

First Aid Delivery Drone



Group Members

Ahsan Basharat	18I-0817
Shahmeer Hussain	19I-0822
Muhammad Bin Asif	19I-0896

Project Supervisor

Dr. Arshad Hassan

Department of Electrical Engineering

National University of Computer and Emerging Sciences, Islamabad
2023

Developer's Submission

"This report is being submitted to the Department of Electrical Engineering of the National University of Computer and Emerging Sciences in partial fulfillment of the requirements for the degree of BS in Electrical Engineering"

Developer's Declaration

"We take full responsibility of the project work conducted during the Final Year Project (FYP) titled "**First Aid Delivery Drone**". We solemnly declare that the project work presented in the FYP report is done solely by us with no significant help from any other person; however, small help wherever taken is duly acknowledged. We have also written the complete FYP report by ourselves. Moreover, we have not presented this FYP (or substantially similar project work) or any part of the thesis previously to any other degree awarding institution within Pakistan or abroad.

We understand that the management of Department of Electrical Engineering of National University of Computer and Emerging Sciences has a zero-tolerance policy towards plagiarism. Therefore, we as an author of the above-mentioned FYP report solemnly declare that no portion of our report has been plagiarized and any material used in the report from other sources is properly referenced. Moreover, the report does not contain any literal citing of more than 70 words (total) even by giving a reference unless we have obtained the written permission of the publisher to do so. Furthermore, the work presented in the report is our own work and we have positively cited the related work of the other projects by clearly differentiating our work from their relevant work.

We further understand that if we are found guilty of any form of plagiarism in our FYP report even after our graduation, the University reserves the right to withdraw our BS degree. Moreover, the University will also have the right to publish our names on its website that keeps a record of the students who committed plagiarism in their FYP reports."

Ahsan Basharat

BS(EE) 2018-0817

Shahmeer Hussain

BS(EE) 2019-0822

Muhammad Bin Asif

BS(EE) 2019-0896

Certified by Supervisor

Verified by Plagiarism Cell Officer

Dated: _____

First Aid Delivery Drone

Sustainable Development Goals

(Please tick the relevant SDG(s) linked with FYDP)

SDG No	Description of SDG	SDG No	Description of SDG
SDG 1	No Poverty	✓ SDG 9	Industry, Innovation, and Infrastructure
SDG 2	Zero Hunger	SDG 10	Reduced Inequalities
✓ SDG 3	Good Health and Well Being	SDG 11	Sustainable Cities and Communities
SDG 4	Quality Education	SDG 12	Responsible Consumption and Production
SDG 5	Gender Equality	SDG 13	Climate Change
SDG 6	Clean Water and Sanitation	SDG 14	Life Below Water
SDG 7	Affordable and Clean Energy	SDG 15	Life on Land
SDG 8	Decent Work and Economic Growth	SDG 16	Peace, Justice and Strong Institutions
		SDG 17	Partnerships for the Goals



Range of Complex Problem Solving			
	Attribute	Complex Problem	
1	Range of conflicting requirements	Involve wide-ranging or conflicting technical, engineering and other issues.	✓
2	Depth of analysis required	Have no obvious solution and require abstract thinking, originality in analysis to formulate suitable models.	✓
3	Depth of knowledge required	Requires research-based knowledge much of which is at, or informed by, the forefront of the professional discipline and which allows a fundamentals-based, first principles analytical approach.	✓
4	Familiarity of issues	Involve infrequently encountered issues	✓
5	Extent of applicable codes	Are outside problems encompassed by standards and codes of practice for professional engineering.	✓
6	Extent of stakeholder involvement and level of conflicting requirements	Involve diverse groups of stakeholders with widely varying needs.	
7	Consequences	Have significant consequences in a range of contexts.	✓
8	Interdependence	Are high level problems including many component parts or sub-problems	
Range of Complex Problem Activities			
	Attribute	Complex Activities	✓
1	Range of resources	Involve the use of diverse resources (and for this purpose, resources include people, money, equipment, materials, information and technologies).	✓
2	Level of interaction	Require resolution of significant problems arising from interactions between wide ranging and conflicting technical, engineering or other issues.	✓
3	Innovation	Involve creative use of engineering principles and research-based knowledge in novel ways.	✓
4	Consequences to society and the environment	Have significant consequences in a range of contexts, characterized by difficulty of prediction and mitigation.	✓
5	Familiarity	Can extend beyond previous experiences by applying principles-based approaches.	✓

Abstract

This project aims to design an automated First Aid Delivery Drone, which could become a critical life-saving equipment of the future. Absence of such devices in the healthcare industry today is a major cause for critical loss of life in cases of sudden cardiac arrest. Ambulance getting stuck in traffic has resulted in loss of countless precious lives. The project is specially designed for helping the human beings in the busiest cities in a faster way (Especially for cardiac patient but not limited to). People on road often face a medical issue or a user books an ambulance for the victim, drones can deliver personalized first aid kit to the user location .

The implementation of the project is explained in detail in Chapter-2.

It is a medical product promising aerial transportation that has an enormous potential for delivery of medical supplies to those in need. The use of drones will prove helpful for rapid delivery of medical equipment and medicines for patient self-administration.

Acknowledgements

We would like to express our sincere gratitude to our supervisor, Dr. Arshad Hassan for providing his invaluable guidance, comments and suggestions throughout the course of this project. We specially thank him for constantly motivating us to work hard and the importance of group work.

Without his guidance and commitment towards us, this project could not have been possible.

We would also like to thank Abdullah Afzal who guided us on numerous occasions on drone assembly.

Table of Contents

DEVELOPER'S SUBMISSION	II
DEVELOPER'S DECLARATION	III
ABSTRACT.....	VI
ACKNOWLEDGEMENTS	VI
TABLE OF CONTENTS.....	VI
LIST OF FIGURES.....	VIII
CHAPTER 1 INTRODUCTION	1
1.1 MOTIVATION.....	ERROR! BOOKMARK NOT DEFINED.
1.2 PROBLEM STATEMENT.....	ERROR! BOOKMARK NOT DEFINED.
1.3 LITERATURE REVIEW	2
1.4 DRONE REGULATION POLICY	3
1.5 REPORT OUTLINE	3
CHAPTER 2 SOLUTION DESIGN & IMPLEMENTATION.....	4
2.1 BLOCK DIAGRAM	4
2.2 FLOW CHART	8
2.3 SOFTWARE IMPLEMENTATION.....	10
2.4 HARDWARE IMPLEMENTATION	15
2.4.1 ASSEMBLING OF DRONE.....	15
2.4.2 DELIVERY MECHANISM.....	20
CHAPTER 3 RESULT AND RECOMMENDATIONS	22
3.1 SAVING LOCATION.....	22
3.2 DEPLOYING FIRST AID KIT.....	24
3.3 BUDGET.....	27
3.4 GANTT CHART.....	28
3.5 MILESTONES ACHIEVED.....	30
3.6 SUSTAINABLE DEVELOPMENT GOALS.....	30
3.7 CONCLUSIONS.....	31
3.8 RECOMMENDATIONS / FUTURE WORK	31

Table of Contents

APPENDIX-A: PROJECT CODES32

A-1 DASHBOARD APP NAME32

A-1.1 JAVA FILE (BACK END).....32

A-1.2 XML FILE (FRONT END)33

A-2 LOGIN..... 34

A-2.1 JAVA FILE (BACK END).....34

A-2.2 XML FILE (FRONT END)36

A-3 REGISTER..... 39

A-3.1 JAVA FILE (BACK END).....39

A-3.2 XML FILE (FRONT END)42

A-4 MAPS..... 45

A-4.1 JAVA FILE (BACK END).....45

A-4.2 XML FILE (FRONT END)47

A-5 REQUEST DRONE..... 48

A-5.1 JAVA FILE (BACK END).....48

A-5.2 XML FILE (FRONT END)51

A-6 PROFILE..... 53

A-6.1 JAVA FILE (BACK END).....53

A-6.2 XML FILE (FRONT END)56

BIBLIOGRAPHY.....59

List of Figures

FIGURE 2.1.1 BLOCK DIAGRAM OF THE PROJECT.	4
FIGURE 2.2.2 FLOW CHART OF PROJECT.	8
FIGURE 2.2.3 FLOW CHART OF MAIN USER APP.	9
FIGURE 2.3.1 ANDROID Studio AND JAVA SYMBOL.	10
FIGURE 2.3.2 DATABASE FOR USER ACCOUNTS.	11
FIGURE 2.3.3 DATABASE FOR USER COORDINATES.	11
FIGURE 2.3.4 APP NAME.....	12
FIGURE 2.3.5 LOGIN AND REGISTER NEW ACCOUNT PAGE.	12
FIGURE 2.3.6 MAIN PAGE OF APP.	13
FIGURE 2.3.7 GET LOCATION FOR DRONE REQUEST.	13
FIGURE 2.3.8 PROFLE VIEW.	14
FIGURE 2.3.9 MISSION PLANNER SOFTWARE.	15
FIGURE 2.4.1.1 PIXHAW 2.4.8 PX4 FLIGHT CONTROLLER.	16
FIGURE 2.4.1.2 ESC.	16
FIGURE 2.4.1.3 BLDC MOTOR.	17
FIGURE 2.4.1.4 PROPELLORS.	17
FIGURE 2.4.1.5 LI-PO BATTERY.	17
FIGURE 2.4.1.6 POWER MODULE.	18
FIGURE 2.4.1.7 RADIO TELEMETRY TRANSMITTER AND RECEIVER.	18
FIGURE 2.4.1.8 SERVO MOTOR.	18
FIGURE 2.4.1.9 COMPETE DRONE.	19
FIGURE 2.4.2.1 MECHANISM TO BE ATTACHED UPSIDE DOWN.	20
FIGURE 2.4.2.2 FRONT VIEW AND SIDE VIEW TO BE ATTACHED UPSIDE DOWN.	20
FIGURE 2.4.2.3 TOP VIEW TO BE ATTACHED UPSIDE DOWN.	20
FIGURE 2.4.2.4 FIRST AID KIT.....	21
FIGURE 3 BLOCK DIAGRAM OF THE WHOLE DELIVERY SYSTEM.	22
FIGURE 3.1.1 LOGIN PAGE.....	22
FIGURE 3.1.2 DATABASE FOR USER ACCOUNTS.	23
FIGURE 3.1.3 ENTERING THE LOCATION BY THE USER.	23
FIGURE 3.1.4 SAVING USER LOCATION.	24
FIGURE 3.2.1 CARRYING OF FIRST AID KIT.	25
FIGURE 3.2.2 DROPPING OF FIRST AID KIT.	25
FIGURE 3.2.3 MISSION PLANNER WAYPOINTS.	26
FIGURE 3.2.4 MISSION PLANNER WAYPOINTS.	26
FIGURE 3.4.1 FYP-I GANTT CHART.	28
FIGURE 3.4.2 FYP-II GANTT CHART.	29
FIGURE 3.6.1 SGD GOAL 1.	30
FIGURE 3.6.2 SGD GOAL 2.	30

Chapter 1 Introduction

Despite the technological revolution the healthcare system has not enhanced and various patients are still unable to receive timely medical assistance. It is quite troublesome to provide vaccines, blood or drugs to remote areas. Likewise, traffic congestion or geographical constraints also contribute to untimely delivery of medicine.

Earlier, drones were mostly used for surveillance purpose due to their high cost and complex technology [1]. However, in the last decade the use of drone has expanded as the technology has become more sophisticated and cheaper. Moreover, in the health sector it was not until recently that research begun. In Pakistan, the first ever health drone prototype was launched in 2019 however it was quite expensive and lacked advance technology. [3]

Therefore, we propose a “Medical Aid Delivery Drone” that will reach the unreached and make isolation relative, not absolute. Through our project we aim to provide a cheaper, faster medical assistance to those in dire need. [6] The drone will have a GPS based navigation system to reach exact location as desired and a camera to provide visuals to a doctor to provide help in terms of making reasonable diagnosis or to give proper advice in case of emergency. Furthermore, motors will be used to adjust speed and the movement of the drone can be easily accessed through a user-friendly mobile application.

1.1 Motivation

Advancement in technology is emerging day by day with the aim of bringing ease in life of people. The invention of drone is one the greatest inventions of modern age, with enhancement to it one can transport needy things one place to other just by handling it. Widespread use of the drone could help and boost emergency survivals rates as high as 80%. The system is still in prototype stage but there is a good chance it will be common in next five years.

1.2 Problem statement

Traffic congestion poses a significant challenge in large cities, primarily attributed to the increasing population. This issue often results in ambulances becoming trapped in traffic, impeding their ability to reach the designated emergency locations within the desired response time. As we know that time is of great essence in medical emergencies and the sooner the treatment begins the more effective it is however, the current modes of transportation of medical aid are quite unreliable. Therefore, to resolve the problem and to provide effective aid we propose a medical product aerial transportation which will deliver medical supplies timely hence leading to a more efficient healthcare system worldwide.

1.3 Literature review

Drones in healthcare:

This article discusses the need of urgent delivery of medical assistance to patients. Study suggests that due to the urgent need of blood and organs in hospitals far away from each other it is difficult to transport them in developing countries. [1]

Ambulance drone delivering medical toolkit:

This study proposes a system in which an ambulance drone delivers essential lifesaving supplies to the victim. It houses a compact defibrillator, medication and CPR aids. This paper clearly states the specifications of different elements of the drone and discuss their implementation in detail. [3]

Flight controlling device, Pixhawk PX4:

This study covers the application of Pixhawk. It is a high-performance autopilot-on-module used in multi rotor, helicopter and vehicle etcetera. This controller gets online support from Ardu-pilot platform which helps in monitoring the drone. [5]

Drone Design for first aid kit delivery:

In this paper the author discusses the need of a drone in health sector and develops a drone that is responsible of dropping 200 grams worth weight to accident location. Deep mathematical calculation on the basis of weight is provided in detail. The paper discusses in detail the technology used for each module as well as alternatives. Analysis of all long-term collected data is then performed to study and identify critical events and their relation to measured parameters. [6]

Methodology:

1. Hardware assembly of Drone.
2. Testing of Drone manually using FS-I6 Flight controller.
3. Aerodynamics (Balancing) of Drone.
4. Creation of App to receive patients live location.
5. Usage of Ardupilot software for Drone Automation.

1.4 Drone Regulation Policy in Islamabad

- Drones must be registered with the CAA, and obtain a drone operating license. They must also ensure that their drones are equipped with the necessary tracking and identification systems, and are operated in accordance with the CAA's regulations.
- In addition, the CAA has also set out guidelines for the safe operation of drones in the city. For example, all drones must be flown at least 30 meters away from people, buildings, and other objects. Furthermore, drones are not allowed to be flown above 200 meters in height, and must be operated within the visual line of sight of the operator.

1.5 Report Outline

This report is further divided into multiple chapters as listed below.

In chapter 2, proposed solution is discussed in detail. It includes details of block diagram, Flow chart of the process.

Chapter 3 discusses the results acquired by testing the device on multiple number of subjects in order to improve the device accuracy. It further discusses the conclusions drawn from the obtained results and the recommendations/future work that is proposed for further enhancements.

Chapter 2 Solution Design & Implementation

This chapter discusses the complete design and implementation of the proposed device. Section 2.1 discusses the block diagram. The details of flow chart are presented in section 2.2 whereas, Section 2.3 discusses in detail the software used in completing the application and carrying out the mission in mission planner. The hardware implementation is presented in Section 2.4.

2.1 Block Diagram

Figure 2.1.1 shows the complete block diagram of the Drone. The details of each block with related technical specifications are discussed below.

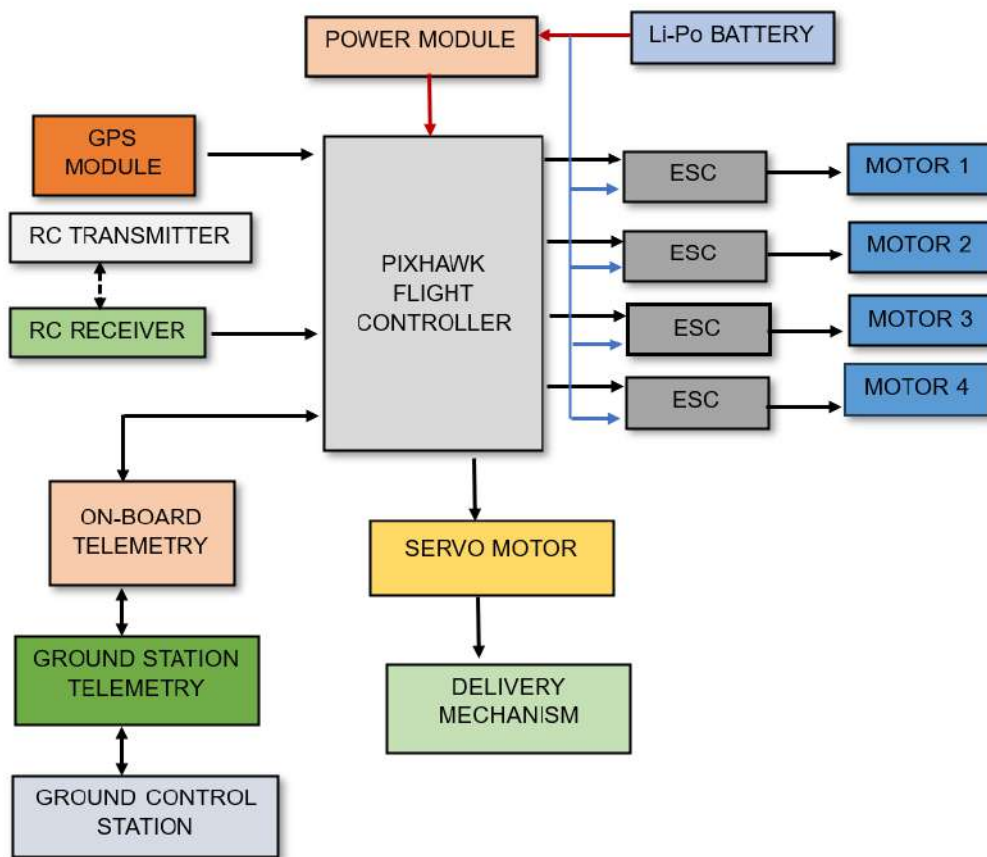


Figure 2.1.1 Block diagram of the project.

Pixhawk 2.4.8 PX4 Flight Controller

The Pixhawk2.4.8 PX4 is an open-source flight controller which gets its online support from mission planner software. It has numerous ports and interfaces as shown in Fig to

connect peripherals such as telemetry radios, RC receivers, GPS module, and other devices. Small planes are remotely piloted as they can be controlled from afar and is mostly used because of its low cost and availability. The waypoints and other action commands are added in the Mission Planner software using the Global Positioning System (GPS) in the Mission Planner software and this data is loaded into the Pixhawk controller.

Electronic Speed Controller (ESC)

An Electronic Speed Controller is a device that interprets the signals from the flight controller, Pixhawk and translates these signals into the electrical pulses to control the speed of the Brushless DC motors.

- **Data input:** It takes power and control signals as an input.
- **Data output:** Sends these signals to control the brushless motors.

Brushless DC (BLDC) Motors

These Brushless Motors are used to drive the propellers the main source of drainage of battery power.

- **Data input:** It takes input power and control signals like speed, direction, and other motor parameters as input from the ESC.
- **Data output:** It produces a force which is used to rotate the propellers to lift the drone.

Propellers

The propellers convert the motion/speed into lifting power. This is due to the special shape of the blades, as because of the uneven shape the air pressure is uneven on two sides while the propellers are in motion. Thus, results in a lifting power. It can be easily modeled as Newton's third law of motion.

- **Data input:** It takes speed as input.
- **Data output:** It produces lifting power/thrust.

Lithium Polymer (Li-Po) Battery

It is the main source of power for the whole quadcopter. It is connected to the power module and provides power to the power module. It is used because of high energy density and higher discharge rate.

- **Data input:** It takes no input.
- **Data output:** It provides power to the whole system.

Power Module

It takes input from the battery and provides power to the power distribution board, which is used to split the battery connections into four for each ESC. This power distribution board provides power to the flight controller by equally distributing power to the whole system.

Global Positioning System (GPS) Module

It is a radio navigation system based on satellites that provide geo location and timing information to a GPS receiver anywhere on or near the earth's surface. It is used to feed the coordinates of the waypoints and to track the location of drone.

- **Data input:** It takes satellite signals as input.
- **Data output:** It provides positioning, timing, speed, and navigation information.

Remote Controlled (RC) Radio Transmitter and Receiver

A Radio Transmitter is used to transmit the input navigational commands to the flight controller which are sensed by the radio receiver attached to the flight controller. The RC receiver receives the control signals from the transmitter, which contains the drone's desired movements such as throttle, pitch, roll, and yaw. These control signals after being decoded are sent to the different channels of Pixhawk, which controls the speed of the ESCs.

On-Board Telemetry

It gets the real-time data and communication signals from the ground control station (GCS) and sends it to the flight controller. This data includes the information related to the altitude, gyroscope, accelerometer, GPS, battery voltage, motor speed and more. It helps the drone to maintain its position, navigate through the waypoints already entered in the mission planner software and perform other tasks accurately.

Ground Telemetry

It sends the real-time data and communication from the ground control station (GCS) to the on-board telemetry which then sends these signals to the autopilot (pixhawk). It helps the drone to verify its position accuracy, set waypoints in the mission planner software and check whether other mission objectives are met.

Ground Control Station (GCS)

It is the main station where telemetry is attached and all the control signals are being sent to the drone through this base station.

Servo-Motor SG90

It is used for the activation and deactivation of the lock for delivery. When the signal is applied servo becomes active, lock opens and the First Aid Kit is deployed at the particular area.

- **Data input:** It takes input from the battery.
- **Data output:** It results in the movement for lock opening and closing.

Dropping Mechanism

An ultrasound transducer is mounted on the axis of a vibrator. Vibrations of mild amplitude and low frequency are transmitted from the vibrator to the tissue via a transducer, thereby including an elastic shear wave that propagate through the tissue.

Figure 2.2 shows the complete flow chart of the project. The details are discussed below.

2.2 Flow Chart

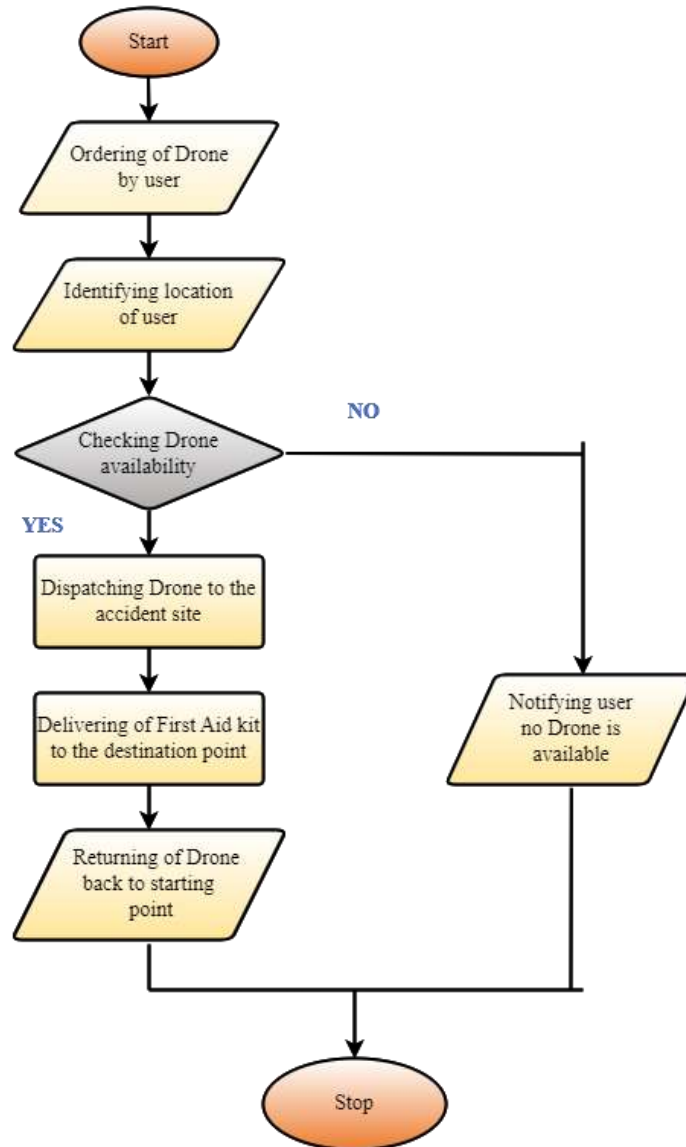


Figure 2.2.2 Flow chart of the project.

Main Process

At first the drone will be ordered by the user through the android application. Global Positioning System (GPS) coordinates of the user will be identified and availability of drone will be checked. If drone is available in the vicinity of the user i.e., if the base from where drone will departure is at a distance of 1 km from the user, then drone will be dispatched to the particular site of accident. After delivering First Aid kit to the user at the desired destination point drone will return back to the starting base point. If drone is not available then app will simply notify the user.

Main User Application

Figure 2.3 shows the complete flow chart of the android application. The details are discussed below.

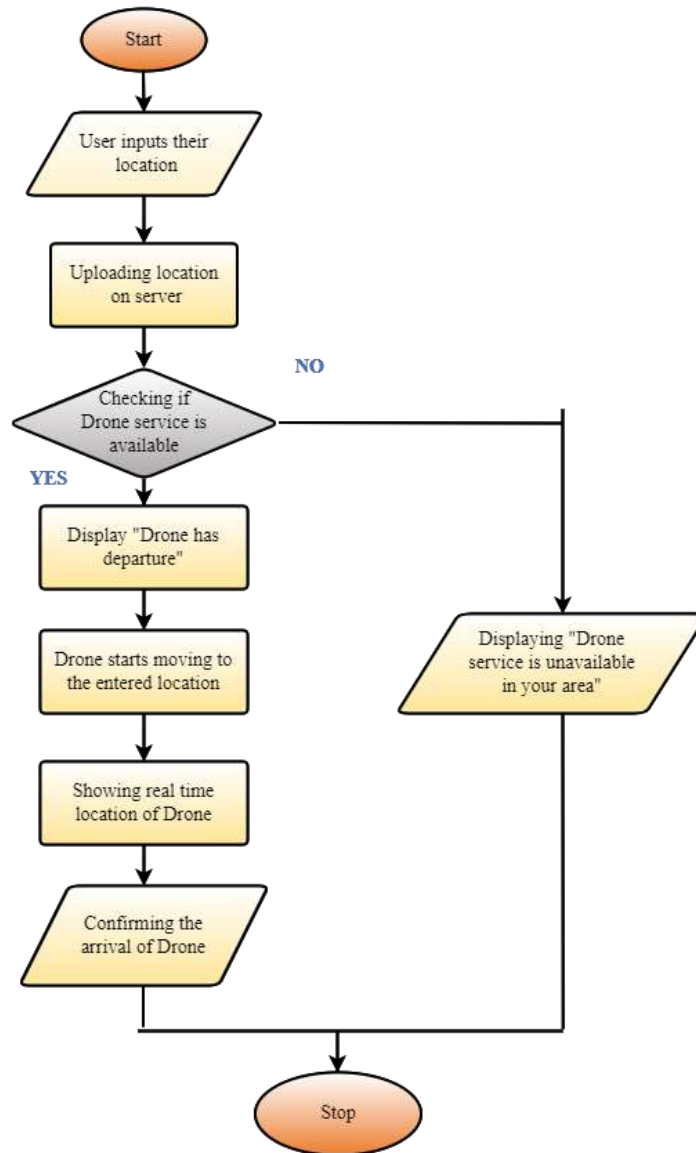


Figure 2.2.3 Flow chart of Main user app.

User Application Process

At first the drone will be ordered by the user through the main user application. The user will enter their Global Positioning System (GPS) location which will be uploaded on the server. After accessing this location from server, the availability of drone service will be checked. If drone

service is available in the area of the user, then app will display “Drone has departure”. Drone starts moving to the entered location of the user, the main user application will show real time location of the drone. The app will then confirm about the arrival of drone and will ask the user to collect the medical kit. If the drone service is not available near the user, then app will simply display “Drone service is currently unavailable in your area”.

2.3 Software Implementation

Android Studio Software

It is an integrated development environment (IDE) provided by google for android app development. It includes various tools which are helpful in making user interface (UI) of android apps and also make back-end development for that ap to function. It also includes an emulator for the testing of app on virtual android devices. All the coding required to make an app is done on this software. Java was used as the programming language to write the code.



Figure 2.3.1 android studio and java symbol.

Firestore Database

User Authentication

The data entered i.e., the user email and the password while registering to the android application is collected on Firestore database which is hosted on Firebase. Similarly, for those who already have an account this data is compared with the already stored data from database and then logs the user if the data entered is accurate. The database also shows the time when the user logs in to the app and continuously updates it every time the user logs in.

The screenshot shows the 'Authentication' console with a search bar and a table of users. The table has columns for Identifier, Providers, Created, Signed In, and User UID.

Identifier	Providers	Created	Signed In	User UID
ahmed@gmail.com	📧	May 22, 2023	May 22, 2023	ep87aON3VXbh82gBRma6wBARR...
fyp@gmail.com	📧	Mar 29, 2023	Mar 29, 2023	Qbf6HDgVNvdvJAwXh7ii3ea5yWh1
mustafa@gmail.com	📧	Mar 28, 2023	Mar 28, 2023	WFhHphkixpxi0lulqHgtBsMsaH3
ahsan@gmail.com	📧	Mar 27, 2023	May 15, 2023	vVXI8wW8SDgvDwrc58UcmlV8sz...
asif@gmail.com	📧	Mar 27, 2023	May 19, 2023	k7ubUiZScRcNksn50vv8VI3YL3I2
ahmad@gmail.com	📧	Mar 27, 2023		o3e7HITQB70bqzohWRBkeCMZ8...
usman@gmail.com	📧	Mar 27, 2023		YYZatATu0nWbm9RZM1dl6YdHQ...
shahmeer@gmail.com	📧	Mar 27, 2023	Jun 14, 2023	MWiiA7Yfr1PC36nTU90c2KJ86yk1
khuzaima@gmail.com	📧	Mar 27, 2023	Jun 14, 2023	DNd9iDusydaNAUTXLdt5bPIA8QN2
wardah@gmail.com	📧	Mar 27, 2023		eYd4vApqVINsWVzuBfb0VxxCom...

Figure 2.3.2 Database for user accounts.

User coordinates

To input the coordinates, the user selects his/her location from the map and saves that location. This location is stored in the real-time database and is continuously changing as the user location changes.

The screenshot shows the 'Realtime Database' console with a tree view of data. The data is organized under a 'Locations' node, with several child nodes containing latitude and longitude values.

```

Locations
├── -Nv1IebDrhYe8JzxMXzw
├── -Nv1Iekc0osE2LUJngSe
│   ├── latitude: 33.65692333333333
│   └── longitude: 73.01865500000001
└── -Nv1If-G6u280wm9Wiy
    
```

Figure 2.3.3 Database for user coordinates.

Dashboard on Application

To display the data collected on database the android application “DroneAid” has been made. Figure 2.3.4 shows the login and a register new account page. If the person already has an account, he/she can simply login to the app otherwise he/she has to register for a new account.



Figure 2.3.4 App name.

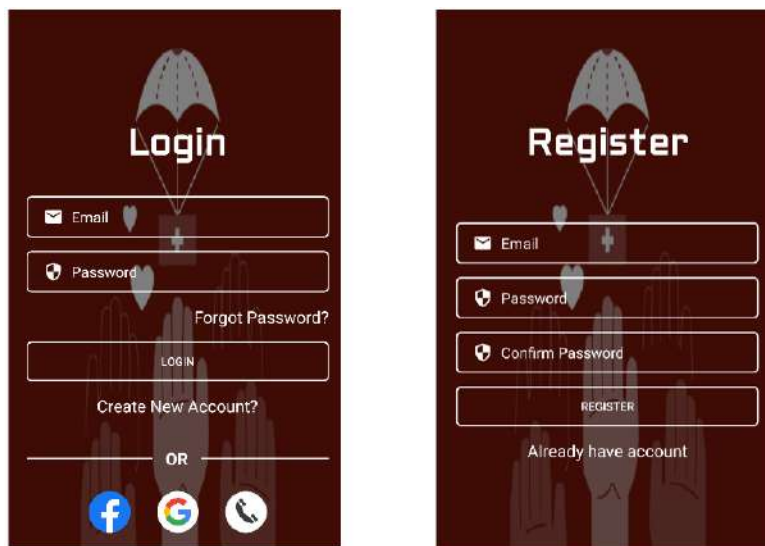


Figure 2.3.5 Login and Register new Account page.

After logging in to the app the main page opens which has options for booking the drone.

Chapter2: Solution Design and Implementation



Figure 2.3.6 main page of app.

After clicking the request drone option another page opens as shown in Figure 2.3.6 where the user can enter his/her coordinates to book for drone. The coordinates are entered and this location is saved in the database. By clicking the profile button, the user data can be seen and saved.

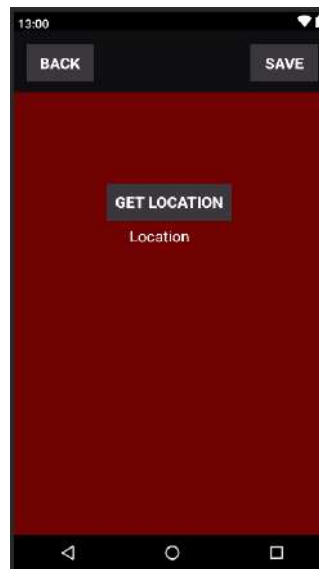


Figure 2.3.7 get location for drone request.

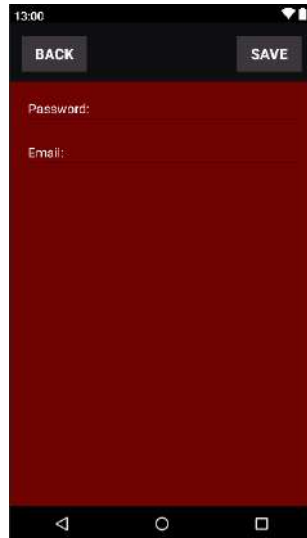


Figure 2.3.8 profile view

Mission Planner Software

It is the open-source ground control station (GCS) software which act as the main platform for planning, monitoring and controlling the automatic missions of the drone. They provide real time telemetry data from the drone showing the altitude, battery performance, ground speed, GPS and more. It is used to set missions and other important actions which the drone has to follow. The figure 2.3.9 shows the initial point marked as 'H'.

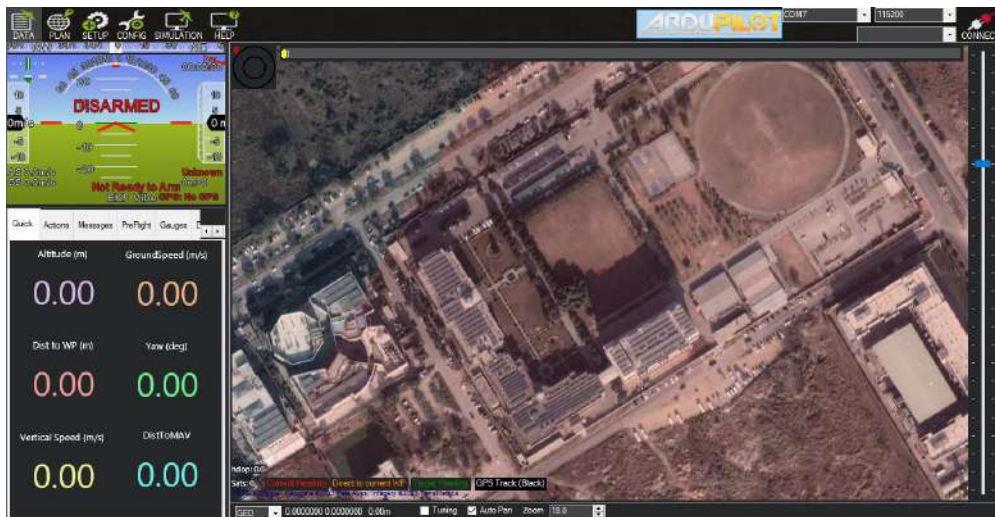




Figure 2.3.9 Mission planner software

2.4 Hardware Implementation

Hardware implementation was carried out in following two stages

1. Assembling of Drone.
2. Delivery Mechanism.

2.4.1 Assembling of Drone

This was the hardware part of our project in which we gathered almost all parts of drone and assembled the drone by first soldering the power module to the power distribution board (soldering positive (red) to positive terminal and negative (black) to negative terminal of board). Then we had soldered Electronic Speed Controllers (ESC) to the power distribution board by using same orientation i.e., red with positive and black with negative. These ESCs are connected to the motors which rotate the propellers. After all the soldering we ensured that no short circuit has occurred while soldering using multi meter. As short circuit can lead to explosion of the Lithium Polymer (LiPo) battery. After this we attached the drone arms and landing gear to the bottom plate with the help of screws. Then we attached the top plate of drone along with motors on the drone arms and propellers on top of motors, such that the spin direction is same for opposite motors/propellers. Then we connected the power module with the lithium polymer battery. The individual parts are connected as follows:

1. Pixhawk 2.4.8 PX4 flight controller:

The Figure 2.4.1.1 shows the ultrasonic sensor module. The following points describe the working principles of this sensor.



Figure 2.4.1.1 Pixhaw 2.4.8 PX4 flight controller.

The power, 4.8 to 5.7V to the controller is provided through power module at the power port. Telemetry port is used to get the ground control signals transmitted through ground telemetry. The CAN pin is used for GPS and a buzzer is connected at the buzzer pin which alerts in case of bad battery. I2c is a serial communication used for interconnecting integrated circuits and for communication with external sensos in this case the GPS sensor.

2. Electronic Speed Controllers:

Figure 2.4.1.2 shows the electronic speed controllers used to drive the brushless DC motors.



Figure 2.4.1.2 ESC.

3. BLDC motors:

The Brushless DC motor has the voltage rating of 1000 KV. It enhances the upward thrust force for propelling the drone.



Figure 2.4.1.3 BLDC Motor.

4. Propellers:

They are the blades used to create a difference in air pressure. The pressure is higher on the bottom side and lower on the top side and due to this the drone lifts in the air.



Figure 2.4.1.4 propellers.

5. Lithium Polymer (Li-Po) battery:

They are used because of higher density to weight ratio so that they can power the drone's onboard systems with fewer cells as compared to other rechargeable batteries.



Figure 2.4.1.5 Li-Po battery.

6. Power module:

It has maximum input voltage of 35V and sensing current of 90A. It provides the output of 5.3V and 3A to the whole power distribution board.



Figure 2.4.1.6 power module.

7. Telemetry Transmitter and Receiver:

They are used to send the navigational commands from the base station to the drone.

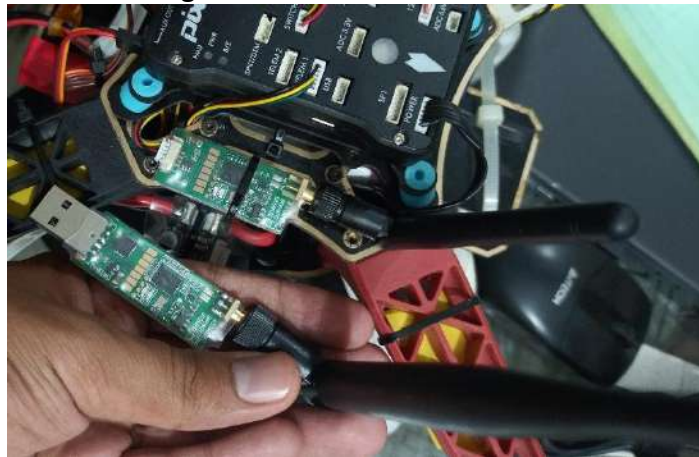


Figure 2.4.1.7 Radio telemetry transmitter and receiver.

8. Servo Motor:

It is operated on 4.8 -6.0 V and the operating speed in case of input 5V is approximately 0.115/60 degree (5.0V).



Figure 2.4.1.8 servo motor.

a. **Complete Drone:**

Figure 2.4.1.3 shows the complete assembled drone with First Aid Kit attached to it.



Figure 2.4.1.9 Compete Drone.

2.4.2 Delivery Mechanism

The First aid kit delivery mechanism is functioned with the help of servo motor. The mechanism to be fitted upside down and it is shown in the Figure 2.4.2.1 the other way around.

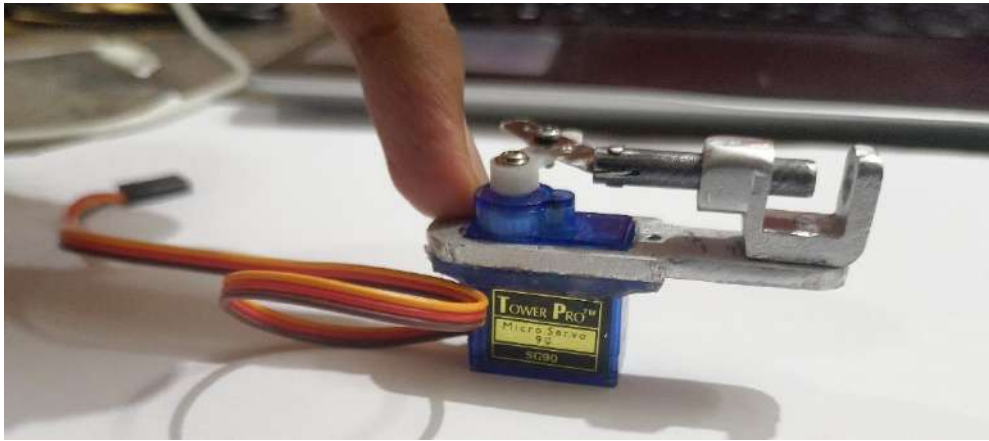


Figure 2.4.2.1 mechanism to be attached upside down.

The input 5V is provided to the servo-motor SG90. When the toggle command is given to the motor it opens the lock and the First aid kit drops.



Figure 2.4.2.2 Front view and Side view to be attached upside down.

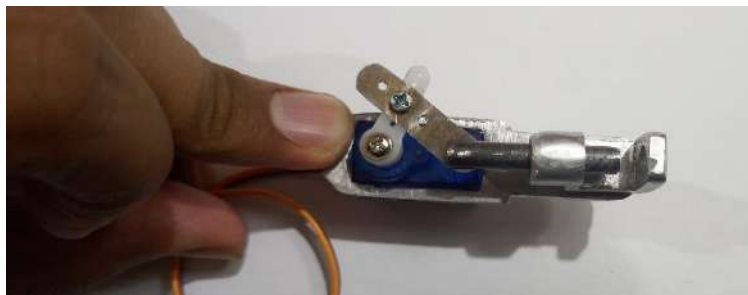


Figure 2.4.2.3 Top view to be attached upside down.

Chapter2: Solution Design and Implementation

Figure 2.4.2.4 shows the First Aid Kit which mainly comprises of 2 cotton swabs, 2 small bandages, 5 antiseptic swabs, 5 alcohol pads, 4 pain killers, 1 roll of hypo-allergenic strapping tape, 10 Saniplast, 5 safety pins, and a small scissor for cutting.



Figure 2.4.2.4 First Aid Kit.

Chapter 3 Result and Recommendations

A First Aid Delivery Demo was conducted on Campus Premises in which a medical bag carrying a weight of around 120g was attached. It took off from a starting point, delivered the medical bag to the desired location and then returned to the takeoff point.

Figure 3 shows the complete block diagram of the First Aid Kit delivery system. The details of each block are discussed below.

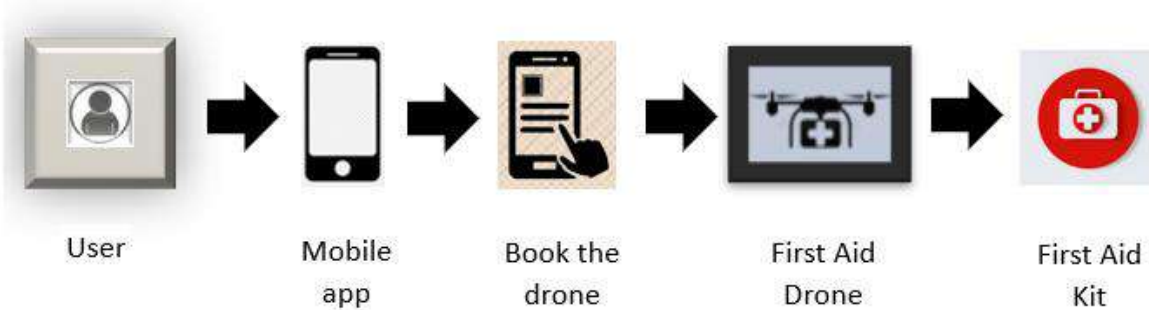


Figure 3 Block diagram of the whole delivery system.

3.1 Saving Location

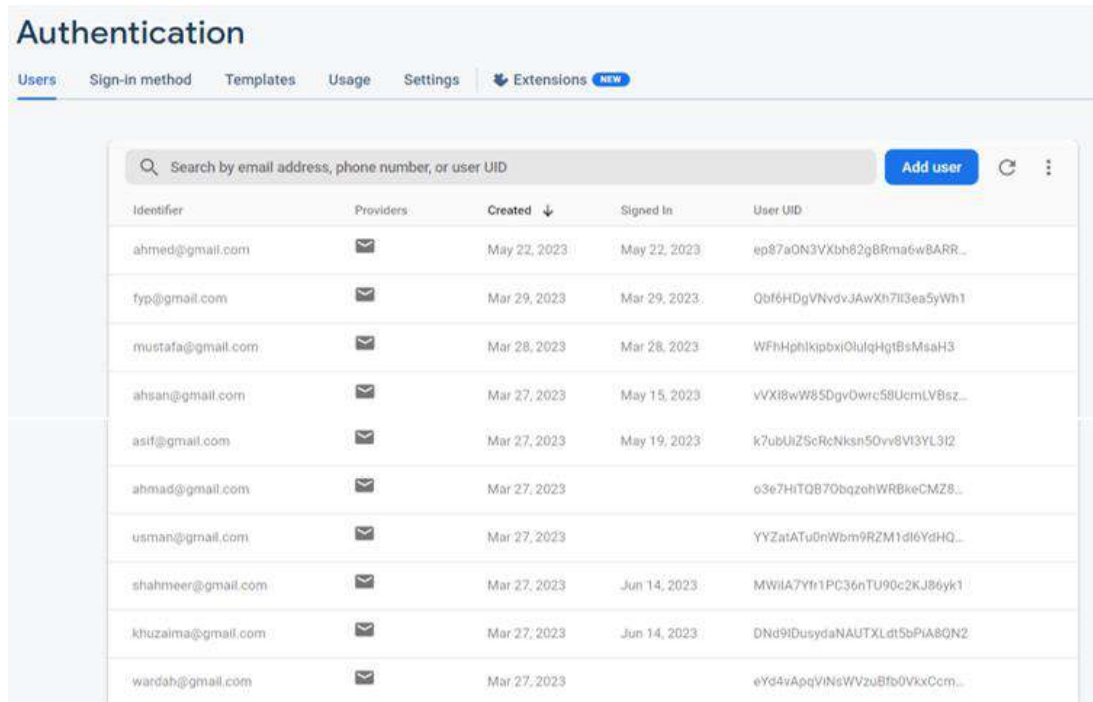
When the user opens the mobile app “DroneAid” he/she enters his/her login credentials as shown in Figure 3.1.1 using his/her email and books the drone through the proposed system’s mobile application. The user enters the desired coordinates from map and this location is saved in the database server as shown in Figure 3.1.3 and Figure 3.1.4 respectively. Then the drone delivers the First Aid Kit and flies back to initial point.



Figure 3.1.1 login page.

Chapter3: Result and Recommendations

Figure 3.1.2 below shows the user database every time a new user registers for an account. The data mainly the email and password are saved in the database and every time the user logs in this data is updated i.e., the Sign-in date changes to the present date of login as shown below.



Identifier	Providers	Created	Signed In	User UID
ahmed@gmail.com		May 22, 2023	May 22, 2023	ep87aON3VXbh82gBRma6wBARR...
fyp@gmail.com		Mar 29, 2023	Mar 29, 2023	Qbf6HDgVNvdvJAwXh7il3Sea5yWh1
mustafa@gmail.com		Mar 28, 2023	Mar 28, 2023	WFHhphkpbxiOlulqHgtBsMsaH3
ahsan@gmail.com		Mar 27, 2023	May 15, 2023	vVXl8wW85DgvOwrc58UcmLV8sz...
asif@gmail.com		Mar 27, 2023	May 19, 2023	k7ubUjZScRcNksn5Ovv8Vl3YL3l2
ahmad@gmail.com		Mar 27, 2023		e3e7HiTQB70bqzohWRBkeCMZ8...
usman@gmail.com		Mar 27, 2023		YYZatATu0nWbm9RZM1dl6YdHQ...
shahmeer@gmail.com		Mar 27, 2023	Jun 14, 2023	MWlIA7Yfr1PC36nTU90c2KJ86yk1
khuzaima@gmail.com		Mar 27, 2023	Jun 14, 2023	DNd9lDusydaNAUTXLdt5bPiA8QNZ
wardah@gmail.com		Mar 27, 2023		eYd4vApgVNsWVzuBfb0VkkxCcm...

Figure 3.1.2 Database for user accounts.

Figure 3.1.3 shows the entering of user location. The user enters the location by dragging the pointer from the app and saving that location in the map. After clicking the Get Location button on the app the location is shown on the app.

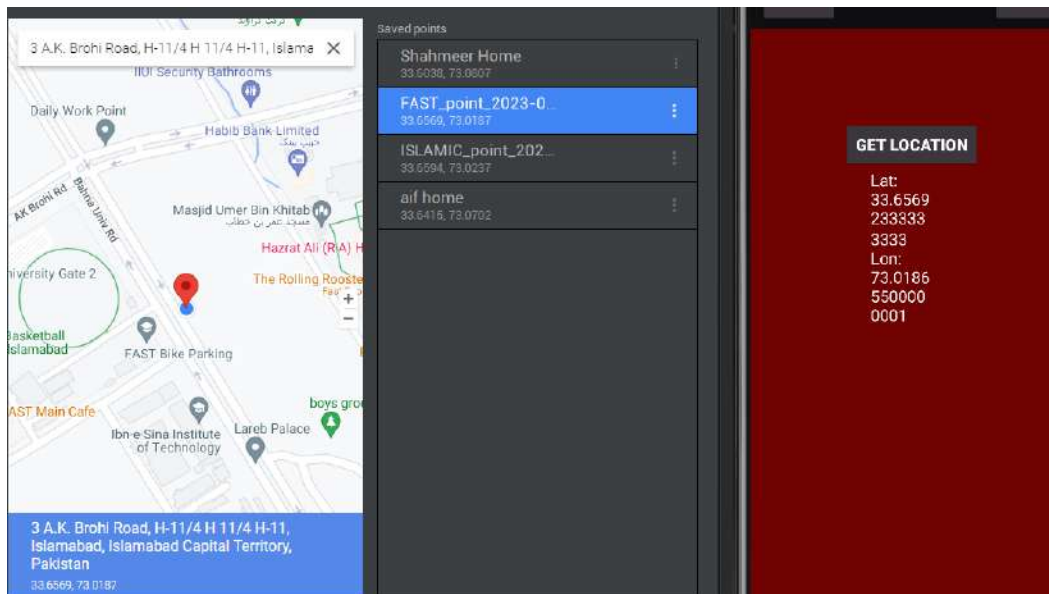


Figure 3.1.3 Entering the location by the user.

Chapter3: Result and Recommendations

Figure 3.1.4 shows the saving of user location. After the user enters his/her location this location is saved on the database server. This real time location changes as the user location changes. The drone accesses this location and then marks the journey to deliver the med-kit.

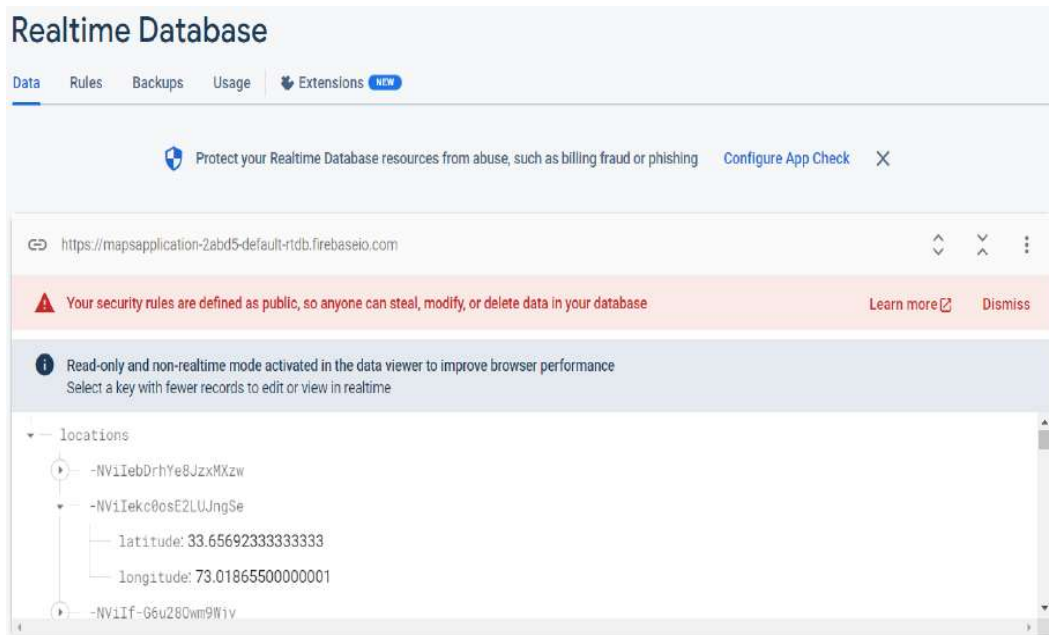


Figure 3.1.4 Saving user location.

3.2 Deploying First Aid Kit

Figure 3.2.1 shows the dropping of first aid kit. Firstly, the first aid kit is attached to the drone and after setting the commands for the whole flight the drone take-offs from initial position. After reaching the desired marked coordinates of the user, who booked the drone, the First Aid Kit is delivered as shown in Figure 3.2.2.

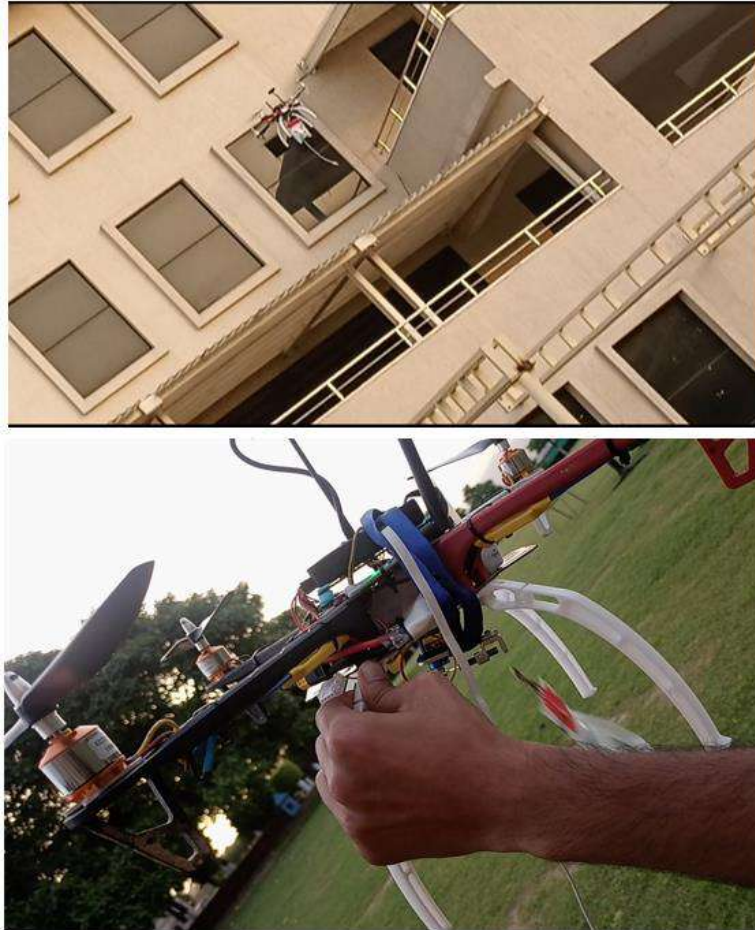


Figure 3.2.1 Carrying of First aid kit.

Figure 3.2.2 shows the dropping of First Aid Kit from a certain height of like 8m. The servo motor is toggled using the mission planner software and when the drone reaches the desired dropping location it drops the med-kit to the user and flows away to the base/take-off point.



Figure 3.2.2 Dropping of First aid kit.

Chapter3: Result and Recommendations

Figure 3.2.3 and Figure 3.2.4 includes entering of waypoints for the whole mission in Mission Planner software. The function and the height are identified at each point. After choosing the point, their latitudes and longitudes at that particular point are saved. These functions are provided in the command option which the drone follows in order to reach the destination point marked by the pilot.



Figure 3.2.3 Mission planner waypoints.

The commands for the drone to perform a task are shown below in Figure 3.2.4, when the drone reaches the particular user location. The servo motor is toggled using the DO_SET_SERVO command and the lock opens through which the First Aid Kit is deployed to the user. After delivering the medical kit the drone reaches the initial position using RETURN_TO_LAUNCH command.



Figure 3.2.4 Mission planner waypoints.

3.3 Budget

S.No.	Items	Price/Unit	Unit	Price / Rs
1.	S450 Frame	5,000	1	5,000
2.	880 kV BLDC Motor	4,500	4	18,000
3.	Telemetry (Receiver and Transmitter)	15,000	1	15,000
4.	Remote Controller (Transmitter)	17,000	1	10,000
5.	Remote Controller (Receiver)	2000	1	2000
6.	Propellers	900	4	3,600
7.	Electronic Speed Controller (ESC)	1,400	4	5,600
8.	3S1P Li-Po Battery	7,500	1	7,500
9.	Pixhawk 2.4.8 PX24 Flight Controller	30,000	1	30,000
10.	GPS Sensor	4,950	1	4,950
11.	Power Module	240	1	240
12.	Dropping Mechanism	800	1	800
	Total cost of Project			102,690≈103,000

3.4 Gantt Chart

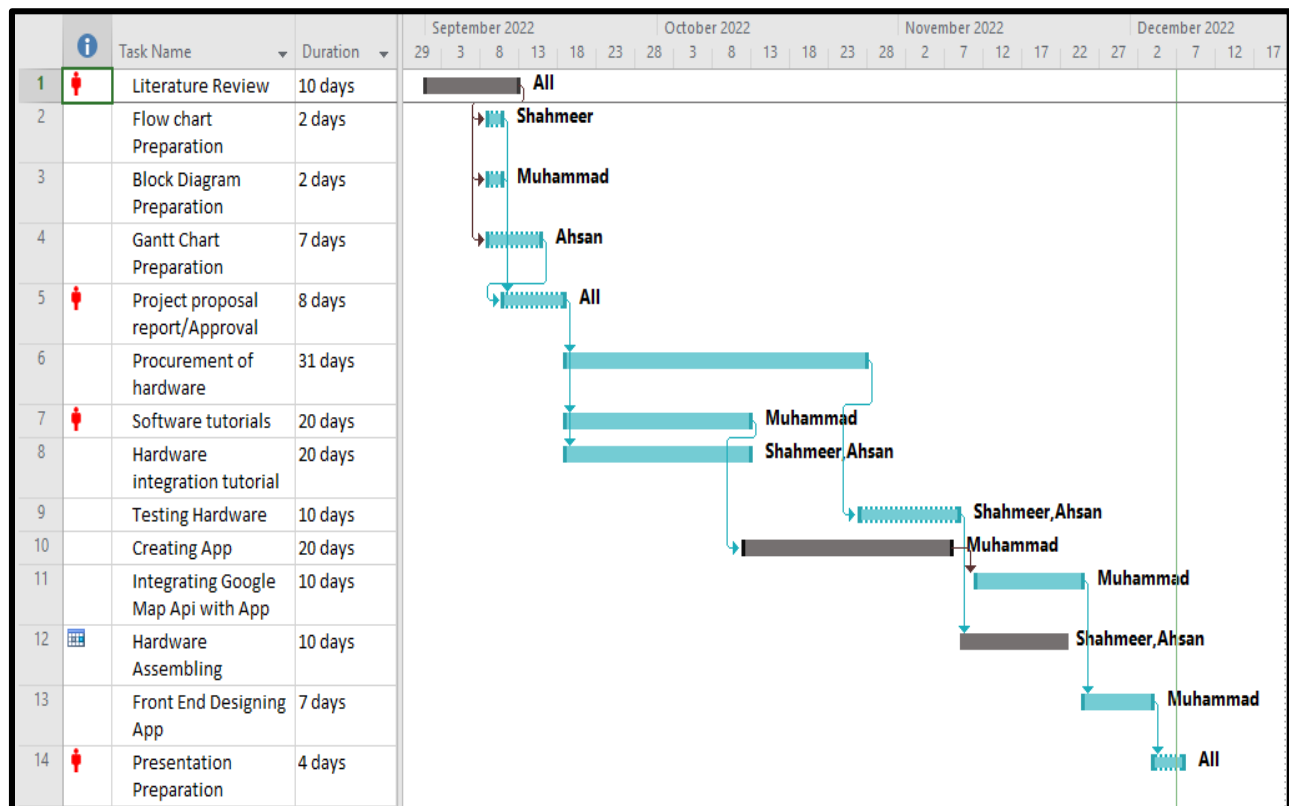


Figure 3.4.1 FYP-I Gantt Chart.

I. FYP-I

The work distribution of FYP-I was divided into four major parts:

- The first part included all of the research work and the preparation needed for the project defense.
- The second part included the software tutorials and the Front End designing of application.
- The third part included the testing of hardware and the hardware assembly of the drone.
- The fourth part included the Google Map API integration with application and the preparation for final FYP-I presentation.

Chapter3: Result and Recommendations

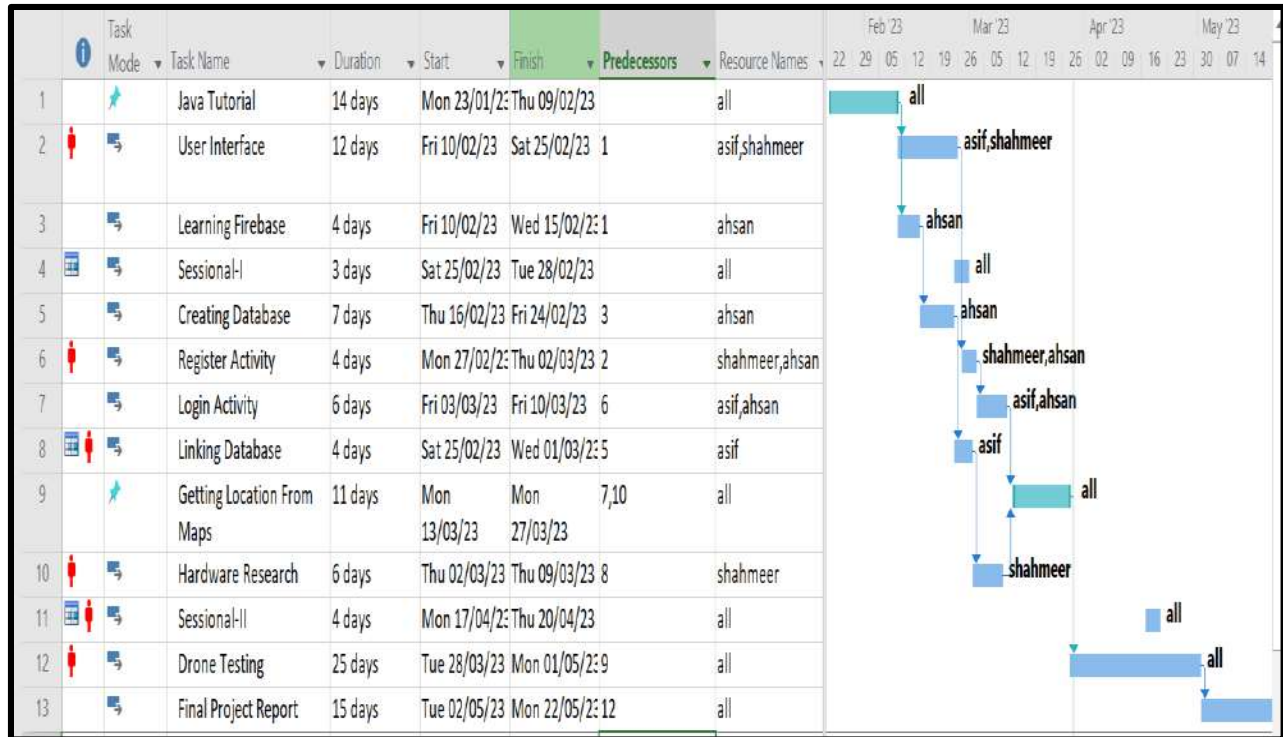


Figure 3.4.2 FYP-II Gantt Chart.

II. FYP-II

The work distribution of FYP-II was divided into four major parts:

- The first part included the complete user interface of the application and the tutorials for back end app development.
- The second part included the creation of Database and linking it with the application.
- The third part included the interfacing of Mission Planner software with the drone and saving the location into the database.
- The fourth part included the final successful testing of the drone and the application, and the preparation for the final FYP-II presentation.

3.5 Milestones Achieved

Our prototype is able to successfully deliver the First Aid Kit to the user. We achieved this by

- Automatically controlled flight.
- Mobile Application.
- Efficient Delivery Mechanism.
- Efficient First Aid Delivery.
- Can carry upto 200g of weight.

3.6 Sustainable Development Goals (SDGs)

The Sustainable Development Goals (SDGs), also known as the Global Goals, were adopted by the United Nations in 2015 as a universal call to action to end poverty, protect the planet, and ensure that by 2030 all people enjoy peace and prosperity. The 17 SDGs are integrated—they recognize that action in one area will affect outcomes in others, and that development must balance social, economic, and environmental sustainability. Among these 17 SDGs following are the goals that our project satisfies:

(i) **Good Health and Well Being:**

Our drone will be a source of remedy. It will help in saving lives. Not only in serious accidents but also delivering basic medical supplies needed.



Figure 3.6.1 SGD Goal 1.

(ii) **Industry, Innovation and Infrastructure:**

Keeping in mind the Pakistani market drones are not used very commonly let alone for medical purposes. The medical industry will not only benefit from this but this also brings innovation on the plate.



Figure 3.6.2 SGD Goal 2.

3.7 Conclusions

First Aid Delivery drones can significantly reduce response times in emergencies by bypassing ground-level traffic and other obstacles. They can quickly reach the scene and provide necessary medical supplies, such as bandages, defibrillators, or even life-saving medications, before professional medical personnel arrive.

3.8 Recommendations / Future Work

In the future, the development of an automated First Aid Delivery Drone holds immense potential for revolutionizing the healthcare industry and saving countless lives. While the current project focuses on the delivery of personalized first aid kits to individuals in need, there are several avenues for future work and recommendations to further enhance the capabilities and impact of this technology.

One key aspect to explore in future iterations is the integration of advanced medical diagnostic tools and real-time communication capabilities within the drone system. This would enable remote medical professionals to assess the situation and provide immediate guidance to the individuals in need, enhancing the quality of care delivered before emergency medical services arrive. Additionally, the drone's payload capacity could be expanded to include essential life-saving equipment, such as automated external defibrillators (AEDs), which could be rapidly deployed to treat sudden cardiac arrest cases, further reducing response times and improving survival rates.

Furthermore, incorporating artificial intelligence (AI) and machine learning algorithms into the drone's system can enable it to navigate complex urban environments more efficiently, optimizing flight paths and avoiding obstacles. This would help mitigate risks associated with congested areas and ensure faster response times, especially in densely populated cities. Additionally, leveraging AI algorithms for predictive analytics could enable the drone to anticipate potential emergency situations based on historical data, proactively positioning itself in high-risk areas to reduce response times even further.

In conclusion, the future work and recommendations for the development of an automated First Aid Delivery Drone involve incorporating advanced medical diagnostics, expanding payload capacity, integrating AI algorithms for efficient navigation, and establishing robust regulations. By continually pushing the boundaries of innovation and collaboration, we can harness the full potential of this technology and significantly reduce response times, saving more lives in critical situations, particularly during sudden cardiac arrest incidents.

Appendix-A: Project Codes

A-1 Dashboard app name

A-1.1 Java File (back end)

```

package com.example.mapsapplication;

import android.content.Intent;
import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class Splash_Activity extends AppCompatActivity {

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

        //class...variable....object created (used if we want to do parallel
work)
        Thread thread = new Thread() {

            public void run(){
                try { // if there can be any error in code write it here
                    sleep(4000);
                }
                catch (Exception e){ // application does not crash in case of
error

                    e.printStackTrace(); // every exception is shown
                }
                finally { // code has to run in any case
                    Intent intent = new Intent(Splash_Activity.this,
LoginActivity.class);
                    startActivity(intent);
                }
            }
        };thread.start();
    }
}

```

A-1.2 XML File (front end)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:background="#710202"
    tools:context=".Splash Activity">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerInParent="true"
        android:foregroundGravity="center"
        app:srcCompat="@drawable/picc_drone" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap content"
        android:layout_height="wrap content"
        android:layout_below="@+id/imageView"
        android:layout_centerInParent="true"
        android:layout_marginLeft="160dp"
        android:rotation="0"
        android:rotationX="-17"
        android:rotationY="27"
        android:text="DroneAid"
        android:textColor="@color/white"
        android:textSize="40sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap content"
        android:layout_height="wrap content"
        android:layout_alignParentBottom="true"
        android:layout_centerInParent="true"
        android:layout_marginBottom="20dp"
        android:shadowColor="@color/teal_200"
        android:text="All rights are reserved"
        android:textSize="18sp"
        android:textStyle="bold" />
</RelativeLayout>
```

A-2 Login

A-2.1 Java File (back end)

```

package com.example.mapsapplication;

import android.app.AlertDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

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;
import com.google.firebase.auth.FirebaseUser;

public class LoginActivity extends AppCompatActivity {

    TextView createNewAccount;

    EditText inputEmail, inputPassword;
    Button btnLogin;

    String emailPattern = "[a-zA-Z0-9, -]+@[a-z]+\\.+[a-z]+";
    ProgressDialog progressDialog;

    FirebaseAuth auth;
    FirebaseUser user;

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

        createNewAccount= findViewById(R.id.createNewAccount);

getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,WindowManager
    .LayoutParams.FLAG_FULLSCREEN);

       inputEmail=findViewById(R.id.inputEmail);
        inputPassword=findViewById(R.id.inputPassword);
        btnLogin=findViewById(R.id.btnLogin);
        progressDialog= new ProgressDialog(this);

        auth=FirebaseAuth.getInstance();
        user= auth.getCurrentUser();

        createNewAccount.setOnClickListener(new View.OnClickListener() {

```

```

        @Override
        public void onClick(View view) {
            startActivity(new
Intent(LoginActivity.this, RegisterActivity.class));
        }
    });

    btnLogin.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            PerformLogin();
        }
    });
}

private void PerformLogin() {
    String email = inputEmail.getText().toString();
    String password = inputPassword.getText().toString();

    if (!email.matches(emailPattern)) { // if email does not match pattern
shows toast msg
        inputEmail.setError("Enter Correct Email");
    } else if (password.isEmpty() || password.length() < 6) { // checking
for the password requirements
        inputPassword.setError("Enter Proper Password");
    } else {
        progressDialog.setMessage("Please wait while Login...");
        progressDialog.setTitle("Login");
        progressDialog.setCanceledOnTouchOutside(false);
        progressDialog.show();

        auth.signInWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful())
                { // if task is successful dismiss dialog
                    progressDialog.dismiss();
                    //if task is successful send user to next activity
                    sendUserToNextActivity();
                    Toast.makeText(LoginActivity.this, "Login
Successful", Toast.LENGTH_SHORT).show();
                }
                else
                {
                    progressDialog.dismiss();
                    Toast.makeText(LoginActivity.this,
""+task.getException(), Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    private void sendUserToNextActivity() {
        Intent intent=new Intent(LoginActivity.this, MapsActivity.class);
intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK|Intent.FLAG_ACTIVITY_NEW_TASK
);
        startActivity(intent);
    }
}

```

A-2.2 XML File (front end)

```

<?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=".LoginActivity"
android:background="@drawable/piccc_loginId">

    <View
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#81000000"/>

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:fillViewport="true"
        tools:layout_editor_absoluteX="16dp"
        tools:layout_editor_absoluteY="29dp">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="#00FFFFFF">

            <TextView
                android:id="@+id/textView"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="132dp"
                android:fontFamily="@font/aldrich"
                android:text="Login"
                android:textColor="@color/white"
                android:textSize="44sp"
                android:textStyle="bold"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="0.498"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />

            <EditText
                android:id="@+id/inputEmail"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginStart="24dp"
                android:layout_marginTop="32dp"
                android:layout_marginEnd="24dp"
                android:background="@drawable/input_bg"
                android:drawableLeft="@drawable/baseline_email_24"
                android:drawablePadding="10dp"
                android:ems="10"
                android:hint="Email"
                android:inputType="textEmailAddress"

```

```

android:paddingLeft="20dp"
android:paddingTop="13dp"
android:paddingRight="20dp"
android:paddingBottom="13dp"
android:textColor="@color/white"
android:textColorHint="@color/white"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.0"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/textView" />

```

<EditText

```

android:id="@+id/inputPassword"
android:layout_width="0dp"
android:layout_height="wrap content"
android:layout_marginStart="24dp"
android:layout_marginTop="16dp"
android:layout_marginEnd="24dp"
android:background="@drawable/input_bg"
android:drawableLeft="@drawable/baseline_security_24"
android:drawablePadding="10dp"
android:ems="10"
android:hint="Password"
android:inputType="textPassword"
android:paddingLeft="20dp"
android:paddingTop="13dp"
android:paddingRight="20dp"
android:paddingBottom="13dp"
android:textColor="@color/white"
android:textColorHint="@color/white"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.497"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/inputEmail" />

```

<TextView

```

android:id="@+id/forgotPassword"
android:layout_width="wrap content"
android:layout_height="wrap content"
android:layout_marginTop="16dp"
android:text="Forgot Password?"
android:textColor="@color/white"
android:textSize="20sp"
app:layout_constraintEnd_toEndOf="@+id/inputPassword"
app:layout_constraintTop_toBottomOf="@+id/inputPassword" />

```

<TextView

```

android:id="@+id/createNewAccount"
android:layout_width="wrap content"
android:layout_height="wrap content"
android:layout_marginTop="16dp"
android:text="Create New Account?"
android:textColor="@color/white"
android:textSize="20sp"
app:layout_constraintEnd_toEndOf="@+id/btnLogin"
app:layout_constraintHorizontal_bias="0.498"

```

```

        app:layout_constraintStart toStartOf="@+id/btnLogin"
        app:layout_constraintTop toBottomOf="@+id/btnLogin" />

<Button
    android:id="@+id/btnLogin"
    android:layout width="0dp"
    android:layout height="wrap_content"
    android:layout_marginTop="16dp"
    android:background="@drawable/input_bg"
    android:text="login"
    android:textColor="@color/white"
    app:layout_constraintEnd toEndOf="@+id/forgotPassword"
    app:layout_constraintStart toStartOf="@+id/inputPassword"
    app:layout_constraintTop toBottomOf="@+id/forgotPassword" />

<TextView
    android:id="@+id/textView4"
    android:layout width="wrap_content"
    android:layout height="wrap_content"
    android:layout_marginTop="36dp"
    android:text="OR"
    android:textColor="@color/white"
    android:textSize="20sp"
    android:textStyle="bold"
    app:layout_constraintEnd toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart toStartOf="parent"
    app:layout_constraintTop toBottomOf="@+id/createNewAccount"
/>

<View
    android:id="@+id/view"
    android:layout width="0dp"
    android:layout height="2dp"
    android:layout_marginStart="16dp"
    android:background="@color/white"
    app:layout_constraintBottom toBottomOf="@+id/textView4"
    app:layout_constraintEnd toEndOf="@+id/btnLogin"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart toEndOf="@+id/textView4"
    app:layout_constraintTop toTopOf="@+id/textView4" />

<View
    android:id="@+id/view4"
    android:layout width="0dp"
    android:layout height="2dp"
    android:layout_marginEnd="16dp"
    android:background="@color/white"
    app:layout_constraintBottom toBottomOf="@+id/textView4"
    app:layout_constraintEnd toStartOf="@+id/textView4"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart toStartOf="@+id/btnLogin"
    app:layout_constraintTop toTopOf="@+id/textView4" />

<ImageView
    android:id="@+id/btnFacebook"
    android:layout width="50dp"

```



```

        android:layout_height="50dp"
        android:layout_marginEnd="16dp"
        app:layout_constraintBottom_toBottomOf="@+id/btnGoogle"
        app:layout_constraintEnd_toStartOf="@+id/btnGoogle"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintHorizontal_chainStyle="packed"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/btnGoogle"
        app:srcCompat="@drawable/picc_fb" />

<ImageView
    android:id="@+id/btnGoogle"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:layout_marginStart="16dp"
    android:layout_marginEnd="16dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/btnPhone"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/btnFacebook"
    app:layout_constraintTop_toBottomOf="@+id/textView4"
    app:srcCompat="@drawable/picc_google" />

<ImageView
    android:id="@+id/btnPhone"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:layout_marginStart="16dp"
    app:layout_constraintBottom_toBottomOf="@+id/btnGoogle"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/btnGoogle"
    app:layout_constraintTop_toTopOf="@+id/btnGoogle"
    app:srcCompat="@drawable/picc_phone" />

</androidx.constraintlayout.widget.ConstraintLayout>

</ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>

```

A-3 Register

A-3.1 Java File (back end)

```

package com.example.mapsapplication;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

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;
import com.google.firebase.auth.FirebaseUser;

public class RegisterActivity extends AppCompatActivity {

    TextView alreadyHaveAccount;
    EditText inputEmail, inputPassword, inputConfirmPassword;
    Button btnRegister;

    String emailPattern = "[a-zA-Z0-9, -]+@[a-z]+\\.+[a-z]+";
    ProgressDialog progressDialog;

    FirebaseAuth auth;
    FirebaseUser user;

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

getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,WindowManager
    .LayoutParams.FLAG_FULLSCREEN);

        alreadyHaveAccount= findViewById(R.id.alreadyHaveAccount);

        inputEmail=findViewById(R.id.inputEmail);
        inputPassword=findViewById(R.id.inputPassword);
        inputConfirmPassword=findViewById(R.id.inputConfirmPassword);
        btnRegister=findViewById(R.id.btnRegister);
        progressDialog= new ProgressDialog(this);

        auth=FirebaseAuth.getInstance();
        user= auth.getCurrentUser();

        alreadyHaveAccount.setOnClickListener(new View.OnClickListener() {
            @Override

```

```

        public void onClick(View view) {
            startActivity(new
Intent(RegisterActivity.this, LoginActivity.class));
        }
    });

    btnRegister.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view)
        {
            Performauthentication();
        }
    });
}

private void Performauthentication() {
    String email = inputEmail.getText().toString();
    String password = inputPassword.getText().toString();
    String confirmPassword = inputConfirmPassword.getText().toString();

    if (!email.matches(emailPattern))
    {
        // if email does not match pattern shows toast msg
        inputEmail.setError("Enter Correct Email");
    }
    else if (password.isEmpty() || password.length() < 6)
    {
        // checking for the password requirements
        inputPassword.setError("Enter Proper Password");
    }
    else if (!password.equals(confirmPassword))
    {
        // if password does not match show errors
        inputConfirmPassword.setError("Password not matched");
    }
    else
    {
        progressDialog.setMessage("Please wait while Registering...");
        progressDialog.setTitle("Registration");
        progressDialog.setCanceledOnTouchOutside(false);
        progressDialog.show();

        auth.createUserWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful())
                {
                    // if task is successful dismiss dialog
                    progressDialog.dismiss();
                    //if task is successful send user to next activity
                    sendUserToNextActivity();
                    Toast.makeText(RegisterActivity.this, "Registration
Successful", Toast.LENGTH_SHORT).show();
                }
                else
                {
                    progressDialog.dismiss();
                    Toast.makeText(RegisterActivity.this,

```

Appendix

```
""+task.getException(), Toast.LENGTH_SHORT).show();
    }
    });
}
}
private void sendUserToNextActivity() {
    Intent intent=new Intent(RegisterActivity.this,MapsActivity.class);
intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK|Intent.FLAG_ACTIVITY_NEW TASK
);
    startActivity(intent);
}
}
```

A-3.2 XML File (front end)

```

<?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=".RegisterActivity"
android:background="@drawable/piccc_login1d">

    <View
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#81000000"/>

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:fillViewport="true"
        tools:layout_editor_absoluteX="0dp"
        tools:layout_editor_absoluteY="-16dp">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="#00FFFFFF">

            <EditText
                android:id="@+id/inputConfirmPassword"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginStart="24dp"
                android:layout_marginTop="16dp"
                android:layout_marginEnd="24dp"
                android:background="@drawable/input_bg"
                android:drawableLeft="@drawable/baseline_security_24"
                android:drawablePadding="10dp"
                android:ems="10"
                android:hint="Confirm Password"
                android:inputType="textPassword"
                android:paddingLeft="20dp"
                android:paddingTop="13dp"
                android:paddingRight="20dp"
                android:paddingBottom="13dp"
                android:textColor="@color/white"
                android:textColorHint="@color/white"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="0.0"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/inputPassword" />

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

```

```

android:layout_marginTop="132dp"
android:fontFamily="@font/aldrich"
android:text="Register"
android:textColor="@color/white"
android:textSize="44sp"
android:textStyle="bold"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.498"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent" />

```

<EditText

```

android:id="@+id/inputEmail"
android:layout_width="0dp"
android:layout_height="wrap content"
android:layout_marginStart="24dp"
android:layout_marginTop="64dp"
android:layout_marginEnd="24dp"
android:background="@drawable/input_bg"
android:drawableLeft="@drawable/baseline_email_24"
android:drawablePadding="10dp"
android:ems="11"
android:hint="Email"
android:inputType="textEmailAddress"
android:paddingLeft="20dp"
android:paddingTop="13dp"
android:paddingRight="20dp"
android:paddingBottom="13dp"
android:textColor="@color/white"
android:textColorHint="@color/white"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.0"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/textView" />

```

<EditText

```

android:id="@+id/inputPassword"
android:layout_width="0dp"
android:layout_height="wrap content"
android:layout_marginStart="24dp"
android:layout_marginTop="16dp"
android:layout_marginEnd="24dp"
android:background="@drawable/input_bg"
android:drawableLeft="@drawable/baseline_security_24"
android:drawablePadding="10dp"
android:ems="10"
android:hint="Password"
android:inputType="textPassword"
android:paddingLeft="20dp"
android:paddingTop="13dp"
android:paddingRight="20dp"
android:paddingBottom="13dp"
android:textColor="@color/white"
android:textColorHint="@color/white"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.497"
app:layout_constraintStart_toStartOf="parent"

```

```

        app:layout_constraintTop toBottomOf="@+id/inputEmail" />

<Button
    android:id="@+id/btnRegister"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:background="@drawable/input_bg"
    android:text="Register"
    android:textColor="@color/white"
    app:layout_constraintEnd toEndOf="@+id/inputConfirmPassword"
    app:layout_constraintHorizontal bias="0.0"
    app:layout_constraintStart toStartOf="@+id/inputPassword"

app:layout_constraintTop toBottomOf="@+id/inputConfirmPassword" />

<TextView
    android:id="@+id/alreadyHaveAccount"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="Already have account"
    android:textColor="@color/white"
    android:textSize="20sp"
    app:layout_constraintEnd toEndOf="@+id/btnRegister"
    app:layout_constraintHorizontal bias="0.498"
    app:layout_constraintStart toStartOf="@+id/btnRegister"
    app:layout_constraintTop toBottomOf="@+id/btnRegister" />

</androidx.constraintlayout.widget.ConstraintLayout>

</ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>

```

A-4 Maps

A-4.1 Java File (back end)

```

package com.example.mapsapplication;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import androidx.appcompat.app.AppCompatActivity;

import com.example.mapsapplication.databinding.ActivityMapsBinding;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.CameraUpdateFactory;

public class MapsActivity extends AppCompatActivity implements
    OnMapReadyCallback {

    private GoogleMap mMap;
    private ActivityMapsBinding binding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());
        // Obtain the SupportMapFragment and get notified when the map is
        // ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment)
            getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        Button button;
        button = findViewById(R.id.button1);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(getApplicationContext(),
                    MainActivity2.class);
                startActivity(intent);
            }
        });
        Button button1;
        button1 = findViewById(R.id.button2);
        button1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(getApplicationContext(),
                    MainActivity3.class);

```



```

        startActivity(intent);
    }
});
}

@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Add a marker in Islamabad
    LatLng islamabad = new LatLng(33.738045, 73.084488);
    mMap.addMarker(new
MarkerOptions().position(islamabad).title("Islamabad"));
    mMap.moveCamera(CameraUpdateFactory.newLatLng(islamabad));
    mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(islamabad,16f));
}
}
}

```

A-4.2 XML File (front end)

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match parent"
    android:layout_height="match parent"
    android:background="#710202"
    android:orientation="vertical" >

    <fragment xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:map="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match parent"
        android:layout_height="680dp"
        tools:context=".MapsActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="380dp"
        android:layout_height="80dp"
        android:layout_gravity="center horizontal"
        android:text="Request Drone"
        android:textSize="18sp"
        android:textColor="@color/white"
        app:backgroundTint="#403D3D" />

    <Button
        android:id="@+id/button2"
        android:layout_width="380dp"
        android:layout_height="80dp"
        android:layout_gravity="center horizontal"
        android:text="Profile"
        android:textSize="18sp"
        android:textColor="@color/white"
        app:backgroundTint="#0E1E61" />

</LinearLayout>

```

A-5 Request Drone

A-5.1 Java File (back end)

```
package com.example.mapsapplication;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.Manifest;
import android.annotation.SuppressLint;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationManager;
import android.os.Build;
import android.os.Bundle;
import android.provider.Settings;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class MainActivity2 extends AppCompatActivity implements
BaseGpsListener{

    //Firebase variables
    private DatabaseReference mDatabase;

    private static final int PERMISSION_LOCATION = 1000;

    TextView textView loc;
    Button button loc;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        mDatabase = FirebaseDatabase.getInstance().getReference();

        Button buttonback2;
        buttonback2 = findViewById(R.id.back_btn2);
        buttonback2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view)
            {
                Intent intent = new
Intent(getApplicationContext(),MapsActivity.class);
                startActivity(intent);
            }
        });

        Button buttonsave;
    }
}
```

```

        buttonsave = findViewById(R.id.save btn);
        buttonsave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view)
            {
                Intent intent = new
Intent(getApplicationContext(), MapsActivity.class);
                startActivity(intent);
            }
        });

        textView loc=findViewById(R.id.textView loc);
        button loc=findViewById(R.id.button loc);

        button loc.setOnClickListener(new View.OnClickListener() {
            //check for location permission
            @Override
            public void onClick(View view) {
                if(Build.VERSION.SDK INT >= Build.VERSION CODES.M &&
checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION)
                != PackageManager.PERMISSION GRANTED){
                    requestPermissions(new
String[]{Manifest.permission.ACCESS_FINE_LOCATION}, PERMISSION LOCATION);
                }else {
                    showLocation();
                }
            }
        });
        // firebase code place here
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        if(requestCode == PERMISSION LOCATION){
            if(grantResults[0] == PackageManager.PERMISSION GRANTED){
                showLocation();
            }else {
                Toast.makeText(this, "Permission not granted!!",
Toast.LENGTH_SHORT).show();
                finish();
            }
        }
    }

    @SuppressWarnings("MissingPermission")
    private void showLocation(){
        LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION SERVICE);
        //check if GPS is enabled
        if(locationManager.isProviderEnabled(LocationManager.GPS PROVIDER)) {
            // start Locating
            textView loc.setText("Loading location...");
        }
    }
}

```

```

locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0,
this);
    } else {
        // enable GPS
        Toast.makeText(this, "Enable GPS!!", Toast.LENGTH_SHORT).show();
        startActivity(new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS));
    }
}

// show location
private String hereLocation(Location location){
    return "Lat: " + location.getLatitude() + "\nLon: " +
location.getLongitude();
}

@Override
public void onLocationChanged(Location location) {
    // update location
    textView loc.setText(hereLocation(location));

sendLocationToFirebase(location.getLatitude(),location.getLongitude());
}
private void sendLocationToFirebase(double latitude, double longitude) {
    // Create a new child node in the "locations" node
    DatabaseReference locationRef = FirebaseDatabase.getInstance().getReference("locations").push();

    // Create a new Location object
    LocationData locationData = new LocationData(latitude, longitude);

    // Set the value of the child node to the LocationData object
    locationRef.setValue(locationData);
}
public class LocationData {
    public double latitude;
    public double longitude;

    public LocationData() {
        // Default constructor required for Firebase
    }

    public LocationData(double latitude, double longitude) {
        this.latitude = latitude;
        this.longitude = longitude;
    }
}

@Override
public void onProviderDisabled(String provider) {
    // empty
}

```

Appendix

```
@Override
public void onProviderEnabled(String provider) {
    // empty
}

@Override
public void onStatusChanged(String provider, int status, Bundle extras) {
    // empty
}

@Override
public void onGpsStatusChanged(int event) {
    // empty
}
}
```

A-5.2 XML File (front end)

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    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:background="#710202"
    tools:context=".MainActivity3">

    <com.google.android.material.appbar.AppBarLayout
        android:id="@+id/app_bar_settings"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar_settings"
            android:layout_width="match_parent"
            android:layout_height="80dp"
            android:background="#0F0E10">

            <RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    >

                    <Button
                        android:id="@+id/back_btn2"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="Back"
                        app:layout_constraintStart toStartOf="parent"
                        app:layout_constraintTop toTopOf="parent"
                        app:layout_constraintRight toLeftOf="@+id/save_btn"
                        android:background="#3B373C"
                        android:textSize="20dp"
                        android:textStyle="bold"
                        android:layout_marginTop="5dp"
                        android:textColor="@color/white"
                    />

                    <Button
                        android:id="@+id/save_btn"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_marginTop="5dp"

```

```

        android:layout_marginEnd="16dp"
        android:background="#3B373C"
        android:layout="@id/toolbar_settings"
        android:text="Save"
        android:textColor="@color/white"
        android:textSize="20dp"
        android:textStyle="bold"
        app:layout_constraintEnd toEndOf="parent"
        app:layout_constraintLeft toRightOf="@+id/back btn2"

        app:layout_constraintTop toTopOf="parent" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</RelativeLayout>

</androidx.appcompat.widget.Toolbar>

</com.google.android.material.appbar.AppBarLayout>

<TextView
    android:id="@+id/textView loc"
    android:layout width="wrap content"
    android:layout height="wrap content"
    android:layout below="@+id/app bar settings"
    android:layout_alignParentStart="true"
    android:layout_alignParentEnd="true"
    android:layout_marginStart="150dp"
    android:layout_marginTop="175dp"
    android:layout_marginEnd="183dp"
    android:text="Location"
    android:textColor="@color/white"
    android:textSize="20sp" />

<Button
    android:id="@+id/button loc"
    android:layout width="163dp"
    android:layout height="wrap content"
    android:layout below="@+id/app_bar_settings"
    android:layout_marginStart="120dp"
    android:layout_marginTop="120dp"
    android:background="#3B373C"
    android:layout="@id/toolbar_settings"
    android:onClick="onLocationButtonClick"
    android:text="Get Location"
    android:textColor="@color/white"
    android:textSize="20dp"
    android:textStyle="bold"
    app:layout_constraintEnd toEndOf="parent"
    app:layout_constraintTop toTopOf="parent" />

</RelativeLayout>

```

A-6 Profile

A-6.1 Java File (back end)

```

package com.example.mapsapplication;

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

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity3 extends AppCompatActivity
{
    private EditText nameEditText, passwordEditText, emailEditText,
    cnicEditText, phoneEditText;
    //    FirebaseAuth mAuth; //////////////// new
    //    FirebaseUser mUser; //////////////// new
    //    DatabaseReference mUserRef; //////////////// new

    Button savebtn;

    private CustomFormValidation Cv;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main3);

        Button buttonback;
        buttonback = findViewById(R.id.back_btn);
        buttonback.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view)
            {
                Intent intent = new
Intent(getApplicationContext(),MapsActivity.class);
                startActivity(intent);
            }
        });

        Button buttonsave;
        buttonsave = findViewById(R.id.save_btn);
        buttonsave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view)
            {
                Intent intent = new
Intent(getApplicationContext(),MapsActivity.class);
                startActivity(intent);
            }
        });

        passwordEditText= findViewById(R.id.password);
        emailEditText= findViewById(R.id.inputEmail);

```



```

savebtn=findViewById(R.id.save btn) ;

Cv=new CustomFormValidation() ;

savebtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String passstore,emailstore;
        int errorcheck=0;
        passstore=passwordEditText.getText().toString();
        emailstore=emailEditText.getText().toString();

        if(passstore.isEmpty() && emailstore.isEmpty()){
            Toast.makeText(MainActivity3.this, "All fields are
mandatory", Toast.LENGTH_SHORT).show();
            // feedback message
        }
        else{
            Cv.passValid(passstore);
            Cv.emailValid(emailstore);
        }

        if(!Cv.passValid(passstore)) {
            Toast.makeText(MainActivity3.this, "Password is
Invalid", Toast.LENGTH_SHORT).show();
            errorcheck++;
        }
        if(!Cv.emailValid(emailstore)) {
            Toast.makeText(MainActivity3.this, "Email is
Invalid", Toast.LENGTH_SHORT).show();
            errorcheck++;
        }
        if(errorcheck==0)
            Toast.makeText(MainActivity3.this, "Validation
Success", Toast.LENGTH_LONG).show();
        } // ending onclick
    }); // ending setOnClickListener
} // ending onCreate

} // ending class

```

CustomFormValidation class

```
package com.example.mapsapplication;

public class CustomFormValidation {
    // Boolean tfpass,tfemail;
    // EditText pass;

    public boolean emailValid(String p) {
        if (p.matches("[a-zA-Z0-9, -]+@[a-z]+\\.+[a-z]+"))

            return true;
        else
            return false;
    }
    public boolean passValid(String p) {
        if (p.matches("(?=.*[0-9]) (?=.[a-z]) (?=.[A-Z]) (?=.[@#$%^&+=]) (?=\\S+$)}.{,}$"))
            // atleast 1 digit, atleast 1 lowercase, atleast 1 uppercase,
            // atleast 1 special char, no white spaces
            return true;
        else
            return false;
    }
}
```

A-6.2 XML File (front end)

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
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:background="#710202"
tools:context=".MainActivity3">

<com.google.android.material.appbar.AppBarLayout
android:id="@+id/app_bar_settings"
android:layout_width="match_parent"
android:layout_height="wrap_content">

<androidx.appcompat.widget.Toolbar
android:id="@+id/toolbar_settings"
android:layout_width="match_parent"
android:layout_height="80dp"
android:background="#0F0E10">

<RelativeLayout
android:layout_width="match_parent"
android:layout_height="wrap_content">

<androidx.constraintlayout.widget.ConstraintLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
>

<Button
android:id="@+id/back_btn"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Back"
app:layout_constraintStart toStartOf="parent"
app:layout_constraintTop toTopOf="parent"
app:layout_constraintRight toLeftOf="@+id/save_btn"
android:textSize="20dp"
android:textStyle="bold"
android:background="#3B373C"
android:layout_marginTop="5dp"
android:textColor="@color/white"
/>

<Button
android:id="@+id/save_btn"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginTop="5dp"

```

```

        android:layout_marginEnd="16dp"
        android:background="#3B373C"
        android:layout="@id/toolbar_settings"
        android:text="Save"
        android:textColor="@color/white"
        android:textSize="20dp"
        android:textStyle="bold"
        app:layout_constraintEnd toEndOf="parent"
        app:layout_constraintLeft toRightOf="@+id/back_btn"

        app:layout_constraintTop toTopOf="parent" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</RelativeLayout>

</androidx.appcompat.widget.Toolbar>

</com.google.android.material.appbar.AppBarLayout>

<EditText
    android:id="@+id/password"
    android:layout_width="match parent"
    android:layout_height="wrap content"
    android:hint="Password:"
    android:inputType="text"
    android:textColor="@color/white"
    android:textColorHint="@color/white"
    android:layout_below="@+id/app_bar_settings"
    android:layout_marginStart="20dp"
    android:layout_marginEnd="20dp"
    android:layout_marginTop="15dp"
    />

<EditText
    android:id="@+id/inputEmail"
    android:layout_width="match parent"
    android:layout_height="wrap content"
    android:hint="Email:"
    android:textColor="@color/white"
    android:textColorHint="@color/white"
    android:layout_below="@+id/password"
    android:layout_marginStart="20dp"
    android:layout_marginEnd="20dp"
    android:layout_marginTop="15dp"
    />

</RelativeLayout>

```

Bibliography

[1] Wimberley Bill, "Drones in healthcare: Blood transportation and beyond", Head of Business Development, WINGCOPTER GmbH, Germany, Issue - 11, June-2021.

[2] Johnson AM, Cunningham CJ, Arnold E, Rosamond WD, Zègre-Hemsey JK, "Impact of Using Drones in Emergency Medicine: What Does the Future Hold?", USA, Issue 16, November-2021.

[3] Alex Monton "Ambulance drone is a flying first aid kit that could save lives", Netherland, October 29, 2014.

[4] Wan Mazlina Wan Mohamed ¹, Muhamad Syaiful Azrie Zulkiflis ², Adibah Shuib ³, "Mechanism For Drones Delivering The Medical First Aid Kits", ¹Malaysia Institute of Transport (MITRANS), ²Faculty of Mechanical Engineering, Tower 1, Engineering Complex, ³Centre for Mathematics Studies, Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (UiTM), 40450 Shah Alam, Selangor, Malaysia, January-2018.

[5] Sasi G, Sankar Arun, Kumar Manoj, Jeeva M, "Autonomous Drone using Pixhawk Flight Controller with Live Stream and Mask Detection Features, Autonomous Drone using Pixhawk Flight Controller with Live Stream and Mask Detection Features", Paavai Engineering College, Namakkal, *India*, ISSN: 2278-018, Vol. 10, Issue May 05, 2021.

[6] Parvathi Sanjana, M. Prathilothamai, "Drone Design for First Aid kit Delivery in Emergency Situation", Department of Computer Science and Engineering Amrita School of Engineering, Coimbatore Amrita Vishwa Vidyapeetham, India, July-2020.