HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Multidisciplinary Project

Clean Air Version 1

Smart Air Purifier System for Household Usage

Instructor: Mai Đức Trung
Students: Cao Minh Quang - 2052221
Trần Cao Duy Trường -2052299
Lâm Quang Khải - 2052128
Văn Ngọc Thanh Tùng - 2052782
Lê Hoàng Minh - 2052595

HO CHI MINH CITY, January 2023

Contents

Members List & Workloads

No.	Full name	Student ID	Task	Contribution
1	Cao Minh Quang	2052221	Sensor System Implementation and Iot Gateway Development	20%
2	Trần Cao Duy Trường	2052299	Back-end Application Development	20%
3	Lâm Quang Khải	2052128	Sensor System Implementation and Iot Gateway Development	20%
4	Văn Ngọc Thanh Tùng	2052782	Iot Gateway and MQTT Server Development	20%
5	Lê Hoàng Minh	2052595	Front-end Application Development	20%

History Log

- **Week 1: From 12/09/2022 to 19/09/2022**
  - **Task:**

    Meeting the instructor.

    Organizing the first group meeting.

    Determining the topic's outlines.
  - **Work Contents:**

    Identifying team members.

    Determining how to work in a team.

    Dividing functional branches for team members.

    Providing a list of device.
- **Week 2: From 20/09/2022 to 27/09/2022**
  - **Task:**

    Proposing user requirements.

    Introducing an overview of developments from the group's choice approved by the instructor.

    Getting familiar with the server and creating a dashboard on the Adafruit server.
  - **Work Contents:**

    General requirements of the topic.

    Referring to the manual on creating Server-Dashboard.
- **Week 3: From 28/09/2022 to 04/10/2022**
  - **Task:**

    Determining system requirements including detailed requirements, use-case and use-case specification.

    Implementing and integrating IoT Gateway with Python.
  - **Work Contents:**

    Detailed requirements of the topic.

    Detailed use-case with interaction and measurable non-functional requirements.

    Referring to the user manual on implementing Iot Gateway.
- **Week 4: From 05/10/2022 to 11/10/2022**
  - **Task:**

    Screen (UI/UX) design.

    Implementing and integrating with Microbit circuit with sensors.
  - **Work Contents:**

    Design User Interfaces.

    Referring to the user manual and implementing sensors.
- **Week 5: From 12/10/2022 to 18/10/2022**

- **Task:**

  Database design.

  Completing the integration with Microbit circuit.

- **Work Contents:**

  Design database for the app.

  Updating the user interface according to the instructor's comments.

  Referring to the user manual and integrating Microbit with sensors.

- **Week 6: From 19/10/2022 to 25/10/2022**

  - **Task:**

    Coding to interface and connecting the application to the database, and connecting the control server to a real device.

  - **Work Contents:**

    Coding to interface and connecting database, and connecting control server with an actual device.

- **Week 7: From 20/10/2022 to 26/10/2022**

  - **Task:**

    Midterm demo.

  - **Work Contents:**

    Demoing the app that connects to the device (at least, works with module 1).

- **Week 8: From 26/10/2022 to 02/11/2022**

  - **Task:**

    Finalizing the app.

  - **Work Contents:**

    Coding testing (fixing bugs, upgrading, and integrating features in the selected direction).

- **Week 9: From 03/11/2022 to 09/11/2022**

  - **Task:**

    Finalizing the app.

  - **Work Contents:**

    Coding testing (fixing bugs, upgrading, and integrating features in the selected direction).

- **Week 10: From 10/11/2022 to 16/11/2022**

  - **Task:**

    Final report.

  - **Work Contents:**

    Submitting the final report and source code.

- **Week 11: From 02/01/2023 to 08/01/2023**
  - **Task:**

    Backup week for final demo.
  - **Work Contents:**

    Demoing the whole system.

Git & Github for version control

Please access the following link to gain a full control of the project source code:https://github.com/cmq2002/IoT_Air_Purifier

1 Introduction

In most parts of the world, especially in big cities, the issue of air pollution is getting progressively worse. One of the economic hubs that contributes to and is affected by this pollution is Ho Chi Minh City. After realizing the gravity and immediacy of the issue, our team set out to create a project called Clean Air. The project's goal is to develop a smart air purification system for domestic usage.

Air pollution leads people to be exposed to fine particles in polluted air that penetrate deep into the lungs and cardiovascular system, causing diseases including stroke, heart disease, lung cancer, chronic obstructive pulmonary diseases and respiratory infections. Industry, transportation, coal power plants and household solid fuel usage are major contributors to air pollution. Air pollution continues to rise at an alarming rate, and affects economies and people's quality of life.

An air purifier or air cleaner is a device which removes contaminants from the air in a room to improve indoor air quality. These devices are commonly marketed as being beneficial to allergy sufferers and asthmatics, and at reducing or eliminating second-hand tobacco smoke.

Our technology connects a home-made air purifier to a mobile device for remote management. Modules, fundamental electronic parts, and a single-board micro-controller make up the purifier. Information is exchanged with the purifier using IoT gateways by the mobile software, which is primarily written in the Flutter programming language. Basically, this system will measure the air indicators, show them on the LCD panel or mobile screen, and let the user manually turn on or off using switches or automatically using a mobile app. Then the purifier will absorb the nearby air, blow it through activated carbon filters and therefore clean the atmosphere.

Our team is hoping that this project will help bring air purification technology into more homes around our nation, specifically in Ho Chi Minh City. Consequently, raising the standard of living and ensuring everyone's health.

The below diagram is a simplified schematic for the construction of our air purifier machine. We expect that it is able to give a general overview of our project.

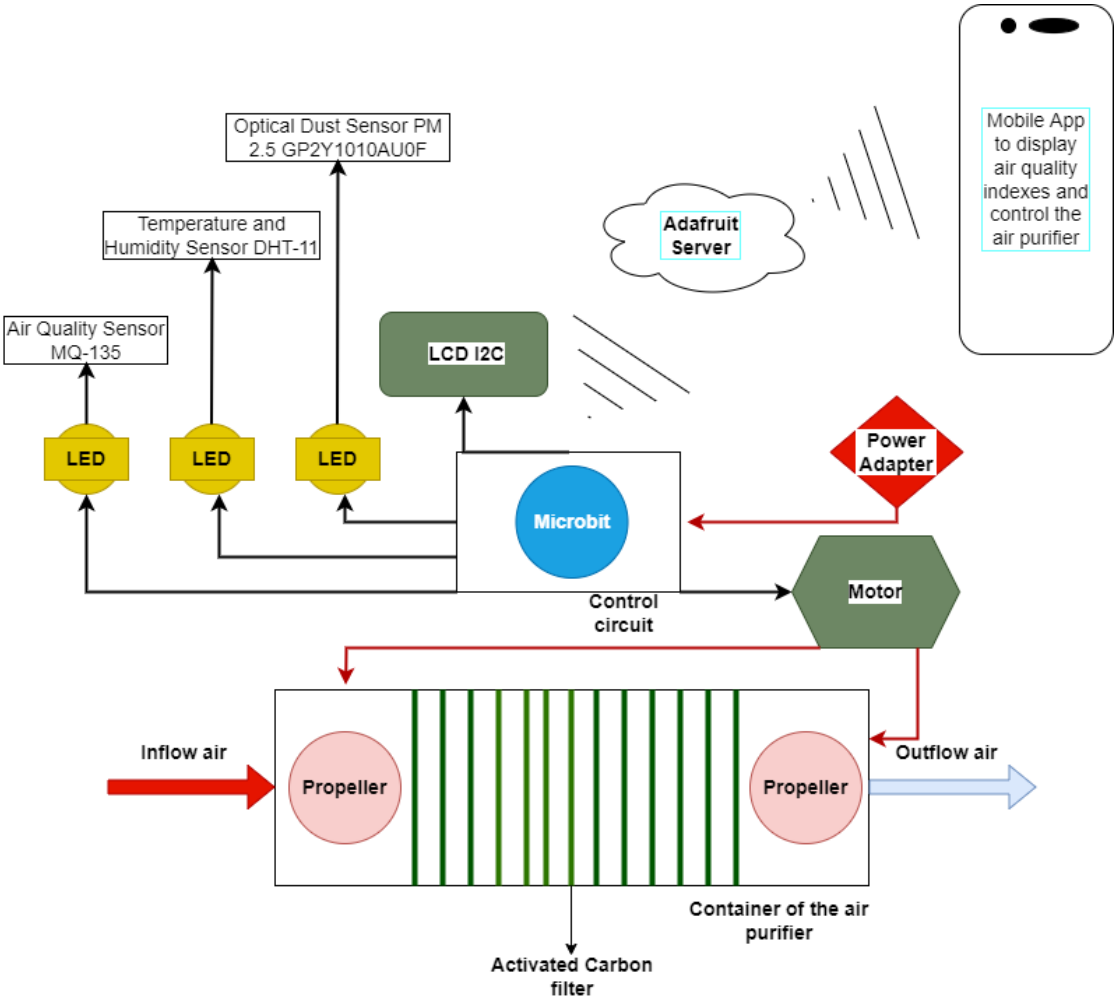


Figure 1: Illustration of the model.

2 Requirements

2.1 Functional Requirements

2.1.1 Iots aspect

- Automatically filters the indoor environment based on the air quality around.
- Offer the machine different modes of control, such as manual mode, automated mode, or regularly by schedule.
- Send out a warning message if the air quality safety level is exceeded.
- Provide the consumer with a real-time air quality information via an LCD display.

2.1.2 Mobile application aspect

- Allow manual control from user of certain functions, such as turning the purifier on or off and setting the machine's automatic operating hours.
- Provide notifications via a mobile app with a real-time air quality report.
- Analyze and report the air conditioning.

2.2 Non-functional Requirements

2.2.1 Iots aspect

- Operate continuously with little planned break time.
- Data is handled using an Adafruit server.
- The maximum actuator delay time should be ten seconds.
- Storing historical system logs for at least 120 reading times in order to use them for diagnostics.
- Easy user interaction using the device's buttons and LCD display.

2.2.2 Mobile application aspect

- Friendly interface for users of all generations.
- Viable on iOS and Android.
- A basic interface interaction should require less than 500 milliseconds and should take less than 5 seconds to connect to the server.

3 Devices

3.1 Microbit

The microbit serves as the hub for all external devices such as sensors, the LCD..., as well as the controller for all of them.

It receives data from the sensors and perform preliminary processing. Pre-processed data then sent to the server for logging and broadcasting to all smart phones currently connected to it.

3.2 Air Quality Sensor MQ-135

Application: Detect gases like Ammonia (NH_3), Sulfur (S), Benzene (C_6H_6), CO_2 and other harmful gases and smoke.

Input The presence of the harmful gases in the air.

Output The MQ-135 supports two types of output. If the gas is detected the indicator LED will turn on and the digital pin will go from logic high to logic low (0V). The analog output pin of the sensor can be used to measure the ppm (mg/L) concentration of the required gas. To do this we need to use the micro-controller.

3.3 Temperature and Humidity Sensor DHT-11

Application: Measure temperature and humidity.

Input: Temperature and humidity of the atmosphere at the moment.

Output: Temperature's value in degree Celsius with accuracy of $\pm 2^oC$ maximum and $\pm 1\%$ for the humidity, which will be reported to the controller.

3.4 Optical Dust Sensor PM 2.5 GP2Y1010AU0F

Application: Measure PM 2.5 dust particles concentration.

Input: The presence of PM 2.5 dust of the atmosphere at the moment.

Output: An analog voltage proportional to the measured dust density, with a sensitivity of $0.5\text{V}/0.1\mu\text{g}/m^3$.

3.5 Power Adapter

Application: A stable continuous power supply for this continuously-operated system.

Input: AC 100V-240V 50-60Hz.

Output: DC 5V.

3.6 Mini Motor and propeller

The main purpose is to inhale contaminated air then let it goes through the activated carbon filter to eliminate the harmful gases and tiny dust particles. At the end, the clean air will be blown out of the machine give us a fresh atmosphere.

3.7 LCD I2C

Application: Display the air quality indexes in characters on two lines, with 16 characters per a line.

Input: DC 3.5/5V

Output: Measured data from the sensors.

3.8 Extension circuit board

This is the backbone for the controller and all external devices.

4 Detailed use-case

4.1 Use-case 1: Dual-mode functionality

Use-case ID	1
Use-case Name	Dual-mode functionality
Stakeholders	User, server and the device
Description	Both completely automatic and fully manual purifying modes are available on the system. When operating in automatic mode, the device will automatically turn on the fan and begin filtering the nearby air whenever the parameters go beyond the set limit. In manual mode, the purifier can only be operated by the user switching physical buttons or clicking buttons on the mobile application. The user can use the LCD screen and mobile application in both modes to view air quality parameters, notifications, and warnings.
Pre-condition	At least two fans, a gas sensor, a temperature sensor, an optical dust sensor, and three activated carbon filter sheets are included in the system, which is operating normally.
Normal flow	To allow the system to operate automatically, the user must do the desired action. + **Step 1:** The user launches the smartphone app. + **Step 2:** The user chooses "Purification" from the main screen's list of functions. + **Step 3:** In the Mode section, the user chooses "Automatic". + **Step 4:** Every measurement that enters the system is evaluated, and if necessary, action is taken. In other words, it will begin filtering if the air quality in the area is above a certain level until the parameters return to normal.
Exception	If the system notices unusual changes (or a lack of changes) in the air quality while filtering at step 4. + **Step 4.1:** The system notices either no change or a slower change than expected in the air quality while filtering. + **Step 4.2:** System notifies users via push notifications from the mobile application, warning that there are problems with the fans or the filter sheets. + **Step 4.3:** The user needs to inspect the devices.

Alternative flow	If the user opts for totally manual purification at step 3: The system monitors air quality via LCD screen as regularly as possible and may issue alerts via push notifications. The user interface of the application may vary as development goes on.
Post-condition	Every time the sensors check the temperature, humidity, dust concentration, and gas density, air quality data is collected. Send all information to the server.

4.2 Use-case 2: Air quality information display

Use-case ID	2
Use-case Name	Air quality information display
Stakeholders	User
Description	The application allows user to view a graph illuminating previously stored recordings as well as a quick overview of the parameters that have been recorded at the present.
Pre-condition	The system is still continuously collecting data, and the mobile application is connected to the online database.
Normal flow	+ **Step 1:**The user launches the mobile program. +**Step 2:**From the list of features on the main screen, the user chooses "Overview." + **Step 3:**The system present the user with the most recent air quality status (based on most recent measurements). A line graph showing the temperature, humidity, dust concentration, and gas density over time is also included. + **Step 4:**By tapping the "Time period" button, the user can choose the chosen time frame. The system offers a thorough calendar so that users may select the reporting period's beginning and finish times. + **Step 5:**The user chooses the window of time and taps "Confirm." + **Step 6:**The correct graph display is restored when the system reloads the report page. + **Step 7:**To return to the home page, the user taps the return button.
Exception	The user-specified time period at step 4 is not present in the database system. The system displays an error box and suggests that the user select a suitable time frame.

Alternative flow	If the user selects "Statistics to report" in step 3's choice bar. + **Step 3.1:** The user can select from a list of parameters that the system will capture (by default, all options are ticked). + **Step 3.2:** The user then selects "Confirm" after checking or unchecking the appropriate items. + **Step 3.3:** The configuration is saved, and the system only reports the requested items the following time.
Post-condition	None

4.3 Use-case 3: Warning messing in case of exceeding safety level

Use-case ID	3
Use-case Name	Warning messing in case of exceeding safety level
Stakeholders	User, server, and devices
Description	The system continuously monitors the air parameters. When the data exceeds the safety level, it quickly sends warning messages through an LCD screen and smartphone notifications in both operating modes.
Pre-condition	The mobile application must have an authorized permission to push notifications and run in the background. The server that stores and transfers data is still functioning normally.
Normal flow	The application sends screen notifications to the mobile device. A quick glance of the air quality status is provided for the user. The user tap on the notification to open the mobile application.
Exception	None
Alternative flow	The warning messages is also display on the LCD screen of the purifier. The user can based on the air quality status to manually turn on the purifier's fans.
Post-condition	

4.4 Use-case diagram for the whole system

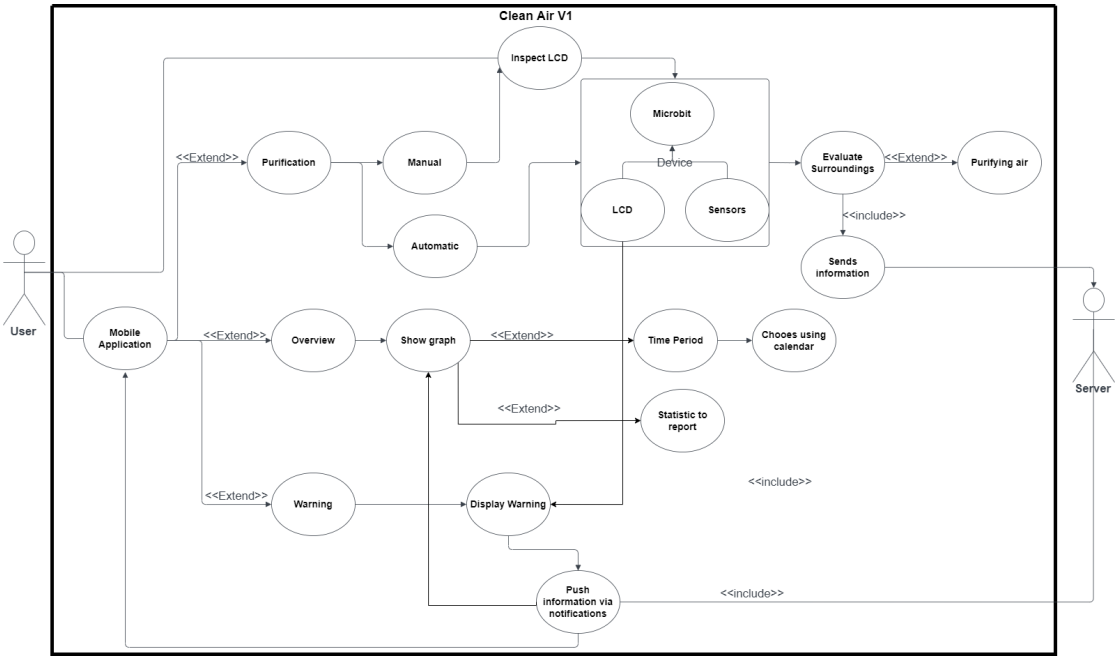


Figure 2: Use-case diagram for Clean Air V1 System.

5 General Design

5.1 Authentication

The username and active key provided by the Ada Fruit database are used to connect the app to the Ada Fruit server instead of attempting to use accounts in this app to obtain access. We just need to enter a new Ada Fruit username and active key to swap between users.

5.2 Connection

The HTTP client is used to retrieve data for the device to display as well as to send signals. The data will be kept in json format for faster retrieval. Additionally, using this approach will reduce the number of packages we need to add to the app and simplify maintenance.

The HTTP protocol also has the advantage of allowing us to use written APIs to gain access by calling the address without using a lot of methods. These APIs efficiently enable the device's code processing function, which in turn optimizes and makes use of the system to ensure smooth operation.

5.3 Business Logic

According to the use-case specification, we have decided to split the business logic into three subsystems: Display, Statistics and History, Control.

- If the concentration of dangerous gases or PM 2.5 dust exceeds the safe limits, the Display subsystem will continuously fetch data from the Ada Fruit server to present in the app and show push notifications.
- A user's activities with the system, such as turning on/off, modifying the propeller speed, etc., are also recorded in the Statistics and History section.
- The Control offers the user a button to switch the system on or off and a slider to manually adjust the speed.

The same HTTP TCP Protocol and database are used by all subsystems. The HTTP TCP protocol exposes straightforward per-device subscription data streams and publish APIs to ensure concurrent connections to the Ada Fruit server while abstracting this from the rest of the logic.

6 Technical Diagrams

6.1 Deployment Diagram

This section shows a package diagram of our system. Here is the demonstrated deployment, which consists of the **System** and **Server/Internet** modules.

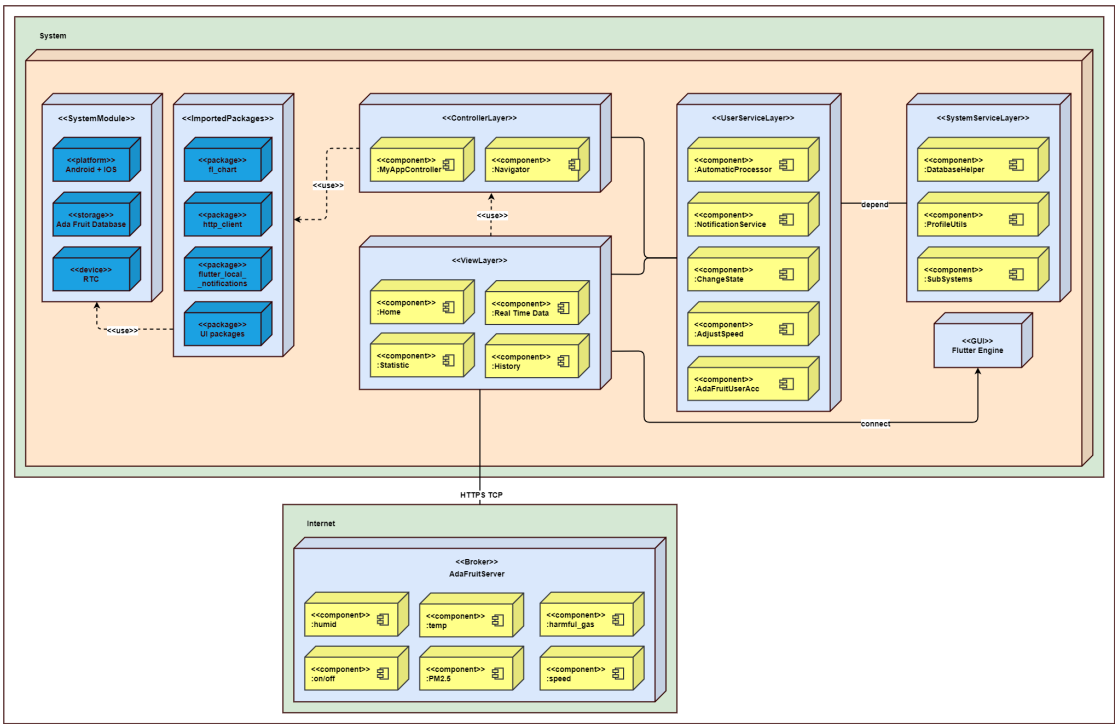


Figure 3: Deployment view of our system

We install all of our packages inside of our mobile device, which is expected to be a Flutter mobile app project, in the system module. The project primarily makes use of currently accessible System Modules, including the Android and iOS Operating Systems, the Ada Fruit Database, the Real-Time Clock, etc. The Flutter framework and its associated Imported Packages depend on these in order to function. We have used a variety of Flutter packages, but mostly useful ones like fl_chart for graphing data, http_client for submitting requests, and flutter_local_notifications for mobile app reminders and alerts. To further enhance the user experience, there are numerous distinct UI libraries.

The primary App controller and the Screen Navigator are now part of the ControllerLayer, which also houses the central logic controller for the GRASS app. This layer is dependent on the following two implementation layers:

– UserServiceLayer, which includes back-end utilities such a processor, notification, state change, speed adjustment, and Ada Fruit User Account. The Database Helper, the Profile Utilities, and the environment-specific Subsystems are a few examples of the services offered by a deeper SystemServiceLayer that manages transactions in the background and are dependent on this layer.

– The front-end UI/UX states of the app widgets are provided by the ViewLayer. Additionally, the screen of the device, which creates the real User Interface and Experience during deployment, does in fact support the ViewLayer. The home page, the real-time data page, the statistics page, and the history page are the primary displays, as described in the use cases. The app's fundamental presentation is made up of auxiliary pages and supporting transition screens.

The online servers we plan to use, which is the AdaFruit Server, may be communicated with and controlled by these system modules. It will establish a connection to the required feeds on the server, which, as seen in our device listing, will include 6 main input/output streams:

– humid
– temp
– harmful_gas
– PM2.5
– on/off
– speed

6.2 Implementation Diagrams

6.2.1 Activity Diagram: Subsystem General Work-Flow

As stated in the use-case section, our system is made up from three main subsystem: display, control, statistics and history. Before delving into the specific workflow and behavior of the system, we would like to provide a brief summary activity diagram for each of these subsystems in order to provide a better understanding of the system.

The mobile application is designed to run in background most of the time without the UI. There are two ways to exit the app: one by tapping Back on the home screen and the other by tapping Return anywhere. Tapping Return simply exits the UI and allows the app to continue running in the background. Tapping the Back button will lead to end monitoring and log out of the current session.

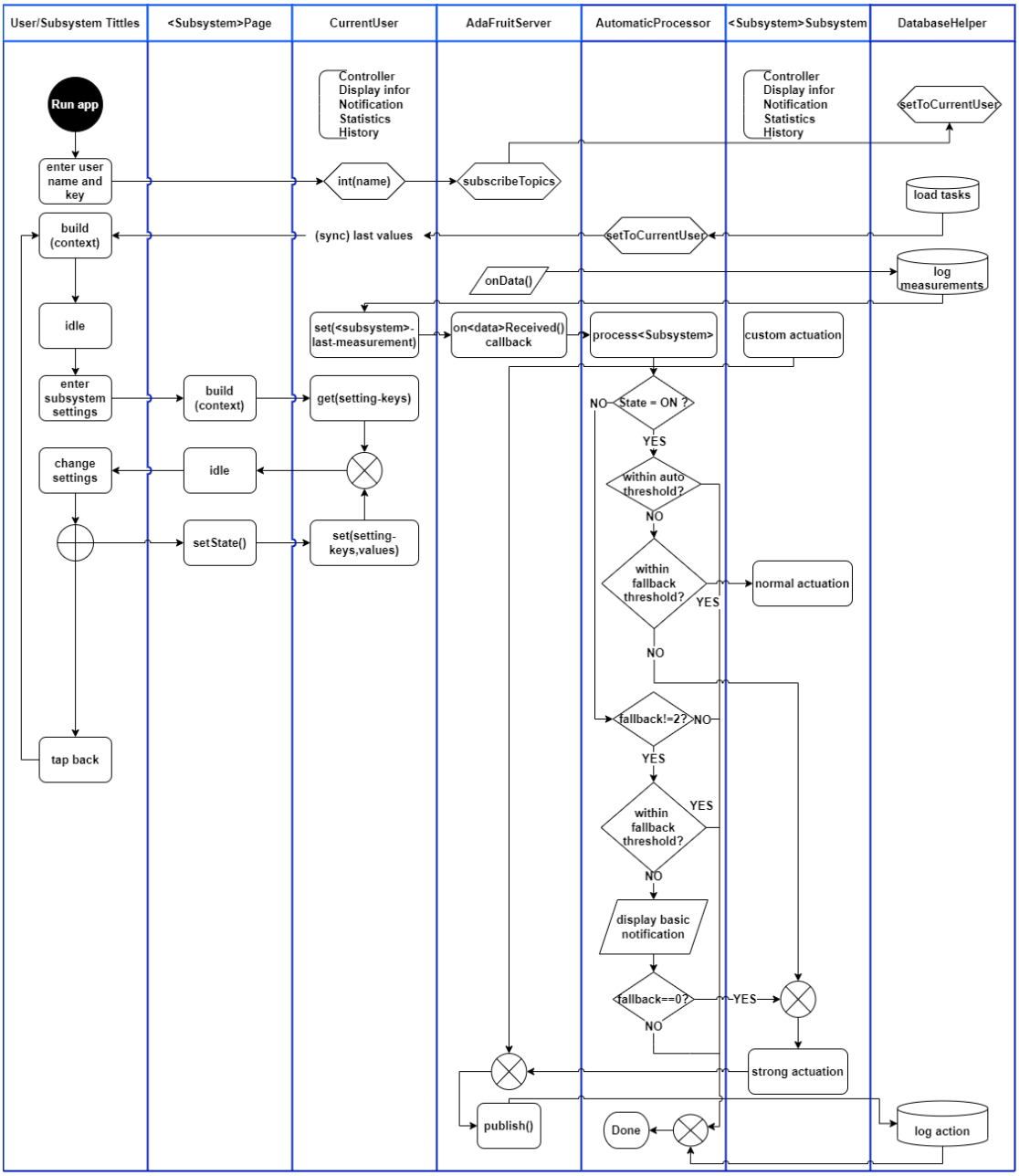


Figure 4: The logic flow of a typical subsystem

Following this section, we will discuss further the activity flow of three main subsystem and also more specifically, we will make some extension by adding some new features to bring the whole picture.

6.2.2 Activity Diagram: Display information and notification

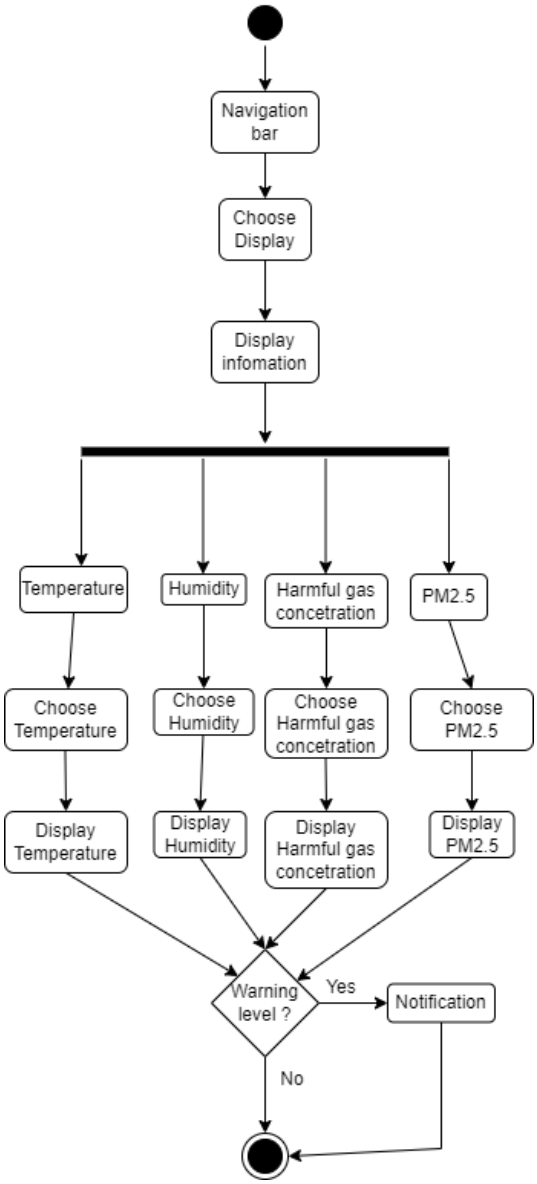


Figure 5: Diagram for display and notification

The user will begin at the navigation bar. There is an option to display data on temperature, humidity, harmful gas concentration, and PM2.5 in the navigation bar. The user can select the information they want to view when they choose each option. A notification will appear if the temperature, humidity, or dangerous gas concentration is at warning level.

6.2.3 Activity Diagram: Show statistics and history log

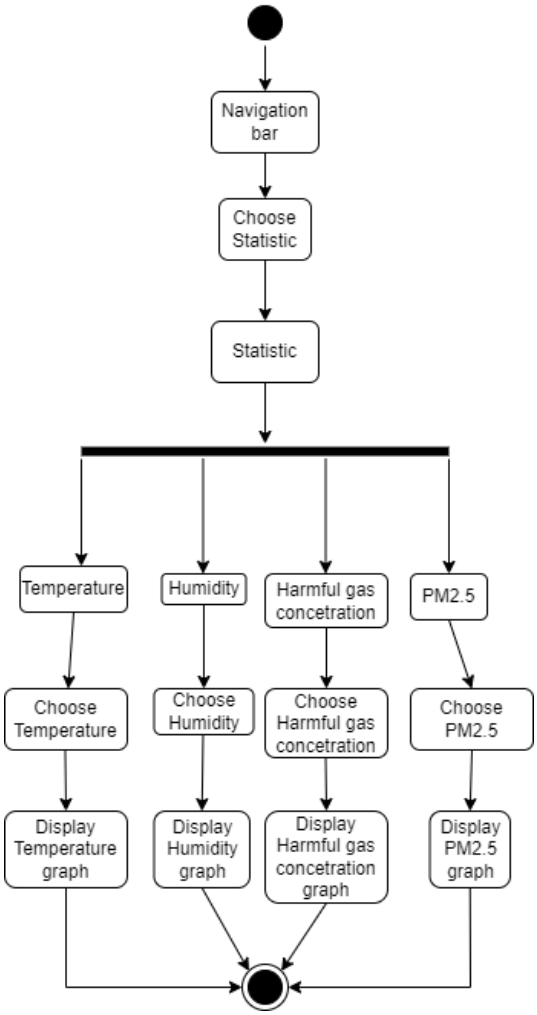


Figure 6: Diagram for statistical graph

The user can select the category of data they want to see if they choose to view the Statistics from the navigation bar. The application will then provide a graph tracking data changes for the selected category.

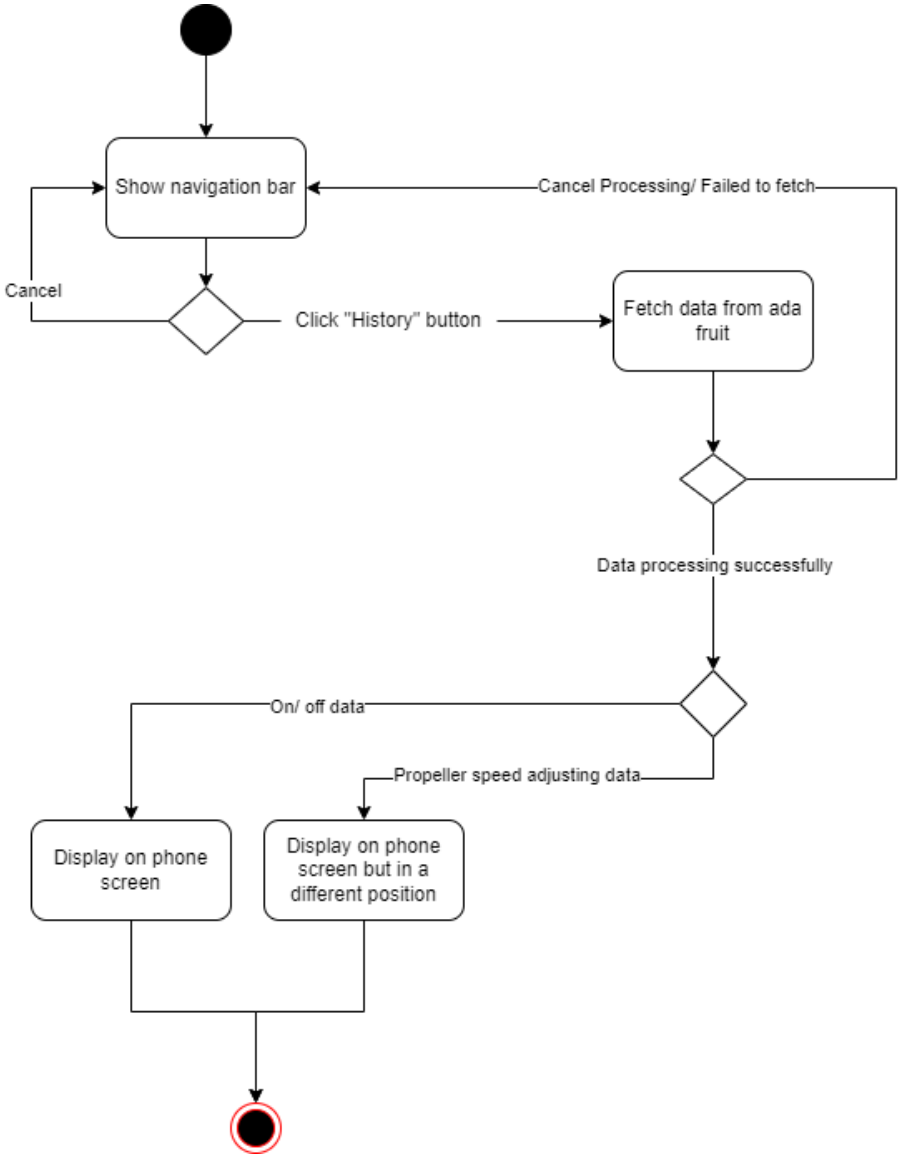


Figure 7: Activity diagram of History log

Once the user selects the "History" option in the navigation bar, the program will begin to process the data it has fetched from the Ada Fruit server, unless the user cancels the process or the app was unable to fetch the data because of a lack of internet. The data is then divided by the app into histories of On/Off and of propeller speeds data.

6.2.4 Activity Diagram: Control the machine

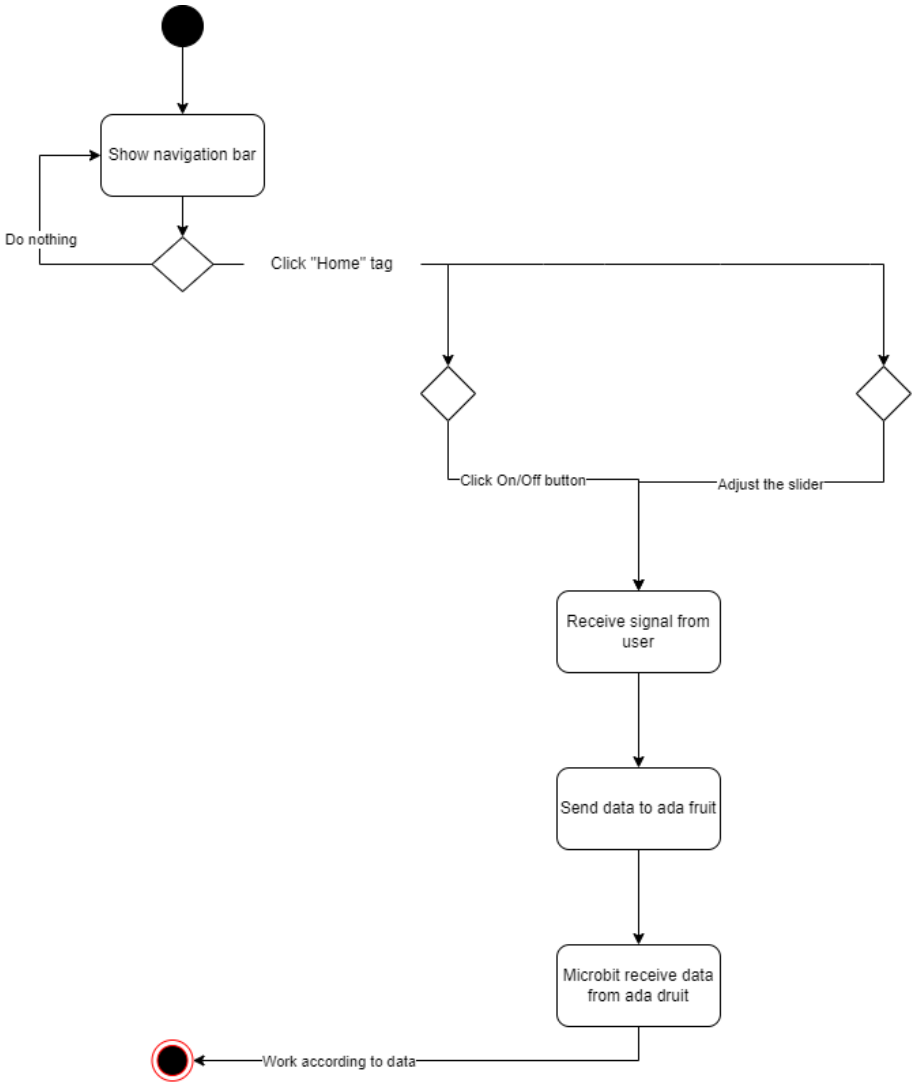


Figure 8: Activity diagram of Control from app

For ease of use, we have designed the system toggling to the auto-mode right after it has been turned on. And whenever the user want to adjust the speed of the propeller, the system now will be in manual mode.

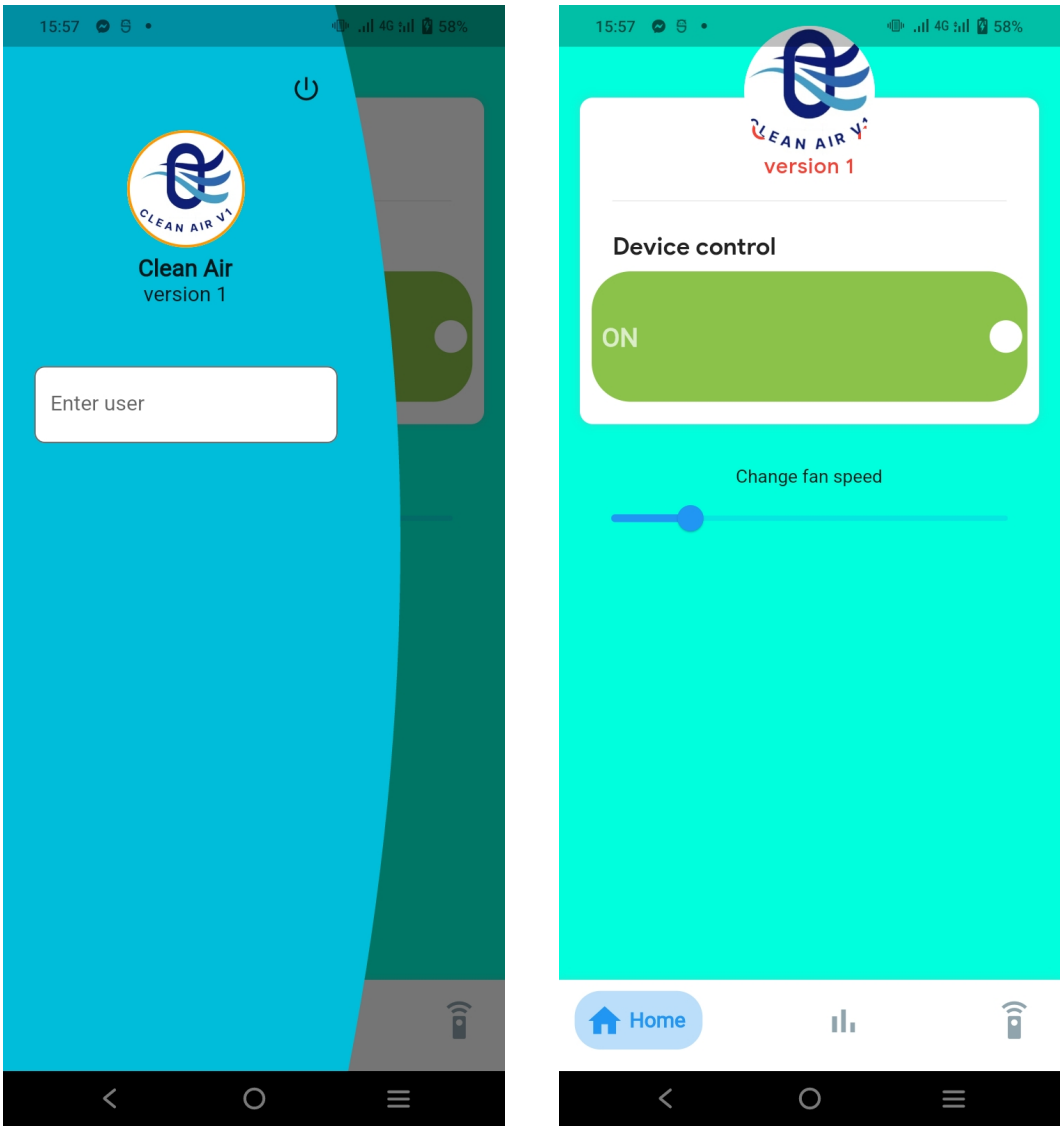
Starting with the inital home page, we have a button for on/off and a slider for modify the pitch of the propeller. The user can adjust using that button and slider to make the machine works in the user's favor. The app will get data accordingly and send it to Ada Fruit so that Ada Fruit will send those signals to the Microbit to run.

7 Finished Product

Here are a few sites of the application that demonstrate our system's primary features including Login, Home, Info, Chart and Statistic.

Initially, users need to enter their AdaFruit user name to login the app.

Then, they will be at the Home page, where there exist a function to turn on or turn off the machine, as well as changing the speed of the propeller.

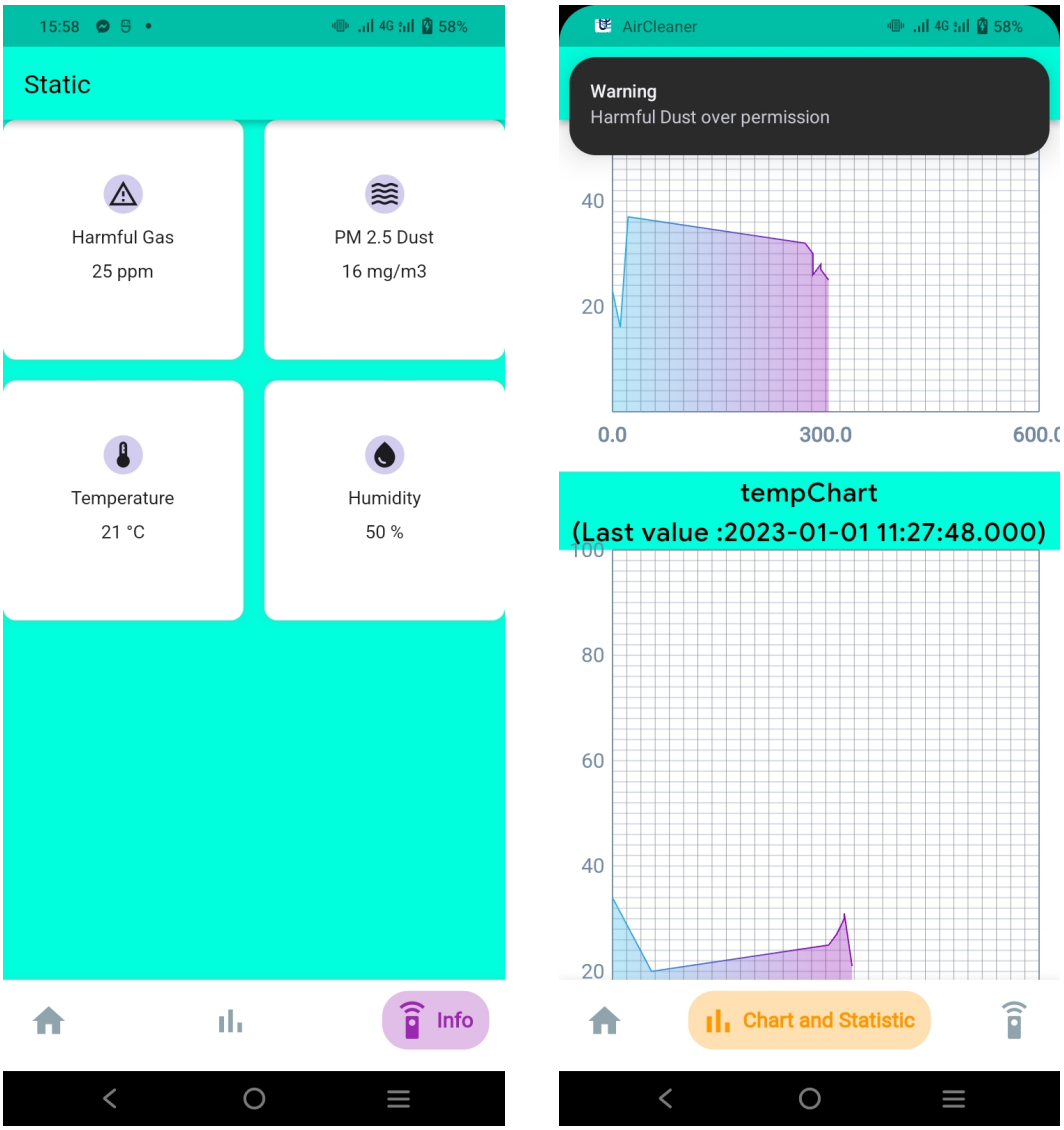


(a) Statistics Page (b) Device control Page

Figure 9: Mobile Application Pages

In the Info page, users can view the latest measurement of the four indexes which are the temperature, humidity, concentration of harmful gases and PM 2.5 dust.

Whenever, any index breach the safety boundary, a notification will pop out.

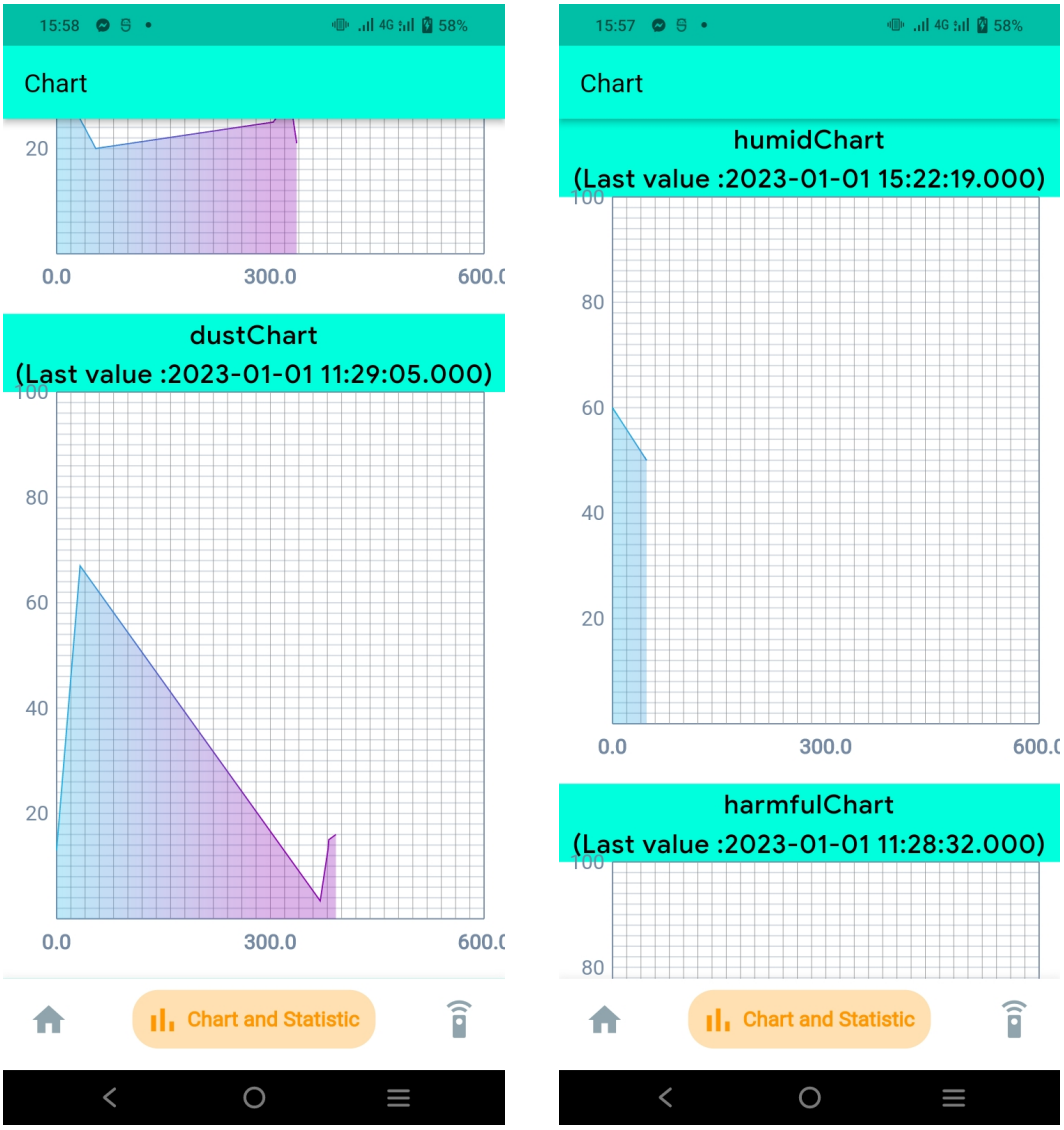


(a) Login Page

(b) Temperature Chart

Figure 10: Mobile Application Pages

The Chart and Statistic page will give us plots for the measurements in the last 10 minutes.



(a) Dust Density Chart (b) Humidity Chart

Figure 11: Mobile Application Pages

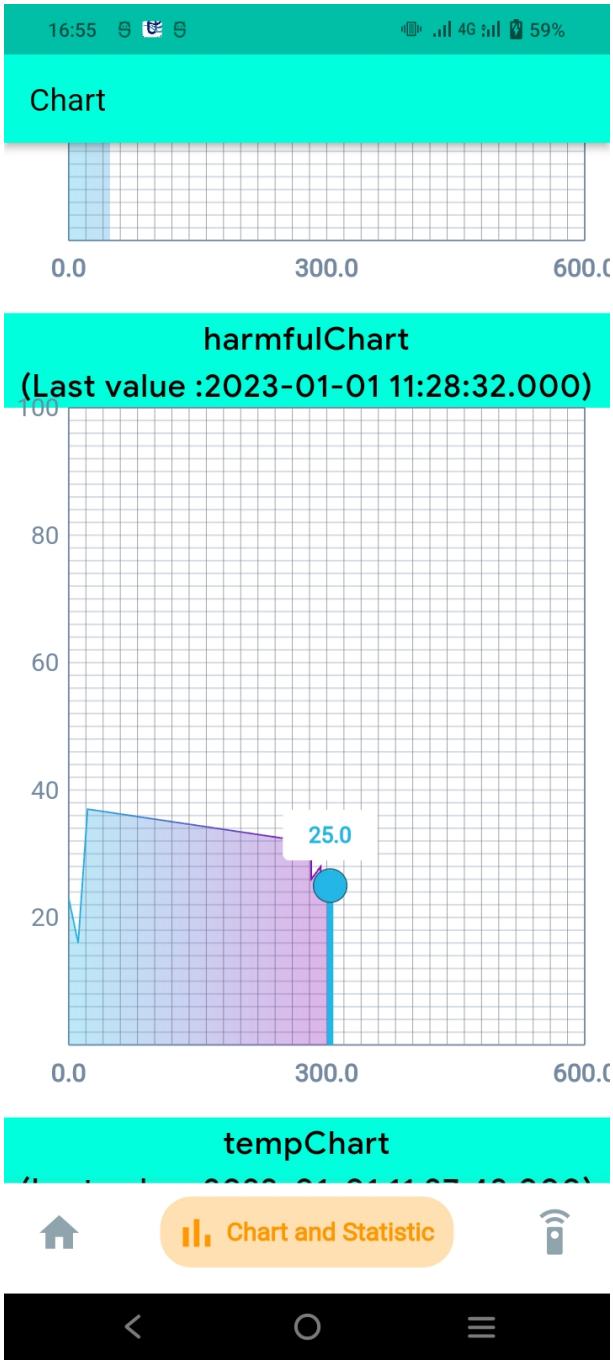


Figure 12: Harmful gases Density Chart