IJMTRS | International Journal of Management and Technology for Research Studies
| Volume 06, Issue 02, August 2023, Number 20238-1

Automated and Self Powered Solar Panel Cleaning Robot

LINH HO KHANH, MINH NGUYEN LE, QUANG CAO NHAT, PHU NGUYEN THANH, PHUC TRAN THIEN, SURENDER RANGARAJU, VY DAN LE, NGOC NGUYEN THI HONG

Abstract— **This comprehensive report encompasses a multifaceted project focused on enhancing solar panel maintenance through robotics, image processing, and innovative control systems. The primary objective was to develop a cutting-edge cleaning robot capable of identifying anomalies on solar panels, ensuring efficient information transmission, optimizing battery management, and providing user-friendly control options.**

The project's challenges encompassed various aspects. The initial hurdle was distinguishing clean and dirty solar panel surfaces, which was addressed by employing sophisticated image processing techniques. Fast Fourier Transform and image enhancement algorithms successfully identified disruptions in the characteristic rectangular pattern, accurately detecting dirty panels.

Efficient information transmission between the robot, server, and Arduino posed another challenge. An optimized data encoding technique enabled seamless real-time video transmission to a mobile app. Integration of the PySerial library established reliable communication with the Arduino for effective data exchange.

Battery management was pivotal for sustaining the robot's functionality. Lithium-ion batteries and a solar panel were utilized to ensure continuous operation. The incorporation of trickle charging during sunlight exposure maintained ample power for extended missions.

The developed system excelled in image processing, information transmission, and battery management. Real-time video transmission provided immersive remote monitoring. Python, OpenCV, Numpy, Arduino, and Raspberry Pi showcased technological versatility.

The robot's modular design facilitated easy maintenance and part replacement. It operated for 8 hours using solar energy and featured both auto and manual modes, with control options via Bluetooth and Wi-Fi.

The project addressed the efficiency challenges of solar panel energy absorption due to dirt accumulation. Methods involving robots for cleaning panels often consume significant water resources. The project aimed to mitigate this challenge using innovative technology.

A solar panel cleaning robot was designed to function without water, guided by an Arduino Due, Bluetooth, and ESP32. The robot's parameters, operating principles, machine elements, and electrical system were elaborated.

This project introduced a mobile application that controls the cleaning robot through WiFi or Bluetooth, offering both manual and automatic modes. The application provided real-time video streaming, a user-friendly interface, and compatibility with solar-powered robots. The application revolutionized cleaning technology, providing unparalleled control and energy efficiency.

Index Terms— **Automatic mode, Battery management, Bluetooth, Cleaning robot, Control application, Dirt detection, Electrical system, Energy efficiency, Fast Fourier Transform, Image processing, Information transmission, Lithium-ion batteries, Machine elements, Manual mode, Modular design, Operating principle, Python, Raspberry Pi, Real-time video, Solar energy, Solar panel, Trickle charging, Turning code, User-friendly interface, WiFi.**

Linh Ho Khanh, Department of Telecommunication Engineering, Faculty of Electrical and Electronic Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam, e-mail: linh.hoak12@hcmut.edu.vn).

Minh Nguyen Le, Department of Computer Engineering, Faculty of Information Technology, Vietnam National University HCMC International University, Ho Chi Minh City, Vietnam, (e-mail: ITITIU20112@student.hcmiu.edu.vn).

Quang Cao Nhat, Department of Robotic, Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam, (e-mail: quang.cao2705@hcmut.edu.vn).

Phu Nguyen Thanh, Department of Computer Engineering, Faculty of Computer Science and Engineering, Vietnam National University HCMC International University, Ho Chi Minh City, Vietnam, (e-mail: Phu.nguyen03022002@hcmut.edu.vn).

Phuc Tran Thien, Department of Computer-Embedded System, Faculty of Electronic and Communication, Ho Chi Minh City University of Science (HCMUS), Ho Chi Minh City, Vietnam, (e-mail: 19200436@student.hcmus.edu.vn).

Surender Rangaraju, SNETEL Technologies, Ho Chi Minh City, Vietnam, (e-mail: surendher23@outlook.com).

Vy Dan Le, SNETEL Technologies, Ho Chi Minh City, Vietnam, (e-mail: vydan.hf@gmail.com).

Ngoc Nguyen Thi Hong, SNETEL Technologies, Ho Chi Minh City, Vietnam, (e-mail: hongngoc2001.work@gmail.com).

I. INTRODUCTION

In response to the increasing adoption of renewable energy sources, particularly solar power, this paper presents the culmination of extensive research and development efforts to create a Solar Panel Cleaning Robot (SPCR), known as the S-Robot. The significance of this project lies in addressing the critical challenges associated with solar panel maintenance, including efficient image processing, seamless information transmission, and innovative battery management.

Traditional methods of manual cleaning have proven time-consuming and impractical, prompting the need for automated solutions. The S-Robot aims to revolutionize the solar panel maintenance landscape by providing a waterless, automated cleaning process. This paper brings together the advancements achieved in image processing, wireless communication, and sustainable energy management to realize this innovative robotic system.

This paper amalgamates various aspects of the project, starting with its primary objectives: anomaly identification on solar panels, real-time video transmission to a mobile app, and extended operational capabilities. The project's focus on efficient image processing techniques, noise reduction, and pattern recognition is critical for accurate anomaly detection. Information transmission is seamlessly enabled through advanced data encoding methods, ensuring reliable real-time video streaming and control commands.

Notably, sustainable battery management is a key enabler of the S-Robot's extended operation. By harnessing Lithium-ion batteries and solar panels for trickle charging, the robot achieves continuous functionality even in outdoor environments with limited sunlight. This integration of cutting-edge technologies, including Arduino boards, Python libraries, and Raspberry Pi, showcases the feasibility and potential of the S-Robot in various practical applications.

Furthermore, the paper introduces the S-Robot's core operational principle-dry cleaning through sweeping, suctioning, and wiping-coupled with its dual modes of operation: autonomous and manual-controlled via a mobile app. The robot's image processing capabilities enhance its autonomous performance by identifying and addressing challenging stains on solar panels.

Ultimately, this paper serves as a comprehensive documentation of the S-Robot project, encompassing image processing, information transmission, and battery management. The development of this robotic solution not only pioneers waterless solar panel cleaning but also underscores the relevance and transformative impact of robotics in addressing real-world challenges associated with renewable energy maintenance.

II. Overview of the System

A. *The general block diagram of the system*

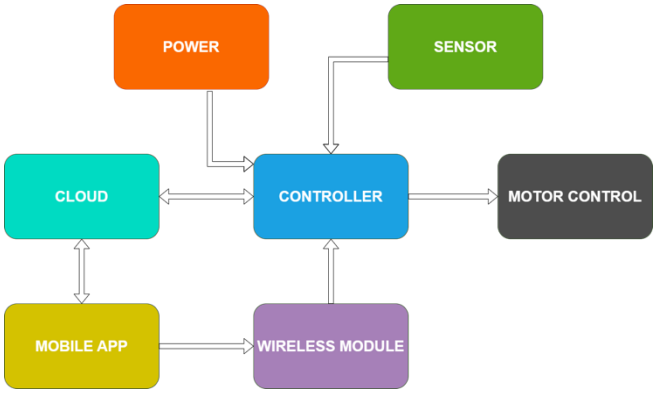

Figure 1 - general block diagram of the system

B. *Detailed block diagram of the system*

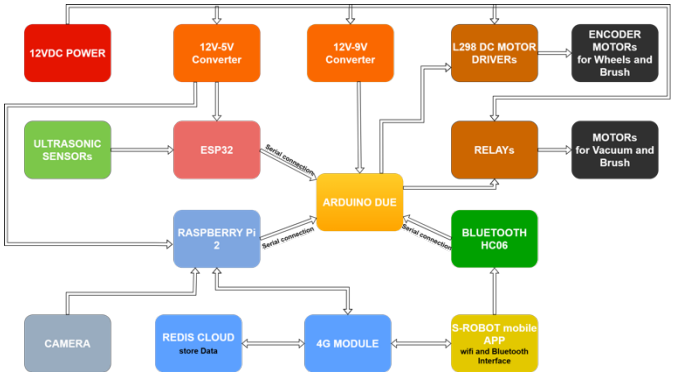

Figure 2 - Detailed block diagram of the system

III. Tools

A. *Arduino IDE*

The Arduino IDE (Integrated Development Environment) is an open-source software application that allows you to program and upload code to Arduino boards. It provides a user-friendly interface for writing, compiling, and uploading code to control various Arduino-based projects. Arduino IDE supports the Arduino programming language, which is based on C and C++.

Figure 3 – Arduino IDE

B. *Python and Python Libraries*

Python is a popular high-level programming language known for its simplicity and versatility. It has a large and active community, making it a preferred choice for a wide range of applications, including robotics, image processing, data analysis, and more. The libraries used were: OpenCV, Numpy, Time, Base64, Matplotlib, PiCamera, PySerial, Redis, Skimage, SciPy, Pandas.

Figure 4 - Python

C. Arduino

Arduino DUE, Arduino Mega: Arduino DUE is to control the robot, the Arduino Mega is for testing code. Arduino DUE and Arduino Mega are two different variants of Arduino boards. The Arduino DUE is based on the Atmel SAM3X8E ARM Cortex-M3 CPU and offers more processing power and memory compared to other Arduino boards. It is suitable for complex projects and tasks that require higher computational capabilities. The Arduino Mega, on the other hand, is based on the ATmega2560 microcontroller and provides a large number of digital and analog I/O pins, making it ideal for projects that require a lot of sensor and actuator connections.

Figure 5 - Mainboard Arduino due

D. Raspberry Pi

Raspberry Pi 2: The Raspberry Pi 2 is a single-board computer developed by the Raspberry Pi Foundation. It features a Broadcom BCM2836 system-on-chip (SoC) with a quad-core ARM Cortex-A7 CPU, 1GB RAM, and various I/O options, including USB ports, HDMI, GPIO pins, Ethernet, and more. The Raspberry Pi 2 is widely used for various computing and IoT projects due to its affordability, small size, and versatility.

Figure 6 - Raspberry pi 2

Raspberry Pi Camera V1: The Raspberry Pi Camera V1 is an official camera module designed specifically for Raspberry Pi boards. It features a 5-megapixel OmniVision OV5647 sensor and can capture still images and record videos at different resolutions. The camera module connects directly to the Raspberry Pi's CSI (Camera Serial Interface) port, making it easy to use for image and video capture in various applications.

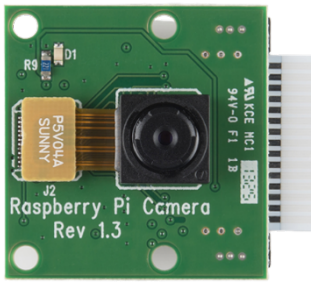

Figure 7 - Raspberry pi camera module

4G module: Used to connect the Raspberry to the Internet, enable it to communicate with the server.

Figure 8 – 4G Module

E. Android Studio

Figure 9 – Android Studio

Android Studio is an integrated development environment (IDE) by Google for creating Android applications. It offers a user-friendly UI designer, efficient code editor, Gradle build system, emulator for testing, debugging tools, version control integration, and support for Kotlin, Java, and C++. Android

Studio simplifies Android app development, making it an ideal choice for developers of all levels.

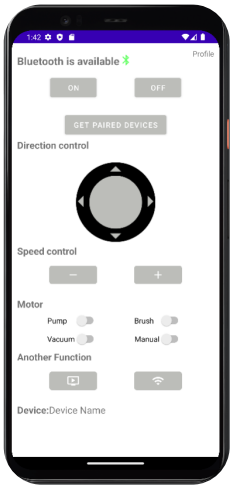

Figure 10 - mobile control app

The mobile application runs on the Android platform and is used to send control signals to the Arduino-based Robot viaBluetooth or Wi-Fi. The app has the flexibility to switch between Bluetooth and Wi-Fi at any time. The maximum

F. Real VNC

Real VNC (Virtual Network Computing) is a remote desktop software that enables users to access and control computers or devices remotely over a network. It allows users to view and interact with a desktop environment from a different location, providing flexibility and convenience for remote troubleshooting, administration, and collaborative work.

Figure 11 – Real VNC

G. Robot Actuators:

Geared Motor with integrated encoder: High-torque geared motor used to operate the robot smoothly without being overloaded. The motor is equipped with an encoder to synchronize the rotational speed of the four motors using PID algorithm, enabling the robot to move in a straight line over long distances.

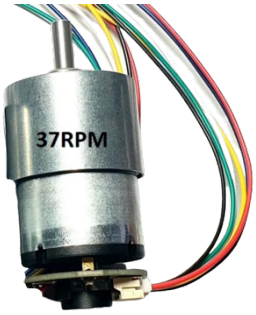

Figure 12 - Geared motor

Geared motor without encoder: Used to rotate the brush for cleaning the solar panels.

Figure 13 - geared motor without encoder

Brushed Motor: Used for the dust suction function of the Robot.

Figure 14 - brushed motor

H. Sensors:

Ultrasonic sensor US-015: Used to detect the edges of the solar panel to ensure that the robot avoids falling during operation, whether in Automatic or Manual mode.

Figure 15 - Ultrasonic sensor

ESP32 devkit V1 30 pins: Used to collect distance data from 4 ultrasonic sensors, then send the distance data to Arduino.

Figure 16 - ESP32 devkit V1

IV. FUNCTION FLOWCHART

- General flowchart of the system

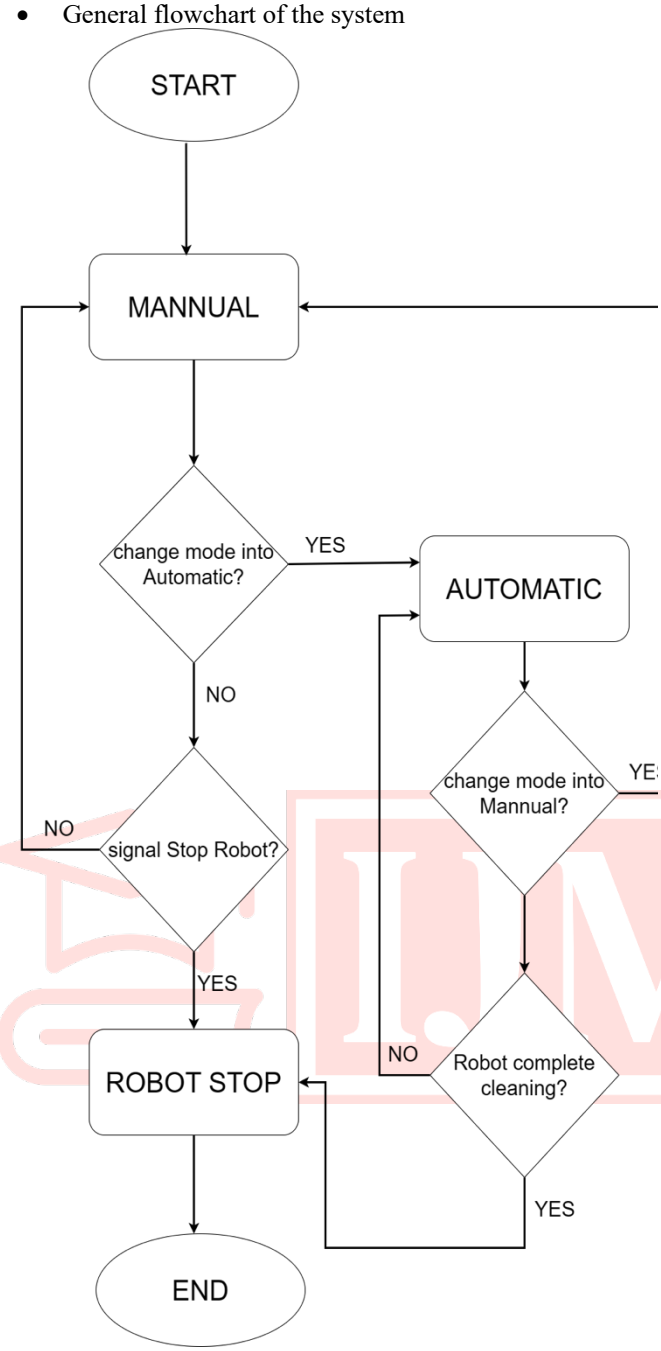

Figure 17 - General flowchart of the system

- Flowchart of the Manual function

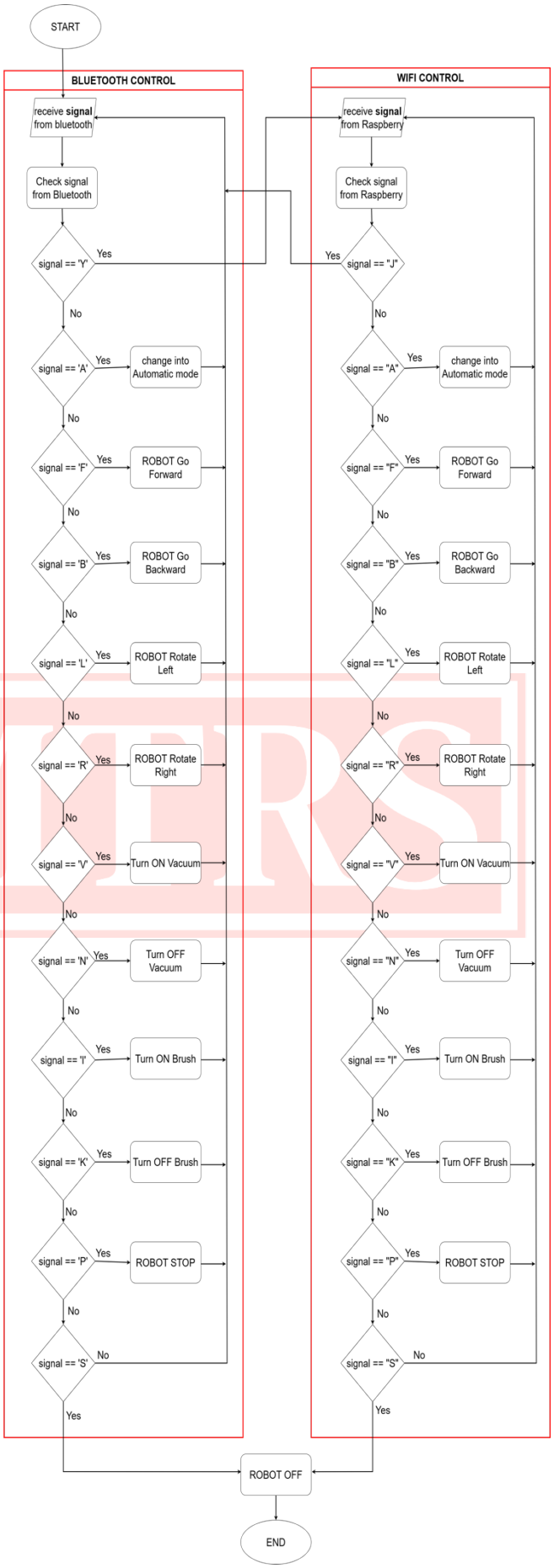

Figure 18 - Flowchart of the manual function

- Flowchart of the Automatic function

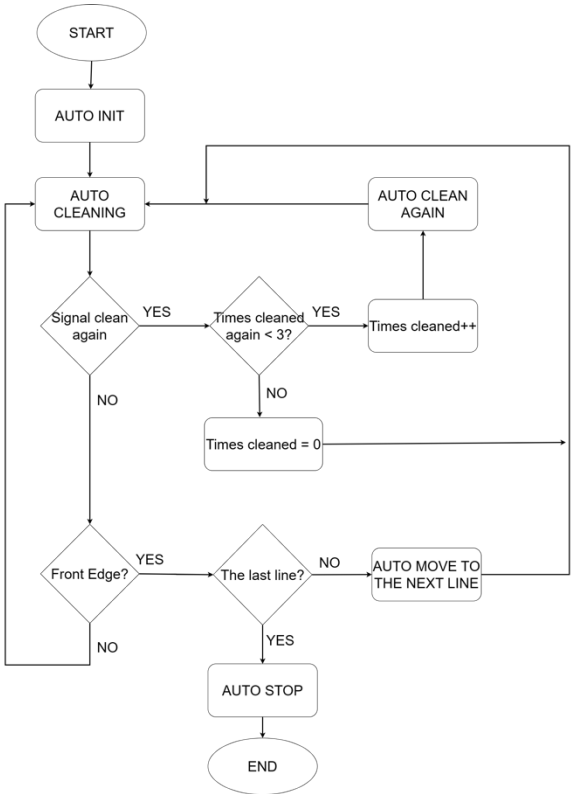

Figure 19 - Flowchart of the Automatic function

- Clarify the block "AUTO CLEANING"

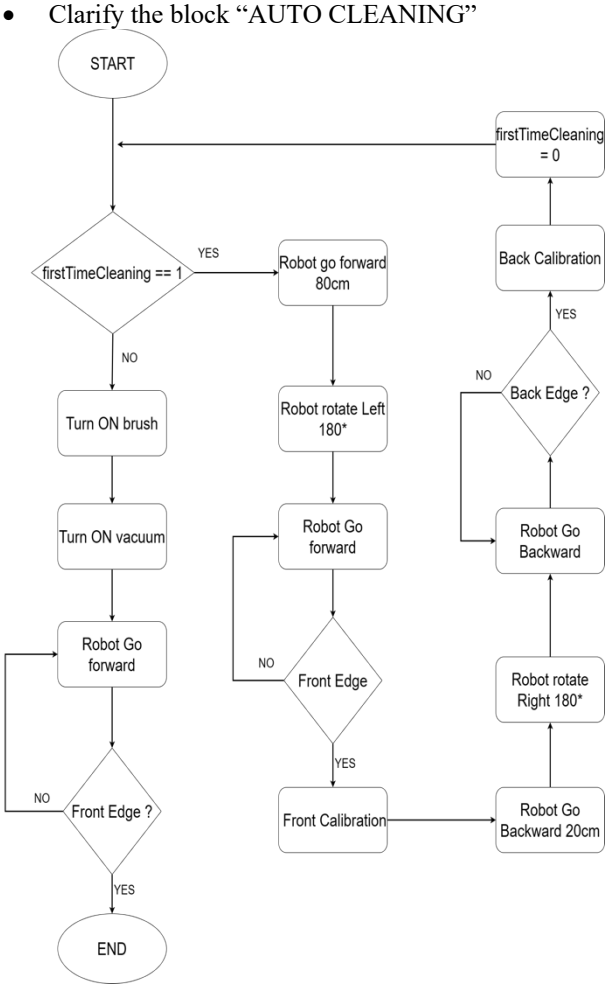

Figure 20 - flowchart of block AUTO CLEANING

- Clarify the block "AUTO MOVE TO THE NEXT LINE"

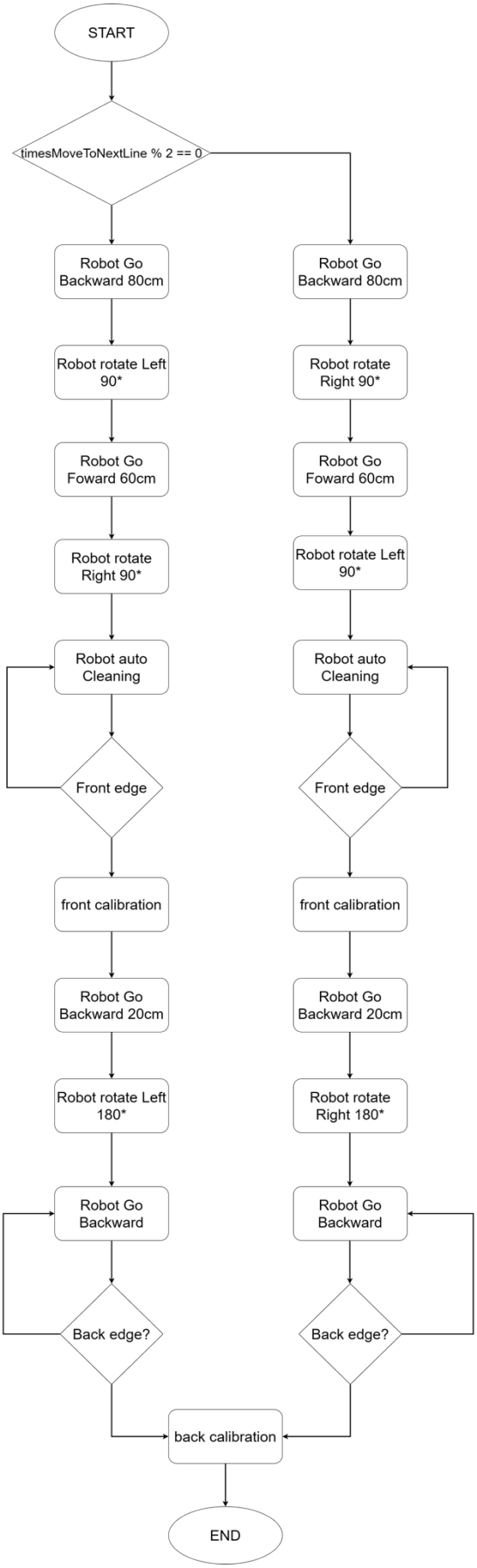

Figure 21 - flowchart of block AUTO MOVE TO THE NEXT LINE

- Clarify the block “AUTO CLEANING AGAIN”

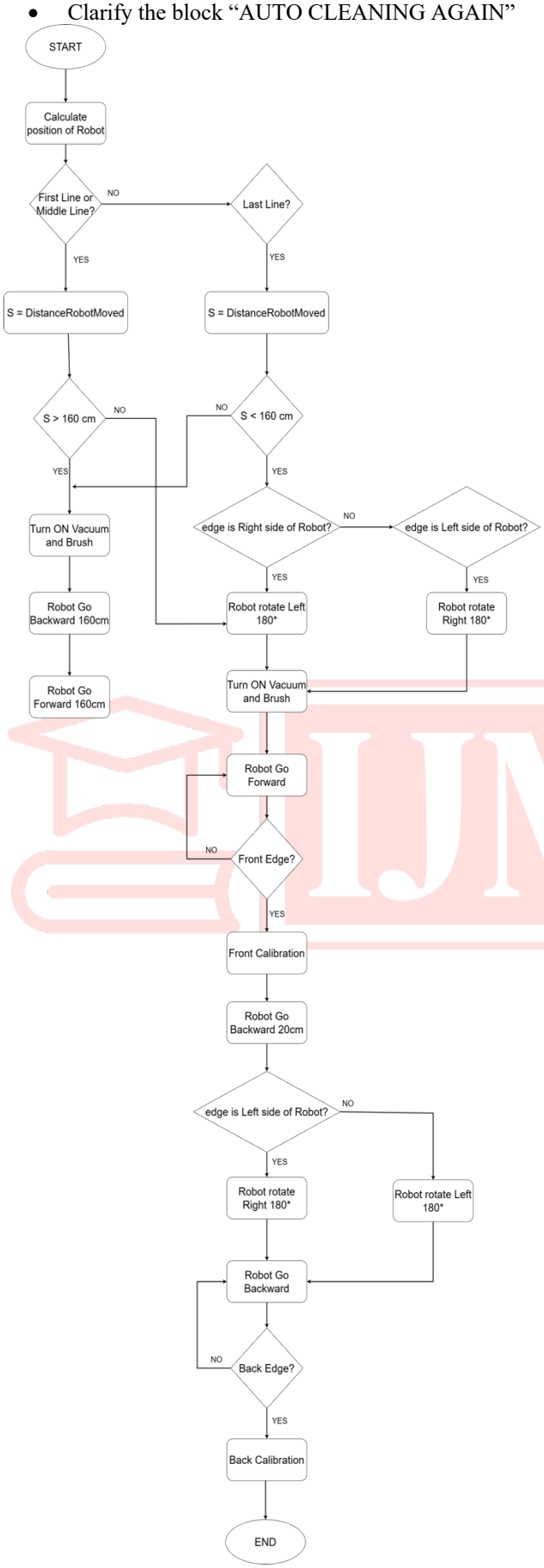

Figure 22 - flowchart of block AUTO CLEANING AGAIN

- General flowchart of the Image Processing system

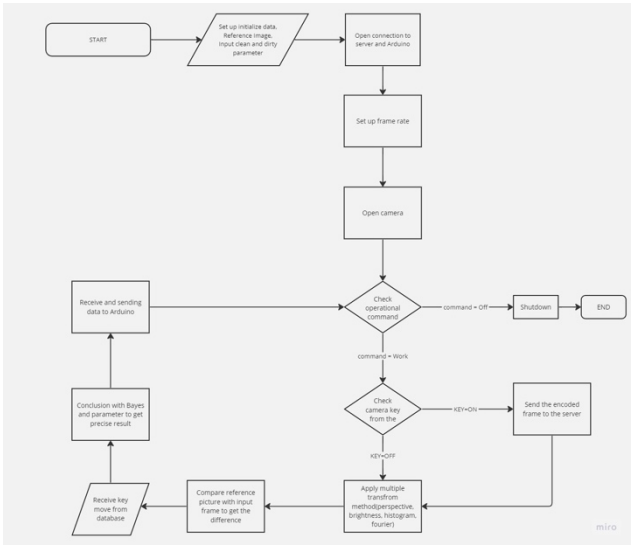

Figure 23 - flowchart of the Image Processing system

- General flowchart of the Control App

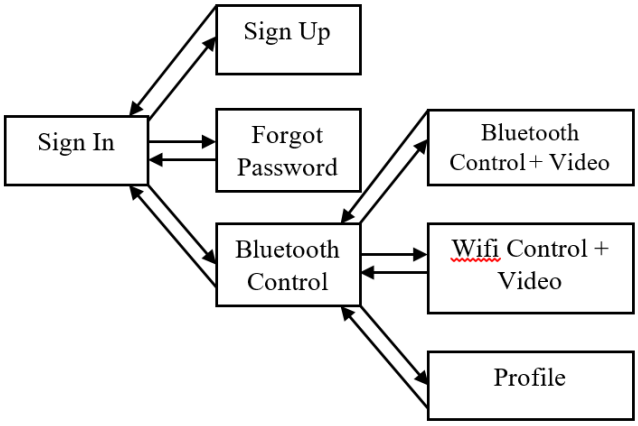

Figure 24 - flowchart of the Control App

V. HARDWARE DESIGNING PROCESSES

A. Learning and researching about the operating principle of the solar panel cleaning robot.

- *Task*

Initially,We have to understand carefully how the SPCR work.We can search the internet to find out the necessary information about the robot. There are several useful *websites and video* [1] thatWe can learn about that kind of robot (appendix 1).

- *Result*

The information show me the operation of each department of the robot. It has 2 main part, there are the body and the brush.

MAIN PART	CARRY
Body	Power supply
	Electrical system
	Distance sensors

[1] The list of websites and videos is in appendix 1.

	Motors for running (2 or 4) and the other component such as shafts, couplings, etc.
	Booster pump
	Pipes
	Pulleys
	Wheels
	Timing belts
Brush	Brush
	Water pipes
	Motor for rotating brush and the other component such as shafts, couplings, etc.
	Timing belts

The information below belong to the most common type of SPCR now:
- Movement system: This is a special kind of robot because it have to work in a glass surface of the solar panels. Moreover, the panel is a incline plane so the robot's percentage of falling down is higher than in a horizontal plane. Therefore, the design of 4 wheels is replaces by 2 timing belts and 1 groups of pulleys-wheels for each belt. This design can reduce the pressure of the robot acting on the glass surface and also increase the friction force to move easily.
- Water pump: Most of the SPCR in the world use water to clean the panel. Because the volume of water it have to use is a large number, it can not contain on its body. The best method is pumping water from the other source. Before going out of the robot, water is pumped one more time by a booster pump to rise the pressure before go outside through 3 to 5 small pipes in front of the robot.
- Brush: The brush is made of snell and have the shape of a cyline. The brush's shaft is parallel to the panel's surface and connecting to driven pulley. The driving pulley connect to the motor and the tramission between two pulley depend on a timing belt.
The operating principle of this kind of SPCR is kindly simple. Water is pumped outside and make the panel wet, then the rotating brush can remove the dust easily. All of dirt will fall down with water.

B. Coming up with the ideas for the new operating principle and design

- *Task*

In the second task,We have to brainstorm to creative a new mechanism:
- Cleaning mechanism: The new one which can clean the panels well without using water.
- The movement system: The robot should be updated to cross the rough terrain.
- Automatic mechanism: It has to work automatically and avoid falling down.
According to that, the robot can help to save water and reduce labour power of human so it will be very cost-effective.

- *Result*

- Cleaning mechanism:Wedecide to remove the booster pump and water pipes. The most serious issue is that if the robot does not use water, the high rotating brush's kinetic energy can make the dust fly around and fall down the panel after the robot leaves. Therefore, the SPCR should hoover and contain the dirt when it works.

To solve that problem,Wecreate one 120W vacuum (P.1) which can be put in the body of the robot. The head of the vacuum is divide into 2 pipes and be assemble close and behind the brush to have the maximum effeciency. When the brush rotates and make the dust fly backward, the vacuum will hoover all of them so the surface of the panels can be cleaned.

Figure 25 – Vacuum Motor

- The movement system: Instead of using the old type of belt (Figure 26),Wedecide to use one more line of wheels under the pulleys for each side (Figure 27). That design make the robot have the 40° slopes in front of and behind the belts. According to that, the SPCR can cross the rough terrain easier. Moreover, the wheel lines - which plays the role as roller supports - are near the center so it can improve the stability and ability to carry more weight of the frame.

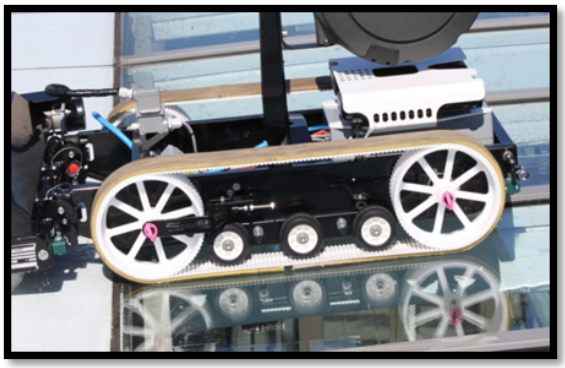

Figure 26 –Previous Belt Types

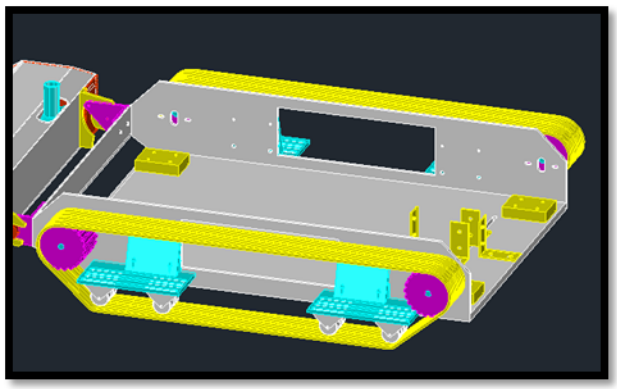

Figure 27 - flowchart of the Control App

- Automatic mechanism: Following the demand, the robot can automatically so it has to have the mechanism which can help itself avoiding falling down. The solution is that we assemble 4 Distance sensors on 4 corners of the robot. We set

up the "safe distance" equal to the distance from the robot to the plane where it works and the tolerance is $\pm$ 50mm. When the robot go to the edge of the panel, the sensor will detect that the distance is longer than the "safe distance" and the SPCR will stop or change the direction.

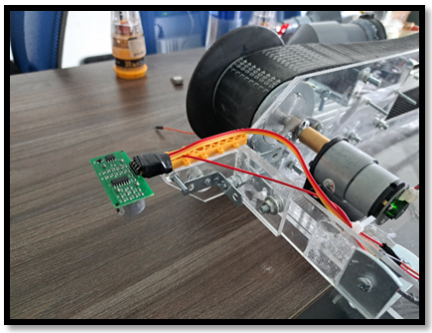

Figure 28 – Sensor Position

- The joints between the brush and the body: It must be stable but also be easy to removable.Wecreacte a mechanism with 2 joints for each side. The big one is the center of the circular motion and the smaller one is used to keep the brush when the robot work. The brush an be taken out of the robot by rotate it up an angle of 90°.

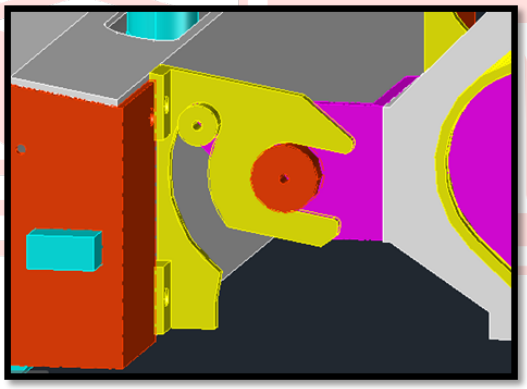

Figure 29 – Joint between the Brush and the Body

C. Designing the frames and choosing the components

- *Task*

After coming up with the ideas,Westar to design the frame for the robot and choose the components for assembling.

- *Result*

- For the frame,Wechoose aluminum with thickness 2mm to make the main frame. The body are made large to carry all of the components. The head of the vacuum is made of PLA plastic by 3D printing.
- For the machine elements,Wechoose the elements for 5 groups of motors:

+ Motor:Wechoose the motor with the speed of 35rpm for movement and 125rpm for the brush. It is enough for the robot to cleam carefully and have strong power.

+ Pulley:Wechoose the aluminum type L pulleys for movement and type XL for rotating the brush.

+ Timing belt: Type L timing belts with large size are chose to move on the panel. Small size type XL pulleys are used for the brush. The length L of the belts are calculate by the formula:

$$L = 2a + \frac{\pi(d_1 + d_2)}{2} + \frac{(d_2 - d_1)^2}{4a}$$

- a: The distance between 2 pulley's centers (mm)
- d_1: The diameter of the first pulley (mm)
- d_1: The diameter of the second pulley (mm)

+ Wheel:Wechoose 8 groups of Ø25mm rubber wheels.

+ Brush: The cyline snell brush is used and the shaft is parallel to the panel's surface.

+ Electrical component: The main control circuit in the robot are belong Arduino. Two another smaller control circuits are used to control the movement system. Bluetooth and wifi circuits are also being used to allow the controller connect and control the SPCR. Four distance sensors are used to avoid falling down and the Raspberry Pi will analyz the photos from the camera to detect the areas which are still dirty.

+ Power supply: The 12V – 20000mAh battery is chose.

+ The other elements such as bearings, shafts, couplings and transistors are chose to be suitable with the previous components.

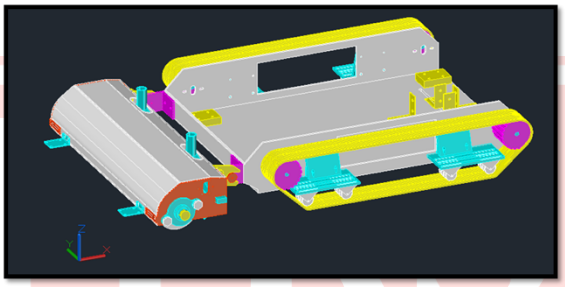

Figure 30 – Overall Design

D. Evaluating the design feasibility and re-design some parts

- *Task*

After surveying and evaluating,Werealize several problems:
- The frame will be heavy and making aluminum are expensive for a propotype.
- The vacuum is weak because the pressure is divide into 2 pipes.
- The timing belt slip out of the pulley when turning
Therefore,Wehave to fix and re-design some parts of the robot.

- *Result*

To solve the problems, our decisions are:
- Frame:Wereplace aluminum by acrylic[2] to reduce the weight. The new issue for that change is that acrylic's strength is not as good as metal soWeincrease the thickness of the main frame up to 5mm. In the parts which are made to assemble the wheels,Weuse more thin iron splints to reinforce. According to those solutions, the frame will be lighter but still be stable.
- Vacuum: For the vacuum,Wedecide to add onr more 120W vacuum motor.

[2] An transparent material

- Timing belts: When the SPCR turn left or right, the belt slip out of the pulleys because of the high friction force between the belt and the panel. To solve the problem,Weput 2 slides which are larger than the area of the pulley for each pulley. It is put close to 2 sides of the pulleys so it can keep the belt in the right position.
- Idler bearing:Werealize that the belt which connect the motor to the brush is still not strained. Therefore,Weput one more idler bearing between 2 pulleys so it can work better.

E. Ordering machine details processing

- *Task*

In the next stage after updating everything, we start to order all of the components that we need to make the SPCR.

- *Result*

Firstly, we search on the internet and connect to the shops and the workshops for the elements by mobile phone, zalo and email. We ask them for the parameters of the components to make sure that they have the product that we need. Secondly, we ask for the quotation and compare between 3 or more shop the have the best choice. Thirdly, we write a payment approval for the manager and wait for his decision. Finally, we order the components and pay for them. Our choices for each type of elements are:
- For the electrical components, we order and buy in various electronic and hi-tech shops[3].
- For the machine elements (Figure 31), we order in the workshops and mechanical shops[4]. There are total 17 technical drawings for the frame.

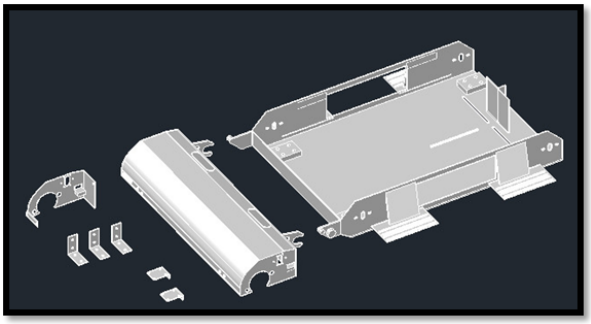

Figure 31 – 3D Design of the frame

F. Assembling hardware

- *Task*

After recieving all of the components, we start to combine it together to be ready for the first trial. The machine elements and the electrical circuit should be completed independently at the same time to save time.

- *Result*

All the parts are assembled in order:
- The machine elements:
  + The frame is assembled first and be kept fixed by M5, M4 and M3 screws and bolts.
  + Motor mounting brackets and the motors are the next part that be assembled.
  + The Ø8mm shafts are connected to the motors by the Ø6mm-Ø8mm couplings. Two supports of each shaft are the bearing of the motor and the Ø8mm bearing which is put on the frame.
  + The L pulleys are kept fixed on the shaft by two M5 screws per pulleys.
  + The wheels are assembled under the splints along 2 sides of the robot.
  + The brush is kept in the brush's frame by a Ø20mm shaft and two Ø20mm bearings. Then the XL pulleys and the belt are assembled.

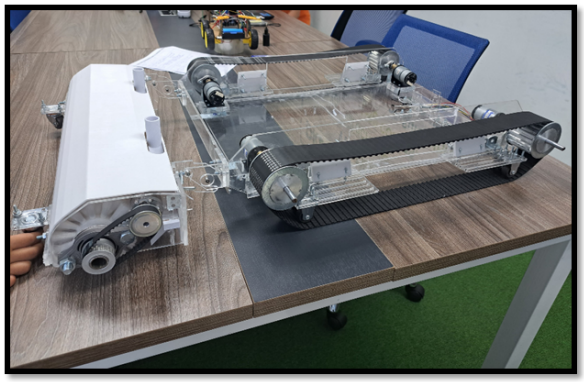

Figure 32 – Assembling the Components

G. Developing the software and the electrical system

- *Task*

The next task is developing the app to control the SPCR and code the program to run it.We are in charge of code the program to recieve the pulses from the encoder of motor, then calculating the pulses to go straight with an unknow distance and turn left and right with an unknow angle. The other members in team carry the other parts of the program.

- *Result*

The perimeter of the pulley is equal to 209.54423mm and the corresponding number of pulses is 1750. We calculate and use the pulses to allow the robot go the distance that it needs to go by the formula:

$$p = \frac{1750 \times d}{209.54423} \ (pulses)$$

- d is the distance we want.
- p is the necessary pulses to go d(mm).

The code for going straight is:

```
#include <iostream>
#include <math.h>
using namespace std;

float Pulse(unsigned long pulse, float distance){
   pulse=0;
   cin>>distance;
   if (distance==0)
     return 0;
   else
     pulse = ceil(1750*distance/209.54423);
   cout << pulse;
}
int main(){
   system("cls");
   unsigned long pulse;
   float distance;
   Pulse(pulse, distance);
}
```

[3] The list of shops is in appendix 2.
[4] The list of workshops is in appendix 3.

- For example: The distance we want is 1604mm and the output is the pulses

$$p = \frac{1750 \times 1604}{209.54423} \approx 13396\ (pulses)$$

+ Input: 1604
+ Output: 13396

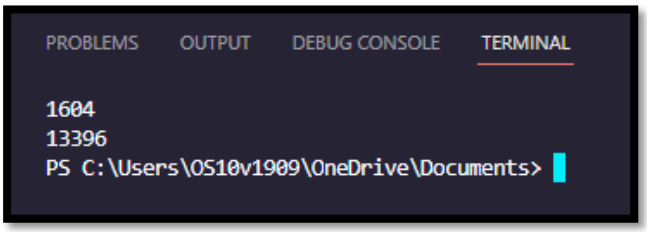

Figure 33 – Result of the calculation

For turning, the SPCR turn around its center. The distances between 4 pulleys are 497.5×481.1842mm so the circumscribed circle around 4 pulleys has the perimeter of 2174.39177mm. Thereforce the distance and the pulses when turning 90° is:

$$d_{90} = \frac{2174.39177}{4} \approx 543.59794\ (mm)$$
$$p = \frac{1750 \times 543.59794}{209.54423} = 4540\ (pulses)$$

The pulses to turn an angle of 90° is 4540. We calculate and use the pulses to allow the robot turn the angle that it needs to do by the formula:

$$p' = \frac{4540 \times \alpha}{90}\ (pulses)$$

- α is the angle we want.
- p' is the necessary pulses to turn angle α (degree).

The code for turning is:

```
#include <iostream>
#include <math.h>
using namespace std;

float Pulse(unsigned long pulse, float angle){
   pulse=0;
   cin>>angle;
   if (angle==0)
     return 0;
   else
     pulse = ceil(4540*angle/90);
   cout << pulse;
}
int main(){
   system("cls");
   unsigned long pulse;
   float angle;
   Pulse(pulse, angle);
}
```

- For example: The angle we want is 16° and the output is the pulses

$$p' = \frac{4540 \times 16}{90} \approx 808\ (pulses)$$

+ Input: 16
+ Output: 808

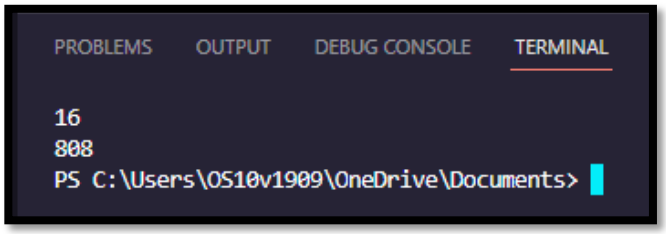

Figure 34 - Result of the calculation

The robot is coded to have capable to work in 2 mode manual running and automatical running. The mobile app (P.11 and P.12) is built to meet both the requirements.

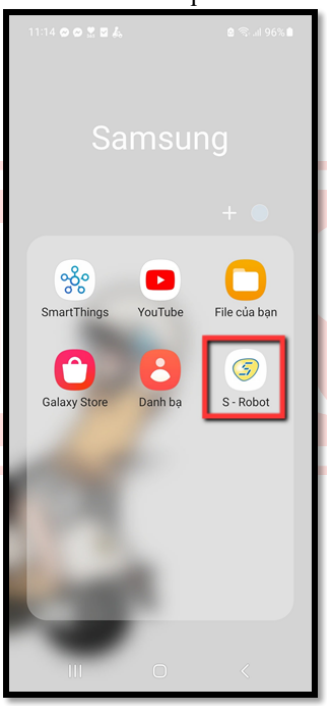

Figure 35 – S-Robot App

Figure 36 – Bluetooth Control Interface

H. Connecting the electrical system and updating the programs

- *Task*

After completing the program, the components of the electrical system are connected together and the program is uploaded to the control circuit.

- *Result*

We start with connecting all of the wires (Figure 37) between the elements like the circuit below (Figure 38)

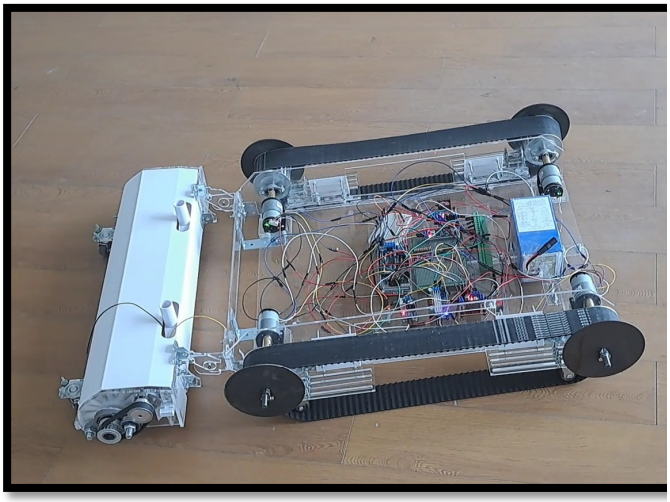

Figure 37 – Add on the electrical system

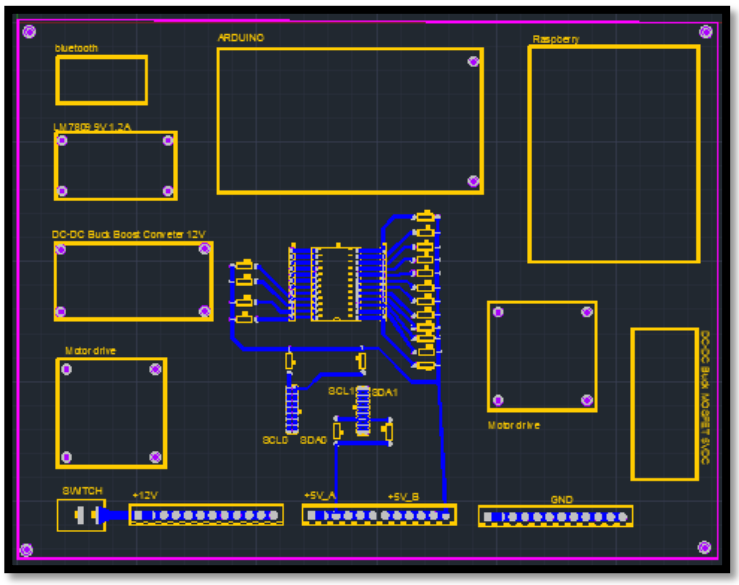

Figure 38 – Circuit

Then we upload the code from VS-Code and Arduino IDE to the circuit by USB-micro wire.

I. Operating the robot and solving the problems

- *Task*

At the final stages, we operate the robot for the first trial to realize and fix the errors before completing the project.

- *Result*

We run the SPCR and test with wide range of test cases:
- Measures:
Measure the robot's weight and size. solar panel's size, height, tilted angle. The thickness and the material of the solar panel's glass. The pillar's endurance
- Solar Panel:
+ Checking the solar panel surface material, get the thickness and what kind of class is that:
  - Step 1: Compare the weight between the robot and the endurance of the solar panel surface
  - Step 2: Checking by increase the weight putting on until it reach the robot weight and the solar panel limit

+ Checking the distribution of the pillars bearing the solar panels and how many there are for 1 solar panel
+ Checking how the efficiency changes after cleaning:
  - Step 1: Check how much the energy that the solar panel produces before cleaning(around 10 minutes). Check the efficiency of the panel .
  - Step 2: Let the robot running (auto/manual, meaning double the steps for this efficiency test).
  - Step 3: checking the watt again to get the different, check the efficiency again.

- Move freely:
+ Rotation angle when turning 180
+ Size of the curve when turning
- Friction force between the SPCR and the solar panel:
+ Step 1: Putting weight on the robotbelt, tilt the solar panel until it slips
+ Step 2: Calculate friction coefficient
+ Step 3: Selection the right belt for the robot

+ Step 4: Testing in go up and go down how the friction affect the robotrun
- Picture:
+ Can the camera take the picture and analyze it to detect the dirty?
+ Can robot stream video?
- For image processing
+ Step 1: Identify the hard stubborn dirt that the brush and the vacuum can not clean
+ Step 2: Let the image of it being processed
+ Step 3: Calculate the time it take to clean one
+ Step 4: Identify what hard stubborn dirt left and what have been clean
+ Step 5: Calculate the percentage to get how much it had cleaned
- For camera streaming:
+ Step 1: calculate the time for startup
+ Step 2: Checking do the robotreal-time response(robotmove user see the video run)
+ Step 3: Check video quality
- For the angle of the camera:
+ Step 1: Get the zone will take the picture
+ Step 2: Calculate the height where to place camera
+ Step 3: Modify the code for the angle
+ Step 4: Checking the image resolution
- Ability to work at night
- Bluetooth mode:
+ Step 1: Connection to robot with bluetooth
+ Step 2: Checking the maximum range it will disconnect
+ Step 3: Checking can it control the robotif there is a wall between connection
+ Step 4: See if the robotwill stop or switch to being controlled by wifi if disconnected.
- Response time of bluetooth:
+ Step 1: control upfront to take the maximum response time
+ Step 2: pull back to monitor that does the distance affect the delay time in control robot
+ Step 3: checking if there is any possible thing in the field can block the bluetooth signal to send
- Response time wifi:
+ Step 1: Control the robotnear the wifi source to get the fastest respond
+ Step 2: Control the robot on the field to get the delay time
- Brushed:
+ Step 1: Checking what dust it can clean
+ Step 2: Can cleaning the small dust
+ Step 3: Consider how big the dirt it can cleaning
+ Step 4: If there is a big dirt then it stop and send signal to the app tell the person to go there and remove the object
- Temperature
+ Step 1: Taking the solar panel's highest and lowest heat to get the reference
+ Step 2: Taking temperature before working at outside
+ Step 3: Let it work outside to see if it work normally, after certain time(around 20-30 minutes) take the temperature again
+ Step 4: If it works normally, then check how long it will work normally in the field to test how it performs in the real environment under the sun heat. There will be 2 possible outcomes
+ Step 5: Taking temperature again to get the temperature that has been affected by the robot, checking if it exceeds the highest temperature of the solar panel or not.
+ Step 6: Modify anything if it necessary
- Climbing
+ The highest hump that the robotcan pass through
+ The shock impact that the robotcan take when passing through the hump
- Stopping
+ Stop when needed
+ Stop when if something on solar panel is broken
+ Stop if something inside the is malfunction
+ Stop and alert the user if there are any stubborn dirt that the robotcan't clean
- Battery
+ Should work for 4-5 hours
- Safet function:
+ Auto shutdown when short circuit
+ Re-open after a short time
- Speed
+ Step 1: Get the fastest time and the slowest time that the robot have on flat surface at tilt angle 0 degree
+ Step 2: Get the fastest time and the slowest time that the robot have when run on solar panel
+ Step 3: Calculate that it should clean X (m^2) in Y (hours) depend on the speed
+ Step 4: Identify the problem that may affect the default speed of the robotfor improvement such as wind, the solar panel surface, tilt angle.
- Automatical mode:
+ Step 1: Putting the robot on the right angle to start
+ Step 2: Calculate the time it take to clean 1 round
- Manual mode:
+ Step 1: Putting the robot on the right angle to start
+ Step 2: Calculate the time a human will clean 1 round with robot's maximum speed
+ Step 3: Compare between manual and automatic to see which is better in time

VI. App Design

A. Sign In

- Initialize Firebase Authentication, different views, SharedPreferences.
- Check if user is logged in by checking SharedPreferences
- If not, you must log in with your registered email and password
- In the login function, check if there is a place that has not been entered, a message asking the user to enter is displayed
- After checking, log in with Firebase Authentication
- If checkbox is checked, save login status to SharedPreferences
- After successful login, switch to Bluetooth Control
- If the login is not successful, a message will be displayed
- There are also textviews to switch to Sign Up, Forgot Password function

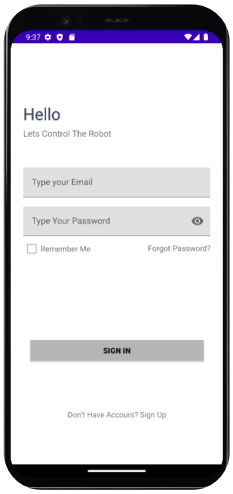

Figure 39 – Sign-in Interface

B. Sign Up

- Consists of 2 parts: Enter personal information and register for an account

1 Personal information

- Initialize the necessary components
- The data to enter includes: Frist Name, Last Name, Birth, Gender, Phone
- With Birth data when clicked will create a Calendar object that takes the current date and year. The user can select the date and after clicking OK, the selected date will be displayed in the EditText.
- With Gender data when clicked, a dialog box will appear for the user to select the gender.
- With Phone data, it will check if the phone number is not enough 10 numbers, it will be invalidated
- After clicking to fill in all the information, click Next to continue. If the data is left blank, a message will be displayed.
- After fully entered, the data is taken to the next place to save to FireStore

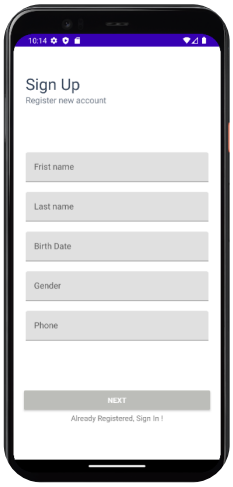

Figure 40 – Sign-up Interface

2 Sign Up

- Initialize Firebase Authentication and Firebase Firestore
- Check Email, Password is valid or not. If not, ask to re-enter. If valid then register user and save information to Firestore.
- When saving data to Firestore Create a document with User Authentication ID as Document ID. Then save the user information to the document.
- If the registration and saving is successful, transfer the user to Bluetooth Control.

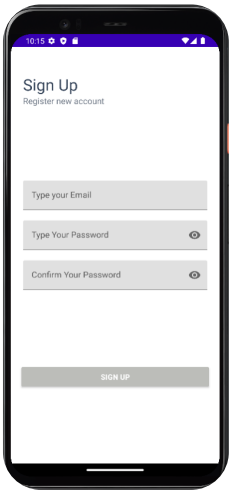

Figure 41 - flowchart of the Control App

C. Forgot Password

- ForgotPass is an Activity to request password reset via email using Firebase Authentication. User enters email into EditText and presses "Reset Password". If the email is valid, a password reset request will be sent via Firebase and a success message will be displayed. Otherwise, display an error message.

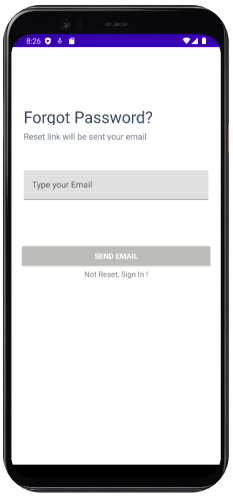

Figure 42 – Forgot Password Interface

D. Bluetooth Control

- Ask the user for permission
- Includes Joystick and Button Components
- Turn Bluetooth on and off via push buttons
- Connect to device connectToDevice()
- Show and select device showDeviceSelectionDialog()
- Send signal writeMessage(String message)

1 Joysticks:

When the user touches the layoutJoystick (ACTION_DOWN), the baseX and baseY variables are set to store the initial offset between the touch position and the center of the joystick. This is done to ensure that the joystick moves smoothly relative to the touch position.

When the user moves their finger on the joystick (ACTION_MOVE), the position of the joystick (joystickX and joystickY) is updated based on the touch position relative to the center of the joystick. The distance and angle between the touch position and the center of the joystick are calculated.

If the distance between the touch position and the center of the joystick exceeds a certain threshold (centerX), the

position of the joystick will be limited to the circumference of the circle (centerX) by trigonometry.
The position of the joystick (joystickX and joystickY) is updated and the view joystick mode is moved to the new position.
DirectionAngle is calculated to determine the angle of the joystick in degrees. This method calculateControlState(directionAngle) is called to convert the angle into some control state (dataTosend).
If the control state (dataTosend) is different from the previous state (not equal to "A"), it will write the new control state to an external device or system using the writeMessage(dataTosend) method.
When the user lifts their finger from the joystick (ACTION_UP), the joystick will animate back to its center position. It sets dataTosend to "P" to indicate that the joystick is in the neutral position.
2 Buttons:
Turn on Bluetooth
When the user taps the "Enable Bluetooth" button (mOnBtn), if Bluetooth is not enabled, the app will display a short message (toast) informing that Bluetooth is enabled and asking the user for permission to enable Bluetooth via a Intents. If Bluetooth is enabled, the application will display another message stating that Bluetooth was previously enabled.
Turn off Bluetooth
When the user taps the "Turn off Bluetooth" button (mOffBtn), if Bluetooth is on, the app closes the Bluetooth socket connection (if available), then turns off Bluetooth and displays a message (toast) saying Bluetooth is off. If Bluetooth is turned off, the application will display another message stating that Bluetooth was previously disabled.
Select device
First, the method checks if Bluetooth is enabled using mBlueAdapter.isEnabled(). The mBlueAdapter variable is a BluetoothAdapter object, which represents Bluetooth on the Android device.
If Bluetooth is enabled, the application will perform an action to display a dialog to let the user select a Bluetooth device from the list of previously paired devices. To do this, the showDeviceSelectionDialog() method is called. In this case, the dialog will contain the paired devices and the user can select a device from the list.
If Bluetooth is not enabled, the application will display a short message (toast) asking the user to enable Bluetooth before selecting a device.
connectToDevice()
Creates a Bluetooth socket to connect to the selected device from the list of previously paired devices, using a predefined UUID.
Connect to the selected Bluetooth device via the newly created socket.
Get the OutputStream object from the socket to send data from the application to the device.
Send two strings "J" and "M" to the selected Bluetooth device, possibly to control or select the "Manual" mode on the device.
3. Other methods
writeMessage(String message)
The "writeMessage" method is a function in Android to send a message via Bluetooth from your Android device to another Bluetooth device.
Check if the Bluetooth connection exists and works.
Send message body as byte array over Bluetooth connection.
Display a success message and record the sent content.
If the connection does not exist or an error occurs, display an error message and close the Bluetooth connection.
showDeviceSelectionDialog()
The "showDeviceSelectionDialog" method in Android displays a dialog box that allows the user to select a paired Bluetooth device from a list of previously paired Bluetooth devices.
Set<BluetoothDevice> devices = mBlueAdapter.getBondedDevices(); : First, this method gets a list of previously paired Bluetooth devices using BluetoothAdapter (mBlueAdapter). These devices have been paired in the past, so there is no need to search for new devices.
if (devices.size() > 0) { ... }: Then the method checks if at least one device is paired.
If there is a paired device, the method will perform the following steps to display the device selection dialog:
Creates an ArrayList (deviceList) to store paired BluetoothDevice objects. Creates an array of deviceNames to store the names of the paired devices as an array of Strings.
Browse through the list of paired devices and add them to the deviceList, and store the name of each device in the deviceNames array.
Display an AlertDialog that allows the user to select a device: Create an AlertDialog.Builder to build the dialog. Set the title of the dialog box to "Select a device". Use setItems to display a list of paired device names in the deviceNames array.
Define a listener (OnClickListener) to handle the event when the user selects a device. When the user selects a device, the method will record the selected device to the mDevice variable, then display the message "Connecting..." and call the "connectToDevice()" method to connect to the selected Bluetooth device. Finally, display the AlertDialog dialog box on the screen.
If there is no paired device, the method will display a toast message saying "No paired devices found".
After the user selects a device from the dialog, the application makes a connection to the selected Bluetooth device and displays the name of the selected device on the user interface.
calculateControlState(double angle)
This function takes an angle (in degrees) as input and calculates the direction of travel based on the position of the joystick. It returns a single character representing the direction of travel (R: Right, B: Bottom, L: Left, F: Forward, P: Pause).
onPause()
This method is called when the application's activity is stopped or lost its "foreground" state. In this case, the main task of the method is to save the Bluetooth connection state to SharedPreferences so that the state can be restored after the application resumes running.
First, the method calls the superclass "super.onPause()" to perform the superclass's onPause action (if any).
Next, the method checks if the Bluetooth connection (mSocket) exists and is connecting (isConnected()). If true, it

saves the connection state to SharedPreferences through an Editor object (editor).
Finally, the method calls editor.apply() to apply the saved changes to the SharedPreferences.

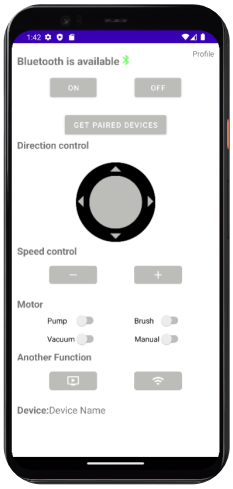

Figure 43 – Bluetooth Control Interface

E. Wifi + Video
- Initialize Redis
- Send on signal to camera key and Y to key move to start control
- Initialize RedisConnectionTask running in the Background.In which reads the data from Redis the key battery, water, time, camera, video.
- The way the button controls is similar to the direction and speed of the bluetooth control, only the way to send data is different
- When closed, send a signal to turn off the camera and send to key move the letter J to change the control and close the Redis connection
receiveRealtimeFrame()
This method is designed to run in a separate thread, as it uses runOnUiThread() to update UI elements.
List<String> fullFrameData = new ArrayList<>(); This creates an ArrayList called fullFrameData to store individual blocks of video frame data.
String frameForDecode = ""; Initialize an empty string called frameForDecode to store the concatenated video frame data.
This method is structured as a while loop (while (isRunning)) runs continuously as long as isRunningboolean is true. This loop receives and processes real-time video frames.
Inside the loop, the method queries Redis to get various pieces of data, such as battery percentage, water level, and time. It also retrieves a control flag named cameraControl.
If cameraControl is not null and equals "on", it means the camera is enabled and this method will continue to receive video frame data from Redis.
This method receives the video frame data in chunks and stores it in the fullFrameData list until it encounters a special chunk called "last frame". When the end of the frame is reached, the blocks in the fullFrameData are concatenated into frameForDecode.
Bitmap videoFrame = decodeFrame(frameForDecode); The frameForDecode connection is passed to a method named decodeFrame, which is expected to decode the frame data and return a Bitmap object representing the video frame.
If the videoFrame is not null, it means the frame has been decoded successfully and this method will keep updating the ImageView in the Android UI with the newly received video frame.
The DataTextView is also updated with battery, water and time values.
fullFrameData.clear(); After processing one frame, the fullFrameData list is cleared in preparation for the next frame.
RedisConnectionTask
Define a nested class RedisConnectionTask that extends AsyncTask<Void, Void, Void>. In doInBackground do: Connect to Redis server using Jedis with specified server, port and password. Implement receiveRealtimeFrame() to read the frame. Close the Redis connection with jedis.close(). If IOException occurs, log the error message and throw the RuntimeException. The above implementation uses AsyncTask to avoid blocking the main UI thread.
- Also use a RedisConnection class to read and write values on Redis.
- In the RedisConnection class include:
RedisConnection(String redisHost, int redisPort, String redisPassword)
Make a connection to Redis using Host, Port and Password
sendSignal(String key, String value)
Check if there is a connection to Redis. If so, use the set command to send a signal to the desired key. If not, it will show on the screen
receiveSignal(String key)
Reads the value from the specified key and returns the value of the key
disconnect()
Close the Redis connection

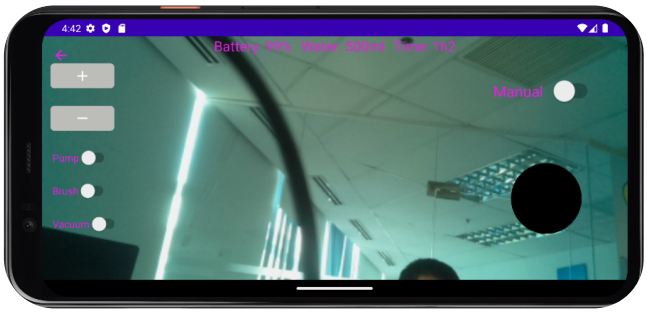

Figure 44 – Wifi Control and Viewing Interface

F. Bluetooth Control + Video
Has the same features as bluetooth control and can watch videos similar to Wifi video.

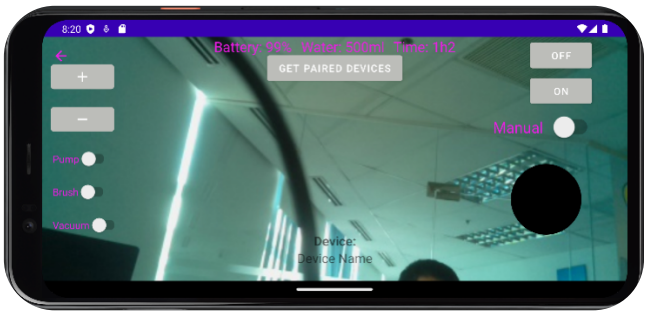

Figure 45 – Bluetooth Control and Viewing Interface

G. Profile
- Initialize Firebase Authentication and Firebase Firestore
- Read UserId from there get data saved from Firestore. Then display the Textview

- There is a function to change the avatar image. The image is then saved in the Firestore so that it appears the next time you log in.
- Can change information such as Frist Name, Last Name, Birth, Gender, Phone.

readUserDataFromFirestore(String userId)
- First, the function initializes a Firestore object by calling FirebaseFirestore.getInstance().
- Then it accesses the collection "users" and the document has the id corresponding to the userId.
- Next, it calls the get() method to get the document's data.
- If the document exists (documentSnapshot.exists() returns true), the function will get the user information from the document and display them to the user interface.
- If the document is not found, it will display a message to the user informing that the document does not exist.
- If there is any error during data reading, the function will display an error message.

showEditDialog()
- First, the function creates a new Dialog object, associated with Profile.this, where this is an Activity containing this function.
- Next, the function uses the setContentView(R.layout.edit_profile_dialog) method to set the appearance of the dialog from the edit_profile_dialog.xml file.
- The function finds views in the dialog interface using the findViewById method.
- Then it extracts the current user information from the TextViews in the UI (nameTv, birthTv, phoneTv, genderTv) and displays it to the corresponding EditTexts in the dialog to allow the user to edit.
- For the gender field (editGender), the function creates a selection dialog (AlertDialog) with gender options ("Male", "Female", "Other"). When the user selects an option, the value is displayed in the EditText editGender.
- For the date of birth field (editBirth), when the user clicks it, a date picker dialog (DatePickerDialog) is displayed allowing the user to select the date. After the user selects the date, the date is displayed in the EditText editBirth.
- When the user clicks the "Save" button (btnSave), the function takes new data from the EditText and stores them into new variables (newFirstName, newLastName, newBirth, newPhone, newGender).

VII. Problems

A. How to differentiate between a clean surface of the solar panel and a dirty one?

What are the characteristics of a clean surface, and a dirty one?

What about the noises such as: overall brightness (too bright or too dark), reflection of the sun (or a strong source of light), etc…?

What methods do we use to decide whether it's clean or not?

B. How to send and receive signal to the server and to the Arduino to control the robot?

To the server: The server is a free server so it's feature are limited (80KB at a time for 1 key). Send and receive control signal here is easy because we only use a few characters for 1 command but for an entire video frame then it's a problem.

To the Arduino:

How the Raspberry find the port of the Arduino to connect?

What kind of data will it send to the Arduino?

It only receives data through one line of connection that is the USB UART Serial port, how will it differentiate the data that the Arduino collected and send to it such as: temperature, the remaining battery capacity, the distance the robot has moved, etc…?

What if we plug out the Arduino to re-embedding another code and then plug it back in again, will it still work?

C. How do we receive data from the server and display it out on the screen of the app?

What data type do we receive from the server?

How do we convert that data to video frame to show?

D. How the battery system fit in?

What kind of battery will we use, how many source of battery, how much is the capacity?

How do we know if it will be capable to work constantly for 4 hours?

How do we utilize a solar panel to maintain the battery capacity?

E. Raspberry Pi 2 hardware limitations

Weak processor

Does not come with webcam like computer

Raspberry pi connection port to Arduino is different from computer

F. How to control the robot using Arduino?

Controlling the robot to move in a straight line over a long distance is a significant challenge.

Precisely controlling the robot to turn at a predetermined angle is another challenge. Although encoders are used for turning, they have some inaccuracies, and the robot's wheels are affected by surface friction on the solar panel.

The robot must clean the entire surface of the solar panel without missing any areas.

G. How to make the control app works smoothly?

When sending data over Bluetooth, an Android app will initially open a Bluetooth connection, then close the connection after the data has been sent. This slows down the process of sending data over Bluetooth and can cause the app to crash unexpectedly. Similarly, opening and closing connections frequently can also slow down data transfer over Wi-Fi.

The cause of this problem is that opening a connection and performing network activities on the main thread of an Android app can cause the user interface to freeze when the app performs long-running network activities.

VIII. Solving The Problems

A. *The characteristics of a solar panel*

If it's clean, the solar panel have a pattern of repeating rectangular shape and white lines between shapes. If it gets dirty, that pattern got disrupted. So if we can transform the image taken to focus on the pattern, it can recognize the disruptions, thus we chose to Fast Fourier Transform the frames taken from the camera and then compare it to the also Fourier Transformed image of a clean solar panel surface.

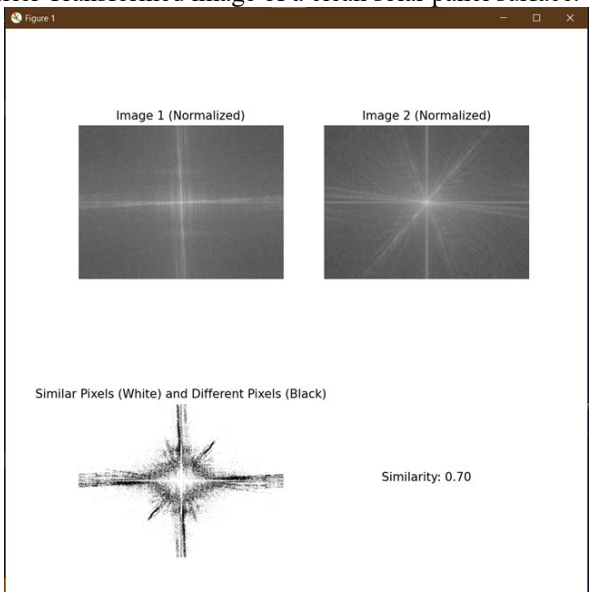

Figure 46 – Analizing two image

To cancel to noises, before we do the Fast Fourier Transform on the frame and the reference image, we must first do an overall average brightness adjustment, a threshold filter to eliminate parts where it's too bright since it's definitely the reflection of the sun or a powerful source of light, then we do a histogram equalization on the frame and that is enough to cancel most of the noise that largely affect the result.

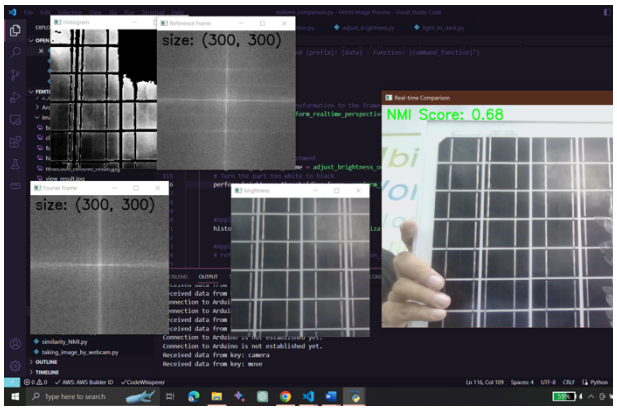

Figure 47 – Warp Perspective and Add on Noise Filter

In order to determine the cleanliness of the captured frame and assess its similarity to the reference image, we employ a Fourier Transform on both images and subsequently normalize the results. This normalization process assigns each pixel a value between 0 (representing black) and 1 (representing white). We then apply a filter, setting pixels below 0.5 to 0, and vice versa. Following the pre-calculation, we compute three key metrics: the structural similarity index (SSIM), the Root Mean Square Error (RMSE), and the Normalized Mutual Information (NMI). To consolidate these metrics into a single score called CombineScore, we employ the formula:

$$CombineScore = \sqrt[3]{SSIM * NMI * (1 - RMSE)}$$

With this CombineScore, we conduct data collection by capturing frames of both clean and dirty solar panel surfaces. Using this data, we calculate the mean and standard deviation of the CombineScore for the frames taken. Utilizing Bayes Rule, we can then classify whether a captured frame belongs to the "clean class" or the "dirty class." This classification enables us to differentiate between clean and dirty solar panel surfaces based on the images taken by the camera. The process remains unchanged, ensuring the accuracy of our determination regarding the cleanliness of the solar panels.

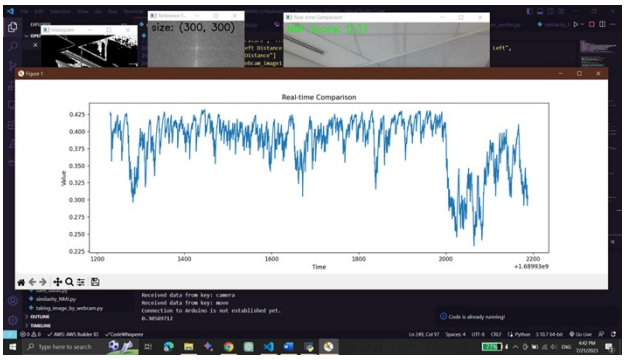

Figure 48 – Data Collection

B. *We used specific encoding to send and receive message*

To the server: We encode the frame taken into a string of characters and then split it up into chunks, each chunk (which is lower than 80KBs) will be adding a prefix and a suffix to ensure the recombination and the decoding of that frame on the receiving end.

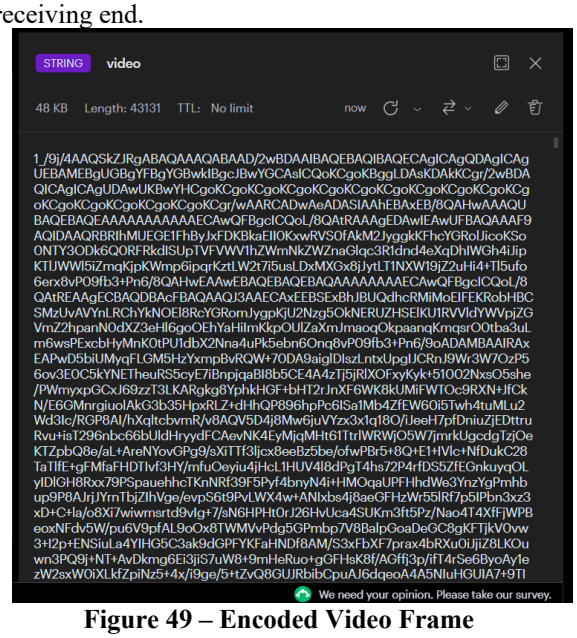

Figure 49 – Encoded Video Frame

To the Arduino:

We use the PySerial library to find the port and port description, and pre-chosen the desire description, for Arduino DUE, it's always have "Arduino" in its

description but for a Raspberry the port for an Arduino is usually contains “ACM” in it.

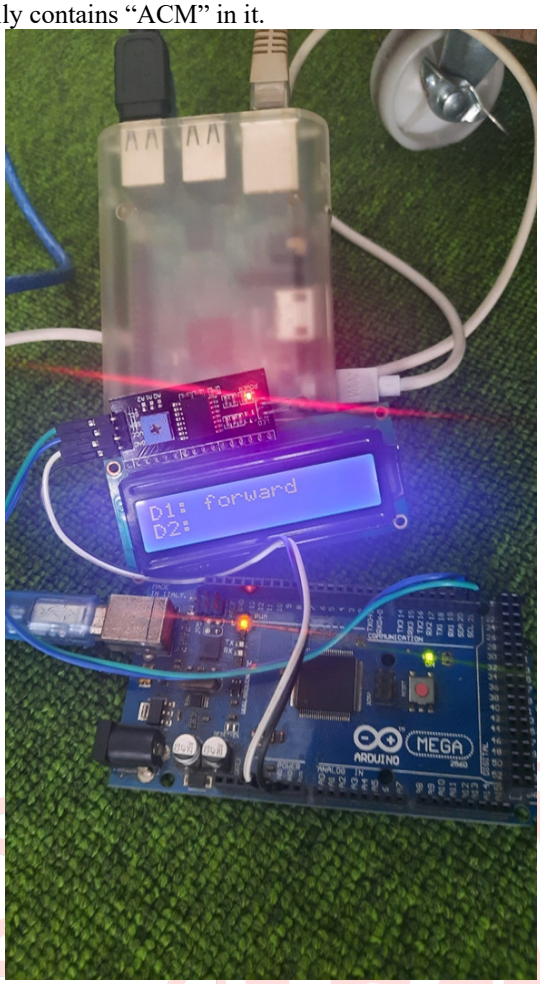

Figure 50 – Sending Signal to Arduino

The data the Raspberry sends to Arduino will be characters and they will be encoded by adding some prefix and suffix, thus also allow us to send different kind of data using only one line of connection.

When we plug the Arduino in and out of the Raspberry, the code that the Raspberry run on has a confirmation part where it checks whether there is connection or not, if there is no connection it will still run the code but don’t send any data through the line, if there is a new connection it will establish the connection with the new device and send data to it.

C. To display the video on the screen of the app?

First the app sent a signal to the server, the Raspberry then read that signal and start to send the video frame to the server. Then we set the app to constantly receive the chunks of strings that we’ve encoded the frame into from the server and decode it, utilizing the encoding method we have perform on the string to ensure the chunks we get belong to the same frame before decoding it back to images.

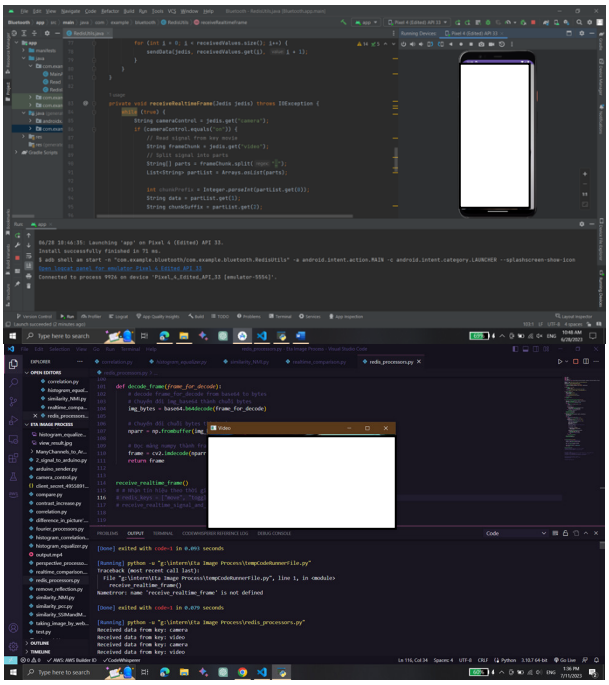

Figure 51 – Sending Video to the App

D. Battery system

Lithium-ion (Li-ion) batteries are commonly used due to their high energy density. We need three 12V batteries, the first one with a capacity of 20Ah for the main control circuit and main motor, 2 batteries with a capacity of 3.6Ah for the 2 vacuum cleaners.

Ensuring constant operation for 4 hours: The average power consumption of the robot is determined by analyzing its performance under different operating conditions, including movement, camera usage, and data transmission. Voltage and amperage measurements are taken using a VOM (Volt-Ohm-Meter) to accurately assess power requirements. Subsequently, based on these calculations and measurements, the appropriate batteries are selected as discussed in the previous section.

Utilizing a solar panel to maintain battery capacity: It turns out the inside the 20Ah battery already has a charging circuit to protect the battery, we just have to mount a solar panel on the robot, plug it to the battery and use it to trickle charge the battery during periods of sunlight exposure. This can help maintain the battery capacity and extend the operating time if the robot operates in outdoor environments with sufficient sunlight.

E. Hardware limitation

By using 4g USB module, raspberry pi can send signal to user even when the robot is in the middle of the field where internet is weak

Each time we plug in a camera, we must go to the terminal of the raspberry pi and type: “sudo raspi config” and then enable support legacy camera then we can use it like a webcam. The using code for raspberry pi camera also must be different because it is completely different than computer webcam.

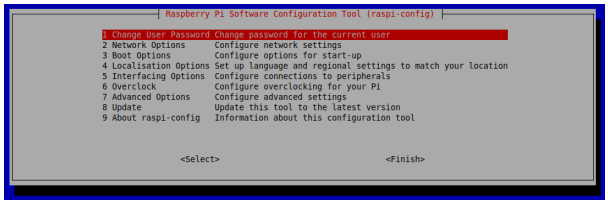

Figure 52 – Enable Pi Camera

The Raspberry pi 2B connect with Arduino do not connect through Serial port like computer. We need to use ACM port to Raspberry pi to Arduino

F. Utilized techniques to control the robot

- Finite state machine

Designing the robot's system using a finite state machine includes major states such as ROBOT_ON, ROBOT_OFF, ROBOT_INIT, AUTO, and MANUAL.

```
void FMS_ROBOT()
{
  if (statusRobot == ROBOT_ON)
  {
    switch (MODE)
    {
    case ROBOT_INIT:
      SelectModeRobot();
      break;
    case AUTO:
      FSM_AUTO();
      break;
    case MANNUAL:
      FSM_MANNUAL();
      break;
    default:
      break;
    }
  }
  else if (statusRobot == ROBOT_OFF)
  {
    TurnOFFRobot();
  }
}
```

- PID Control with Encoder Motor

Purpose of use: Used to synchronize the speed of the motors to enable the robot to move on long distances without deviating from its path.

A proportional–integral–derivative controller (PID controller or three-term controller) is a control loop mechanism employing feedback that is widely used in industrial control systems and a variety of other applications requiring continuously modulated control. A PID controller continuously calculates an error value e(t) as the difference between a desired setpoint (SP) and a measured process variable (PV) and applies a correction based on proportional, integral, and derivative terms (denoted P, I, and D respectively)

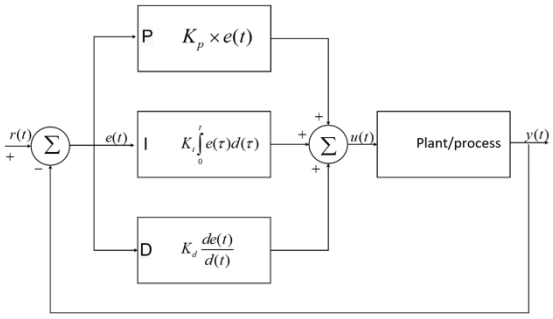

Figure 53 - PID controller

The overall control function:

$$u(t) = K_p \times e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{d(t)}$$

Introduction to how the encoder work:

Encoders use different types of technologies to create a signal, including: mechanical, magnetic, resistive, and optical – optical being the most common. In optical sensing, the encoder provides feedback based on the interruption of light, as illustrated in Figure .

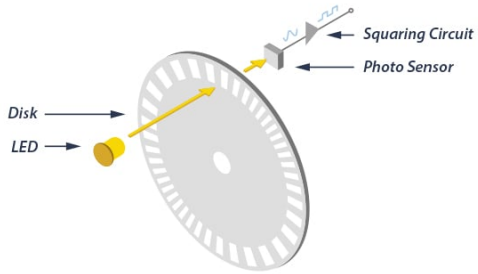

Figure 54 - ENCODER

A beam of light emitted from an LED passes through the Code Disk (see Figure 1), which is patterned with opaque lines , much like the spokes on a bike wheel. As the encoder shaft rotates, the light beam from the LED is interrupted by the opaque lines on the Code Disk before being picked up by the Photodetector Assembly. This produces a pulse signal: light = on; no light = off. The signal is sent to the counter or controller, which will then send the signal to produce the desired function.

Applied to the Robot system

Using the AttachInterrupt() function to detect the signal returned from the encoder:

attachInterrupt(encoderPinA_backRightWheel, ReadEncoderBackRightWheel, FALLING);

The ReadEncoderBackRightWheel() function will be called every time there is a pulse from the encoder. This function is responsible for counting the number of pulses returned in 1 second, which is used as input for the PID controller.

```
void ReadEncoderBackRightWheel()
{
  if (MODE == AUTO && (statusAuto ==
AUTO_CLEANING || statusAuto ==
AUTO_MOVE_TO_THE_NEXT_LINE || statusAuto ==
AUTO_CLEAN_AGAIN || statusAuto == AUTO_INIT))
  {
    if (encoderCount_BackRightWheel == 0)
```

```
        {
            startTimeBackRightWheel = millis();
        }
        encoderCount_BackRightWheel++;
        if (encoderCount_BackRightWheel % 80 == 0)
        {
            stopTimeBackRightWheel = millis();
            encoderCount_BackRightWheel = 0;
            PID_input_BackRightWheel = (int)80 /
((float)(stopTimeBackRightWheel -
startTimeBackRightWheel) / 1000);
            startTimeBackRightWheel = 0;
            stopTimeBackRightWheel = 0;
        }
    }
}
```

After every 0.01 seconds, the PID control function will be executed once. The output of this function is the PWM value used to adjust the motor speed to reach the target value.

```
void PIDExcuteBackRightWheel()
{

    PID_error_BackRightWheel =
PID_setpoint_speedWheel - PID_input_BackRightWheel;
    PID_integral_BackRightWheel +=
(PID_error_BackRightWheel * 0.01);
    PID_derivative_BackRightWheel =
(PID_error_BackRightWheel -
PID_lastError_BackRightWheel) / 0.01;

    PID_output_BackRightWheel = (Kp_Wheel *
PID_derivative_BackRightWheel) + (Ki_Wheel *
PID_integral_BackRightWheel) + (Kd_Wheel *
PID_derivative_BackRightWheel);
    if (PID_output_BackRightWheel > 255)
        PID_output_BackRightWheel = 255;
    if (PID_output_BackRightWheel < 0)
        PID_output_BackRightWheel = 0;

    PID_lastError_BackRightWheel =
PID_error_BackRightWheel;
    speedBackRightWheel =
PID_output_BackRightWheel;
}
```

Above is the PID controller for the rear right motor of the Robot. Similarly, we will have three more PID controllers for the front left, front right, and rear left motors.

- Timing using Timer

  A timer is a specialized type of clock which is used to measure time intervals. A timer that counts from zero upwards for measuring time elapsed is often called a stopwatch. It is a device that counts down from a specified time interval and used to generate a time delay, for example, an hourglass is a timer.

  In this Robot system, the timer is used to periodically receive and process data from ESP32, Raspberry Pi, and Bluetooth. Additionally, it is utilized to perform tasks with time constraints. The TimerRun function will be executed once every 10ms.

```
myTimer.attachInterrupt(TimerRun).setPeriod(1000
0).start();
void TimerRun()
{
    (timer0_counter > 0) ? timer0_counter-- :
timer0_flag = 1;
    (timer1_counter > 0) ? timer1_counter-- :
timer1_flag = 1;
    (timer2_counter > 0) ? timer2_counter-- :
timer2_flag = 1;
    (timer3_counter > 0) ? timer3_counter-- :
timer3_flag = 1;
    (timer4_counter > 0) ? timer4_counter-- :
timer4_flag = 1;
    (timer5_counter > 0) ? timer5_counter-- :
timer5_flag = 1;
    (timer6_counter > 0) ? timer6_counter-- :
timer6_flag = 1;
    ReceiveDataFromRasp();
    ReadDataFromBluetooth();
    ReadDistanceFromEsp();

    if (MODE == MANNUAL)
    {
        if (!isSafeDistance(frontLeftDistance) ||
!isSafeDistance(frontRightDistance) ||
!isSafeDistance(backLeftDistance) ||
!isSafeDistance(backRightDistance))
        {
            dataRecvFromBluetooth = 'P';
            dataRecvFromCloud[0] = "P";
            RobotStop();
        }
    }
}
```

- Uart communication

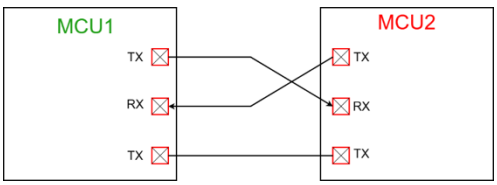

Figure 55 - UART communication

Universal Asynchronous Receive Transmit (UART) or Serial communication is one of the most simple communication protocols between two devices. It transfers data between devices by connecting two wires between the devices, one is the

transmission line while the other is the receiving line. The data transfers bit by bit digitally in form of bits from one device to another. The main advantage of this communication protocol is that its not necessary for both the devices to have the same operating frequency. However, a predefined bit rate that is referred to as baud rate usually set in the flash memory of both microcontrollers for the instruction to be understood by both the devices.

In this project used for communication between arduino due and ESP32, Raspberry pi 2 and module bluetooth HC06.

Example function receive data from bluetooth use uart:

```
void ReadDataFromBluetooth()
{
  if (Serial1.available())
  {
    dataRecvFromBluetooth = Serial1.read();
  }
```

G. *How to make the control app works smoothly?*

There are two ways to solve this problem:

For Bluetooth: Instead of opening a connection each time data is sent, only open a Bluetooth connection once after establishing a connection with the device.

For Wi-Fi: Network-related activities should be moved to the background thread or use asynchronous methods.

To ensure that the app functions correctly and consistently, developers should:

Grant the app the necessary permissions related to Bluetooth, Wi-Fi, and the internet.

Use asynchronous methods to perform network activities.

Avoid performing network activities on the main thread of the app.

IX. Final product

Finally, the SPCR has the parameter below:

- Total lenght: 870mm
- Total width: 625mm
- Height: 150mm
- Weight: 16kg
- Rated voltage: 12VDC

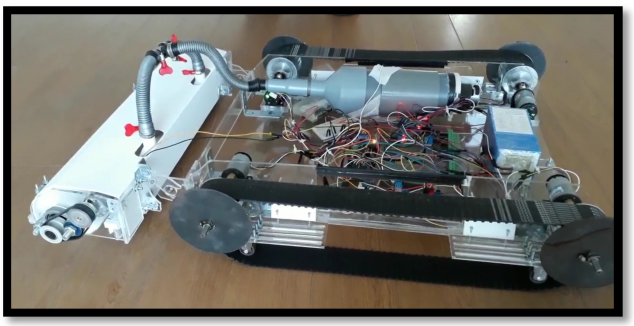

Figure 56 – Final Product

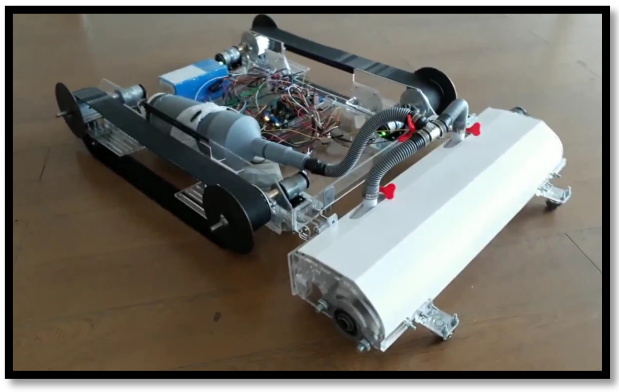

Figure 57 – Final Product

The final product will be tested in a solar panel in Tri Ton District, An Giang Province.

X. Conclusion

The intern project successfully addressed the challenges in image processing, information transmission, and battery management for the robot. The developed system effectively identifies the cleanliness of solar panel surfaces, transmits real-time video to a mobile app, and ensures the robot's functionality for extended periods. The implemented solutions showcase the feasibility of the proposed approach, demonstrating promising results in the field of robotics and image processing

Image Processing Program Source Code

File: **Premium_realtime_comparison.py**:

```
import cv2 as cv
import numpy as np
import time
import base64
import picamera
import csv
#from fourier_processors import
perform_fourier_transform_and_compare_to_reference_fou
rier_image, process_image_fourier
from arduino_sender import open_serial_connection,
close_serial_connection, send_data,
command_to_send_to_arduino, find_port, receive_data
from perspective_processor import
perform_realtime_perspective_transform,
process_image_perspective
from redis_processors import connect_redis, receive_signal,
disconnect_redis, send_signal
from histogram_equalizer import histogram_equalization,
histogram_equalization_on_frame
# from remove_reflection  import remove_reflection,
remove_reflection_on_frame
from adjust_brightness import adjust_brightness_on_frame,
adjust_brightness_on_image
from correlation import extract_spectrum,
extract_spectrum_on_frame, spectrum_to_see
```

```
from similarity_NMI import compare_images
# from remove_reflection import
remove_reflection_on_frame, remove_reflection
from light_to_dark import perform_brightness_thresholding,
perform_brightness_thresholding_on_image
from Bayes_class_decision import predict_group,
get_parameters
#Main Run: Nhận tín hiệu từ Redis, biến đổi fourier và so
sánh từng frame lấy từ Camera với ảnh sạch, rồi chuyển lệnh
đến Arduino

def Main_Run(reference_image, baud_rate,
redis_receive_keys, arduino_data_keys):
    #Open Redis Connection
    RedisHost =
'redis-17060.c299.asia-northeast1-1.gce.cloud.redislabs.com'
    RedisPort = 17060
    RedisPassword =
'YS9EKyvuTG5Q3HGxKFem48ePa7ykaV93'
    redis_conn = connect_redis(RedisHost , RedisPort,
RedisPassword)
    # Open Serial connection
    # receive_data=""
    arduino_port = find_port()
    ser = open_serial_connection(arduino_port, baud_rate)

    # Load the reference image for comparison
    load_reference_image = cv.imread(reference_image,
cv.IMREAD_GRAYSCALE)
    reference_fourier_frame =
spectrum_to_see(load_reference_image)
    cv.imwrite("image/fourier_image.jpg",
reference_fourier_frame)
    # Open the camera for video capture
    #cap = cv.VideoCapture(0)  # Use 0 for the default camera,
or specify the camera index

    # Get the camera's frame rate and dimensions
    #fps = cap.get(cv.CAP_PROP_FPS)
    #width = int(cap.get(cv.CAP_PROP_FRAME_WIDTH))
    #height =
int(cap.get(cv.CAP_PROP_FRAME_HEIGHT))

    #turn on module camera raspberry
    #set video resolution

    with picamera.PiCamera() as camera:
        camera.resolution = (480,240)
        camera.framerate = 10
    #start the video preview
        camera.start_preview()
        width = camera.resolution[0]
        height = camera.resolution[1]


    # Initialize variables for Kalman Filter
        kalman = cv.KalmanFilter(1, 1, 0)
        kalman.transitionMatrix = np.array([[1]],
dtype=np.float32)
        kalman.measurementMatrix = np.array([[1]],
dtype=np.float32)
        kalman.processNoiseCov = np.array([[1e-5]],
dtype=np.float32)
        kalman.measurementNoiseCov = np.array([[1e-3]],
dtype=np.float32)
        kalman.errorCovPost = np.array([[1]],
dtype=np.float32)
        kalman.statePost = np.array([[0]], dtype=np.float32)

        frame_rate_limit = 10  # Giới hạn số khung hình mỗi
giây
        frame_interval = 1 / frame_rate_limit
        last_frame_time = time.time()

        timestamps= []
        values=[]
        first_frame_time=time.time()

        mean_clean, std_clean =
get_parameters('clean_parameters.csv')
        mean_dirty, std_dirty =
get_parameters('dirty_parameters.csv')
        #count = 0

        csvfile = open('data_test.csv', 'w', newline='')
        csvwriter = csv.writer(csvfile)
        csvwriter.writerow(['Timestamps', 'Values'])

        while True:
        # Read the next frame from the camera
        #ret, frame = cap.read()
        #if not ret:
              # Error in capturing frame
        #       break

            current_time = time.time()
            elapsed_time = current_time - last_frame_time
            if elapsed_time >= frame_interval:
                last_frame_time = current_time
```

```
            stream
=np.empty((camera.resolution[1]*camera.resolution[0]*3),
dtype=np.uint8)
            camera.capture(stream, 'bgr')
            frame =stream.reshape ((camera.resolution[1],
camera.resolution[0], 3))

            camera_control = receive_signal(redis_conn,
"camera")
            if camera_control == "on":
            # Chuyển đổi frame thành dạng nhị phân
                _, img_encoded = cv.imencode('.jpg', frame)
                img_bytes = img_encoded.tobytes()

                # Chuyển đổi img_bytes thành chuỗi base64
                img_base64 =
base64.b64encode(img_bytes).decode('utf-8')

               chunk_size = 60000  # Độ dài của mỗi phần chuỗi
                chunks = [img_base64[i:i+chunk_size] for i in
range(0, len(img_base64), chunk_size)]

                # Gắn thêm flag để xác định thứ tự của các chuỗi
mới
               flagged_chunks = [f"{i+1}_{chunk}" for i, chunk
in enumerate(chunks)]  # example chunk: "[1_data 2_info"
                flagged_chunks[-1] =
f"{flagged_chunks[-1]}_endframe"

               # Thêm hậu tố "_notyet" cho các phần tử, trừ phần
tử cuối cùng
                flagged_chunks = [f"{chunk}_notyet" if i !=
len(flagged_chunks) - 1 else chunk for i, chunk in
enumerate(flagged_chunks)]

                # Gửi frame đến Redis chỉ khi đã trôi qua đủ
khoảng thời gian

                for flagged_chunk in flagged_chunks:
                   redis_conn.set('video', flagged_chunk)




            #Receive data from Arduino

            # received_datas = receive_data(ser)
            # print("Thông tin nhận được:")
            # for prefix, data in enumerate(received_data):
            #     if data is not None:
            #         command_function =
arduino_data_keys[prefix] if prefix <
len(CommandFunction) else "Unknown Function"
            #         print(f"Command {prefix}: {data} -
Function: {command_function}")




            # Apply perspective transformation to the frame
            #transformed_frame =
perform_realtime_perspective_transform(frame, width,
height)



            # # Turn the part too white to black
            # perform_brightness_thresholding_frame=
perform_brightness_thresholding(transformed_frame, 100)

            # Apply brightness adjustment
            brightness_adjusted_frame =
adjust_brightness_on_frame(frame, 100)



            # Turn the part too white to black
            perform_brightness_thresholding_frame=
perform_brightness_thresholding(brightness_adjusted_fram
e, 150)



            #Apply histogram equalizer to the frame
            histograme_equalized_frame =
histogram_equalization_on_frame(perform_brightness_thres
holding_frame)

            #Apply reflection remove to the frame
            # reflection_removed_frame =
remove_reflection_on_frame(histograme_equalized_frame,
50)

            #Apply extract_spectrum

            fourier_frame =
spectrum_to_see(histograme_equalized_frame)

            #if count != -1
            #    count += 1
            #    if count == 50:
            #        cv.imwrite("image/fourier_frame.jpg",
fourier_frame)
```

```
            #       count = -1

            NMI_Score=
compare_images(reference_fourier_frame,fourier_frame)

   #          fourier_frame_shape = fourier_frame.shape
   #          reference_fourier_frame_shape =
reference_fourier_frame.shape
   #          cv.putText(fourier_frame, f"size:
{fourier_frame_shape}", (10,30),
cv.FONT_HERSHEY_SIMPLEX, 1,
   #                (0, 255, 0), 2)
   #          cv.putText(reference_fourier_frame, f"size:
{reference_fourier_frame_shape}", (10,30),
cv.FONT_HERSHEY_SIMPLEX, 1,
   #                (0, 255, 0), 2)

            #Nmi comparison to get the difference
   #          cv.imshow("Fourier Frame", fourier_frame)
   #          cv.imshow("brightness",
brightness_adjusted_frame)
   #          cv.imshow("Histogram",
histograme_equalized_frame)
   #          cv.imshow("Reference Frame",
reference_fourier_frame)


            # Apply Kalman Filter to diff_percentage
            kalman_prediction = kalman.predict()
            kalman_corrected =
kalman.correct(np.array([[NMI_Score]], dtype=np.float32))
            NMI_Score_filtered = kalman_corrected[0, 0]

            #list of time and values
            if not isinstance(values, list):
               values = [values]
            timestamp = current_time-first_frame_time

            csvwriter.writerow([timestamp,
NMI_Score_filtered])
            #timestamps.append(timestamp)
            #values.append(NMI_Score_filtered)

            # Receive data from Redis
            received_values = []
            for key in redis_receive_keys:
              received_value = receive_signal(redis_conn, key)
               received_values.append(received_value)
            # Send the diff_percentage_filtered to
Arduino

            for i, value in enumerate(received_values):
               send_data(ser,baud_rate ,value, i)
            TheCommand =
predict_group(NMI_Score_filtered, mean_clean, std_clean,
mean_dirty, std_dirty)
            ser = send_data(ser, baud_rate, TheCommand, 1)
            # Display the frame and difference percentage in
real-time
   #          cv.putText(frame, f"NMI Score:
{NMI_Score_filtered:.2f}", (10, 30),
cv.FONT_HERSHEY_SIMPLEX, 1,
   #                (0, 255, 0), 2)
   #          cv.imshow('Real-time Comparison', frame)

            # Send datas from Arduino to Redis
            #for prefix, data in enumerate(received_datas):
            #    if data is not None:
            #       send_signal(redis_conn,
arduino_data_keys[prefix], data)

            print(f"NMI_Score: {NMI_Score_filtered:.2f}")
            kalman.statePost = kalman_corrected
            print(f"TheCommand: {TheCommand}")


   # Release the camera
      camera.stop_preview()
      camera.close()
   # Close connection to Arduino
      close_serial_connection(ser)
      disconnect_redis(redis_conn)

   # Close the 'Real-time Comparison' window
   #    cv.destroyAllWindows()


if __name__ == '__main__':
   redis_keys = ["move"]
   arduino_data_keys = ["Temperature", "Sensor1",
"Sensor2", "Sensor3", "Sensor4", "water", "battery", "Moved
Distance"]
   input_image_file = 'image/webcam_image5.jpg'
   #view_image_file =
process_image_perspective(input_image_file)
   #remove_reflection_file =
remove_reflection(input_image_file, 50)
```

```
    brightness_adjusted_file =
adjust_brightness_on_image(input_image_file, 100)
    brightness_threshold_file =
perform_brightness_thresholding_on_image(brightness_adj
usted_file, 150)
    histogram_equalized_file =
histogram_equalization(brightness_threshold_file)
    baud_rate = 9600
    Main_Run(histogram_equalized_file, baud_rate,
redis_keys, arduino_data_keys)
```

File: **arduino_sender.py**:

```
import serial
import serial.tools.list_ports
import time
def getch():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN,
old_settings)
    return ch
# 'USB-SERIAL CH-340'
def find_port():
    arduino_port = None
    # Tìm kiếm cổng kết nối
    ports = serial.tools.list_ports.comports()

    # Lặp qua danh sách các cổng
    for port in ports:
        # List available ports and their descriptions
        print("Port:", port.device)
        print("Description:", port.description)
        print()
        # Kiểm tra tên và mô tả cổng
        if 'Arduino' or 'ACM' in port.description:
            # Xác định cổng kết nối của Arduino
            arduino_port = port.device
            print("Arduino is connected to", arduino_port)
            break

    # Nếu không tìm thấy cổng kết nối Arduino
    if arduino_port is None:
        print("Arduino is not connected to any port")

    return arduino_port
```

```
def open_serial_connection(arduino_port, baud_rate):
    if arduino_port is None:
        return None

    try:
        ser = serial.Serial(arduino_port, baud_rate,
exclusive=True)
        time.sleep(2)
        return ser
    except serial.SerialException as e:
        print("An error occurred while opening the serial
connection:", e)
        return None


def send_data(ser, baudrate, data, channel = 0):
    if ser is None:
        print("Connection to Arduino is not established yet.")
        arduino_port = find_port()
        ser= open_serial_connection(arduino_port, baudrate)
        return ser

    try:
        # Check if data is already in bytes format
        if isinstance(data, bytes):
            data_bytes = data
        else:
            # Convert the data to string format if it's not already a
string
            if not isinstance(data, str):
                data_str = str(data)
            else:
                data_str = data
            # Convert the data to bytes format
            data_bytes = data_str.encode()

        # Convert the channel to bytes format and add newline
character to the data
        data_with_newline = str(channel).encode() +
str(":").encode() + data_bytes + b'#'


        # Send data to Arduino
        ser.write(data_with_newline)
        print(f"Data: {data} has been sent to channel
{channel}")
        return ser
    except serial.SerialException as e:
        print("An error occurred while sending data:", e)
```

```
def receive_data(ser):
    if ser is None:
        print("Connection is not established yet.")
        return []

    received_data = []

    while ser.in_waiting:
        line = ser.readline().decode('latin-1').strip()
        parts = line.split(":")
        if len(parts) == 2:
            prefix, data = parts[0], parts[1]
            if prefix.isdigit():  # Kiểm tra tính hợp lệ của tiền tố
                prefix = int(prefix)
                if len(received_data) > prefix:
                    received_data[prefix] = data
                else:
                    received_data.extend([None] * (prefix -
len(received_data)))
                    received_data.append(data)

    if len(received_data) > 0:
        print("Đã nhận được tín hiệu từ Arduino")
    else:
        print("Không có tín hiệu từ Arduino")

    return received_data



def close_serial_connection(ser):
    if ser is not None:
        ser.close()


def send_to_arduino(data, baud_rate, channel = 1):
    # Tìm Port kết nối
    arduino_port = find_port()
    # Open serial connection to Arduino
    ser = open_serial_connection(arduino_port, baud_rate)

    # Send data to Arduino
    send_data(ser, data, channel)

    # Close serial connection to Arduino
    close_serial_connection(ser)

def receive_from_arduino(baud_rate):
    # Tìm Port kết nối
    arduino_port = find_port()

    # Open serial connection to Arduino
    ser = open_serial_connection(arduino_port, baud_rate)

    # Receive data from Arduino
    received_commands = receive_data(ser)

    # Close serial connection to Arduino
    close_serial_connection(ser)

    return received_commands

def command_to_send_to_arduino(float_value, base_data):
    if float_value < 0:
        return "G"
    else:
        return base_data


#float_value = 7
#data = command_to_send_to_arduino(float_value)
#send_to_arduino(data, 9600)

# # Gọi hàm để nhận dữ liệu từ Arduino
# received_commands = receive_from_arduino(9600)

# # In ra các tín hiệu nhận được
# for command in received_commands:
#     print("Received command:", command)



## Test Nhận tín hiệu từ Arduino
# Kết nối đến Serial và khởi tạo ser
#arduino_port = find_port()
#baud_rate = 115200
#ser = open_serial_connection(arduino_port, baud_rate)
#CommandFunction = ["Work Time", "Temperature",
"Front-Left Distance Sensor", "Front-Right Distance
Sensor",
#                   "Back-Left Distance Sensor", "Back-Right
Distance Sensor", "Left Over Water", "Battery Left",
#                   "Moved Distance", "Over Work"]

# Kết nối đến Serial và khởi tạo ser

#received_data = receive_data(ser)
```

```
#print("Thông tin nhận được:")
#for prefix, data in enumerate(received_data):
#    if data is not None:
#        command_function = CommandFunction[prefix] if
prefix < len(CommandFunction) else "Unknown Function"
#        print(f"Command {prefix}: {data} - Function:
{command_function}")


#close_serial_connection(ser)
```

File: **adjust_brightness.py**:

```
import cv2
import numpy as np

def adjust_brightness_on_frame(frame, target_brightness):
    # Tính độ sáng trung bình của hình ảnh
    avg_brightness = np.mean(frame)

    # Tính tỷ lệ điều chỉnh dựa trên độ sáng trung bình hiện tại
và độ sáng mục tiêu
    adjustment_ratio = target_brightness / avg_brightness

    # # Giới hạn độ sáng lớn nhất của mỗi pixel là 150
    adjusted = np.clip(frame, None, 150)

    # Điều chỉnh độ sáng của hình ảnh mà không thay đổi kiểu
dữ liệu
    adjusted = cv2.convertScaleAbs(frame,
alpha=adjustment_ratio)

    return adjusted

def adjust_brightness_on_image(image_file,
target_brightness):
    output_file = "image/brightness_adjusted_image.jpg"
    image = cv2.imread(image_file)
    adjusted = adjust_brightness_on_frame(image,
target_brightness)
    cv2.imwrite(output_file, adjusted)
    return output_file

def main():
    # Mở webcam
    cap = cv2.VideoCapture(0)
    target_brightness = 100
    # Kiểm tra xem webcam đã được mở chưa
    if not cap.isOpened():
        print("Không thể mở webcam.")
        return

    while True:
        # Đọc khung hình từ webcam
        ret, frame = cap.read()

        if not ret:
            print("Không thể đọc khung hình.")
            break

        # Điều chỉnh độ sáng
        adjusted_frame = adjust_brightness_on_frame(frame,
target_brightness)

        # Tính độ sáng trung bình của hai hình ảnh
        avg_brightness_original = np.mean(frame)
        avg_brightness_adjusted = np.mean(adjusted_frame)

        # Hiển thị độ sáng trung bình lên trên hình ảnh
        cv2.putText(frame, f'Original:
{avg_brightness_original}', (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
        cv2.putText(adjusted_frame, f'Adjusted:
{avg_brightness_adjusted}', (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

        # Hiển thị hình ảnh gốc và hình ảnh đã được điều chỉnh
độ sáng
        cv2.imshow('Original', frame)
        cv2.imshow('Adjusted', adjusted_frame)

        # Chờ nhấn phím 'q' để thoát
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # Giải phóng tài nguyên
    cap.release()
    cv2.destroyAllWindows()

if __name__ == '__main__':
    main()
```

File: **Bayes_class_decision.py**:

```
import pandas as pd
import numpy as np

def save_parameters_to_csv(parameter_file_path, mean,
std):
    # Tạo DataFrame chứa thông tin về các tham số
    parameters_df = pd.DataFrame({
```

```
        'Parameter': ['mean', 'std'],
        'Values': [mean, std]
    })
    # Xuất DataFrame vào file CSV
    parameters_df.to_csv(parameter_file_path, index=False)

def estimate_gaussian_parameters(file_path,
Parameter_file_path, feature_name = 'Values'):
    # Đọc dữ liệu từ file CSV
    data = pd.read_csv(file_path)
    # Trích xuất đặc trưng từ dữ liệu
    feature_values = data[feature_name].values
    # Ước lượng tham số cho phân phối Gaussian (mean và
standard deviation)
    mean = np.mean(feature_values)
    std = np.std(feature_values)
    save_parameters_to_csv(Parameter_file_path, mean, std)

def get_parameters(parameter_file_path):
    # Đọc dữ liệu từ file CSV
    data = pd.read_csv(parameter_file_path)
    # Lấy giá trị mean và std từ DataFrame
    mean = data[data['Parameter'] ==
'mean']['Values'].values[0]
    std = data[data['Parameter'] == 'std']['Values'].values[0]
    return mean, std


# Dự đoán nhóm của số mới
def predict_group(value, mean_clean, std_clean, mean_dirty,
std_dirty):
    p_clean = 1 / (np.sqrt(2 * np.pi) * std_clean) * np.exp(-0.5
* ((value - mean_clean) / std_clean) ** 2)
    p_dirty = 1 / (np.sqrt(2 * np.pi) * std_dirty) * np.exp(-0.5 *
((value - mean_dirty) / std_dirty) ** 2)

    if p_clean > p_dirty:
        return "Clean"
    else:
        return "G"

if __name__ == '__main__':
    new_value = 0.67
    # Sử dụng hàm predict_group để dự đoán nhóm của số mới
    clean_file = 'data_100_150.csv'
    dirty_file = 'data_100_150_dirty.csv'
    clean_parameters_file = 'clean_parameters.csv'
    dirty_parameters_file = 'dirty_parameters.csv'

    estimate_gaussian_parameters(clean_file,
clean_parameters_file)
    estimate_gaussian_parameters(dirty_file,
dirty_parameters_file)
     # Get the parameters for the Bayes Classifier
    mean_clean, std_clean =
get_parameters('clean_parameters.csv')
    mean_dirty, std_dirty =
get_parameters('dirty_parameters.csv')

    predicted_group = predict_group(new_value,
'base_command',  mean_clean, std_clean,  mean_dirty,
std_dirty)

    print("Number {} is belong to group :
{}".format(new_value, predicted_group))
```

File: **correlation.py**:

```
import numpy as np
import cv2

def extract_spectrum(image):
    load_image = cv2.imread(image)
    spectrum = np.abs(np.fft.fftshift(np.fft.fft2(load_image)))
    return spectrum

def extract_spectrum_on_frame(frame):
    spectrum = np.abs(np.fft.fftshift(np.fft.fft2(frame)))
    return spectrum

def spectrum_to_see(picture):
    spectrum = np.fft.fftshift(np.fft.fft2(picture))
    # Calculate the magnitude spectrum
    magnitude_spectrum = np.abs(spectrum)

    # Handle zero values
    small_value = 1e-10  # You can adjust this value as needed,
it should be small enough not to affect the overall spectrum
significantly
    magnitude_spectrum[magnitude_spectrum == 0] =
small_value

    # Apply logarithmic transformation
    magnitude_spectrum = 20 * np.log(magnitude_spectrum)
    magnitude_spectrum[np.isinf(magnitude_spectrum)] =
0  # Replace -inf values with 0
```

```
    magnitude_spectrum[np.isnan(magnitude_spectrum)] =
0  # Replace nan values with 0
    magnitude_spectrum =
cv2.normalize(magnitude_spectrum, None, 0, 255,
cv2.NORM_MINMAX, cv2.CV_8U)
    # magnitude_spectrum_normalized =
(magnitude_spectrum - np.min(magnitude_spectrum)) /
(np.max(magnitude_spectrum) -
np.min(magnitude_spectrum))
    return magnitude_spectrum


def compare_spectra(spectrum1, spectrum2):
    correlation = np.corrcoef(spectrum1.flatten(),
spectrum2.flatten())[0, 1]
    return correlation

if __name__ == '__main__':
    image1 = ...  # Đường dẫn đến hình ảnh thứ nhất
    image2 = ...  # Đường dẫn đến hình ảnh thứ hai

    spectrum1 = extract_spectrum(image1)
    spectrum2 = extract_spectrum(image2)

    correlation = compare_spectra(spectrum1, spectrum2)
    print("Correlation:", correlation)
```

File: **histogram_equalizer.py**:

```
import cv2

def histogram_equalization(image_path):
    # Đọc ảnh từ đường dẫn
    image = cv2.imread(image_path)

    #thiết lập output
    output_image_path =
'image/histogram_equalized_result.jpg'

    # Chuyển ảnh sang ảnh xám
    gray_image = cv2.cvtColor(image,
cv2.COLOR_BGR2GRAY)

    # Cân bằng histogram
    equalized_image = cv2.equalizeHist(gray_image)

    # color_equalized_image =
cv2.cvtColor(equalized_image, cv2.COLOR_GRAY2BGR)

    cv2.imwrite(output_image_path, equalized_image)

    # Trả về ảnh đã được cân bằng histogram
    return output_image_path

def histogram_equalization_on_frame(frame):
    # Chuyển ảnh sang ảnh xám
    gray_frame = cv2.cvtColor(frame,
cv2.COLOR_BGR2GRAY)

    # Cân bằng histogram
    equalized_frame = cv2.equalizeHist(gray_frame)

    # color_equalized_frame = cv2.cvtColor(equalized_frame,
cv2.COLOR_GRAY2BGR)

    # Trả về ảnh đã được cân bằng histogram
    return equalized_frame
```

File: **fourier_processor.py**:

```
import cv2 as cv
import numpy as np

def process_image_fourier(input_file):
    output_file = "image/fourier_result.jpg"

    # Load the input image
    img = cv.imread(input_file, cv.IMREAD_GRAYSCALE)
    assert img is not None, "Image file could not be read"

    # Perform Fourier transform
    f = np.fft.fft2(img)
    fshift = np.fft.fftshift(f)
    magnitude_spectrum = 20 * np.log(np.abs(fshift))

    # Convert the magnitude spectrum to uint8 for writing to
image
    magnitude_spectrum_uint8 =
cv.normalize(magnitude_spectrum, None, 0, 255,
cv.NORM_MINMAX, cv.CV_8U)

    # Save the result image
    cv.imwrite(output_file, magnitude_spectrum_uint8)

    return output_file

def process_video_fourier(input_file):
    output_file = "video/fourier_result.mp4"

    # Open the input video
    cap = cv.VideoCapture(input_file)

    # Check if the video file was successfully opened
```

```
    assert cap.isOpened(), "Video file could not be opened"

    # Get the video properties
    fps = cap.get(cv.CAP_PROP_FPS)
    width = int(cap.get(cv.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv.CAP_PROP_FRAME_HEIGHT))

    # Create a VideoWriter object for the output video
    fourcc = cv.VideoWriter_fourcc(*"mp4v")
    out = cv.VideoWriter(output_file, fourcc, fps, (width,
height), False)

    # Process each frame in the video
    while True:
        ret, frame = cap.read()

        # Check if there are no more frames to read
        if not ret:
            break

        # Convert the frame to grayscale
        img = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)

        # Perform Fourier transform
        f = np.fft.fft2(img)
        fshift = np.fft.fftshift(f)
        magnitude_spectrum = 20 * np.log(np.abs(fshift))

        # Convert the magnitude spectrum to uint8 for writing to
video
        magnitude_spectrum_uint8 =
cv.normalize(magnitude_spectrum, None, 0, 255,
cv.NORM_MINMAX, cv.CV_8U)

        # Write the frame to the output video
        out.write(magnitude_spectrum_uint8)

    # Release the video file and the output video
    cap.release()
    out.release()

    return output_file

def
perform_fourier_transform_and_compare_to_reference_fou
rier_image(frame, reference_image):
    # Convert the frame to grayscale
    gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)

    # Perform Fourier transform
    dft = cv.dft(np.float32(gray),
flags=cv.DFT_COMPLEX_OUTPUT)
    dft_shift = np.fft.fftshift(dft)

    # Compute magnitude spectrum
    magnitude_spectrum = 20 *
np.log(cv.magnitude(dft_shift[:, :, 0], dft_shift[:, :, 1]))

    # Normalize the magnitude spectrum for display
    magnitude_spectrum =
cv.normalize(magnitude_spectrum, None, 0, 255,
cv.NORM_MINMAX, dtype=cv.CV_8U)

    # Compare the magnitude spectrum with the reference
image
    diff = cv.absdiff(reference_image, magnitude_spectrum)
    diff_percentage = np.mean(diff) / 255 * 100

    # Convert back to BGR for display
    result = cv.cvtColor(magnitude_spectrum,
cv.COLOR_GRAY2BGR)

    return result, diff_percentage
```

File: **light_to_dark.py**:

```
import numpy as np
import cv2
import __main__

def perform_brightness_thresholding(frame,
brightness_threshold):
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    value = hsv[:, :, 2]
    value = np.where(value > brightness_threshold, 0, value)
    hsv[:, :, 2] = value
    processed_frame = cv2.cvtColor(hsv,
cv2.COLOR_HSV2BGR)
    return processed_frame

def perform_brightness_thresholding_on_image(image_file,
brightness_threshold):
    output_file = "image/brightness_threshold_image.jpg"
    image = cv2.imread(image_file)
    processed_image =
perform_brightness_thresholding(image,
brightness_threshold)
    cv2.imwrite(output_file, processed_image)
    return output_file
```

```
# Example usage:
if __main__ == '__main__':
    frame = cv2.imread('input_frame.jpg')  # Replace 'input_frame.jpg' with the path to your input frame
    threshold_value = 200
    result_frame = perform_brightness_thresholding(frame, threshold_value)
    cv2.imshow('Result Frame', result_frame)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

File **perspective_processor.py**:

```
import cv2 as cv
import numpy as np

def process_video_perspective(input_file):
    output_file = "video/view_result.mp4"
    multiplier_width = 0.246
    multiplier_height = 0.1875

    # Open the video file
    cap = cv.VideoCapture(input_file)

    # Get the video's frame rate and dimensions
    fps = cap.get(cv.CAP_PROP_FPS)
    width = int(cap.get(cv.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv.CAP_PROP_FRAME_HEIGHT))

    # Define the transformation function
    def getTransform(points):
        pts1 = np.float32(points)
        pts2 = np.float32([[0, 0], [300, 0], [300, 300], [0, 300]])
        return cv.getPerspectiveTransform(pts1, pts2)

    # Create video writer
    fourcc = cv.VideoWriter_fourcc(*'mp4v')
    output = cv.VideoWriter(output_file, fourcc, fps, (300, 300))

    while True:
        # Read the next frame from the video
        ret, frame = cap.read()

        if not ret:
            # End of video
            break

        # Copy the frame
        img_ori = frame.copy()

        # Define the points for transformation
        points = np.array([[width * multiplier_width, height * multiplier_height],
                           [width * (1 - multiplier_width), height * multiplier_height],
                           [width * (1 - multiplier_width), height * (1 - multiplier_height)],
                           [width * multiplier_width, height * (1 - multiplier_height)]], dtype=np.float32)

        m = getTransform(points)

        # Apply perspective transformation
        dst = cv.warpPerspective(np.float32(img_ori), m, (300, 300))
        dst = np.array(dst, dtype='uint8')

        # Write the processed frame to the output video
        output.write(dst)

    # Release the video file and writer
    cap.release()
    output.release()

    # Close any open windows
    cv.destroyAllWindows()

    return output_file

def process_image_perspective(input_file):
    output_file = "image/view_result.jpg"
    multiplier_width = 0.246
    multiplier_height = 0.1875

    # Load the image
    img = cv.imread(input_file)

    # Define the points for transformation
    height, width = img.shape[:2]
    points = np.array([[width * multiplier_width, height * multiplier_height],
                       [width * (1 - multiplier_width), height * multiplier_height],
                       [width * (1 - multiplier_width), height * (1 - multiplier_height)],
                       [width * multiplier_width, height * (1 - multiplier_height)]], dtype=np.float32)

    # Define the transformation function
    def getTransform(points):
        pts1 = np.float32(points)
        pts2 = np.float32([[0, 0], [300, 0], [300, 300], [0, 300]])
```

```
        return cv.getPerspectiveTransform(pts1, pts2)

    m = getTransform(points)

    # Apply perspective transformation to the image
    dst = cv.warpPerspective(img, m, (300, 300))

    # Save the processed image
    cv.imwrite(output_file, dst)

    return output_file


def perform_realtime_perspective_transform(frame, width,
height):
    # Define the points for perspective transformation
    multiplier_width = 0.246
    multiplier_height = 0.1875
    points = np.array([[width * multiplier_width, height *
multiplier_height],
                       [width * (1 - multiplier_width), height *
multiplier_height],
                       [width * (1 - multiplier_width), height * (1 -
multiplier_height)],
                       [width * multiplier_width, height * (1 -
multiplier_height)]], dtype=np.float32)

    # Define the transformation function
    def get_transform(points):
        pts1 = np.float32(points)
        pts2 = np.float32([[0, 0], [300, 0], [300, 300], [0, 300]])
        return cv.getPerspectiveTransform(pts1, pts2)

    m = get_transform(points)
    transformed_frame = cv.warpPerspective(frame, m, (300,
300))

    return transformed_frame
```

File **redis_processor.py**:

```
import redis
import cv2
import numpy as np
import base64
from arduino_sender import send_to_arduino,
open_serial_connection, send_data, close_serial_connection,
find_port
import termios
import sys
import tty

RedisHost =
'redis-17060.c299.asia-northeast1-1.gce.cloud.redislabs.com'
RedisPort = 17060
RedisPassword =
'YS9EKyvuTG5Q3HGxKFem48ePa7ykaV93'
#hàm bấm nút để ngừng
def getch():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN,
old_settings)
    return ch
#mở cổng kết nối với redis
def connect_redis(RedisHost, RedisPort, RedisPassword):
    r = redis.Redis(RedisHost, RedisPort, db=0, password =
RedisPassword)
    return r
#gởi tín hiệu đến redis
def send_signal(r, key, value):
    if r.ping():
        r.set(key, value)
        print(f"Tín hiệu đã được gửi thành công đến key:
{key}.")
    else:
        print("Không có kết nối đến Redis. Tín hiệu không được
gửi.")
# nhận mảng tín hiệu từ redis
def receive_signal(r, key):
    value = r.get(key)
    value = value.decode('utf-8')
    if value is None:
        print(f"Nothing from key: {key}")
    else:
        print(f"Received data from key: {key}")
    return value

def disconnect_redis(r):
    r.close()

def
receive_realtime_signal_and_send_to_arduino(redis_keys):
```

```
    redis_conn = connect_redis(RedisHost, RedisPort,
RedisPassword)
    arduino_port = find_port()
    ser = open_serial_connection(arduino_port, 9600)

    while True:
      # if msvcrt.kbhit() and msvcrt.getch() == b'q':
       #    break

       received_values = []

       for key in redis_keys:
          received_value = receive_signal(redis_conn, key)
          received_values.append(received_value)

       for i, value in enumerate(received_values):
          send_data(ser, value, i+1)

    close_serial_connection(ser)
    disconnect_redis(redis_conn)

def receive_realtime_frame():

    redis_conn = connect_redis(RedisHost, RedisPort,
RedisPassword)
    full_frame_data = ""
    frame_for_decode = ""
    while True:
       # Thoát đọc khi bấm q từ bàn phím
       char = getch()
       if char == 'q':
          break
       # Đọc tín hiệu từ key movie
       frame_chunk = receive_signal(redis_conn, "video")
       # tách tín hiệu thành các phần
       parts = frame_chunk.split("_")
       chunk_prefix = int(parts[0])
       data = parts[1]
       chunk_suffix = parts[2]
       # Nối các phần data lại theo thứ tự (chưa đúng lắm)

       full_frame_data += data

       # Reset lại cho vòng lặp mới
       if chunk_suffix == "endframe":
          frame_for_decode = full_frame_data
          full_frame_data = ""

       video_frame = decode_frame(frame_for_decode)
       if video_frame is not None:
          cv2.imshow("Video", video_frame)
       if cv2.waitKey(1) & 0xFF == ord('q'):
          # Exit loop if 'q' key is pressed
          break

    cv2.destroyAllWindows()
    disconnect_redis(redis_conn)




def decode_frame(frame_for_decode):
    # decode frame_for_decode from base64 to bytes
    # Chuyển đổi img_base64 thành chuỗi bytes
    img_bytes = base64.b64decode(frame_for_decode)

    # Chuyển đổi chuỗi bytes thành mảng numpy
    nparr = np.frombuffer(img_bytes, np.uint8)

    # Đọc mảng numpy thành frame
    frame = cv2.imdecode(nparr, cv2.IMREAD_COLOR)
    return frame


#receive_realtime_frame()
# # Nhận tín hiệu theo thời gian thực từ key "move" &
"toggle"
# redis_keys = ["move", "toggle"]
# receive_realtime_signal_and_send_to_arduino(redis_keys)

# # Kết nối tới Redis
# redis_host =
'redis-12030.c73.us-east-1-2.ec2.cloud.redislabs.com'  # Địa
chỉ Redis server
# redis_port = 12030  # Cổng Redis
# redis_password =
'ZYtsQ1tAG4osutc294eHcQLRFKdxKbE3'  # Mật khẩu
Redis (nếu có)
# redis_conn = connect_redis(redis_host, redis_port,
redis_password)

# # # Lấy hình ảnh từ key 'movie' và giải mã
# test_string = receive_signal(redis_conn, "video")
# print(test_string)


# disconnect_redis(redis_conn)
```

File **remove_reflection.py**:

```
import cv2
import numpy as np
```

```
def remove_reflection(image_path, threshold_adder):
    # Đọc ảnh từ đường dẫn
    image = cv2.imread(image_path)
    #thiết lập output
    output_image_path = 'image/reflection_remove_result.jpg'

    # Chuyển đổi ảnh sang không gian màu LAB
    lab_image = cv2.cvtColor(image,
cv2.COLOR_BGR2LAB)

    # Tính toán ngưỡng dựa trên giá trị sáng trung bình của ảnh
    mean_brightness = np.mean(lab_image[:, :, 0])
    threshold = mean_brightness + threshold_adder  # Có thể
điều chỉnh giá trị ngưỡng tùy ý

    # Xác định vùng phản chiếu bằng cách tìm các điểm sáng
hơn ngưỡng
    reflection_mask = cv2.threshold(lab_image[:, :, 0],
threshold, 255, cv2.THRESH_BINARY)[1]
    reflection_mask = cv2.dilate(reflection_mask, None,
iterations=2)  # Mở rộng vùng phản chiếu

    # Tạo hình chữ nhật ước lượng chứa vùng phản chiếu
    contours, _ = cv2.findContours(reflection_mask,
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    x, y, w, h = cv2.boundingRect(max(contours,
key=cv2.contourArea))

    # Cắt bỏ phần phản chiếu khỏi ảnh gốc
    processed_image = image.copy()
    processed_image[y:y+h, x:x+w] =
cv2.medianBlur(processed_image[y:y+h, x:x+w], 15)

    # Lưu ảnh đã được xử lý
    cv2.imwrite(output_image_path, processed_image)

    # Trả về ảnh đã xử lý
    return output_image_path

def remove_reflection_on_frame(frame, threshold_adder):
    # Chuyển đổi ảnh sang không gian màu BGR nếu ảnh đầu
vào là ảnh grayscale
    if len(frame.shape) == 2:
        frame = cv2.cvtColor(frame,
cv2.COLOR_GRAY2BGR)

    # Chuyển đổi ảnh sang không gian màu LAB
    lab_frame = cv2.cvtColor(frame,
cv2.COLOR_BGR2LAB)

    # Tính toán ngưỡng dựa trên giá trị sáng trung bình của ảnh
    mean_brightness = np.mean(lab_frame[:, :, 0])
    threshold = mean_brightness + threshold_adder  # Có thể
điều chỉnh giá trị ngưỡng tùy ý

    # Xác định vùng phản chiếu bằng cách tìm các điểm sáng
hơn ngưỡng
    reflection_mask = cv2.threshold(lab_frame[:, :, 0],
threshold, 255, cv2.THRESH_BINARY)[1]
    reflection_mask = cv2.dilate(reflection_mask, None,
iterations=2)  # Mở rộng vùng phản chiếu

    # Tạo hình chữ nhật ước lượng chứa vùng phản chiếu
    contours, _ = cv2.findContours(reflection_mask,
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    if contours:  # Check if contours is not empty
        x, y, w, h = cv2.boundingRect(max(contours,
key=cv2.contourArea))
        # Cắt bỏ phần phản chiếu khỏi ảnh gốc
        processed_frame = frame.copy()
        processed_frame[y:y+h, x:x+w] =
cv2.medianBlur(processed_frame[y:y+h, x:x+w], 15)
    else:
        processed_frame = frame.copy()  # Assign a copy of the
input frame when contours is empty

    # Trả về ảnh đã xử lý
    return processed_frame
```

File **similarity_NMI.py**:

```
import skimage
from skimage.metrics import structural_similarity as ssim
import cv2 as cv
from scipy.stats import entropy
#from skimage.io import imread
import numpy as np
#import matplotlib.pyplot as plt

def compare_images(frame1, frame2):
    if frame1 is None or frame2 is None:
        return 0
    size = (300, 300)
    image1 = cv.resize(frame1, size)
    image2 = cv.resize(frame2, size)

    # Normalize the pixel values between 0 and 1
    if (np.max(image1) - np.min(image1))==0:
        image1_normalized = image1
```

```
    else:
        image1_normalized = (image1 - np.min(image1)) /
(np.max(image1) - np.min(image1))
    if (np.max(image2) - np.min(image2)) ==0:
        image2_normalized= image2
    else:
        image2_normalized = (image2 - np.min(image2)) /
(np.max(image2) - np.min(image2))

    # Convert image1 to black and white
    image1_bw = np.where(image1_normalized >= 0.5, 1, 0)

    # Convert image2 to black and white
    image2_bw = np.where(image2_normalized >= 0.5, 1, 0)

    # Calculate the structural similarity index (SSIM)
    similarity = ssim(image1_bw, image2_bw,
data_range=1.0)

    # Calculate the Mean square error (MSE)
    se = np.mean((image1_bw - image2_bw) ** 2)
    mse = se**(1/2)
    # Calculate the histograms
    histogram1 = np.histogram(image1_normalized,
bins=256)[0]
    histogram2 = np.histogram(image2_normalized,
bins=256)[0]
    joint_histogram =
np.histogram2d(image1_normalized.flatten(),
image2_normalized.flatten(), bins=256)[0]

    # Calculate the normalized mutual information (NMI)
    nmi = ((entropy(histogram1) + entropy(histogram2)) /
entropy(joint_histogram.flatten()))



    # Combine the normalized metrics
    combined_score = (nmi*similarity*(1-mse))**(1/3)
    if np.isnan(combined_score):
        combined_score = 0

    return combined_score


#display image
# # Paths to the two black and white images
# image1_path = 'image/fourier_result2.jpg'
# image2_path = 'image/fourier_result3.jpg'
# # Compare the images and get the similarity score, mask,
and images
# similarity_score, similarity_mask, image1_bw,
image2_bw, image1_normalized, image2_normalized =
compare_images(image1_path, image2_path)


# # Create a figure to display the images
# fig, axes = plt.subplots(2, 2, figsize=(10, 10))
# # Display image1
# axes[0, 0].imshow(image1_normalized, cmap='gray',
vmin=0, vmax=1)
# axes[0, 0].axis('off')
# axes[0, 0].set_title('Image 1 (Normalized)')
# # Display image2
# axes[0, 1].imshow(image2_normalized, cmap='gray',
vmin=0, vmax=1)
# axes[0, 1].axis('off')
# axes[0, 1].set_title('Image 2 (Normalized)')
# # Display the similarity mask
# axes[1, 0].imshow(similarity_mask, cmap='gray')
# axes[1, 0].axis('off')
# axes[1, 0].set_title('Similar Pixels (White) and Different
Pixels (Black)')
# # Add a text box to show the similarity score
# axes[1, 1].text(0.5, 0.5, f'Similarity: {similarity_score:.2f}',
ha='center', fontsize=12)
# axes[1, 1].axis('off')
# # Adjust the spacing between subplots
# plt.subplots_adjust(wspace=0.2, hspace=0.4)
# # Show the figure
# plt.show()
```

File **picamera_picture.py**:

```
import picamera
import cv2
import picamera.array
import numpy as np


#Hiên thị hình ảnh từ camera trên màn hình
def show_camera_preview():
    cv2.namedWindow("Camera Preview",
cv2.WINDOW_NORMAL)
    cv2.resizeWindow("Camera Preview", 480,240)

    #biến để lưu ảnh
    frame= None

    with picamera.PiCamera() as camera:
        camera.resolution = (480,240)
        camera.framerate = 24
        camera.start_preview()
```

```
    while True:

        stream
=np.empty((camera.resolution[1]*camera.resolution[0]*3),
dtype=np.uint8)
        camera.capture(stream, 'bgr')
        frame =stream.reshape ((camera.resolution[1],
camera.resolution[0], 3))

        #show picture to the screen
        cv2.imshow("Camera Preview",frame)

        #wait key
        key= cv2.waitKey(1)
        if key == ord ('q'):
            image_path="/home/snetel/Desktop/GammaImage
Process/image/webcam_image5.jpg"
            cv2.imwrite(image_path, frame)
            print("already taken picture: ",image_path)
            break
    camera.stop_preview()
    camera.close
  cv2.destroyAllWindows()
if __name__== '__main__':
  show_camera_preview()
```

S-ROBOT APP SOURCE CODE

File: **SignIn.java**:

```
package com.example.bluetooth;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class SignIn extends AppCompatActivity {
  private EditText emailEditText, passwordEditText;
  private Button signInButton;
  private TextView tvSignIn, tvforgot;
  private CheckBox rememberCheckbox;
  private FirebaseAuth firebaseAuth;
  private SharedPreferences sharedPreferences;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sign_in);

    // Khởi tạo Firebase Authentication
    firebaseAuth = FirebaseAuth.getInstance();

    // Ánh xạ các View
    emailEditText = findViewById(R.id.emailEt);
    passwordEditText = findViewById(R.id.passEt);
    signInButton = findViewById(R.id.button);
    tvSignIn = findViewById(R.id.tvSignIn);
    tvforgot = findViewById(R.id.tvforgot);
    rememberCheckbox =
findViewById(R.id.rememberMeCheckbox);

    // Khởi tạo SharedPreferences
    sharedPreferences =
getSharedPreferences("loginPrefs", MODE_PRIVATE);

    // Kiểm tra trạng thái đăng nhập được lưu trữ
    if (isUserLoggedIn()) {
      // Đăng nhập được duy trì, chuyển hướng người dùng
đến màn hình điều khiển (MainActivity)
      Intent intent = new Intent(SignIn.this,
MainActivity.class);
      startActivity(intent);
      finish(); // Kết thúc activity hiện tại để ngăn người
dùng quay lại màn hình đăng nhập
    } else {
      // Trạng thái đăng nhập không được lưu trữ hoặc
không hợp lệ, người dùng cần đăng nhập
      signInButton.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
          String email =
```

```
emailEditText.getText().toString();
                String password =
passwordEditText.getText().toString();
                // Gọi hàm đăng nhập
                signIn(email, password);
            }
        });
        tvSignIn.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(SignIn.this,
Information.class);
                startActivity(intent);
            }
        });
        tvforgot.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(SignIn.this,
ForgotPass.class);
                startActivity(intent);
            }
        });
    }
  }
  private void signIn(String email, String password) {
    // Kiểm tra các trường nhập không được trống
    if (email.isEmpty()) {
      showToast("Please enter email!");
      return;
    }
    if (password.isEmpty()) {
      showToast("Please enter password!");
      return;
    }
    // Thực hiện đăng nhập bằng Firebase Authentication
    firebaseAuth.signInWithEmailAndPassword(email,
password)
        .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(Task<AuthResult>
task) {
              if (task.isSuccessful()) {
                // Đăng nhập thành công
                showToast("Logged in successfully!");
                // Lưu trạng thái đăng nhập nếu CheckBox
được chọn
                if (rememberCheckbox.isChecked()) {
                  saveLoginStatus(true);
                }
                // Tiếp tục xử lý sau khi đăng nhập thành
công
                Intent intent = new Intent(SignIn.this,
MainActivity.class);
                startActivity(intent);
                finish();
              } else {
                // Đăng nhập thất bại
                showToast("Login failed!");
                Log.e("SignInActivity", "Đăng nhập thất
bại!", task.getException());
              }
            }
        });
  }
  private boolean isUserLoggedIn() {
    // Kiểm tra trạng thái đăng nhập trong
SharedPreferences
    return sharedPreferences.getBoolean("loggedIn", false);
  }
  private void saveLoginStatus(boolean isLoggedIn) {
    // Lưu trạng thái đăng nhập vào SharedPreferences
    SharedPreferences.Editor editor =
sharedPreferences.edit();
    editor.putBoolean("loggedIn", isLoggedIn);
    editor.apply();
  }
  private void showToast(String message) {
    Toast toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);
    toast.show();

    Handler handler = new Handler();
    handler.postDelayed(new Runnable() {
      @Override
      public void run() {
        toast.cancel();
      }
    }, 500);
  }
}
```

File: **activity_sign_in.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
  xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".SignIn">

<TextView
   android:id="@+id/hello"
   android:layout_width="78dp"
   android:layout_height="36dp"
   android:layout_alignParentLeft="true"
   android:layout_alignParentTop="true"
   android:layout_marginStart="23dp"
   android:layout_marginTop="110dp"
   android:gravity="top"
   android:text="@string/hello"
   android:textAppearance="@style/hello"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent"
   tools:ignore="MissingConstraints" />

<TextView
   android:id="@+id/let_s_control"
   android:layout_width="227dp"
   android:layout_height="20dp"
   android:layout_alignParentLeft="true"
   android:layout_alignParentTop="true"
   android:layout_marginStart="23dp"
   android:layout_marginTop="156dp"
   android:alpha="0.8"
   android:gravity="top"

   android:text="@string/let_s_control"
   android:textAppearance="@style/let_s_control"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent"
   tools:ignore="MissingConstraints" />

<com.google.android.material.textfield.TextInputLayout
   android:id="@+id/emailLayout"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:layout_marginHorizontal="16dp"
   android:layout_marginStart="23dp"
   android:layout_marginTop="230dp"
   android:layout_marginEnd="23dp"
   app:layout_constraintEnd_toEndOf="parent"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
      android:id="@+id/emailEt"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Type your Email"
      android:inputType="textEmailAddress" />

</com.google.android.material.textfield.TextInputLayout>

  <com.google.android.material.textfield.TextInputLayout
     android:id="@+id/passwordLayout"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_marginHorizontal="16dp"
     android:layout_marginStart="23dp"
     android:layout_marginTop="302dp"
     android:layout_marginEnd="23dp"
     app:layout_constraintEnd_toEndOf="parent"
     app:layout_constraintHorizontal_bias="0.5"
     app:layout_constraintStart_toStartOf="parent"
     app:layout_constraintTop_toTopOf="parent"
     app:passwordToggleEnabled="true">


<com.google.android.material.textfield.TextInputEditText
      android:id="@+id/passEt"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Type Your Password"
      android:inputType="textPassword" />

</com.google.android.material.textfield.TextInputLayout>

  <androidx.appcompat.widget.AppCompatButton
     android:id="@+id/button"
     android:layout_width="329dp"
     android:layout_height="50dp"
     android:layout_marginHorizontal="16dp"
     android:backgroundTint="#B5B5B5"
     android:text="Sign In"
     android:textColor="@color/black"
     android:textStyle="bold"
     app:layout_constraintBottom_toBottomOf="parent"
     app:layout_constraintEnd_toEndOf="parent"
     app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/passwordLa
yout"
```

```
        app:layout_constraintVertical_bias="0.501" />

    <TextView
        android:id="@+id/tvSignIn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Don’t Have Account? Sign Up"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button"
/>

    <TextView
        android:id="@+id/tvforgot"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginTop="370dp"
        android:gravity="right|top"
        android:text="@string/forgot_pass"
        android:textAppearance="@style/forgot_pass"
        android:textSize="14sp"

app:layout_constraintEnd_toEndOf="@+id/passwordLayout
"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />

    <CheckBox
        android:id="@+id/rememberMeCheckbox"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="357dp"
        android:buttonTint="#BCBDB9"
        android:text="Remember Me"
        android:textColor="#6A6F7D"

app:layout_constraintStart_toStartOf="@+id/passwordLayo
ut"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

File: **Information.java**:

```
package com.example.bluetooth;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import
com.google.android.material.textfield.TextInputEditText;
import com.google.firebase.firestore.CollectionReference;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.HashMap;
import java.util.Map;

import android.app.DatePickerDialog;
import android.icu.util.Calendar;

public class Information extends AppCompatActivity {

    private TextInputEditText fnameEditText, lnameEditText,
birthEditText, genderEditText, phoneEditText;
    private Button finishButton;
    private TextView tvSignIn;
    private DatePickerDialog.OnDateSetListener
birthDateListener;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_information);

        fnameEditText = findViewById(R.id.FnameEt);
        lnameEditText = findViewById(R.id.LnameEt);
        birthEditText = findViewById(R.id.birthEt);
        genderEditText = findViewById(R.id.genderEt);
        phoneEditText = findViewById(R.id.phoneEt);

        finishButton = findViewById(R.id.button);
        tvSignIn = findViewById(R.id.tvSignIn);
        birthDateListener = new
DatePickerDialog.OnDateSetListener() {
            @Override
            public void onDateSet(DatePicker view, int year, int
monthOfYear, int dayOfMonth) {
                // Handle the selected date
                // You can format the date as desired and set it to
```

```
the birth date EditText field
            String selectedDate = dayOfMonth + "/" +
(monthOfYear + 1) + "/" + year;
            birthEditText.setText(selectedDate);
          }
      };

      birthEditText.setOnClickListener(new
View.OnClickListener() {
          @Override
          public void onClick(View v) {
            // Get the current date
            Calendar calendar = Calendar.getInstance();
            int year = calendar.get(Calendar.YEAR);
            int month = calendar.get(Calendar.MONTH);
            int day =
calendar.get(Calendar.DAY_OF_MONTH);

            // Create and show the date picker dialog
            DatePickerDialog datePickerDialog = new
DatePickerDialog(Information.this, birthDateListener, year,
month, day);
            datePickerDialog.show();
          }
      });
      String[] genderOptions = {"Male", "Female", "Other"};

      // Thiết lập sự kiện khi bấm vào TextInputEditText
      genderEditText.setOnClickListener(new
View.OnClickListener() {
          @Override
          public void onClick(View v) {
            AlertDialog.Builder builder = new
AlertDialog.Builder(Information.this);
            builder.setTitle("Select Gender")
                .setItems(genderOptions, new
DialogInterface.OnClickListener() {
                  @Override
                  public void onClick(DialogInterface
dialog, int which) {
                    String selectedGender =
genderOptions[which];

genderEditText.setText(selectedGender);
                  }
                })
                .setNegativeButton("Cancel", null)
                .create()
                .show();
          }
      });
      finishButton.setOnClickListener(new
View.OnClickListener() {
          @Override
          public void onClick(View view) {
            saveUserData();
          }
      });
      tvSignIn.setOnClickListener(new
View.OnClickListener() {
          @Override
          public void onClick(View v) {
            Intent intent = new Intent(Information.this,
SignIn.class);
            startActivity(intent);
            finish();
          }
      });
  }
  private void saveUserData(){
      String fname = fnameEditText.getText().toString();
      String lname = lnameEditText.getText().toString();
      String birth = birthEditText.getText().toString();
      String gender = genderEditText.getText().toString();
      String phone = phoneEditText.getText().toString();

      // Kiểm tra không có trường nào để trống
      if (TextUtils.isEmpty(fname) ||
TextUtils.isEmpty(lname) || TextUtils.isEmpty(birth)
            || TextUtils.isEmpty(gender) ||
TextUtils.isEmpty(phone)) {
          Toast.makeText(this, "Empty Fields Are not Allowed
!!", Toast.LENGTH_SHORT).show();
          return;
      }
      if (!isValidPhoneNumber(phone)) {
          Toast.makeText(this, "Invalid Phone Number",
Toast.LENGTH_SHORT).show();
          return;
      }

      // Tạo Intent để chuyển sang activity khác
      Intent intent = new Intent(Information.this,
SignUp.class);

      // Đưa dữ liệu vào Intent
      intent.putExtra("fname", fname);
      intent.putExtra("lname", lname);
```

```
        intent.putExtra("birth", birth);
        intent.putExtra("gender", gender);
        intent.putExtra("phone", phone);

        startActivity(intent);
        finish();
    }
    private boolean isValidPhoneNumber(String phone) {
        // Kiểm tra số điện thoại hợp lệ theo yêu cầu của ứng
dụng của bạn.
        // Ví dụ: kiểm tra xem số điện thoại có chứa 10 chữ số
hay không.
        // Bạn có thể thay đổi quy tắc kiểm tra theo nhu cầu của
mình.
        return phone.length() == 10;
    }
}
```

File: **activity_information.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Information">

    <TextView
        android:id="@+id/hello"
        android:layout_width="106dp"
        android:layout_height="40dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="23dp"
        android:layout_marginTop="50dp"
        android:gravity="top"
        android:text="@string/signup"
        android:textAppearance="@style/signup"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />

    <TextView
        android:id="@+id/register_ne"
        android:layout_width="163dp"
        android:layout_height="20dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="23dp"
        android:layout_marginTop="90dp"
        android:alpha="0.8"
        android:gravity="top"

        android:text="@string/register_ne"
        android:textAppearance="@style/register_ne"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />
    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/FnameLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="200dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/FnameEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Frist name"
            android:inputType="textPersonName" />

</com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/LnameLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="275dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/LnameEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Last name"
            android:inputType="textPersonName" />
```

```
</com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/birthLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="350dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/birthEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Birth Date"
            android:focusable="false"
            android:clickable="true"
            android:inputType="date" />

</com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/GenderLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="425dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/genderEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Gender"
            android:focusable="false"
            android:clickable="true"
            android:inputType="text" />

</com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/phoneLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="500dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/phoneEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Phone"
            android:inputType="phone" />

</com.google.android.material.textfield.TextInputLayout>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:backgroundTint="#BCBDB9"
        android:text="Next"
        android:textColor="@color/white"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/phoneLayou
t" />
    <TextView
        android:id="@+id/tvSignIn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Already Registered, Sign In !"
        app:layout_constraintTop_toBottomOf="@id/button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintVertical_bias="0.5"/>
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

File: **SignUp.java**:

```
package com.example.bluetooth;

import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.text.TextUtils;
import android.util.Patterns;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.CollectionReference;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.HashMap;
import java.util.Map;
import java.util.regex.Pattern;

public class SignUp extends AppCompatActivity {
    private EditText emailEt, passEt, confirmPass;
    private Button button;
    private FirebaseAuth firebaseAuth;
    private FirebaseFirestore firestore;
    private CollectionReference usersCollection;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up);

        emailEt = findViewById(R.id.emailEt);
        passEt = findViewById(R.id.passEt);
        confirmPass = findViewById(R.id.confirmPass);
        button = findViewById(R.id.button);

        firebaseAuth = FirebaseAuth.getInstance();
        // Lấy tham chiếu đến collection "users" từ Firestore
        firestore = FirebaseFirestore.getInstance();
        usersCollection = firestore.collection("users");

        button.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String email = emailEt.getText().toString();
                String pass = passEt.getText().toString();
                String confirmPassText =
confirmPass.getText().toString();

                if (!email.isEmpty() && !pass.isEmpty() &&
!confirmPassText.isEmpty()) {
                    if (isEmailValid(email)) {
                        if (pass.equals(confirmPassText)) {
                            if (isPasswordValid(pass)) {
                                // Đăng ký người dùng và lưu thông tin
vào Firestore
                                signUpAndSaveUserData(email, pass);
                            } else {
                                showToast("Password must be at least 11
characters long and contain at least one lowercase letter, one
uppercase letter, one digit, and one special character.");
                            }
                        } else {
                            showToast("Password is not matching");
                        }
                    } else {
                        showToast("Invalid email format");
                    }
                } else {
                    showToast("Empty Fields Are not Allowed !!");
                }
            }
        });
    }

    private boolean isPasswordValid(String password) {
        Pattern passwordPattern =
Pattern.compile("^(?=.*[a-z])(?=.*[A-Z])(?=.*\\d)(?=.*[@$!
%*?&])[A-Za-z\\d@$!%*?&]{11,}$");
        return passwordPattern.matcher(password).matches();
    }

    private boolean isEmailValid(String email) {
        return
Patterns.EMAIL_ADDRESS.matcher(email).matches();
    }
    private void showToast(String message) {
        Toast toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);
        toast.show();

        Handler handler = new Handler();
        handler.postDelayed(new Runnable() {
```

```
        @Override
        public void run() {
            toast.cancel();
        }
    }, 1000);
  }
  private void signUpAndSaveUserData(String email,
String password) {
    // Đăng ký người dùng bằng Firebase Authentication

firebaseAuth.createUserWithEmailAndPassword(email,
password)
        .addOnCompleteListener(task -> {
          if (task.isSuccessful()) {
            Intent intent = new Intent(SignUp.this,
MainActivity.class);
            intent.putExtra("email", email);
            intent.putExtra("password", password);
            // Đăng ký thành công, lưu thông tin người
dùng vào Firestore
            saveUserData(email, password);
          } else {
            showToast("Failed to sign up: " +
task.getException().getMessage());
          }
        });
  }

  private void saveUserData(String email, String password)
{
    Intent intent = getIntent();
    String fname = intent.getStringExtra("fname");
    String lname = intent.getStringExtra("lname");
    String birth = intent.getStringExtra("birth");
    String gender = intent.getStringExtra("gender");
    String phone = intent.getStringExtra("phone");

    // Tạo một document với User Authentication ID làm
Document ID
    String userAuthId =
firebaseAuth.getCurrentUser().getUid();
    DocumentReference userDocRef =
firestore.collection("users").document(userAuthId);

    // Lưu thông tin người dùng vào tài liệu
    Map<String, Object> user = new HashMap<>();
    user.put("first_name", fname);
    user.put("last_name", lname);
    user.put("birth", birth);
    user.put("gender", gender);
    user.put("phone", phone);
    user.put("email", email);
    user.put("pass", password);

    // Thêm dữ liệu vào Firestore
    userDocRef.set(user)
        .addOnSuccessListener(documentReference -> {
          // Xử lý thành công
          showToast("User signed up and data saved
successfully!");
          finish();
        })
        .addOnFailureListener(e -> {
          // Xử lý lỗi
          showToast("Failed to save user data: " +
e.getMessage());
        });
  }
}
```

File: **activity_sign_up.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:visibility="visible"
  tools:context=".SignUp">

  <TextView
    android:id="@+id/hello"
    android:layout_width="106dp"
    android:layout_height="40dp"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginStart="23dp"
    android:layout_marginTop="50dp"
    android:gravity="top"
    android:text="@string/signup"
    android:textAppearance="@style/signup"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    tools:ignore="MissingConstraints" />

  <TextView
```

```
        android:id="@+id/register_ne"
        android:layout_width="163dp"
        android:layout_height="20dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="23dp"
        android:layout_marginTop="90dp"
        android:alpha="0.8"
        android:gravity="top"

        android:text="@string/register_ne"
        android:textAppearance="@style/register_ne"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />


    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/emailLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="250dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/emailEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Type your Email"
            android:inputType="textEmailAddress" />

</com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/passwordLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="325dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:passwordToggleEnabled="true">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/passEt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Type Your Password"
            android:inputType="textPassword" />

</com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/confirmPassLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:layout_marginStart="23dp"
        android:layout_marginTop="400dp"
        android:layout_marginEnd="23dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:passwordToggleEnabled="true">


<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/confirmPass"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Confirm Your Password"
            android:inputType="textPassword" />

</com.google.android.material.textfield.TextInputLayout>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="16dp"
        android:backgroundTint="#BCBDB9"
        android:text="Sign Up"
        android:textColor="@color/white"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/confirmPass
Layout" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

File: **ForgotPass.java**:

```
package com.example.bluetooth;

import android.content.Intent;
import android.os.Bundle;
import android.util.Patterns;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;

public class ForgotPass extends AppCompatActivity {

    private EditText emailEditText;
    private FirebaseAuth firebaseAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_forgot_pass);

        emailEditText = findViewById(R.id.emailEt);
        firebaseAuth = FirebaseAuth.getInstance();


findViewById(R.id.textviewSI).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(ForgotPass.this,
SignIn.class);
                startActivity(intent);
                finish();
            }
        });


findViewById(R.id.resetButton).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String email = emailEditText.getText().toString();

                if (!email.isEmpty()) {
                    if (isEmailValid(email)) {
                        resetPassword(email);
                    } else {
                        Toast.makeText(ForgotPass.this, "Invalid
email format", Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText(ForgotPass.this, "Please enter
your email", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    private boolean isEmailValid(String email) {
        return
Patterns.EMAIL_ADDRESS.matcher(email).matches();
    }

    private void resetPassword(String email) {

firebaseAuth.sendPasswordResetEmail(email).addOnCompl
eteListener(task -> {
            if (task.isSuccessful()) {
                // Yêu cầu đặt lại mật khẩu thành công
                Toast.makeText(ForgotPass.this, "Password reset
email sent", Toast.LENGTH_SHORT).show();
            } else {
                // Yêu cầu đặt lại mật khẩu thất bại
                Toast.makeText(ForgotPass.this, "Failed to send
password reset email", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

File: **activity_forgot_pass.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ForgotPass">

    <TextView
```

```
    android:id="@+id/forgotPass"
    android:layout_width="253dp"
    android:layout_height="36dp"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginStart="23dp"
    android:layout_marginTop="110dp"
    android:gravity="top"
    android:text="@string/forgot_pass"
    android:textAppearance="@style/forget_pass"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    tools:ignore="MissingConstraints" />

  <TextView
    android:id="@+id/resetLink"
    android:layout_width="245dp"
    android:layout_height="20dp"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginStart="23dp"
    android:layout_marginTop="156dp"
    android:alpha="0.8"
    android:gravity="top"

    android:text="@string/reset_link_"
    android:textAppearance="@style/reset_link_"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <com.google.android.material.textfield.TextInputLayout
    android:id="@+id/emailLayout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginHorizontal="16dp"
    android:layout_marginStart="23dp"
    android:layout_marginTop="230dp"
    android:layout_marginEnd="23dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">


<com.google.android.material.textfield.TextInputEditText
        android:id="@+id/emailEt"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Type your Email"
        android:inputType="textEmailAddress" />

</com.google.android.material.textfield.TextInputLayout>

  <androidx.appcompat.widget.AppCompatButton
    android:id="@+id/resetButton"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginHorizontal="16dp"
    android:backgroundTint="#BCBDB9"
    android:text="SEND EMAIL"
    android:textColor="@color/white"
    android:textStyle="bold"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <TextView
    android:id="@+id/textviewSI"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"

    android:text="Not Reset, Sign In !"

app:layout_constraintTop_toBottomOf="@id/resetButton"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintVertical_bias="0.5"   />

</androidx.constraintlayout.widget.ConstraintLayout>
```

File: **MainActivity.java**:

```
package com.example.bluetooth;

import android.Manifest;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.annotation.SuppressLint;
import android.app.AlertDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.os.Handler;
```

```
import android.provider.Settings;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.auth.FirebaseAuth;

import java.io.IOException;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.Set;
import java.util.UUID;

public class MainActivity extends AppCompatActivity {
    private static final int REQUEST_ENABLE_BT = 0;//yêu cầu bật Bluetooth
    private static final int REQUEST_BLUETOOTH_PERMISSION = 1;//yêu cầu cấp quyền cho bluetooth
    TextView mStatusBlueTv, status, tvSignOut;
    ImageView mBlueIv, joystickBase, joystick;
    Button mOnBtn, mOffBtn, mPairedBtn;
    ImageButton mSendBtn5, mSendBtn6, mSendBtn7, mSendBtn8;
    Switch mSendSwitch1, mSendSwitch2, mSendSwitch3, mSendSwitch4;
    BluetoothAdapter mBlueAdapter;
    BluetoothDevice mDevice;
    BluetoothSocket mSocket;
    OutputStream mOutputStream;
    RelativeLayout layoutJoystick;
    private static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB"); // UUID của dịch vụ Bluetooth Serial Port
    private String dataTosend = "M";
    private float baseX = 0f;
    private float baseY = 0f;
    private float joystickX = 0f;
    private float joystickY = 0f;
    private SharedPreferences sharedPreferences;

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mStatusBlueTv = findViewById(R.id.statusBluetoothTv);
        status = findViewById(R.id.status);
        mPairedBtn = findViewById(R.id.pairedBtn);
        mBlueIv = findViewById(R.id.bluetoothIv);
        mOnBtn = findViewById(R.id.onBtn);
        mOffBtn = findViewById(R.id.offBtn);

        mSendBtn5 = findViewById(R.id.sendBtn5);
        mSendBtn6 = findViewById(R.id.sendBtn6);
        mSendBtn7 = findViewById(R.id.sendBtn7);
        mSendBtn8 = findViewById(R.id.sendBtn8);

        mSendSwitch1 = findViewById(R.id.switch1);
        mSendSwitch2 = findViewById(R.id.switch2);
        mSendSwitch3 = findViewById(R.id.switch3);
        mSendSwitch4 = findViewById(R.id.switch4);

        joystickBase = findViewById(R.id.joystickBase);
        joystick = findViewById(R.id.joystick);
        layoutJoystick = findViewById(R.id.layoutJoystick);

        tvSignOut = findViewById(R.id.tvSignOut);

        mBlueAdapter = BluetoothAdapter.getDefaultAdapter();//quản lý các chức năng liên quan đến Bluetooth
        // Khởi tạo SharedPreferences
        sharedPreferences = getSharedPreferences("loginPrefs", MODE_PRIVATE);
        // Check Bluetooth permission
        if (ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT)
                != PackageManager.PERMISSION_GRANTED) {
            // Quyền chưa được cấp, yêu cầu từ người dùng
            ActivityCompat.requestPermissions(this,
                    new String[]{Manifest.permission.BLUETOOTH_CONNECT},
                    REQUEST_BLUETOOTH_PERMISSION);
        } else {
            // Quyền đã được cấp, tiếp tục thực hiện tác vụ Bluetooth
            performBluetoothTask();
        }
```

```
//       // Check if Bluetooth is available
//       if (mBlueAdapter == null) {
//          mStatusBlueTv.setText("Bluetooth is Not
available");
//       } else {
//          mStatusBlueTv.setText("Bluetooth is available");
//       }
      // Set status icon/image based on Bluetooth state
//       if (mBlueAdapter.isEnabled()) {
//
mBlueIv.setImageResource(R.drawable.ic_action_on);
//       } else {
//
mBlueIv.setImageResource(R.drawable.ic_action_off);
//       }
      //Joystick
      layoutJoystick.setOnTouchListener(new
View.OnTouchListener() {
         @Override
         public boolean onTouch(View v, MotionEvent event)
{
            float centerX = layoutJoystick.getWidth() / 2f;
            float centerY = layoutJoystick.getHeight() / 2f;

            switch (event.getAction()) {
               case MotionEvent.ACTION_DOWN:
                  baseX = event.getX() - centerX;
                  baseY = event.getY() - centerY;
                  break;
               case MotionEvent.ACTION_MOVE:
                  joystickX = event.getX() - centerX;
                  joystickY = event.getY() - centerY;

                  double distance =
Math.sqrt(Math.pow(joystickX, 2) + Math.pow(joystickY,
2));
                  double angle = Math.atan2(joystickY,
joystickX);

                  if (distance > centerX) {
                     joystickX = (float) (centerX *
Math.cos(angle));
                     joystickY = (float) (centerX *
Math.sin(angle));
                  }

                  joystick.setX(joystickX + baseX);
                  joystick.setY(joystickY + baseY);

                  double directionAngle =
Math.toDegrees(angle);

                  if (!dataTosend.equals("A")) {
                     dataTosend =
calculateControlState(directionAngle);
                     writeMessage(dataTosend);
                  }
                  break;
               case MotionEvent.ACTION_UP:
                  joystick.animate()
                        .x(centerX - joystick.getWidth() / 2f)
                        .y(centerY - joystick.getHeight() / 2f)
                        .setDuration(200)
                        .start();

                  if (!dataTosend.equals("A"))
                     dataTosend = "P";
                  Log.d("datasend", dataTosend);
                  writeMessage(dataTosend);
                  break;
            }
            return true;
         }
      });
      // Turn On Bluetooth
      mOnBtn.setOnClickListener(new
View.OnClickListener() {
         @Override
         public void onClick(View view) {

mBlueIv.setImageResource(R.drawable.ic_action_on);

            // Kiểm tra xem ứng dụng đã có quyền
BLUETOOTH_ADMIN chưa
            if
(ContextCompat.checkSelfPermission(MainActivity.this,

Manifest.permission.BLUETOOTH_ADMIN) ==
PackageManager.PERMISSION_GRANTED) {

               // Kiểm tra xem Bluetooth đã bật chưa
               if (!mBlueAdapter.isEnabled()) {
                  Toast.makeText(MainActivity.this, "Turning
Bluetooth On...", Toast.LENGTH_SHORT).show();

                  // Gửi yêu cầu bật Bluetooth
                  Intent intent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                  startActivityForResult(intent,
REQUEST_ENABLE_BT);

               } else {
                  Toast.makeText(MainActivity.this,
"Bluetooth is already On",
```

```
Toast.LENGTH_SHORT).show();
                }
            } else {
                // Yêu cầu quyền BLUETOOTH_ADMIN nếu
chưa được cấp

ActivityCompat.requestPermissions(MainActivity.this,
                        new
String[]{Manifest.permission.BLUETOOTH_ADMIN},
                        REQUEST_ENABLE_BT);
            }
        }
    });

// Turn Off Bluetooth
    mOffBtn.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View view) {

mBlueIv.setImageResource(R.drawable.ic_action_off);

            // Kiểm tra xem ứng dụng đã có quyền
BLUETOOTH_ADMIN chưa
            if
(ContextCompat.checkSelfPermission(MainActivity.this,

Manifest.permission.BLUETOOTH_ADMIN) ==
PackageManager.PERMISSION_GRANTED) {

                // Kiểm tra xem Bluetooth đã bật chưa
                if (mBlueAdapter.isEnabled()) {
                    closeSocket();
                    mBlueAdapter.disable();
                    Toast.makeText(MainActivity.this, "Turning
Bluetooth Off...", Toast.LENGTH_SHORT).show();

                } else {
                    Toast.makeText(MainActivity.this,
"Bluetooth is already Off...",
Toast.LENGTH_SHORT).show();
                }
            } else {
                // Yêu cầu quyền BLUETOOTH_ADMIN nếu
chưa được cấp

ActivityCompat.requestPermissions(MainActivity.this,
                        new
String[]{Manifest.permission.BLUETOOTH_ADMIN},
                        REQUEST_ENABLE_BT);
            }
        }
    });
    // Select device button
    mPairedBtn.setOnClickListener(new
View.OnClickListener() {
        @SuppressLint("MissingPermission")
        @Override
        public void onClick(View view) {
            if (mBlueAdapter.isEnabled()) {
                showDeviceSelectionDialog();
            } else {
                showToast("Turn On Bluetooth to get paired
devices");
            }
        }
    });
    setButtonTouchListener(mSendBtn5, "D", "");
    setButtonTouchListener(mSendBtn6, "U", "");
    mSendSwitch1.setOnCheckedChangeListener(new
SwitchCheckedChangeListener("E", "H"));
    mSendSwitch2.setOnCheckedChangeListener(new
SwitchCheckedChangeListener("I", "K"));
    mSendSwitch3.setOnCheckedChangeListener(new
SwitchCheckedChangeListener("V", "N"));
    //  Switch to change mode from Manual to Auto
    mSendSwitch4.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
        @Override
        public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
            if (isChecked) {
                String newData = "A";
                dataTosend = newData;
                mSendSwitch4.setText("Auto    ");
                writeMessage(dataTosend);
            } else {
                String newData = "M";
                dataTosend = newData;
                mSendSwitch4.setText("Manual");
                writeMessage(dataTosend);
            }
        }
    });
    mSendBtn7.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // Chuyển sang màn hình xem Video từ Redis
            showToast("Video Mode");
```

```
            Intent intent = new Intent(MainActivity.this,
BluetoothVideo.class);
            intent.putExtra("bluetooth_device", mDevice);
            startActivity(intent);
        }
    });
    mSendBtn8.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            dataTosend = "Y";
            try{
                writeMessage(dataTosend);
            } catch (Exception e) {
                showToast("NotConnect");
            }
            // Chuyển sang màn hình xem Video từ Redis
            showToast("Wifi Mode");
            Intent intent = new Intent(MainActivity.this,
WifiVideo.class);
            startActivity(intent);
            //finish(); // Đóng hoạt động hiện tại nếu bạn không
muốn quay lại hoạt động trước
        }
    });
    tvSignOut.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(MainActivity.this,
Profile.class);
            startActivity(intent);
        }
    });

    if (mDevice != null) {
        if (mSocket != null && mSocket.isConnected()) {
            // Thiết bị đang kết nối
            // Gửi dữ liệu qua Bluetooth
            writeMessage(dataTosend);
        } else {
            // Thiết bị đang kết nối bị tắt
            showToast("HC-06 is not paired or Bluetooth is
off");
        }
    }
  }
  //  Show device and select device
  private void showDeviceSelectionDialog() {
    Set<BluetoothDevice> devices =
mBlueAdapter.getBondedDevices();
    if (devices.size() > 0) {
        //  Show device
        ArrayList<BluetoothDevice> deviceList = new
ArrayList<>();
        final String[] deviceNames = new
String[devices.size()];
        int index = 0;
        for (BluetoothDevice device : devices) {
            deviceList.add(device);
            deviceNames[index] = device.getName();
            index++;
        }
        //Select device
        AlertDialog.Builder builder = new
AlertDialog.Builder(this);
        builder.setTitle("Select a device")
                .setItems(deviceNames, new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog,
int which) {
                        mDevice = deviceList.get(which);
                        try {

Toast.makeText(getApplicationContext(), "Connecting...",
Toast.LENGTH_LONG).show();
                            connectToDevice();
                        } catch (IOException e) {
                            e.printStackTrace();
                        }
                        showToast("Selected device: " +
mDevice.getName());
                        status.setText(mDevice.getName());
                    }
                });
        builder.show();
    } else {
        Toast.makeText(MainActivity.this, "No paired
devices found", Toast.LENGTH_SHORT).show();
    }
  }
  //  Kết nối tới thiết bị
  private void connectToDevice() throws IOException {
    mSocket =
mDevice.createRfcommSocketToServiceRecord(MY_UUI
D);
    mSocket.connect();
    mOutputStream = mSocket.getOutputStream();
    //  Khi kết nối tới thiết bị thì gửi "M" để chọn chế độ
Manual
    writeMessage("J");
    writeMessage("M");
  }
  // Gửi tín hiệu qua bluetooth
```

```
    private void writeMessage(String message) {
        try {
            // Kiểm tra xem BluetoothSocket và kết nối đã tồn tại
và đang hoạt động không
            if (mSocket != null && mSocket.isConnected() &&
mOutputStream != null) {
                // Gửi dữ liệu tới thiết bị Bluetooth
                mOutputStream.write(message.getBytes());
                mOutputStream.flush();
                showToast("Message sent: " + message);
                Log.d("datasend", message);
            } else {
                status.setText("Not Connected");
                showToast("Device is not paired or Bluetooth is
off");
            }
        } catch (IOException e) {
            // Xử lý ngoại lệ IOException
            showToast("Failed to send message: " +
e.getMessage());
            e.printStackTrace();
            // Đóng kết nối Bluetooth
            closeSocket();
        }
    }
    //  Đóng kết nối
    private void closeSocket() {
        if (mSocket != null) {
            try {
                mSocket.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
    @Override
    protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
        switch (requestCode) {
            case REQUEST_ENABLE_BT:
                if (resultCode == RESULT_OK) {
                    // Bluetooth is ON

mBlueIv.setImageResource(R.drawable.ic_action_on);
                    showToast("Bluetooth is ON");
                } else {
                    showToast("Failed to enable Bluetooth");
                }
                break;
        }
        super.onActivityResult(requestCode, resultCode, data);
    }
    // Hiển thị thông báo ra màn hình trong 100ms
    private void showToast(String message) {
        Toast toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);
        toast.show();

        Handler handler = new Handler();
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                toast.cancel();
            }
        }, 100);
    }
    private String calculateControlState(double angle) {
        if (angle >= -22.5 && angle <= 22.5) {
            return "R";//Right
        } else if (angle > 22.5 && angle <= 67.5) {
            return "R";//Bottom Right
        } else if (angle > 67.5 && angle <= 112.5) {
            return "B";//Bottom
        } else if (angle > 112.5 && angle <= 157.5) {
            return "L";//Bottom Left
        } else if ((angle > 157.5 && angle <= 180) || (angle >=
-180 && angle <= -157.5)) {
            return "L";//Left
        } else if (angle > -157.5 && angle <= -112.5) {
            return "L";//Forward Left
        } else if (angle > -112.5 && angle <= -67.5) {
            return "F";//Forward
        } else if (angle > -67.5 && angle <= -22.5) {
            return "R";//Forward Right
        } else {
            return "P";//Pause
        }
    }
    @Override
    public void onRequestPermissionsResult(int requestCode,
@NonNull String[] permissions, @NonNull int[]
grantResults) {
        super.onRequestPermissionsResult(requestCode,
permissions, grantResults);

        if (requestCode == REQUEST_ENABLE_BT) {
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                // Bluetooth Admin permission granted, continue
Bluetooth tasks
```

```
            performBluetoothTask();
        } else {
            // Bluetooth Admin permission denied, handle the situation accordingly
            handleBluetoothPermissionDenied();
        }
    }
}
private void performBluetoothTask() {
    // TODO: Thực hiện các tác vụ Bluetooth ở đây
    if (mBlueAdapter == null) {
        mStatusBlueTv.setText("Bluetooth is Not available");
    } else {
        mStatusBlueTv.setText("Bluetooth is available");
    }
    // Set status icon/image based on Bluetooth state
    if (mBlueAdapter.isEnabled()) {

mBlueIv.setImageResource(R.drawable.ic_action_on);
    } else {

mBlueIv.setImageResource(R.drawable.ic_action_off);
    }
}
// Phương thức xử lý khi người dùng từ chối cấp quyền Bluetooth
private void handleBluetoothPermissionDenied() {
    // TODO: Xử lý khi người dùng từ chối cấp quyền Bluetooth
    showBluetoothPermissionRequestDialog();
}
// Xử lý khi người dùng từ chối cấp quyền Bluetooth
private void showBluetoothPermissionRequestDialog() {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Yêu cầu cấp quyền Bluetooth");
    builder.setMessage("Ứng dụng cần cấp quyền Bluetooth để thực hiện các tác vụ liên quan đến Bluetooth. Vui lòng cấp quyền để tiếp tục.");
    builder.setPositiveButton("Cấp quyền", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            // Chuyển đến màn hình cấu hình quyền Bluetooth
            navigateToBluetoothPermissionSettings();
        }
    });
    builder.setNegativeButton("Hủy", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            // Xử lý khi người dùng từ chối cấp quyền Bluetooth
            handleBluetoothPermissionDenied();
        }
    });
    builder.show();
}
private void navigateToBluetoothPermissionSettings() {
    // Tạo một Intent để chuyển đến màn hình cài đặt Bluetooth
    Intent intent = new Intent(Settings.ACTION_BLUETOOTH_SETTINGS);
    // Kiểm tra phiên bản Android để chuyển trực tiếp đến màn hình cấp quyền Bluetooth
    // Chuyển đến màn hình cấp quyền Bluetooth trong cài đặt hệ thống

intent.setAction(Settings.ACTION_BLUETOOTH_SETTINGS);
    intent.putExtra(Settings.EXTRA_APP_PACKAGE, getPackageName());
    intent.putExtra(Settings.EXTRA_CHANNEL_ID, getApplicationInfo().uid);
    // Khởi chạy Intent để mở màn hình cấp quyền Bluetooth
    startActivity(intent);
}
// Nút nhấn
private void setButtonTouchListener(ImageButton button, final String dataOnTouchDown, final String dataOnTouchUp) {
    button.setOnTouchListener((v, event) -> {
        if (event.getAction() == MotionEvent.ACTION_DOWN) {
            writeMessage(dataOnTouchDown);
        } else if (event.getAction() == MotionEvent.ACTION_UP) {
            writeMessage(dataOnTouchUp);
        }
        return true;
    });
}
private class SwitchCheckedChangeListener implements CompoundButton.OnCheckedChangeListener {
    private String dataOn;
    private String dataOff;
    public SwitchCheckedChangeListener(String dataOn, String dataOff) {
        this.dataOn = dataOn;
        this.dataOff = dataOff;
        showToast(dataOff);
```

```
        }
        @Override
        public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
            if (isChecked) {
                writeMessage(dataOn);
                showToast(dataOn);
            } else {
                writeMessage(dataOff);
                showToast(dataOff);
            }
        }
    }
    protected void onPause() {
        super.onPause();
        // Luu trạng thái kết nối vào SharedPreferences
        SharedPreferences.Editor editor =
sharedPreferences.edit();
        editor.putBoolean("isConnected", (mSocket != null &&
mSocket.isConnected()));
        editor.apply();
    }
    @Override
    protected void onResume() {
        super.onResume();
        if (dataTosend.equals("P")) {
            writeMessage("P");
        }
    }
}
```

File: **activity_main.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/statusBluetoothTv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="20dp"
        android:text="Bluetooth Status"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/FTv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="480dp"
        android:text="Motor"
        android:textSize="18sp"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/FTv2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="580dp"
        android:text="Another Function"
        android:textSize="18sp"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/FTv3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="680dp"
        android:text="Device:"
        android:textSize="18sp"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/status"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="680dp"
        android:text="Device Name"
        android:textSize="18sp"
        app:layout_constraintStart_toEndOf="@+id/FTv3"
```

```
    app:layout_constraintTop_toTopOf="parent" />

  <ImageView
    android:id="@+id/bluetoothIv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:src="@drawable/ic_action_off"

app:layout_constraintStart_toEndOf="@+id/statusBluetooth
Tv"
    app:layout_constraintTop_toTopOf="parent" />

  <Button
    android:id="@+id/onBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="60dp"
    android:text="On"
    app:layout_constraintEnd_toStartOf="@+id/offBtn"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <Button
    android:id="@+id/offBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="60dp"
    android:text="Off"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/onBtn"
    app:layout_constraintTop_toTopOf="parent" />

  <Button
    android:id="@+id/pairedBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="130dp"
    android:text="Get Paired Devices"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <TextView
    android:id="@+id/DCTv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="10dp"
    android:layout_marginTop="180dp"
    android:text="Direction control"
    android:textSize="18sp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="175dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_forward"-->
  <!--    android:text="F"-->
  <!--    app:layout_constraintEnd_toEndOf="parent"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--    app:layout_constraintStart_toStartOf="parent"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->

  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn2"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="250dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_back"-->
  <!--    android:text="L"-->
  <!--
app:layout_constraintEnd_toStartOf="@+id/sendBtn3"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--    app:layout_constraintStart_toStartOf="parent"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->

  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn3"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="250dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_backward"-->
  <!--    android:text="B"-->
  <!--
app:layout_constraintEnd_toStartOf="@+id/sendBtn4"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--
app:layout_constraintStart_toEndOf="@+id/sendBtn2"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->

  <!--  <ImageButton-->
```

```
    <!--        android:id="@+id/sendBtn4"-->
    <!--        android:layout_width="wrap_content"-->
    <!--        android:layout_height="wrap_content"-->
    <!--        android:layout_marginTop="250dp"-->
    <!--        android:background="@drawable/circle"-->
    <!--        android:src="@drawable/arrow_right"-->
    <!--        android:text="R"-->
    <!--        app:layout_constraintEnd_toEndOf="parent"-->
    <!--        app:layout_constraintHorizontal_bias="0.5"-->
    <!--
app:layout_constraintStart_toEndOf="@+id/sendBtn3"-->
    <!--        app:layout_constraintTop_toTopOf="parent"
/>-->

    <TextView
        android:id="@+id/STv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="380dp"
        android:text="Speed control"
        android:textSize="18sp"
        android:textStyle="bold"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageButton
        android:id="@+id/sendBtn5"
        android:layout_width="90dp"
        android:layout_height="35dp"
        android:layout_marginTop="420dp"
        android:src="@drawable/minus"
        android:background="@drawable/rounded_corner"
        app:layout_constraintEnd_toStartOf="@+id/sendBtn6"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageButton
        android:id="@+id/sendBtn6"
        android:layout_width="90dp"
        android:layout_height="35dp"
        android:layout_marginTop="420dp"
        android:background="@drawable/rounded_corner"
        android:src="@drawable/add"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/sendBtn5"
        app:layout_constraintTop_toTopOf="parent" />

    <Switch
        android:id="@+id/switch1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="510dp"
        android:text="Pump    "
        app:layout_constraintEnd_toStartOf="@+id/switch2"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />


    <Switch
        android:id="@+id/switch3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="545dp"
        android:text="Vacuum"
        app:layout_constraintEnd_toStartOf="@+id/switch4"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Switch
        android:id="@+id/switch4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="545dp"
        android:text="Manual"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/switch3"
        app:layout_constraintTop_toTopOf="parent" />

    <Switch
        android:id="@+id/switch2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="510dp"
        android:text="Brush   "
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/switch1"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageButton
        android:id="@+id/sendBtn7"
        android:layout_width="90dp"
        android:layout_height="35dp"
        android:layout_marginTop="620dp"
```

```
    android:background="@drawable/rounded_corner"
    android:src="@drawable/video"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/sendBtn8"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />


<ImageButton
    android:id="@+id/sendBtn8"
    android:layout_width="90dp"
    android:layout_height="35dp"
    android:layout_marginTop="620dp"
    android:background="@drawable/rounded_corner"
    android:src="@drawable/wifi"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/sendBtn7"
    app:layout_constraintTop_toTopOf="parent" />

<RelativeLayout
    android:id="@+id/layoutJoystick"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="225dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">

    <ImageView
        android:id="@+id/joystickBase"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:src="@drawable/circle_big" />

    <ImageView
        android:id="@+id/joystick"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_centerInParent="true"
        android:src="@drawable/circle" />

</RelativeLayout>


<ImageView
    android:id="@+id/arrow_up"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="213dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/arrow_up" />

    <ImageView
        android:id="@+id/arrow_down"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="75dp"

app:layout_constraintStart_toEndOf="@+id/arrow_left"

app:layout_constraintTop_toBottomOf="@+id/arrow_up"
        app:srcCompat="@drawable/arrow_down" />

    <ImageView
        android:id="@+id/arrow_left"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:layout_marginEnd="15dp"
        app:layout_constraintEnd_toStartOf="@+id/arrow_up"

app:layout_constraintTop_toBottomOf="@+id/arrow_up"
        app:srcCompat="@drawable/arrow_left" />

    <ImageView
        android:id="@+id/arrow_right"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="10dp"
        app:layout_constraintStart_toEndOf="@+id/arrow_up"

app:layout_constraintTop_toBottomOf="@+id/arrow_up"
        app:srcCompat="@drawable/arrow_right" />

    <TextView
        android:id="@+id/tvSignOut"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:layout_marginEnd="10dp"
        android:text="Profile"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent" />


</androidx.constraintlayout.widget.ConstraintLayout>
```

File: **BluetoothVideo.java**:

```
package com.example.bluetooth;

import android.Manifest;
import android.annotation.SuppressLint;
import android.app.AlertDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.DialogInterface;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import java.io.IOException;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Base64;
import java.util.Set;
import java.util.UUID;

import redis.clients.jedis.Jedis;

public class BluetoothVideo extends AppCompatActivity {

    private static final String redisHost =
"redis-17060.c299.asia-northeast1-1.gce.cloud.redislabs.com";
    private static final int redisPort = 17060;
    private static final String redisPassword =
"YS9EKyvuTG5Q3HGxKFem48ePa7ykaV93";
    private Jedis jedis;
    private RedisConnectionTask connectionTask;

    private static final int REQUEST_ENABLE_BT = 0;//yêu
cầu bật Bluetooth
    private static final int
REQUEST_BLUETOOTH_PERMISSION = 1;//yêu cầu cấp
quyền cho bluetooth
    TextView status, dataTextView;
    ImageView joystickBase, joystick, Back;
    Button mOnBtn, mOffBtn, mPairedBtn;
    ImageButton mSendBtn5, mSendBtn6;
    Switch mSendSwitch1, mSendSwitch2, mSendSwitch3,
mSendSwitch4;
    BluetoothAdapter mBlueAdapter;
    BluetoothDevice mDevice;
    BluetoothSocket mSocket;
    OutputStream mOutputStream;
    RelativeLayout layoutJoystick;
    private static final UUID MY_UUID =
UUID.fromString("00001101-0000-1000-8000-00805F9B3
4FB"); // UUID của dịch vụ Bluetooth Serial Port
    private String dataTosend = "M";
    private float baseX = 0f;
    private float baseY = 0f;
    private float joystickX = 0f;
    private float joystickY = 0f;
    private boolean isRunning = true;
    private String battery = "";
    private String water = "";
    private String time = "";
    private RedisConnection redisConnection; // Thêm biến
redisConnection
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bluetooth_video);
        Back = findViewById(R.id.Back);

        status = findViewById(R.id.status);

        mPairedBtn = findViewById(R.id.pairedBtn);
        mOnBtn = findViewById(R.id.onBtn);
        mOffBtn = findViewById(R.id.offBtn);
```

```
        mSendBtn5 = findViewById(R.id.sendBtn5);
        mSendBtn6 = findViewById(R.id.sendBtn6);

        mSendSwitch1 = findViewById(R.id.switch1);
        mSendSwitch2 = findViewById(R.id.switch2);
        mSendSwitch3 = findViewById(R.id.switch3);
        mSendSwitch4 = findViewById(R.id.switch4);

        joystickBase = findViewById(R.id.joystickBase);
        joystick = findViewById(R.id.joystick);
        layoutJoystick = findViewById(R.id.layoutJoystick);

        dataTextView = findViewById(R.id.dataTextView);

        mBlueAdapter =
BluetoothAdapter.getDefaultAdapter();//quản lý các chức
năng liên quan đến Bluetooth

        final Handler handler = new Handler();
        final int delayMs = 100; // Độ trễ giữa các lần đọc (ms)

        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                connectionTask = new RedisConnectionTask();
                connectionTask.execute();
                handler.postDelayed(this, delayMs); // Lặp lại sau
khoảng thời gian delayMs
            }
        }, delayMs);



        if (ContextCompat.checkSelfPermission(this,
android.Manifest.permission.BLUETOOTH_CONNECT)
                != PackageManager.PERMISSION_GRANTED)
{
            // Quyền chưa được cấp, yêu cầu từ người dùng
            ActivityCompat.requestPermissions(this,
                    new
String[]{Manifest.permission.BLUETOOTH_CONNECT},
                    REQUEST_BLUETOOTH_PERMISSION);
        } else {
            // Quyền đã được cấp, tiếp tục thực hiện tác vụ
Bluetooth
//            performBluetoothTask();
        }

        //Show devide and choise device to connect
        mPairedBtn.setOnClickListener(new
View.OnClickListener() {
            @SuppressLint("MissingPermission")
            @Override
            public void onClick(View view) {
                if (mBlueAdapter.isEnabled()) {
                    showDeviceSelectionDialog();
                } else {
                    showToast("Turn On Bluetooth to get paired
devices");
                }
            }
        });
        //Joystick
        layoutJoystick.setOnTouchListener(new
View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event)
{
                float centerX = layoutJoystick.getWidth() / 2f;
                float centerY = layoutJoystick.getHeight() / 2f;

                switch (event.getAction()) {
                    case MotionEvent.ACTION_DOWN:
                        baseX = event.getX() - centerX;
                        baseY = event.getY() - centerY;
                        break;
                    case MotionEvent.ACTION_MOVE:
                        joystickX = event.getX() - centerX;
                        joystickY = event.getY() - centerY;

                        double distance =
Math.sqrt(Math.pow(joystickX, 2) + Math.pow(joystickY,
2));
                        double angle = Math.atan2(joystickY,
joystickX);

                        if (distance > centerX) {
                            joystickX = (float) (centerX *
Math.cos(angle));
                            joystickY = (float) (centerX *
Math.sin(angle));
                        }

                        joystick.setX(joystickX + baseX);
                        joystick.setY(joystickY + baseY);

                        double directionAngle =
Math.toDegrees(angle);

                        if (!dataTosend.equals("A")) {
                            dataTosend =
calculateControlState(directionAngle);
                            try {
                                writeMessage(dataTosend);
                            } catch (IOException e) {
```

```
                    throw new RuntimeException(e);
                }
            }
            break;
        case MotionEvent.ACTION_UP:
            joystick.animate()
                    .x(centerX - joystick.getWidth() / 2f)
                    .y(centerY - joystick.getHeight() / 2f)
                    .setDuration(200)
                    .start();

            if (!dataTosend.equals("A"))
                dataTosend = "P";
            Log.d("datasend", dataTosend);
            try {
                writeMessage(dataTosend);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
            break;
    }
    return true;
  }
});
// Switch
mSendSwitch1.setOnCheckedChangeListener(new
BluetoothVideo.SwitchCheckedChangeListener("E", "H"));
mSendSwitch2.setOnCheckedChangeListener(new
BluetoothVideo.SwitchCheckedChangeListener("I", "K"));
mSendSwitch3.setOnCheckedChangeListener(new
BluetoothVideo.SwitchCheckedChangeListener("V", "N"));
// Switch to change mode from Manual to Auto
mSendSwitch4.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
        if (isChecked) {
            String newData = "A";
            dataTosend = newData;
            mSendSwitch4.setText("Auto    ");
            try {
                writeMessage(dataTosend);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        } else {
            String newData = "M";
            dataTosend = newData;
            mSendSwitch4.setText("Manual");
            try {
                writeMessage(dataTosend);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        }
    }
});

Back.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View view) {
//          RedisConnection.sendSignal("camera", "off");
        finish();
    }
});

if (mDevice != null) {
    if (mSocket != null && mSocket.isConnected()) {
        // Thiết bị đang kết nối
        // Gửi dữ liệu qua Bluetooth
        try {
            writeMessage(dataTosend);
        } catch (IOException e) {
            showToast("Failed to send message");
            e.printStackTrace();
        }
    } else {
        // Thiết bị đang kết nối bị tắt
        showToast("HC-06 is not paired or Bluetooth is
off");
    }
}
@Override
protected void onResume() {
    super.onResume();
    if (connectionTask != null &&
connectionTask.getStatus() ==
AsyncTask.Status.FINISHED) {
        connectionTask = new RedisConnectionTask();
        connectionTask.execute();
    }
}
@Override
protected void onDestroy() {
    super.onDestroy();
    //RedisConnection.sendSignal("camera", "off");
    //jedis.set("camera","off");
```

```
        isRunning = false; // Stop the frame reception loop
        if (connectionTask != null &&
connectionTask.getStatus() !=
AsyncTask.Status.FINISHED) {
            connectionTask.cancel(true);
        }
        //closeRedisConnection();
        RedisConnection.disconnect();
    }
    private class RedisConnectionTask extends
AsyncTask<Void, Void, Void> {
        @Override
        protected Void doInBackground(Void... voids) {
            Log.d("RedisConnectionTaskBV", "Connecting to
Redis...");
            try {
                jedis = new Jedis(redisHost, redisPort);
                jedis.auth(redisPassword);
                Log.d("RedisConnectionTaskBV", "Connected to
Redis");

//                // Receive realtime signal
//                List<String> redisKeys = new ArrayList<>();
//                redisKeys.add("move");
//                redisKeys.add("toggle");
                jedis.set("camera","on");
                Log.d("RedisConnectionTaskBV", "Starting
receiveRealtimeFrame()...");
                receiveRealtimeFrame();
                Log.d("RedisConnectionTaskBV",
"receiveRealtimeFrame() completed");
                jedis.close();
                Log.d("RedisConnectionTaskBV", "Jedis
connection closed");
            } catch (IOException e) {
                Log.e("RedisConnectionTaskBV", "Error in
receiving realtime frame: " + e.getMessage());
                throw new RuntimeException(e);
            }
            return null;
        }
    }

    private void receiveRealtimeFrame() throws IOException
{
        List<String> fullFrameData = new ArrayList<>();
        String frameForDecode = "";

        while (isRunning) {
            battery = jedis.get("battery");
            water = jedis.get("water");
            time = jedis.get("time");
            String cameraControl = jedis.get("camera");
            if (cameraControl != null &&
cameraControl.equals("on")) {
                String frameChunk = jedis.get("video");
                Log.d("RedisConnectionTaskBV", frameChunk);
                String[] parts = frameChunk.split("_");
                List<String> partList = Arrays.asList(parts);

                String data = partList.get(1);
                String chunkSuffix = partList.get(2);

                fullFrameData.add(data);
                if (chunkSuffix.equals("endframe")) {
                    StringBuilder frameBuilder = new
StringBuilder();
                    Log.d("RedisConnectionTaskBV",
"RedisConnectionTaskBV");
                    for (String chunk : fullFrameData) {
                        frameBuilder.append(chunk);
                    }
                    frameForDecode = frameBuilder.toString();
                    Log.d("RedisConnectionTaskBV",
frameForDecode);
                    Bitmap videoFrame =
decodeFrame(frameForDecode);
                    if (videoFrame != null) {
                        runOnUiThread(() -> {
                            Log.d("RedisConnectionTaskBV",
videoFrame.toString());
                            ImageView imageView =
findViewById(R.id.imageView);
                            imageView.setImageBitmap(videoFrame);

                        });
                    }
                    runOnUiThread(() -> {
                        // Update the dataTextView with battery, time,
and water values
                        String displayText = "Battery: " + battery +
"%\t\tWater: " + water + "ml\t\tTime: " + time;
                        dataTextView.setText(displayText);
                    });
                    fullFrameData.clear();
                }
            }
        }
    }
    private void closeRedisConnection() {
        if (jedis != null) {
            jedis.close();
        }
    }
```

```
    private Bitmap decodeFrame(String frameForDecode) {
        byte[] imgBytes =
Base64.getDecoder().decode(frameForDecode);
        Bitmap bitmap =
BitmapFactory.decodeByteArray(imgBytes, 0,
imgBytes.length);
        return bitmap;
    }
    private void showToast(String message) {
        Toast toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);
        toast.show();

        Handler handler = new Handler();
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                toast.cancel();
            }
        }, 100);
    }
    private String calculateControlState(double angle) {
        if (angle >= -22.5 && angle <= 22.5) {
            return "R";//Right
        } else if (angle > 22.5 && angle <= 67.5) {
            return "R";//Bottom Right
        } else if (angle > 67.5 && angle <= 112.5) {
            return "B";//Bottom
        } else if (angle > 112.5 && angle <= 157.5) {
            return "L";//Bottom Left
        } else if ((angle > 157.5 && angle <= 180) || (angle >=
-180 && angle <= -157.5)) {
            return "L";//Left
        } else if (angle > -157.5 && angle <= -112.5) {
            return "L";//Forward Left
        } else if (angle > -112.5 && angle <= -67.5) {
            return "F";//Forward
        } else if (angle > -67.5 && angle <= -22.5) {
            return "R";//Forward Right
        } else {
            return "P";//Pause
        }
    }
    private void writeMessage(String message) throws
IOException {
        // Kiểm tra xem socket đã được kết nối chưa
        if (mSocket != null && mSocket.isConnected()) {
            // Gửi dữ liệu tới thiết bị Bluetooth
            mOutputStream.write(message.getBytes());
            mOutputStream.flush();
            showToast("Message sent: " + message);
            Log.d("datasend", message);
        } else {
            status.setText("Not Connected");
            showToast("Device is not paired or Bluetooth is
off");
        }
    }

    private void connectToDevice() throws IOException {
        mSocket =
this.mDevice.createRfcommSocketToServiceRecord(MY_U
UID);
        mSocket.connect();
        mOutputStream = mSocket.getOutputStream();
        //  Khi kết nối tới thiết bị thì gửi "M" để chọn chế độ
Manual
        try {
            writeMessage("M");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
    //Show devide and choise device to connect
    private void showDeviceSelectionDialog() {
        Set<BluetoothDevice> devices =
mBlueAdapter.getBondedDevices();
        if (devices.size() > 0) {
            //  Show device
            ArrayList<BluetoothDevice> deviceList = new
ArrayList<>();
            final String[] deviceNames = new
String[devices.size()];
            int index = 0;
            for (BluetoothDevice device : devices) {
                deviceList.add(device);
                deviceNames[index] = device.getName();
                index++;
            }
            //Select device
            AlertDialog.Builder builder = new
AlertDialog.Builder(this);
            builder.setTitle("Select a device")
                    .setItems(deviceNames, new
DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
int which) {
                            mDevice = deviceList.get(which);
                            try {
```

```
Toast.makeText(getApplicationContext(), "Connecting...",
Toast.LENGTH_LONG).show();
                        connectToDevice();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                    showToast("Selected device: " +
mDevice.getName());
                    status.setText(mDevice.getName());
                }
            });
        builder.show();
    } else {
        showToast("No paired devices found");
    }
  }
  //Switch
  private class SwitchCheckedChangeListener implements
CompoundButton.OnCheckedChangeListener {
    private String dataOn;
    private String dataOff;
    public SwitchCheckedChangeListener(String dataOn,
String dataOff) {
        this.dataOn = dataOn;
        this.dataOff = dataOff;
        showToast(dataOff);
    }
    @Override
    public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
        if (isChecked) {
            try {
                writeMessage(dataOn);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
            showToast(dataOn);
        } else {
            try {
                writeMessage(dataOff);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
            showToast(dataOff);
        }
    }
  }
}
```

File: **activity_bluetooth_video.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".BluetoothVideo">

  <ImageView
    android:id="@+id/imageView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:scaleType="centerCrop"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0" />

  <ImageView
    android:id="@+id/Back"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="15dp"
    android:layout_marginTop="15dp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/back" />


  <TextView
    android:id="@+id/FTv3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Device:"
    android:textSize="18sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/status"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent" />

  <TextView
    android:id="@+id/status"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="20dp"
    android:text="Device Name"
    android:textSize="18sp"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent" />


  <Button
    android:id="@+id/onBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="On"

app:layout_constraintBottom_toTopOf="@+id/switch4"
    app:layout_constraintEnd_toEndOf="@+id/offBtn"
    app:layout_constraintStart_toStartOf="@+id/offBtn"
    app:layout_constraintTop_toBottomOf="@+id/offBtn"
    app:layout_constraintVertical_bias="0.5" />

  <Button
    android:id="@+id/offBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="50dp"
    android:text="Off"
    app:layout_constraintBottom_toTopOf="@+id/onBtn"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.5" />

  <Button
    android:id="@+id/pairedBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:text="Get Paired Devices"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="175dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_forward"-->
  <!--    android:text="F"-->
  <!--    app:layout_constraintEnd_toEndOf="parent"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--    app:layout_constraintStart_toStartOf="parent"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->


  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn2"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="250dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_back"-->
  <!--    android:text="L"-->
  <!--
app:layout_constraintEnd_toStartOf="@+id/sendBtn3"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--    app:layout_constraintStart_toStartOf="parent"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->


  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn3"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="250dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_backward"-->
  <!--    android:text="B"-->
  <!--
app:layout_constraintEnd_toStartOf="@+id/sendBtn4"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--
app:layout_constraintStart_toEndOf="@+id/sendBtn2"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->


  <!--  <ImageButton-->
  <!--    android:id="@+id/sendBtn4"-->
  <!--    android:layout_width="wrap_content"-->
  <!--    android:layout_height="wrap_content"-->
  <!--    android:layout_marginTop="250dp"-->
  <!--    android:background="@drawable/circle"-->
  <!--    android:src="@drawable/arrow_right"-->
  <!--    android:text="R"-->
  <!--    app:layout_constraintEnd_toEndOf="parent"-->
  <!--    app:layout_constraintHorizontal_bias="0.5"-->
  <!--
app:layout_constraintStart_toEndOf="@+id/sendBtn3"-->
  <!--    app:layout_constraintTop_toTopOf="parent"
/>-->

  <ImageButton
```

```
        android:id="@+id/sendBtn5"
        android:layout_width="90dp"
        android:layout_height="35dp"
        android:layout_marginBottom="25dp"
        android:background="@drawable/rounded_corner"
        android:src="@drawable/minus"

app:layout_constraintBottom_toTopOf="@+id/switch1"
        app:layout_constraintEnd_toEndOf="@+id/switch1"
        app:layout_constraintStart_toStartOf="@+id/switch1"
/>

    <ImageButton
        android:id="@+id/sendBtn6"
        android:layout_width="90dp"
        android:layout_height="35dp"
        android:layout_marginBottom="25dp"
        android:background="@drawable/rounded_corner"
        android:src="@drawable/add"

app:layout_constraintBottom_toTopOf="@+id/sendBtn5"
        app:layout_constraintEnd_toEndOf="@+id/sendBtn5"

app:layout_constraintStart_toStartOf="@+id/sendBtn5" />

    <Switch
        android:id="@+id/switch1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:text="Pump"
        android:textColor="#F01CE8"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Switch
        android:id="@+id/switch3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="20dp"
        android:text="Vacuum"
        android:textColor="#F01CE8"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/switch2" />

    <Switch
        android:id="@+id/switch4"
        android:layout_width="100dp"
        android:layout_height="50dp"
        android:layout_marginEnd="65dp"
        android:scaleX="1.5"
        android:scaleY="1.5"
        android:text="Manual"
        android:textColor="#F01CE8"

app:layout_constraintBottom_toTopOf="@+id/layoutJoystick"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/onBtn"
/>

    <Switch
        android:id="@+id/switch2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="20dp"
        android:text="Brush"
        android:textColor="#F01CE8"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/switch1" />

    <RelativeLayout
        android:id="@+id/layoutJoystick"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="40dp"
        android:layout_marginBottom="40dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent">

        <ImageView
            android:id="@+id/joystickBase"
            android:layout_width="150dp"
            android:layout_height="150dp" />

        <ImageView
            android:id="@+id/joystick"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_centerInParent="true"
            android:src="@drawable/circle_big" />

    </RelativeLayout>

    <TextView
        android:id="@+id/dataTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Information"
```

```
        android:textSize="20sp"
        android:textColor="#F01CE8"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

File: **WifiVideo.java**:

```
package com.example.bluetooth;

import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.os.StrictMode;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.CompoundButton;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Base64;

import redis.clients.jedis.Jedis;

public class WifiVideo extends AppCompatActivity {

    private static final String redisHost =
"redis-17060.c299.asia-northeast1-1.gce.cloud.redislabs.co
m";
    private static final int redisPort = 17060;
    private static final String redisPassword =
"YS9EKyvuTG5Q3HGxKFem48ePa7ykaV93";
    private Jedis jedis;
    TextView dataTextView;
    private RedisConnectionTask connectionTask;
    private RedisConnection redisConnection; // Thêm biến
redisConnection
    ImageView joystickBase, joystick, Back;
    RelativeLayout layoutJoystick;
    private float baseX = 0f;
    private float baseY = 0f;
    private float joystickX = 0f;
    private float joystickY = 0f;
    private String baseData = "P";

    private Switch switch1, switch2, switch3, switch4;
    private ImageButton button1, button2, button6, button7;
    private boolean isRunning = true;
    private String battery = "";
    private String water = "";
    private String time = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_wifi_video);
        Back = findViewById(R.id.Back);

        switch1= findViewById(R.id.switch1);
        switch2= findViewById(R.id.switch2);
        switch3= findViewById(R.id.switch3);
        switch4= findViewById(R.id.switch4);

        button1 = findViewById(R.id.button1);
        button2 = findViewById(R.id.button2);

        joystickBase = findViewById(R.id.joystickBase);
        joystick = findViewById(R.id.joystick);
        layoutJoystick = findViewById(R.id.layoutJoystick);

        dataTextView = findViewById(R.id.dataTextView);

        StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);

        connectionTask = new RedisConnectionTask();
        connectionTask.execute();

        // Khởi tạo đối tượng redisConnection
        redisConnection = new RedisConnection(redisHost,
redisPort, redisPassword);
```

```
//        RedisConnection.sendSignal("move", "start");
        RedisConnection.sendSignal("camera", "on");
        RedisConnection.sendSignal("move", "Y");
        layoutJoystick.setOnTouchListener(new
View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event)
{
                float centerX = layoutJoystick.getWidth() / 2f;
                float centerY = layoutJoystick.getHeight() / 2f;

                switch (event.getAction()) {
                    case MotionEvent.ACTION_DOWN:
                        baseX = event.getX() - centerX;
                        baseY = event.getY() - centerY;
                        break;
                    case MotionEvent.ACTION_MOVE:
                        joystickX = event.getX() - centerX;
                        joystickY = event.getY() - centerY;

                        double distance =
Math.sqrt(Math.pow(joystickX, 2) + Math.pow(joystickY,
2));
                        double angle = Math.atan2(joystickY,
joystickX);

                        if (distance > centerX) {
                            joystickX = (float) (centerX *
Math.cos(angle));
                            joystickY = (float) (centerX *
Math.sin(angle));
                        }

                        joystick.setX(joystickX + baseX);
                        joystick.setY(joystickY + baseY);

                        double directionAngle =
Math.toDegrees(angle);

                        if (!baseData.equals("A")) {
                            baseData =
calculateControlState(directionAngle);
                            RedisConnection.sendSignal("move",
baseData);
                            //showToast(baseData);
                        }
                        break;
                    case MotionEvent.ACTION_UP:
                        joystick.animate()
                                .x(centerX - joystick.getWidth() / 2f)
                                .y(centerY - joystick.getHeight() / 2f)
                                .setDuration(200)
                                .start();

                        if (!baseData.equals("A"))
                            baseData = "P";
                        Log.d("datasend", baseData);
                        RedisConnection.sendSignal("move",
baseData);
                        //showToast(baseData);
                        break;
                }
                return true;
            }
        });

        Back.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View view) {
//                Intent intent = new Intent(WifiVideo.this,
Control.class);
//                startActivity(intent);
                finish();
            }
        });
        switch1.setOnCheckedChangeListener(new
WifiVideo.SwitchCheckedChangeListener("E", "H"));
        switch2.setOnCheckedChangeListener(new
WifiVideo.SwitchCheckedChangeListener("I", "K"));
        switch3.setOnCheckedChangeListener(new
WifiVideo.SwitchCheckedChangeListener("V", "N"));
        switch4.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
                if (isChecked) {
                    String newData = "A";
                    baseData = newData;
                    switch4.setText("Auto    ");
                    RedisConnection.sendSignal("move",
baseData);
                } else {
                    String newData = "M";
                    baseData = newData;
                    switch4.setText("Manual");
                    RedisConnection.sendSignal("move",
baseData);
                }
                //showToast(baseData);
            }
        });
```

```
        button1.setOnTouchListener(new
WifiVideo.ButtonTouchListener("D", baseData));
        button2.setOnTouchListener(new
WifiVideo.ButtonTouchListener("U", baseData));

    }
    private String calculateControlState(double angle) {
        if (angle >= -22.5 && angle <= 22.5) {
            return "R";//Right
        } else if (angle > 22.5 && angle <= 67.5) {
            return "R";//Bottom Right
        } else if (angle > 67.5 && angle <= 112.5) {
            return "B";//Bottom
        } else if (angle > 112.5 && angle <= 157.5) {
            return "L";//Bottom Left
        } else if ((angle > 157.5 && angle <= 180) || (angle >=
-180 && angle <= -157.5)) {
            return "L";//Left
        } else if (angle > -157.5 && angle <= -112.5) {
            return "L";//Forward Left
        } else if (angle > -112.5 && angle <= -67.5) {
            return "F";//Forward
        } else if (angle > -67.5 && angle <= -22.5) {
            return "R";//Forward Right
        } else {
            return "P";//Pause
        }
    }

    @Override
    protected void onResume() {
        super.onResume();
        if (connectionTask != null &&
connectionTask.getStatus() ==
AsyncTask.Status.FINISHED) {
            connectionTask = new RedisConnectionTask();
            connectionTask.execute();
        }
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        RedisConnection.sendSignal("camera", "off");
        RedisConnection.sendSignal("move", "J");
        isRunning = false; // Stop the frame reception loop
        if (connectionTask != null &&
connectionTask.getStatus() !=
AsyncTask.Status.FINISHED) {
            connectionTask.cancel(true);
        }
        closeRedisConnection();
        RedisConnection.disconnect();
    }
    private class RedisConnectionTask extends
AsyncTask<Void, Void, Void> {
        @Override
        protected Void doInBackground(Void... voids) {
            Log.d("RedisConnectionTaskWV", "Connecting to
Redis...");
            try {
                jedis = new Jedis(redisHost, redisPort);
                jedis.auth(redisPassword);
                Log.d("RedisConnectionTaskWV", "Connected to
Redis");

                Log.d("RedisConnectionTaskWV", "Starting
receiveRealtimeFrame()...");
                receiveRealtimeFrame();
                Log.d("RedisConnectionTaskWV",
"receiveRealtimeFrame() completed");
                jedis.close();
                Log.d("RedisConnectionTaskWV", "Jedis
connection closed");
            } catch (IOException e) {
                Log.e("RedisConnectionTaskWV", "Error in
receiving realtime frame: " + e.getMessage());
                throw new RuntimeException(e);
            }
            return null;
        }
    }

    private void receiveRealtimeFrame() throws IOException
{
        List<String> fullFrameData = new ArrayList<>();
        String frameForDecode = "";

        while (isRunning) {
            battery = jedis.get("battery");
            water = jedis.get("water");
            time = jedis.get("time");
            String cameraControl = jedis.get("camera");
            if (cameraControl != null &&
cameraControl.equals("on")) {
                String frameChunk = jedis.get("video");
                Log.d("RedisConnectionTaskWV", frameChunk);
                String[] parts = frameChunk.split("_");
                List<String> partList = Arrays.asList(parts);
```

```
            String data = partList.get(1);
            String chunkSuffix = partList.get(2);

            fullFrameData.add(data);
            if (chunkSuffix.equals("endframe")) {
               StringBuilder frameBuilder = new
StringBuilder();
               Log.d("RedisConnectionTaskWV",
"RedisConnectionTaskWV");
               for (String chunk : fullFrameData) {
                  frameBuilder.append(chunk);
               }
               frameForDecode = frameBuilder.toString();
               Log.d("RedisConnectionTaskWV",
frameForDecode);
               Bitmap videoFrame =
decodeFrame(frameForDecode);
               if (videoFrame != null) {
                  runOnUiThread(() -> {
                     Log.d("RedisConnectionTaskWV",
videoFrame.toString());
                     ImageView imageView =
findViewById(R.id.imageView);
                     imageView.setImageBitmap(videoFrame);

                  });
               }
               runOnUiThread(() -> {
                  // Update the dataTextView with battery, time,
and water values
                  String displayText = "Battery: " + battery +
"%\t\tWater: " + water + "ml\t\tTime: " + time;
                  dataTextView.setText(displayText);
               });
               fullFrameData.clear();
            }
         }
      }
   }

   private void closeRedisConnection() {
      if (jedis != null) {
         jedis.close();
      }
      redisConnection.disconnect();
   }

   private Bitmap decodeFrame(String frameForDecode) {
      byte[] imgBytes =
Base64.getDecoder().decode(frameForDecode);
      Bitmap bitmap =
BitmapFactory.decodeByteArray(imgBytes, 0,
imgBytes.length);
      return bitmap;
   }
   private class SwitchCheckedChangeListener implements
CompoundButton.OnCheckedChangeListener {
      private String dataOn;
      private String dataOff;

      public SwitchCheckedChangeListener(String dataOn,
String dataOff) {
         this.dataOn = dataOn;
         this.dataOff = dataOff;
         //showToast(dataOff);
      }

      @Override
      public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
         if (isChecked) {
            redisConnection.sendSignal("move", dataOn);
            //showToast(dataOn);
         } else {
            redisConnection.sendSignal("move", dataOff);
            //showToast(dataOff);
         }
      }
   }
   private class ButtonTouchListener implements
View.OnTouchListener {
      private String dataDown;
      private String dataUp;

      public ButtonTouchListener(String dataDown, String
dataUp) {
         this.dataDown = dataDown;
         this.dataUp = dataUp;
         //showToast(dataDown);
      }

      @Override
      public boolean onTouch(View v, MotionEvent event) {
         switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:
               RedisConnection.sendSignal("move",
dataDown);
               //showToast(dataDown);
               break;
            case MotionEvent.ACTION_UP:
               RedisConnection.sendSignal("move", dataUp);
               //showToast(dataUp);
               break;
         }
```

```
            return false;
        }

    }
    private void showToast(String message) {
        Toast toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);
        toast.show();

        Handler handler = new Handler();
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                toast.cancel();
            }
        }, 100);
    }

}
```

File: **activity_wifi_video.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".WifiVideo">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="centerCrop"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/Back"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="15dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/back" />


    <Switch
        android:id="@+id/switch1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginBottom="1dp"
        android:text="Pump"
        android:textColor="#F01CE8"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Switch
        android:id="@+id/switch2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="20dp"
        android:text="Brush"
        android:textColor="#F01CE8"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/switch1" />

    <Switch
        android:id="@+id/switch3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="15dp"
        android:layout_marginTop="20dp"
        android:text="Vacuum"
        android:textColor="#F01CE8"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/switch2" />

    <RelativeLayout
        android:id="@+id/layoutJoystick"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="40dp"
        android:layout_marginBottom="40dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent">
```

```
    <ImageView
        android:id="@+id/joystickBase"
        android:layout_width="150dp"
        android:layout_height="150dp" />

    <ImageView
        android:id="@+id/joystick"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_centerInParent="true"
        android:src="@drawable/circle_big" />

  </RelativeLayout>


  <ImageButton
    android:id="@+id/button1"
    android:layout_width="90dp"
    android:layout_height="35dp"
    android:layout_marginBottom="25dp"
    android:background="@drawable/rounded_corner"
    android:src="@drawable/add"

app:layout_constraintBottom_toTopOf="@+id/button2"
    app:layout_constraintEnd_toEndOf="@+id/button2"
    app:layout_constraintStart_toStartOf="@+id/button2"
/>

  <ImageButton
    android:id="@+id/button2"
    android:layout_width="90dp"
    android:layout_height="35dp"
    android:layout_marginBottom="25dp"
    android:background="@drawable/rounded_corner"
    android:src="@drawable/minus"

app:layout_constraintBottom_toTopOf="@+id/switch1"
    app:layout_constraintEnd_toEndOf="@+id/switch1"
    app:layout_constraintStart_toStartOf="@+id/switch1"
/>

  <Switch
    android:id="@+id/switch4"
    android:layout_width="100dp"
    android:layout_height="50dp"
    android:layout_marginEnd="65dp"
    android:scaleX="1.5"
    android:scaleY="1.5"
    android:text="Manual"
    android:textColor="#F01CE8"

app:layout_constraintBottom_toTopOf="@+id/layoutJoysti
ck"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <TextView
    android:id="@+id/dataTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Information"
    android:textSize="20sp"
    android:textColor="#F01CE8"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

APPENDIX

[1] The list of websites:
- https://www.ecoppia.com/
- https://www.pmu.edu.sa/attachments/academics/
- https://vuphong.vn/robot-ve-sinh-tam-pin-mat-troi/
- https://www.youtube.com/watch?v=p5jQDos14Rg
- https://www.youtube.com/watch?v=7cBEXbhJUJg

[2] The list of shops:
- https://hshop.vn/
- https://nshopvn.com/
- https://www.dosangtao.vn/
- https://tae.vn/goi-do-ngang
- https://haophong.com/

[3] The list of workshops:
- https://www.cokhibk.com/puly-r%C4%83ng-l
- https://cokhiphucthanh.com/
- https://3dthinking.vn/
- https://phanlonglaser.vn/

ACKNOWLEDGMENT

The authors would like to thank Snetel Technologies Company Limited for their support on creating this project, this support was crucial to materialize this project.

REFERENCES

[1] Nguyen Huu Loc, Textbook of Machine Design, Publishing Company of Vietnam National University Ho Chi Minh City, 2020.
[2] Trinh Chat, Le Van Uyen, Calculation and design of mechanical drive systems (Part 1 and 2), Publishing Company of Educational, 2006.
[3] The Basics of How an Encoder Works. Available: https://www.encoder.com/wp2011-basics-how-an-encoder-works
[4] PID controller. Available: https://en.wikipedia.org/wiki/PID_controller
[5] R. C. Gonzalez, R. E. Woods, Digital Image Processing, Addison Wesley, 2nd edition (2002), 3rd edition (2008).
[6] R. C. Gonzalez, R. E. Woods, S. L. Eddins, Digital Image Processing using Matlab, Gatesmark Publishing, 2nd edition (2009) H. Poor, *An*

Introduction to Signal Detection and Estimation. New York: Springer-Verlag, 1985, ch. 4.

[7] "Self-Driving Course - Part 1 - Perception - Advanced Lane Detection - 1 (5)", Youtube, May 21,2021, Available at: https://www.youtube.com/watch?v=f41AAKV2CFM&list=PLbv11v7kH7vU85BfOS65HA8DjKgxu5hi9&index=5