PDE 2101
ENGINEERING SOFTWARE DEVELOPMENT

IFIT STUDIO

ANUSHKA NARSIMA – M00851749
RURAMAI MUCHENGA – M00912425
JOELLA JOSE – M00850468

OVERVIEW

1 ABSTRACT

2 PROBLEM DEFINITION & BENEFITS

3 REQUIREMENTS SPECIFICATION

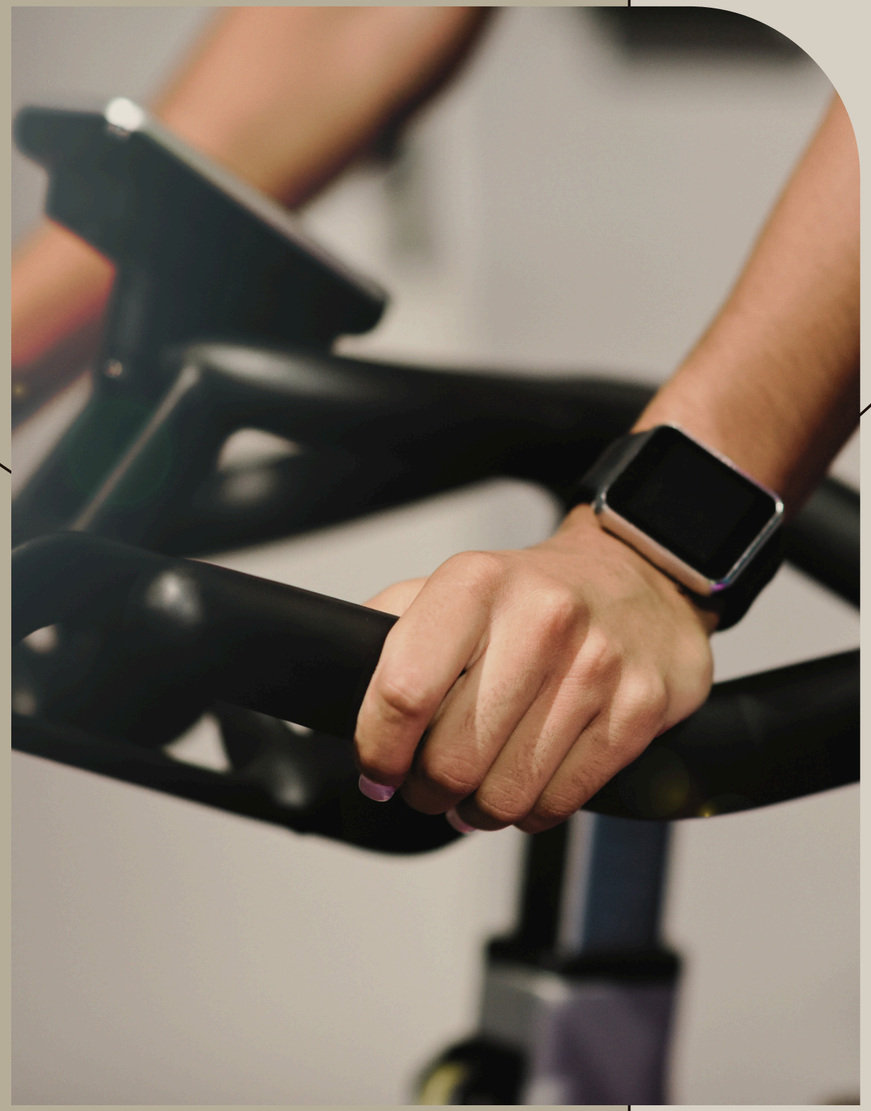
4 TECHNOLOGIES & DEVELOPMENT TOOLS

5 FLOWCHART

6 PROJECT MANAGEMENT DIAGRAMS

7 IMPLEMENTATION

8 CONCLUSION



ABSTRACT

Our project introduces an innovative IoT-based fitness management system comprising a smartwatch and a companion mobile application.

The system seamlessly integrates these components to enhance the fitness tracking experience and optimize gym operations.

PROBLEM DEFINITION

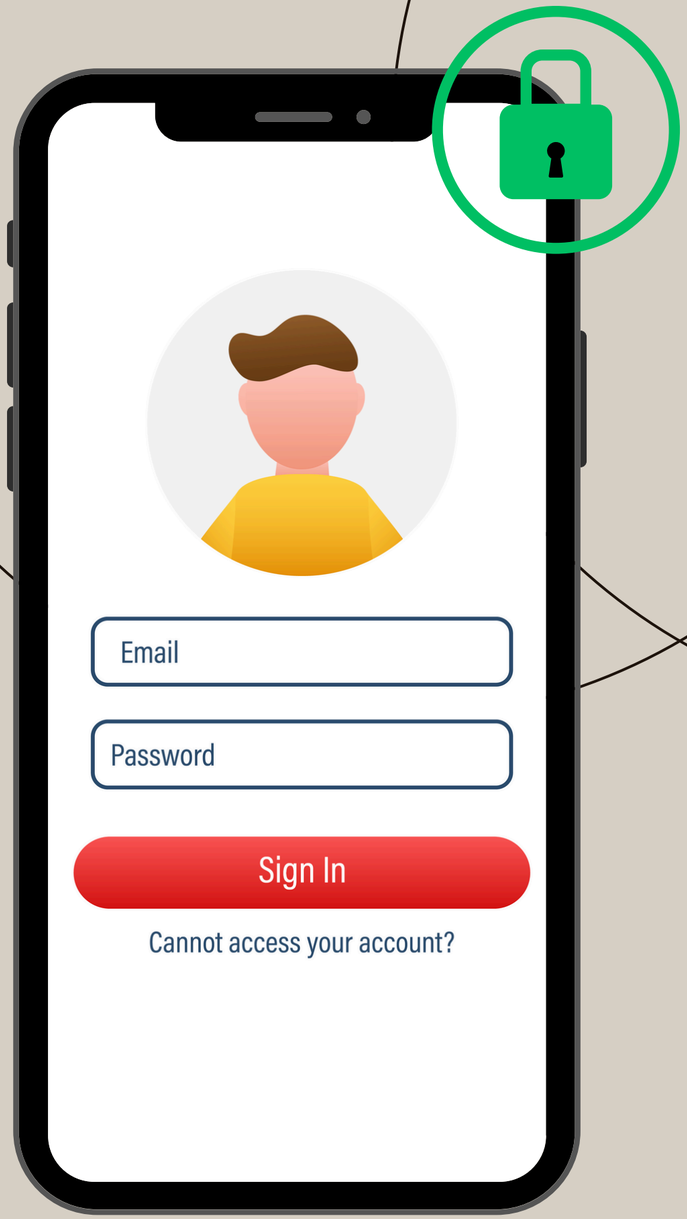
In the realm of fitness management, a key challenge is seamlessly integrating equipment availability tracking and workout guidance.

Based on research from Healthline and IEEE Xplore, we can effectively address these challenges by developing a comprehensive IoT-based fitness management system that displays present and upcoming workouts on a smartwatch.

This system aims to streamline gym operations, optimise equipment usage, and enhance the overall fitness experience for users.

BENEFITS

- Exercise Tracking: Tracks exercise with minimal data for efficient monitoring and progress tracking.
- New workout addition Feature: No current products have this option, & it is essential since exercises are inexhaustive.
- Personalized Workout Guidance: The smartwatch displays present and upcoming exercises, providing personalized guidance tailored to users' fitness goals, keeping them motivated and on track.
- Improved Community Engagement: The mobile app's group chat fosters a sense of community, enabling users to share tips, motivation, and fitness-related discussions, promoting accountability.



REQUIREMENTS SPECIFICATION

FUNCTIONAL REQUIREMENTS:

- Workout tracking on the smartwatch
- Plan Generation
- Present and following scheduled exercises displayed based on the user's workout plan
- User authentication on the mobile application
- Group chat feature for community engagement

NON-FUNCTIONAL REQUIREMENTS:

- Reliability and accurate data tracking
- User-friendly interfaces on smartwatch and mobile app
- Low latency in displaying real-time data and notifications
- Robust user authentication mechanisms
- Safe keeping of sensitive data
- Scalable to accommodate future updates and expanding user base

TECHNOLOGIES & DEVELOPMENT TOOLS

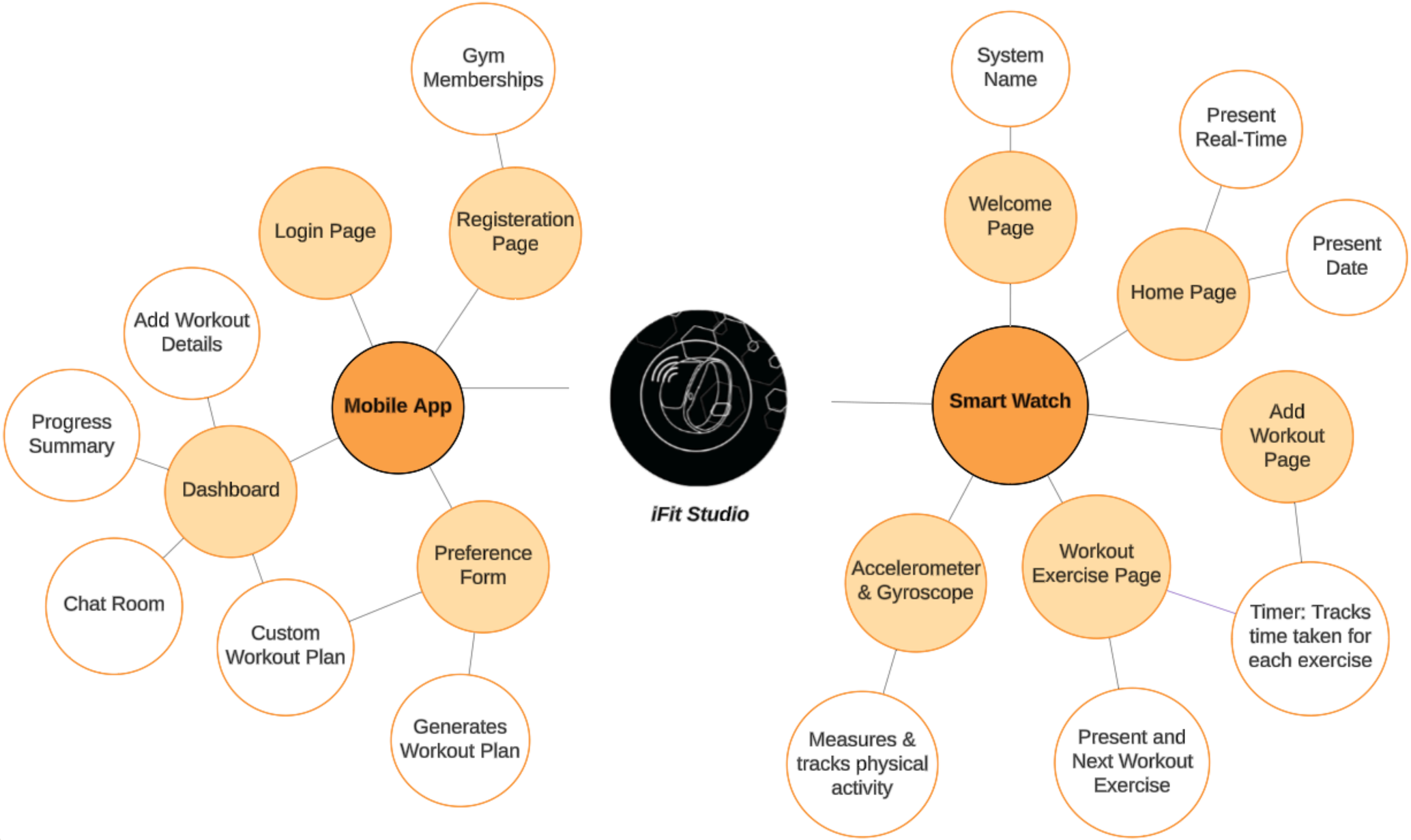
SOFTWARE

- Flutter
- Firebase Firestore (NoSQL DB)
- Arduino IDE
- Python – Flask, Matplotlib, Pandas
- Git/GitHub

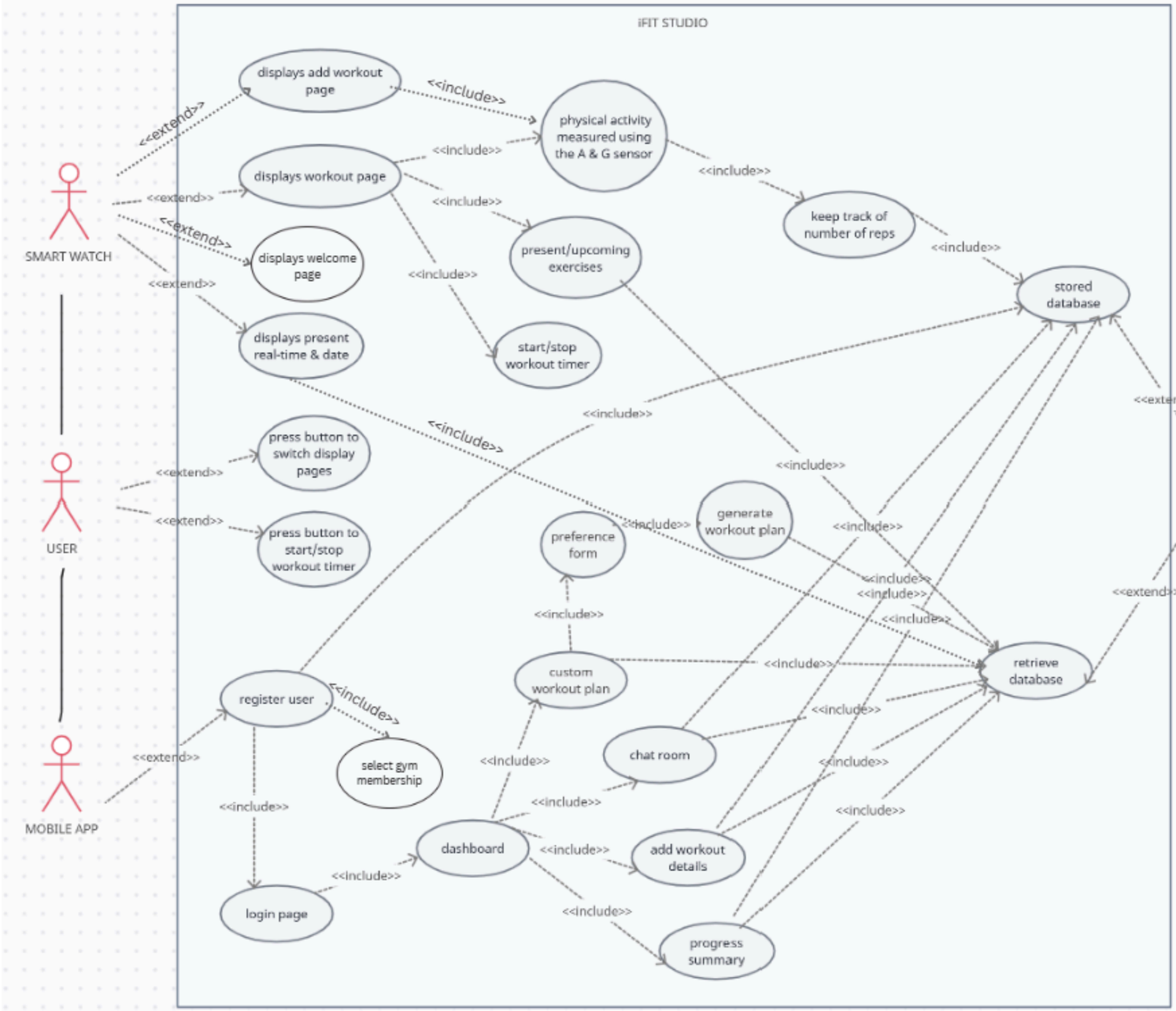
HARDWARE

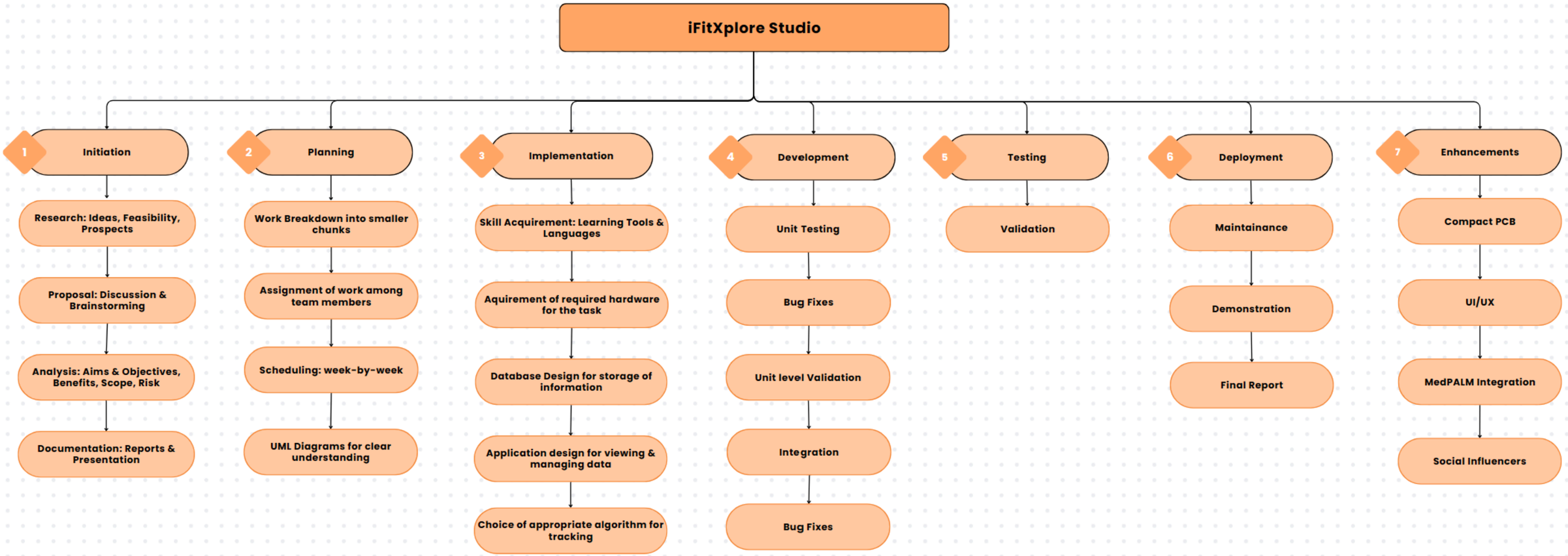
- LCD 1.28inch waveshare
- Accelerometer & Gyroscope MPU 6050
- ESP 32 Board
- Buttons
- Power Supply
- Jumper wires and a Breadboard

FEATURES MINDMAP



USE CASE DIAGRAM

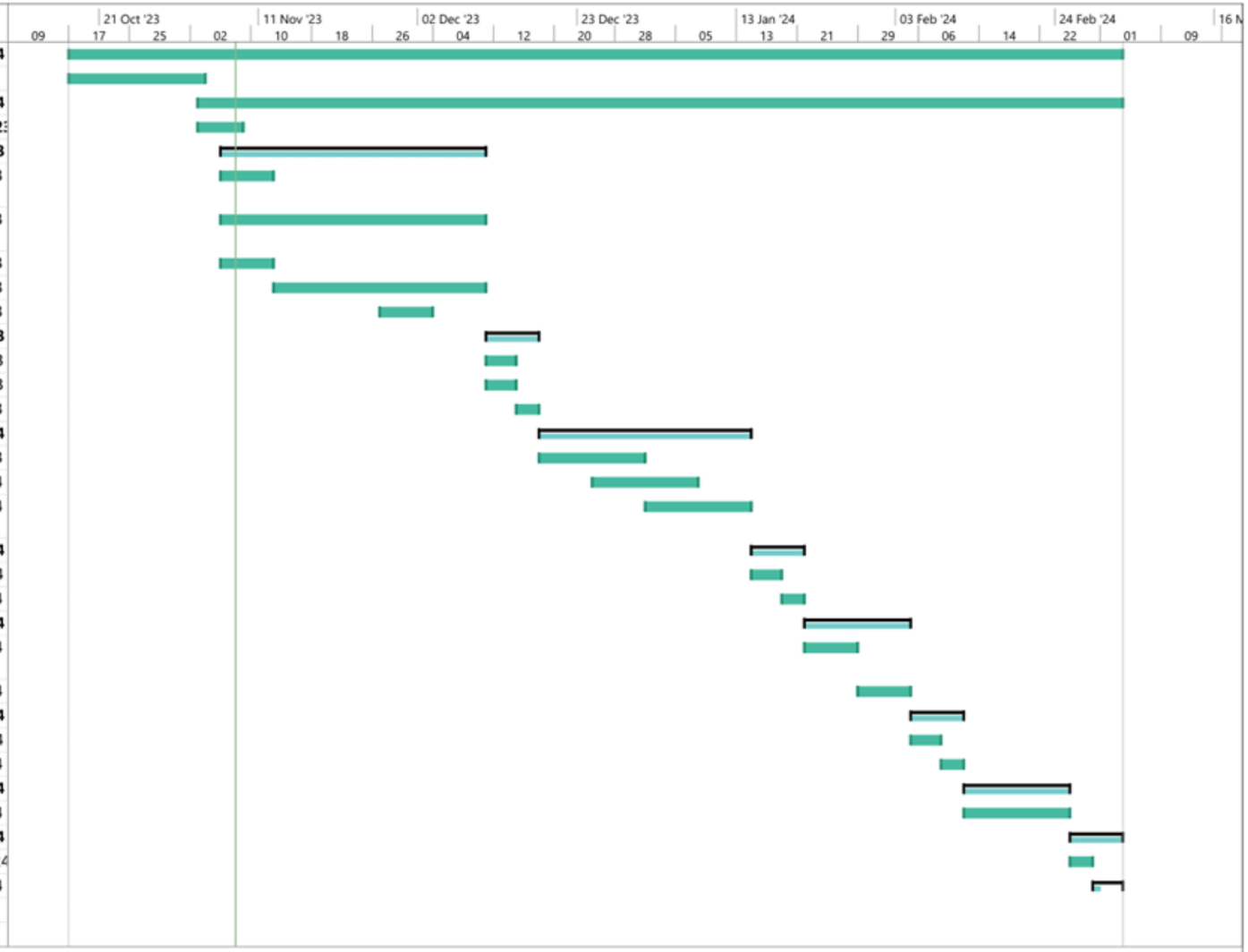




WORK BREAKDOWN STRUCTURE

GANTT CHART

ID	Task Name	Duration	Start	Finish
1	**FitXplore Studio**	**100 day**	**Tue 17/10/23**	**Sun 03/03/24**
2	**1. Project Approval**	**14 days**	**Tue 17/10/23**	**Fri 03/11/23**
3	**2. Documentation**	**87 days**	**Fri 03/11/23**	**Sun 03/03/24**
4	**3. Proposal Presentation**	**4 days**	**Fri 03/11/23**	**Wed 08/11/2**
5	**4. Development Phase 1**	**26 days**	**Mon 06/11/2**	**Sun 10/12/23**
6	4.1. Research & decide detailed Plan of Action	6 days	Mon 06/11/23	Sun 12/11/23
7	4.2. Acquire required skills needed to proceed	26 days	Mon 06/11/23	Sun 10/12/23
8	4.3. Decide Component List & Or	6 days	Mon 06/11/2	Sun 12/11/23
9	4.4. Eqpt. Hardware Dev.	21 days	Mon 13/11/2	Sun 10/12/23
10	4.5. Basic App Design & Dev.	6 days	Mon 27/11/2	Sun 03/12/23
11	**5. Testing Phase 1**	**6 days**	**Mon 11/12/2**	**Sun 17/12/23**
12	5.1. Test Eqpt. Hardware	4 days	Mon 11/12/2	Thu 14/12/23
13	5.2. Check accuracy of Readings	4 days	Mon 11/12/2	Thu 14/12/23
14	5.3. Validate app pages	2 days	Fri 15/12/23	Sun 17/12/23
15	**6. Development Phase 2**	**21 days**	**Mon 18/12/2**	**Sun 14/01/24**
16	6.1. Functional App Dev.	11 days	Mon 18/12/2	Sun 31/12/23
17	6.2. Cloud DB Setup	11 days	Mon 25/12/2	Sun 07/01/24
18	6.3. Watch Software Design & Implementation	11 days	Mon 01/01/24	Sun 14/01/24
19	**7. Testing Phase 2**	**6 days**	**Mon 15/01/2**	**Sun 21/01/24**
20	7.1 App Testing & Validation	4 days	Mon 15/01/2	Thu 18/01/24
21	7.2. Cloud DB Testing	2 days	Fri 19/01/24	Sun 21/01/24
22	**8. Development Phase 3**	**11 days**	**Mon 22/01/2**	**Sun 04/02/24**
23	8.1. Watch Hardware Design & Implementation	6 days	Mon 22/01/24	Sun 28/01/24
24	8.2. Chatroom Dev.	6 days	Mon 29/01/2	Sun 04/02/24
25	**9. Testing Phase 3**	**6 days**	**Mon 05/02/2**	**Sun 11/02/24**
26	9.1. Check watch Performance	4 days	Mon 05/02/2	Thu 08/02/24
27	9.2. Test chatroom features	2 days	Fri 09/02/24	Sun 11/02/24
28	**10. Development Phase 4**	**11 days**	**Mon 12/02/2**	**Sun 25/02/24**
29	10.1. Integration	11 days	Mon 12/02/2	Sun 25/02/24
30	**11. Testing Phase 3**	**6 days**	**Mon 26/02/2**	**Sun 03/03/24**
31	11.1. Integrated Testing	3 days	Mon 26/02/2	Wed 28/02/2
32	11.2. System Validation	3 days	Thu 29/02/24	Sun 03/03/24
33				
34				



Project: Project1
Date: Wed 08/11/23

Task	Project Summary	Manual Task	Start-only	Deadline
Split	Inactive Task	Duration-only	Finish-only	Progress
Milestone	Inactive Milestone	Manual Summary Rollup	External Tasks	Manual Progress
Summary	Inactive Summary	Manual Summary	External Milestone

RACI MATRIX

Task / Role	Project Manager (Anushka)	Hardware Engineer (Joella)	Software Developer (Ruramai)	Module Tutor (Dr. Sumitra Kotipalli)
Planning and Requirements	R	A	C	I
Front end	A	I	R	I
Back end	R	I	A	I
Hardware Integration	A	R	I	I
Documentation	A	R	A	I

KEY:
R (Responsible)
A (Accountable)
C (Consulted)
I (Informed)

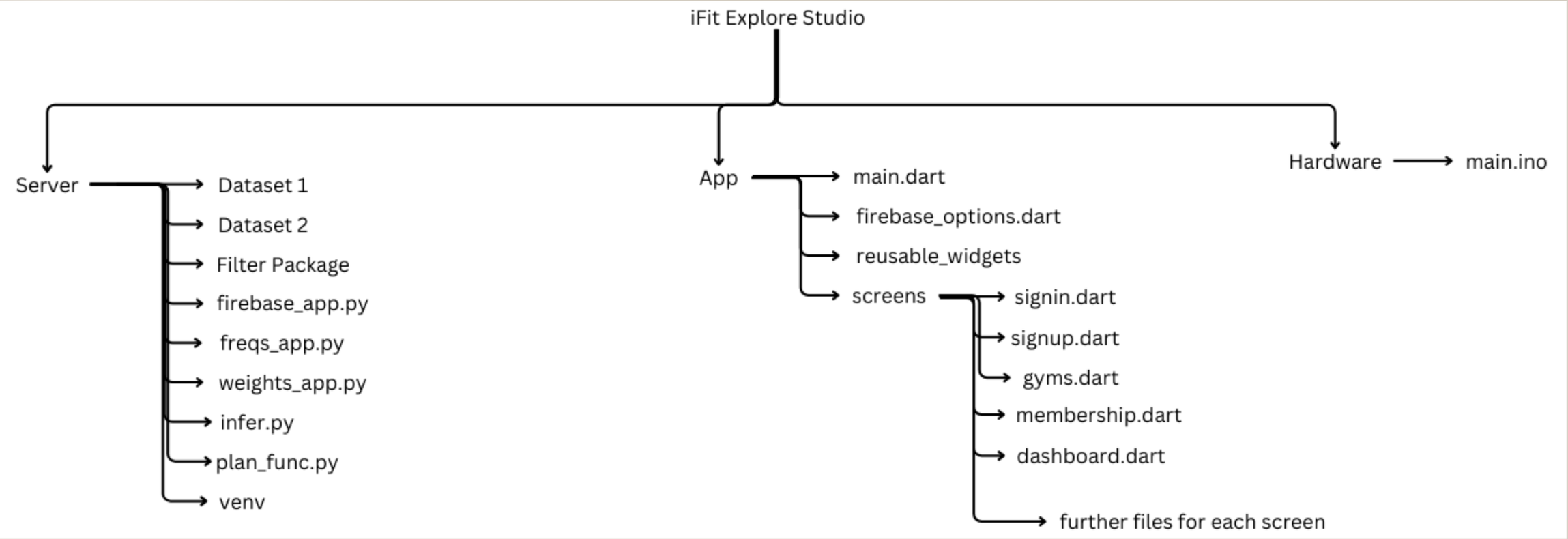
Tool used to clarify roles and responsibilities within a project

RESEARCH & DEVELOPMENT METHODOLOGY

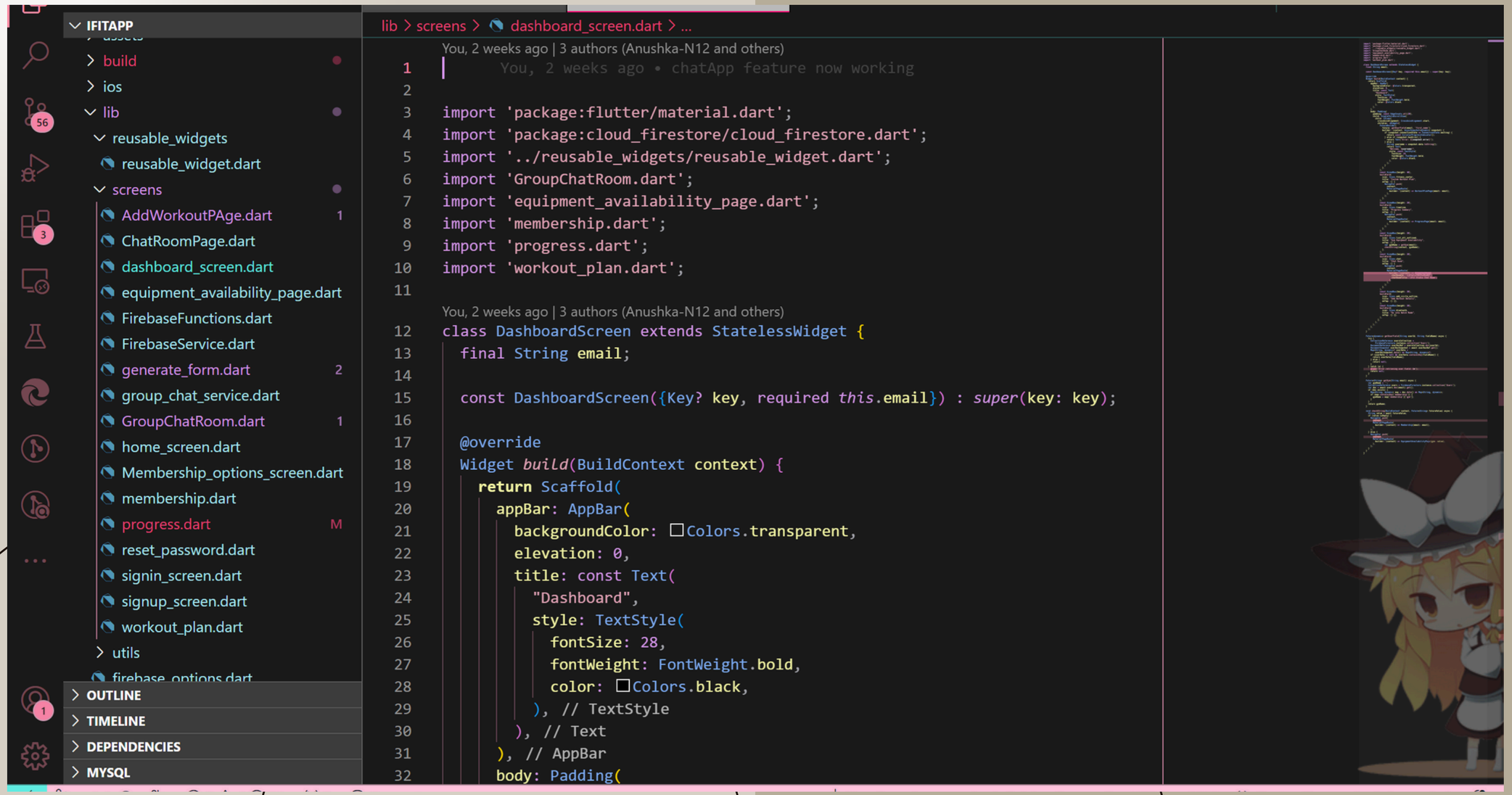
For the research methodology, we conducted exploratory research to understand IoT trends and challenges in the fitness sector. This involved observing industry trends, identifying challenges, and exploring opportunities. Applied research involved looking for areas of improvement in existing methods for the task. This research guided our decision-making throughout the project lifecycle, ensuring that we were well-informed about the landscape.

For the development methodology, we adopted the agile approach, specifically the Scrum framework, due to the duration & collaboration involved. This method suited our project's multiple small components, allowing for easier debugging and integration. The project was divided into sprints with regular meetings to update progress, plan actions, and address issues. This fast-paced and iterative approach facilitated rapid implementation, ensuring the production of a quality product.

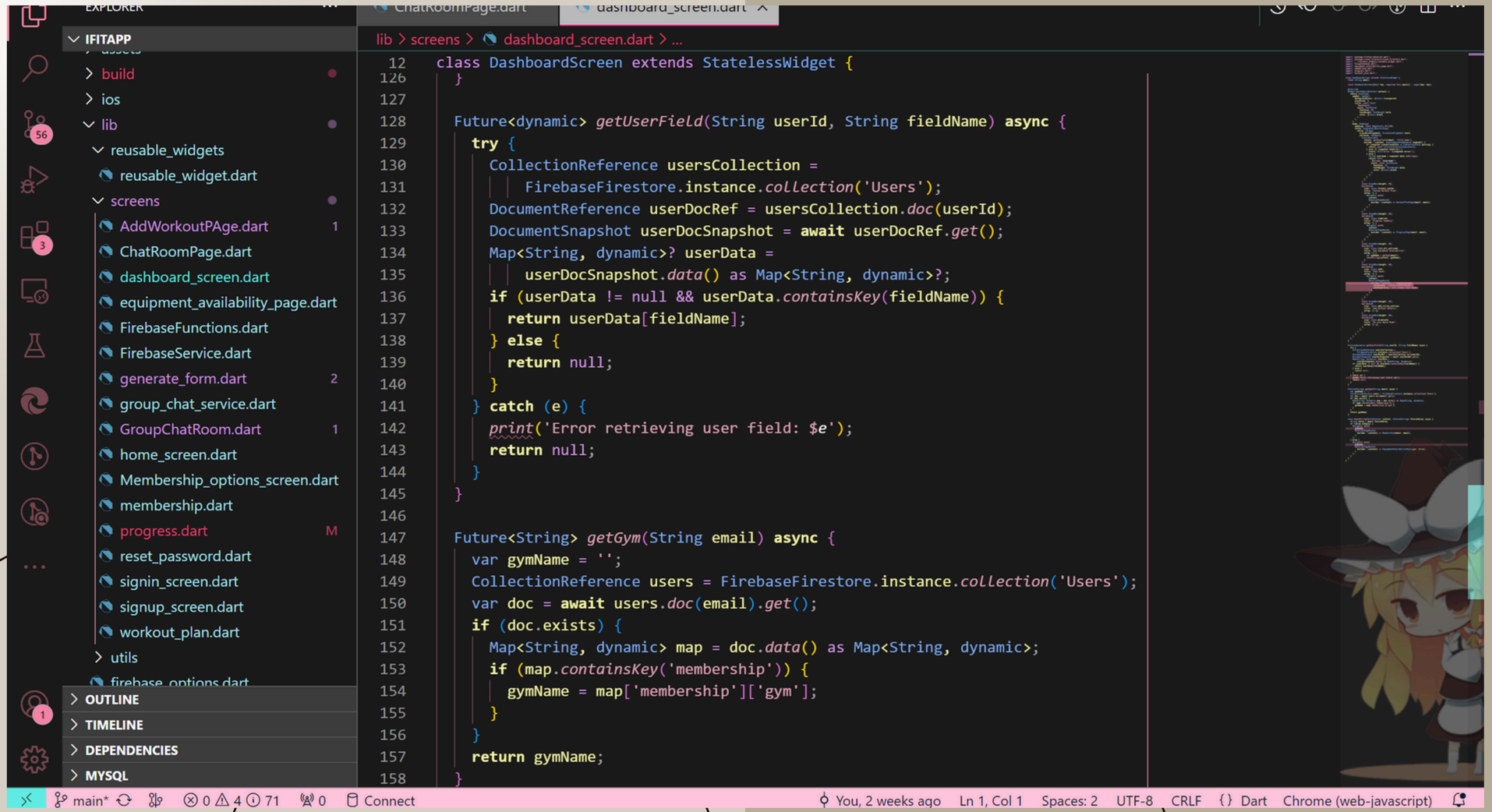
IMPLEMENTATION OVERVIEW



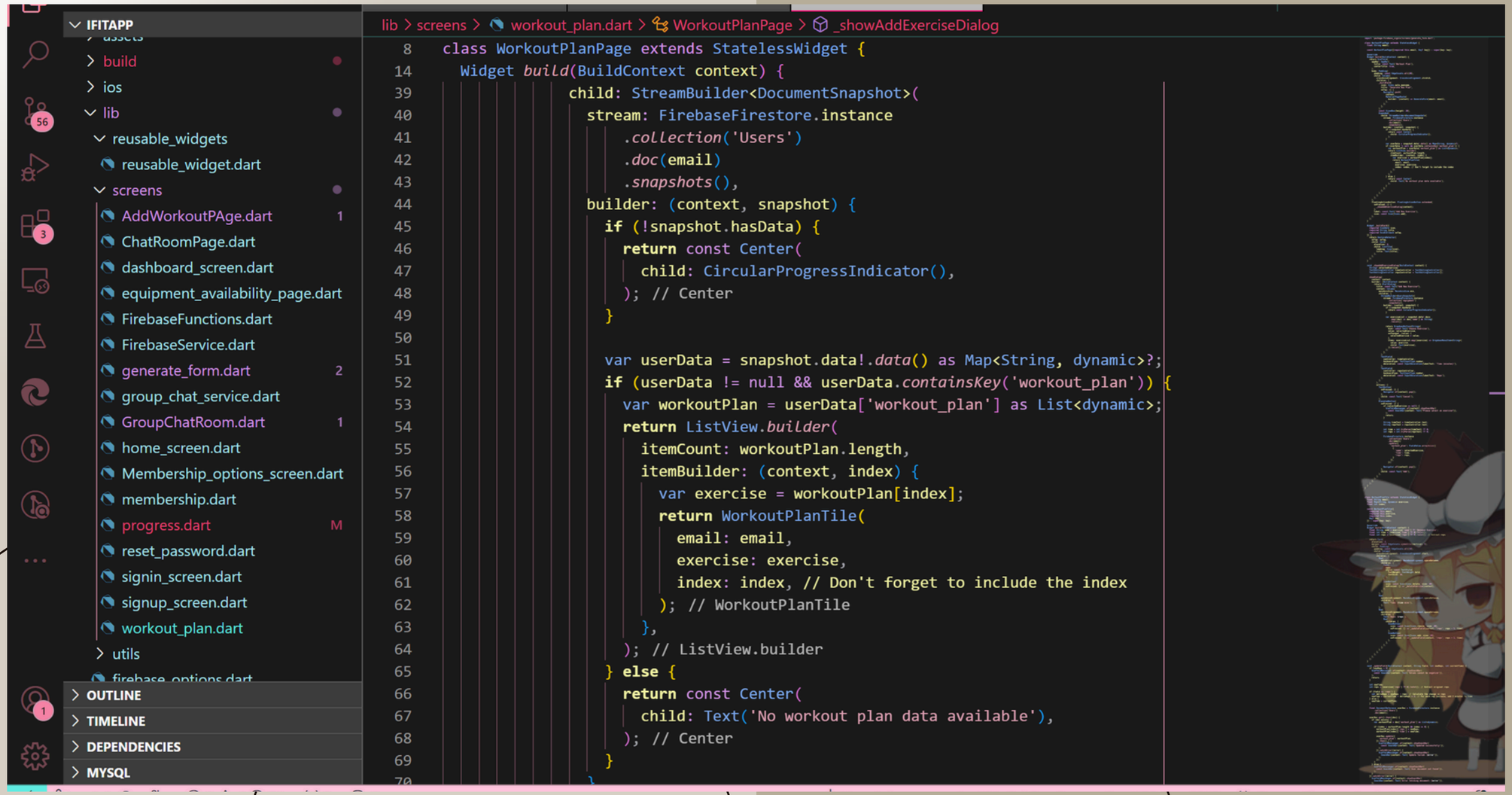
IMPLEMENTATION: APP



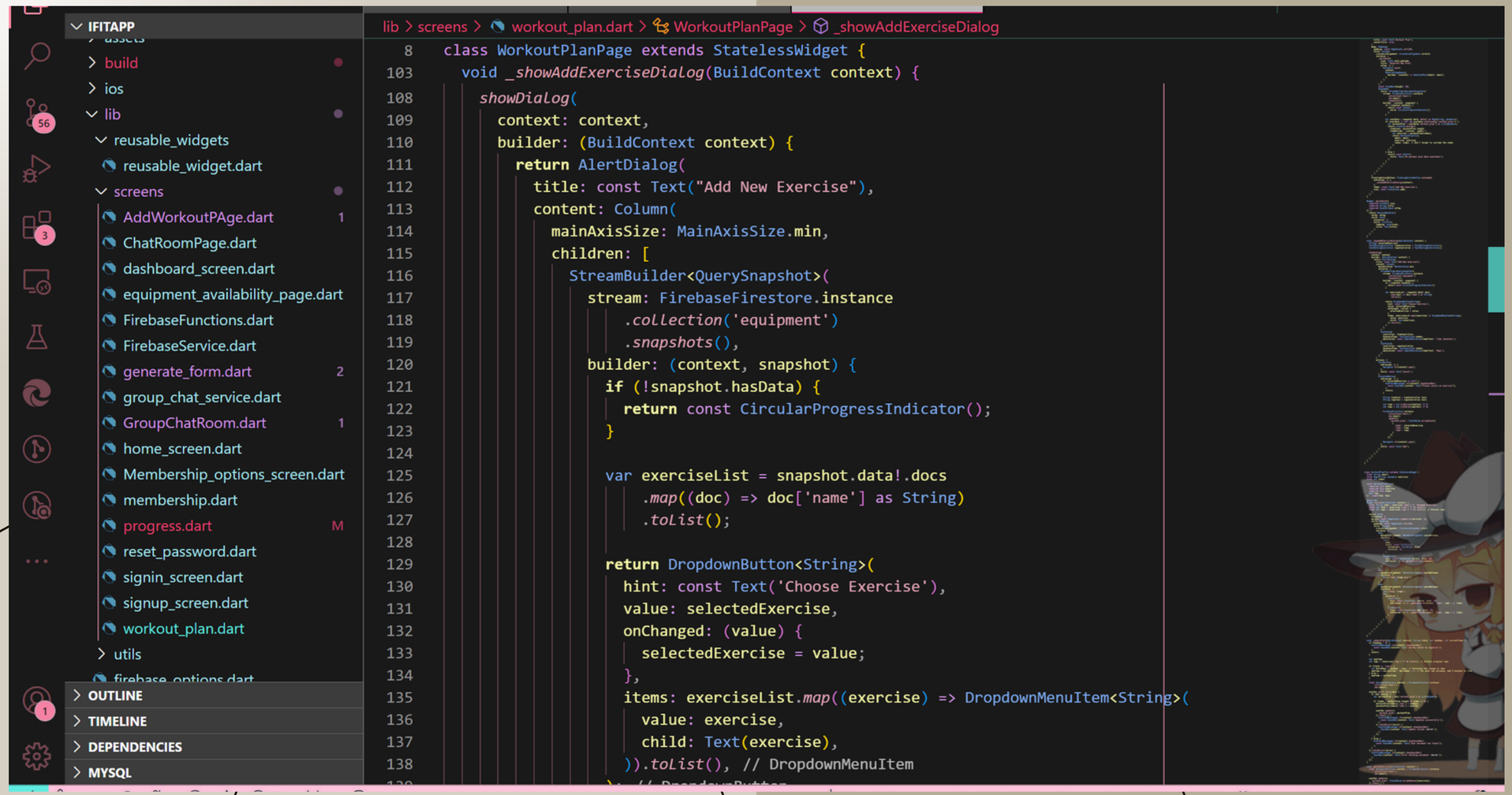
IMPLEMENTATION: APP



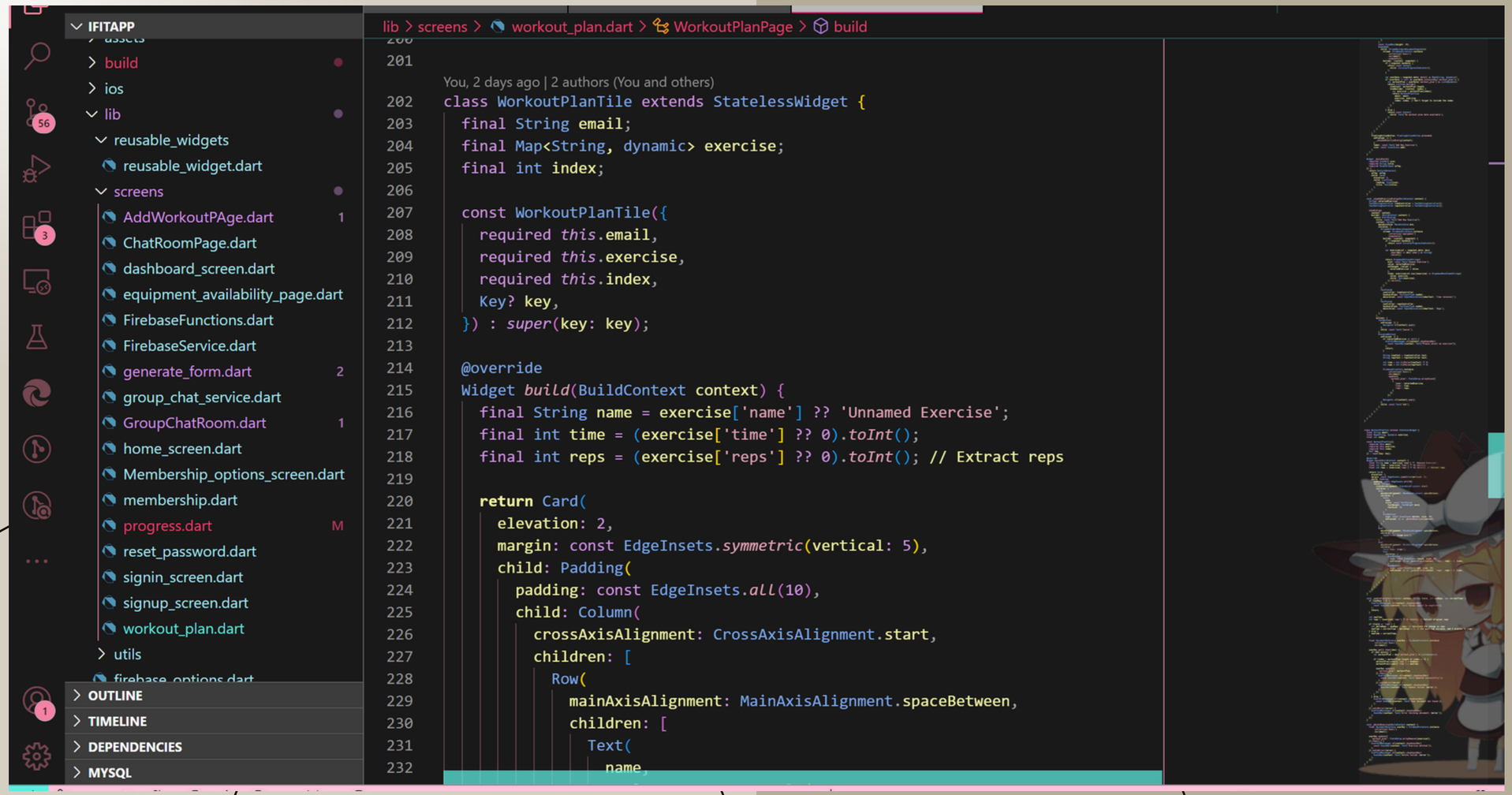
IMPLEMENTATION: APP



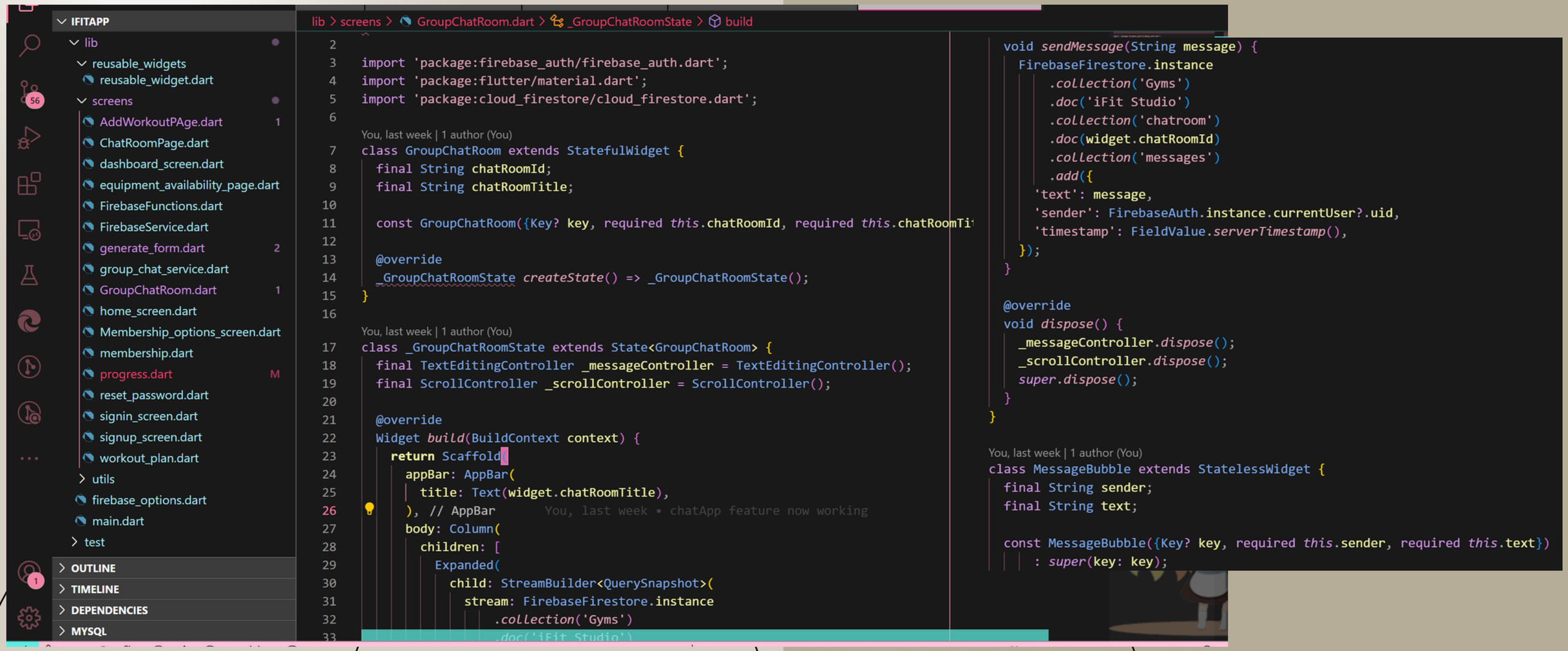
IMPLEMENTATION: APP



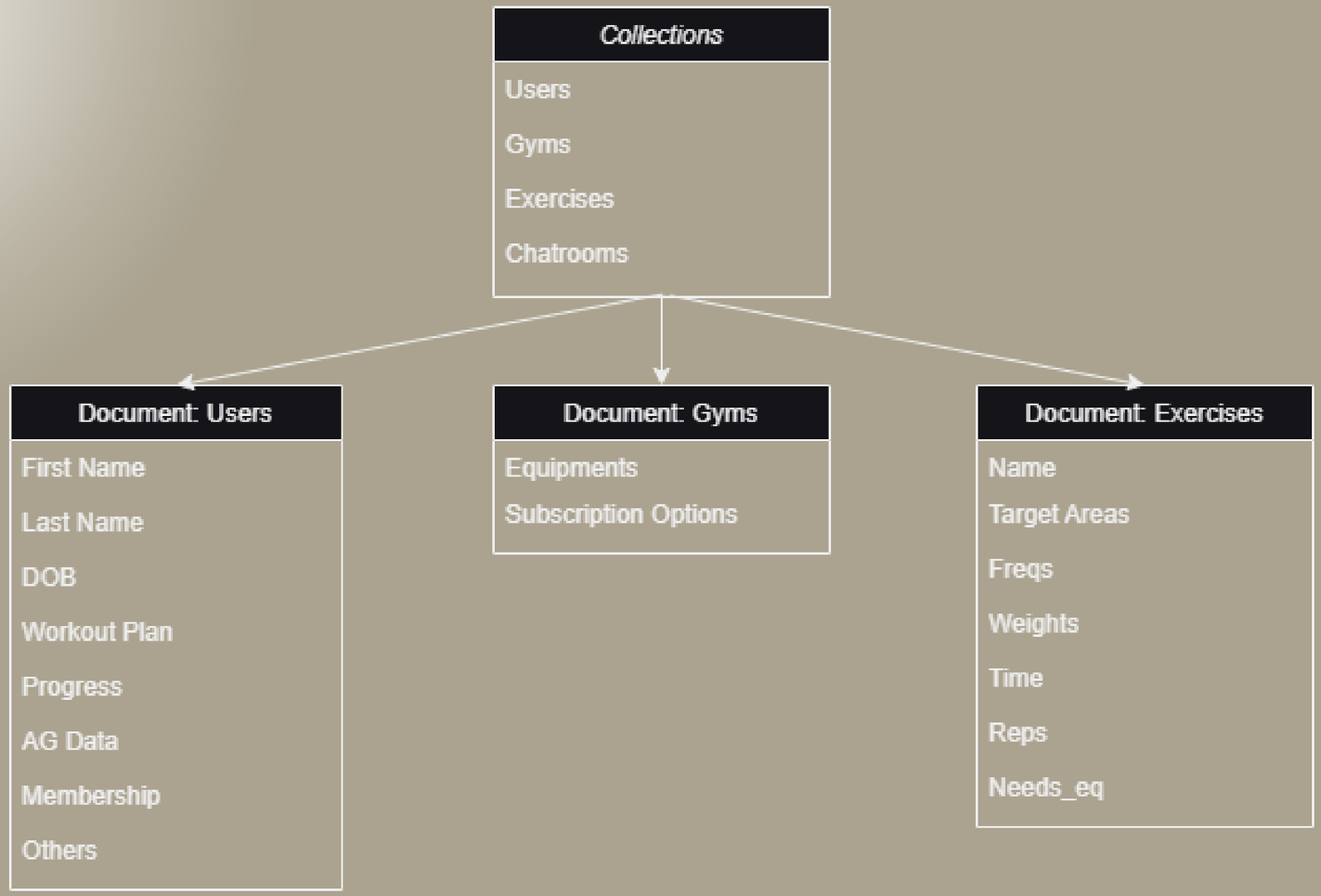
IMPLEMENTATION: APP



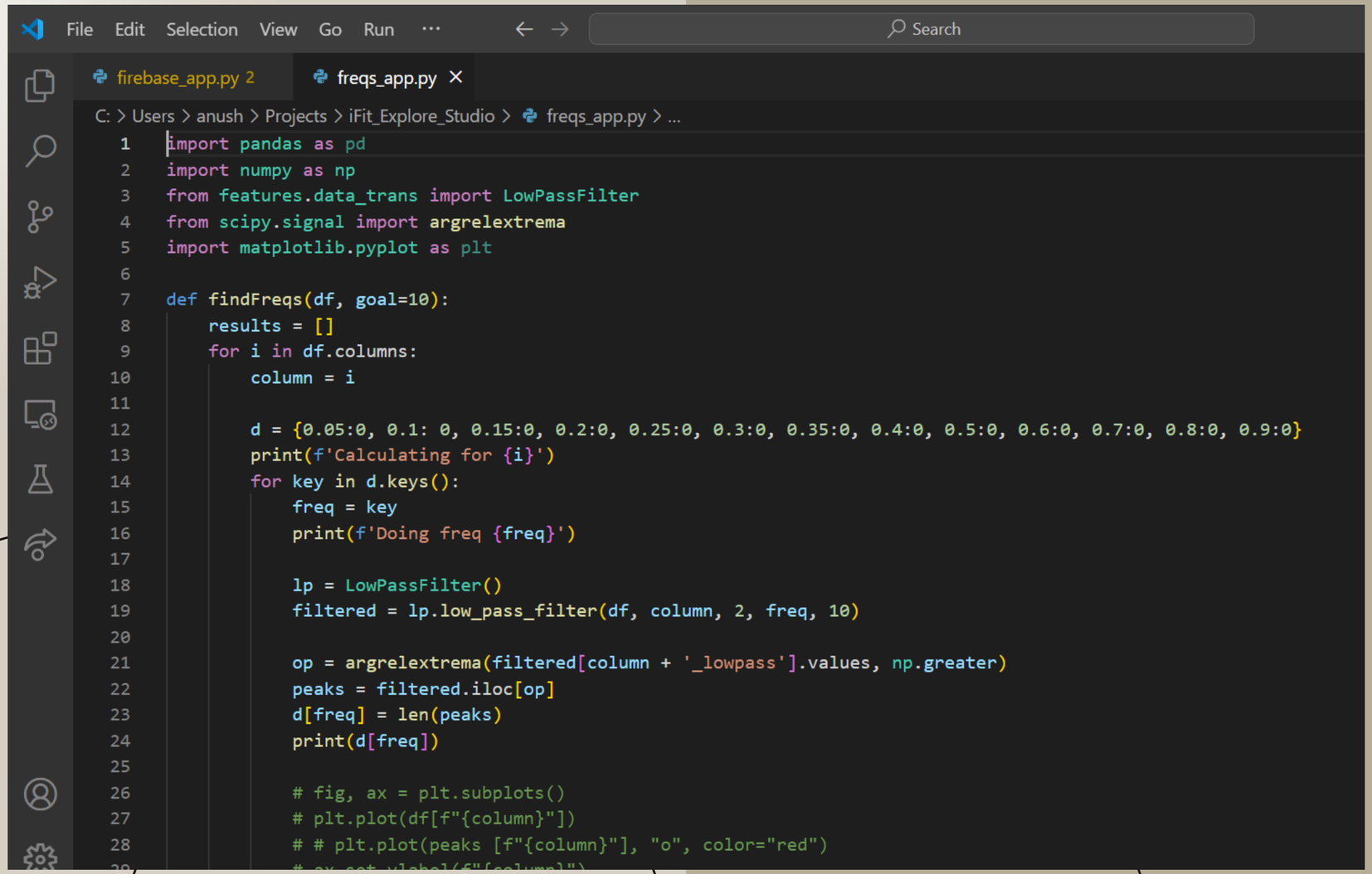
IMPLEMENTATION: APP



IMPLEMENTATION: DATA



IMPLEMENTATION: SERVER



IMPLEMENTATION: SERVER

File Edit Selection View Go Run ···

firebase_app.py 2 | freqs_app.py | weights_app.py

C: > Users > anush > Projects > iFit_Explore_Studio > weights_ap

```python
def findWeights(df, freqs, goal=10):
        # freq = freqs.get(col)
        # for key in freqs.keys():
            # print(f'comparing {key} and {col}')
        #     if key[0] == col[0]:
        #         freq = df[key]
        # print('checking for freq ', freq)
        filtered = lp.low_pass_filter(df, col, 2, freqs[col], 10)
        op = argrelextrema(filtered[col + '_lowpass'].values, np.greater)
        peaks = filtered.iloc[op]
        preds.append(len(peaks))
    print(preds)
    weights = []
    for i in preds:
        if i == goal:
            weights.append(1)
        else:
            weights.append(round(1/(abs(i - goal)+1), 2))
    weights = {'x_a':weights[0], 'y_a':weights[1], 'z_a':weights[2], 'x_g':weights[3], 'y_g':weights[4], 'z_g':weights[5]}
    return weights
```

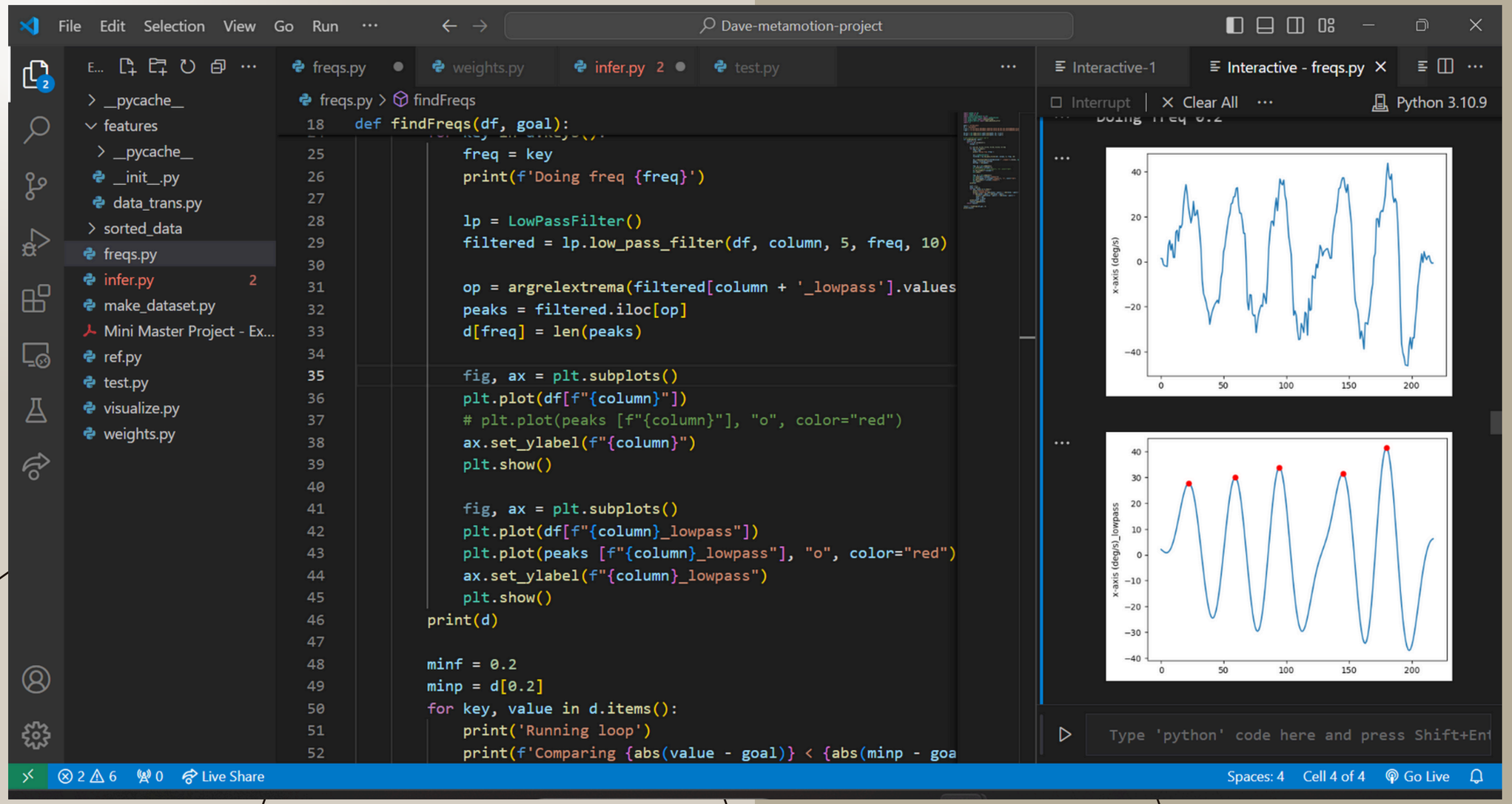
File Edit Selection View Go Run ··· Search

firebase_app.py 2 | freqs_app.py

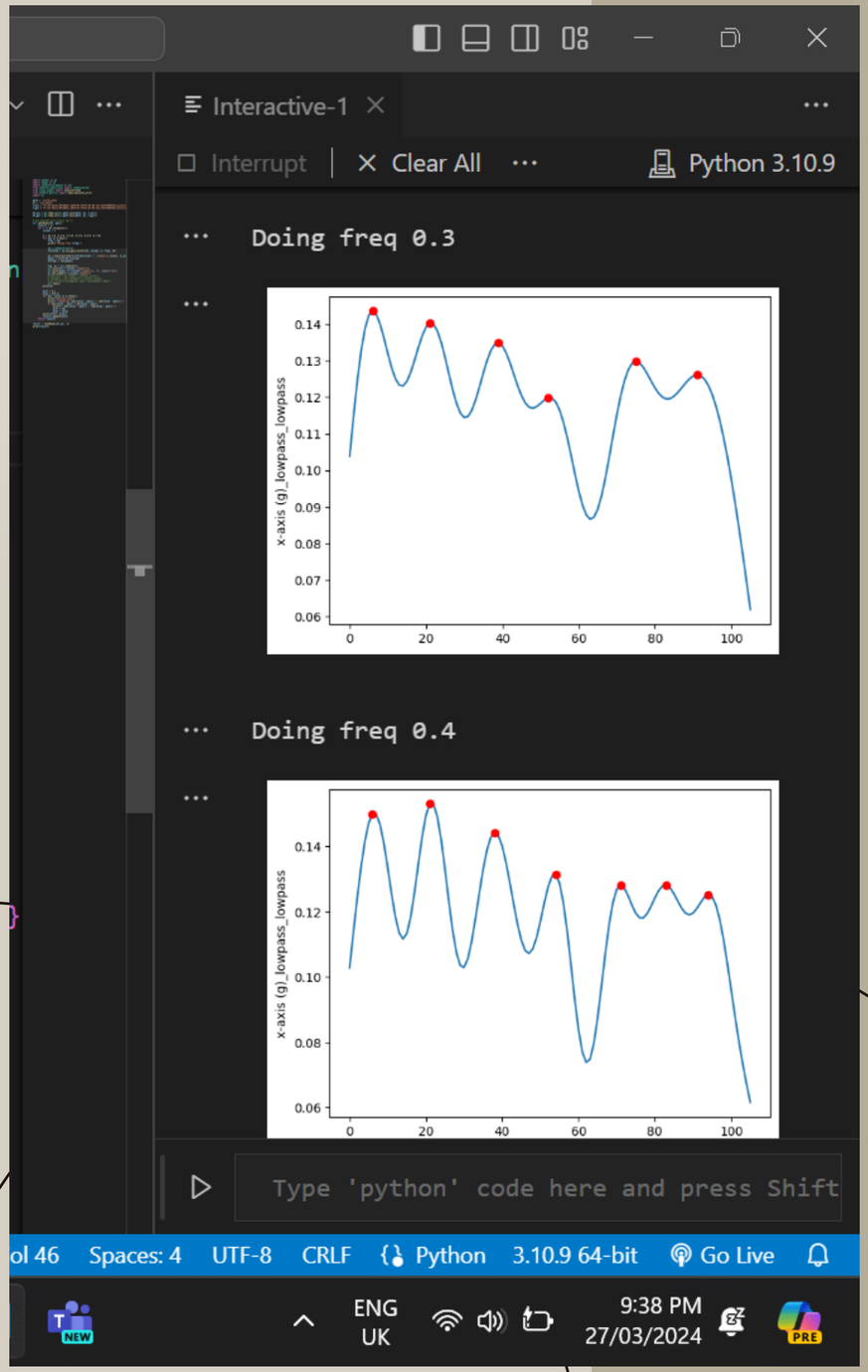
C: > Users > anush > Projects > iFit_Explore_Studio > freqs_app.py > ...

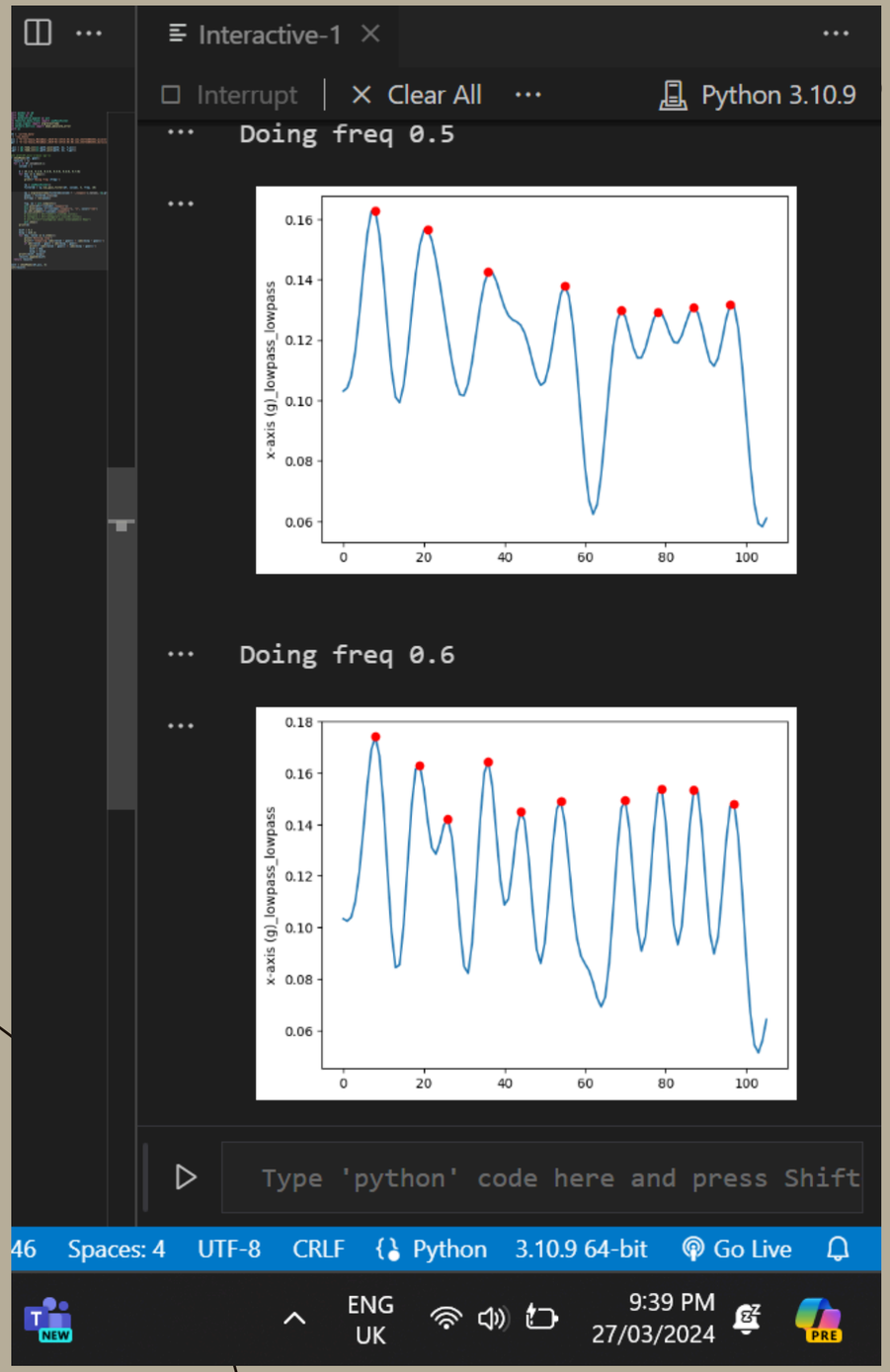
```python
def findFreqs(df, goal=10):
            # fig, ax = plt.subplots()
            # plt.plot(df[f"{column}_lowpass"])
            # plt.plot(peaks [f"{column}_lowpass"], "o", color="red")
            # ax.set_ylabel(f"{column}_lowpass")
            # plt.show()

        print(d)

        minf = 0.1
        minp = d[0.1]
        for key, value in d.items():
            print('Running loop')
            print(f'Comparing {abs(value - goal)} < {abs(minp - goal)}')
            if abs(value - goal) < abs(minp - goal):
                print(f'{abs(value - goal)} < {abs(minp - goal)}')
                minf = key
                minp = value
        print((minf, minp))
        results.append(minf)
    results = {'x_a':results[0], 'y_a':results[1], 'z_a':results[2], 'x_g':results[3], 'y_g':results[4], 'z_g':results[5]}
    return results
```

IMPLEMENTATION: SERVER

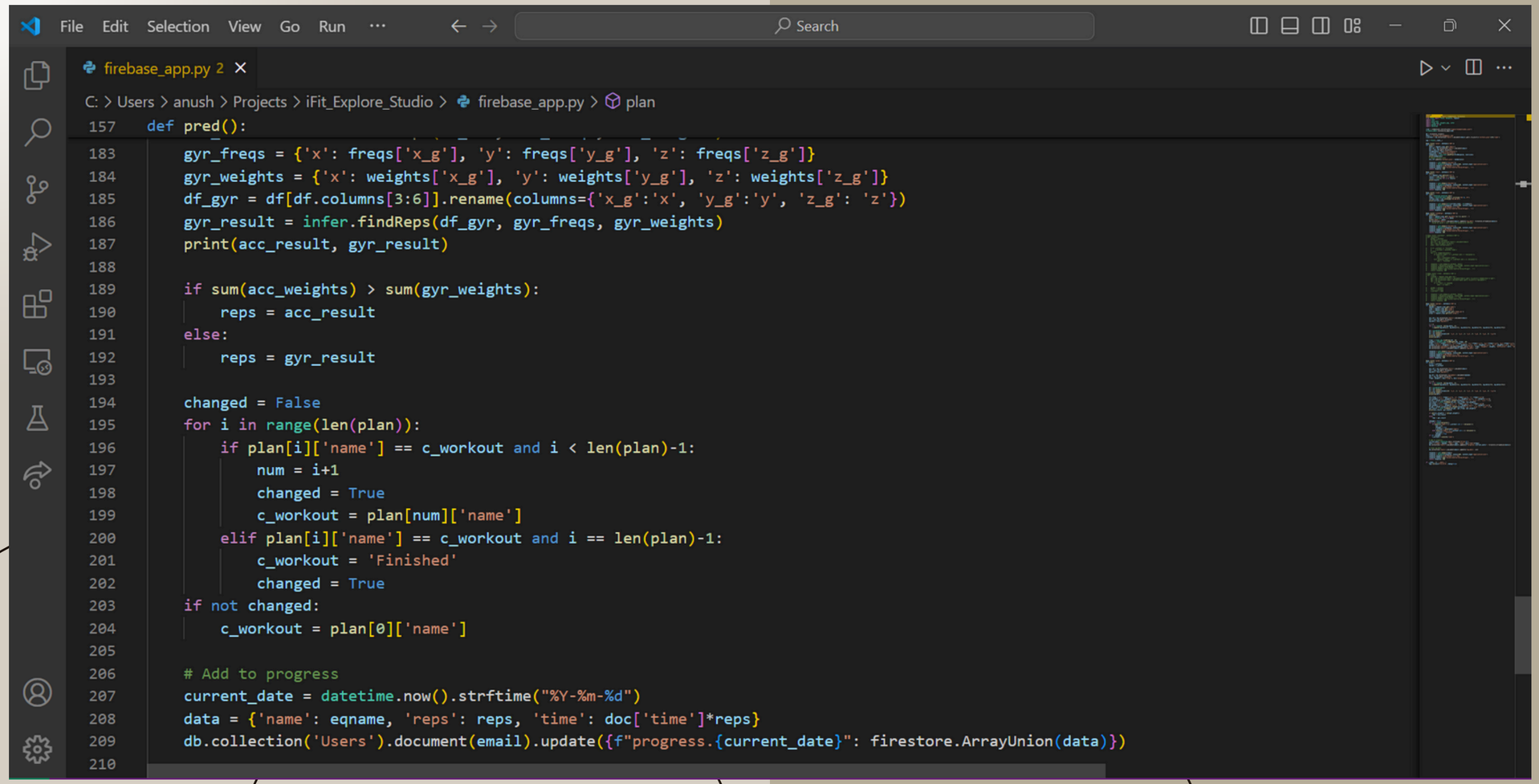


IMPLEMENTATION: SERVER



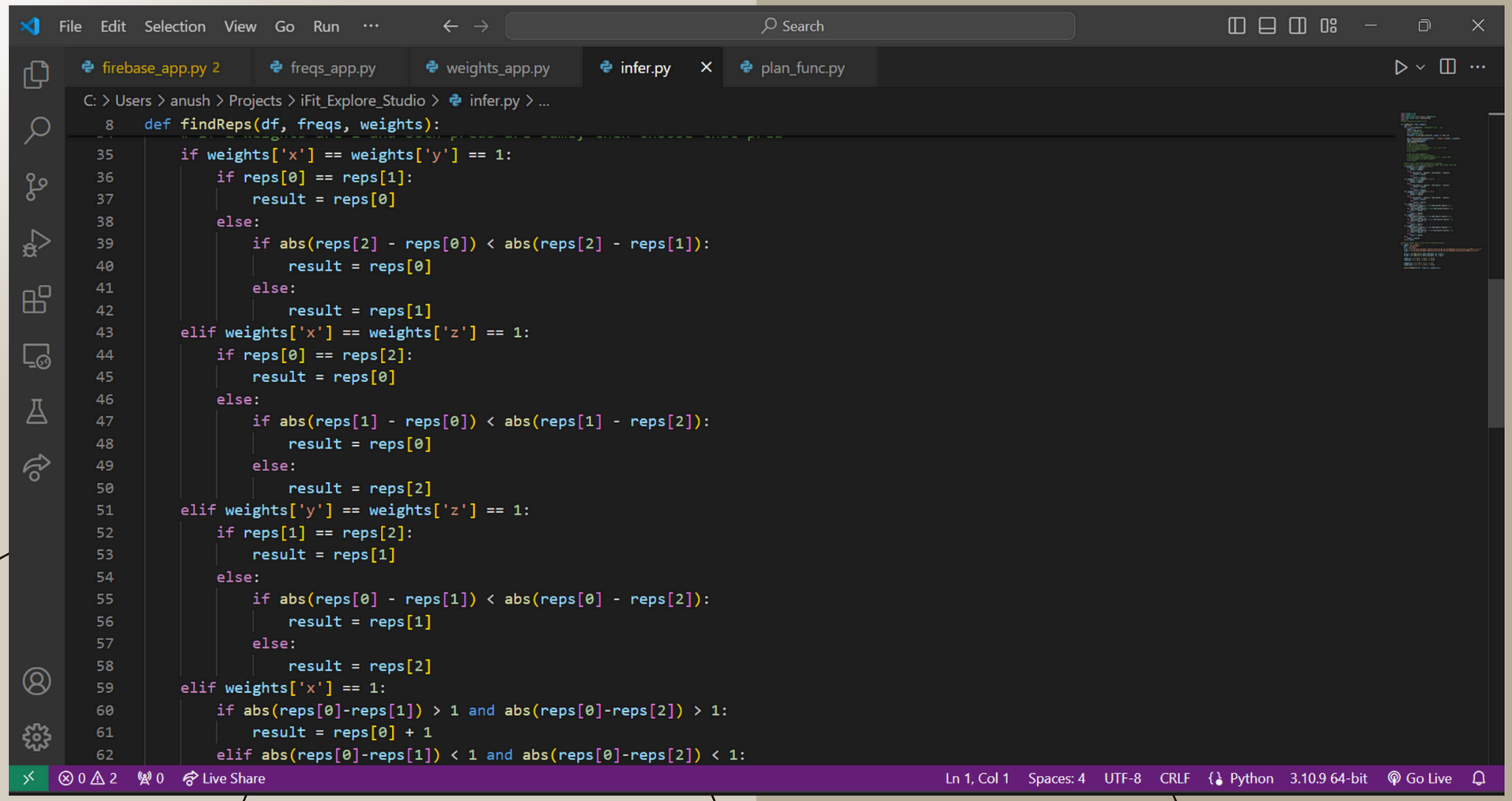


IMPLEMENTATION: SERVER

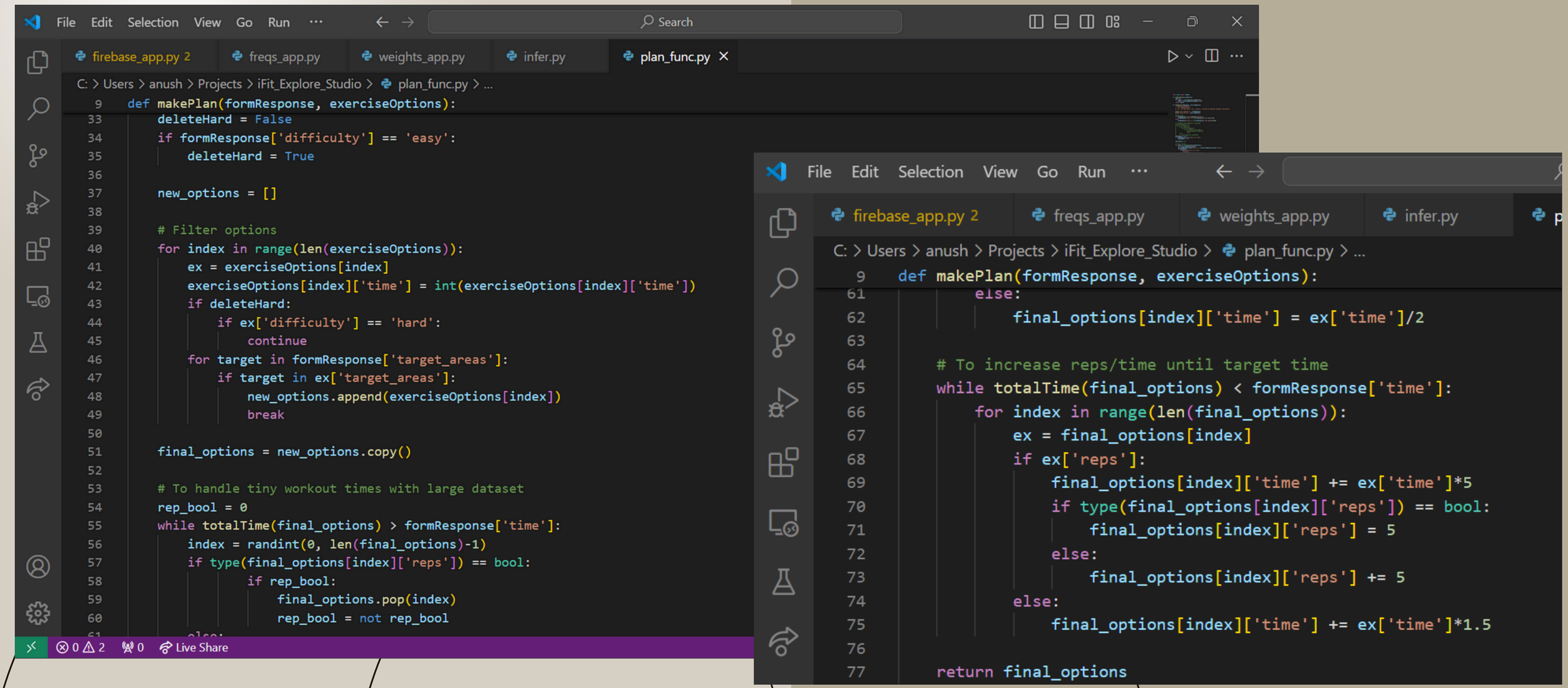



```python
def pred():
    gyr_freqs = {'x': freqs['x_g'], 'y': freqs['y_g'], 'z': freqs['z_g']}
    gyr_weights = {'x': weights['x_g'], 'y': weights['y_g'], 'z': weights['z_g']}
    df_gyr = df[df.columns[3:6]].rename(columns={'x_g':'x', 'y_g':'y', 'z_g': 'z'})
    gyr_result = infer.findReps(df_gyr, gyr_freqs, gyr_weights)
    print(acc_result, gyr_result)

    if sum(acc_weights) > sum(gyr_weights):
        reps = acc_result
    else:
        reps = gyr_result

    changed = False
    for i in range(len(plan)):
        if plan[i]['name'] == c_workout and i < len(plan)-1:
            num = i+1
            changed = True
            c_workout = plan[num]['name']
        elif plan[i]['name'] == c_workout and i == len(plan)-1:
            c_workout = 'Finished'
            changed = True
    if not changed:
        c_workout = plan[0]['name']

    # Add to progress
    current_date = datetime.now().strftime("%Y-%m-%d")
    data = {'name': eqname, 'reps': reps, 'time': doc['time']*reps}
    db.collection('Users').document(email).update({f"progress.{current_date}": firestore.ArrayUnion(data)})
```

IMPLEMENTATION: SERVER



IMPLEMENTATION: SERVER



IMPLEMENTATION: WATCH

main | Arduino IDE 2.3.2

File Edit Sketch Tools Help

ESP32 Wrover Module

main.ino

```ino
#include <WiFi.h>
#include <Wire.h>
#include <WiFiMulti.h>
#include <HTTPClient.h>
#include <SPI.h>
#include <TFT_eSPI.h>       // Hardware-specific library
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <stdio.h>

WiFiMulti wifiMulti;
TFT_eSPI tft = TFT_eSPI();  // Invoke custom library
Adafruit_MPU6050 mpu;

// Variables for timer and button state
unsigned long startTime = 0;
unsigned long elapsedTime = 0;
bool timerRunning = false;
bool buttonPressed = true;
float data[6];
char cdata[50];
unsigned int count_ag = 1;
// Define button pin
#define BUTTON_PIN_AG 15
```

main | Arduino IDE 2.3.2

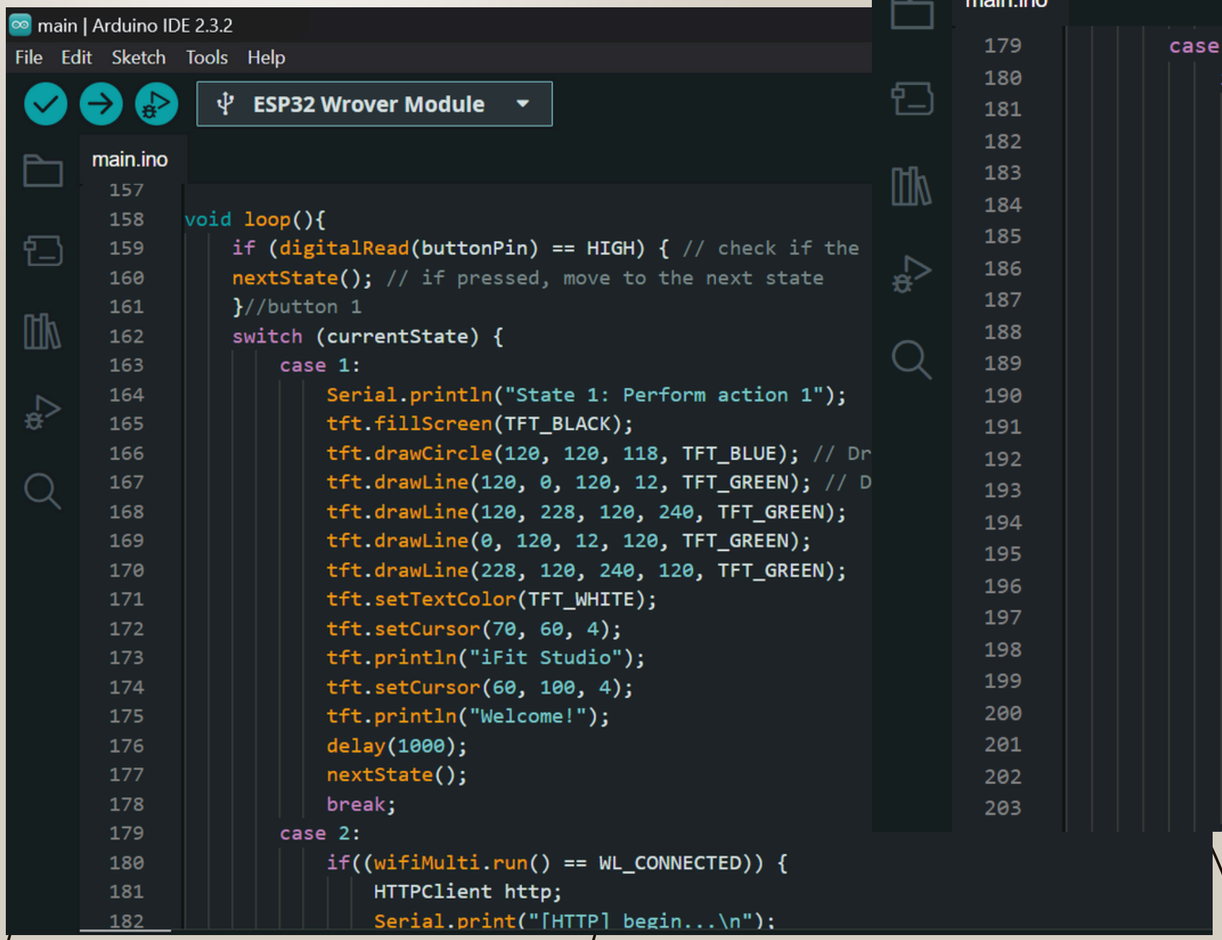
File Edit Sketch Tools Help

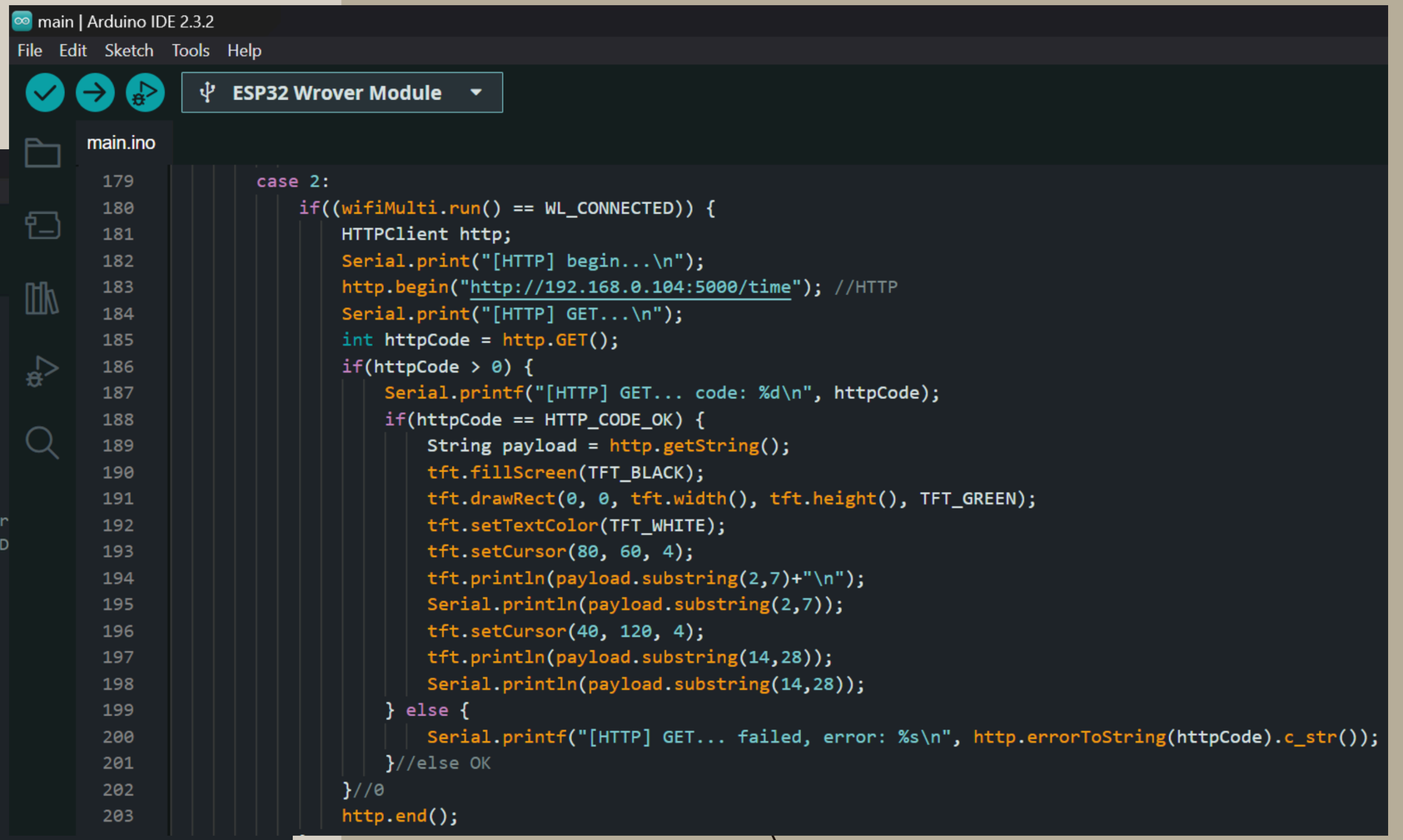
ESP32 Wrover Module

main.ino

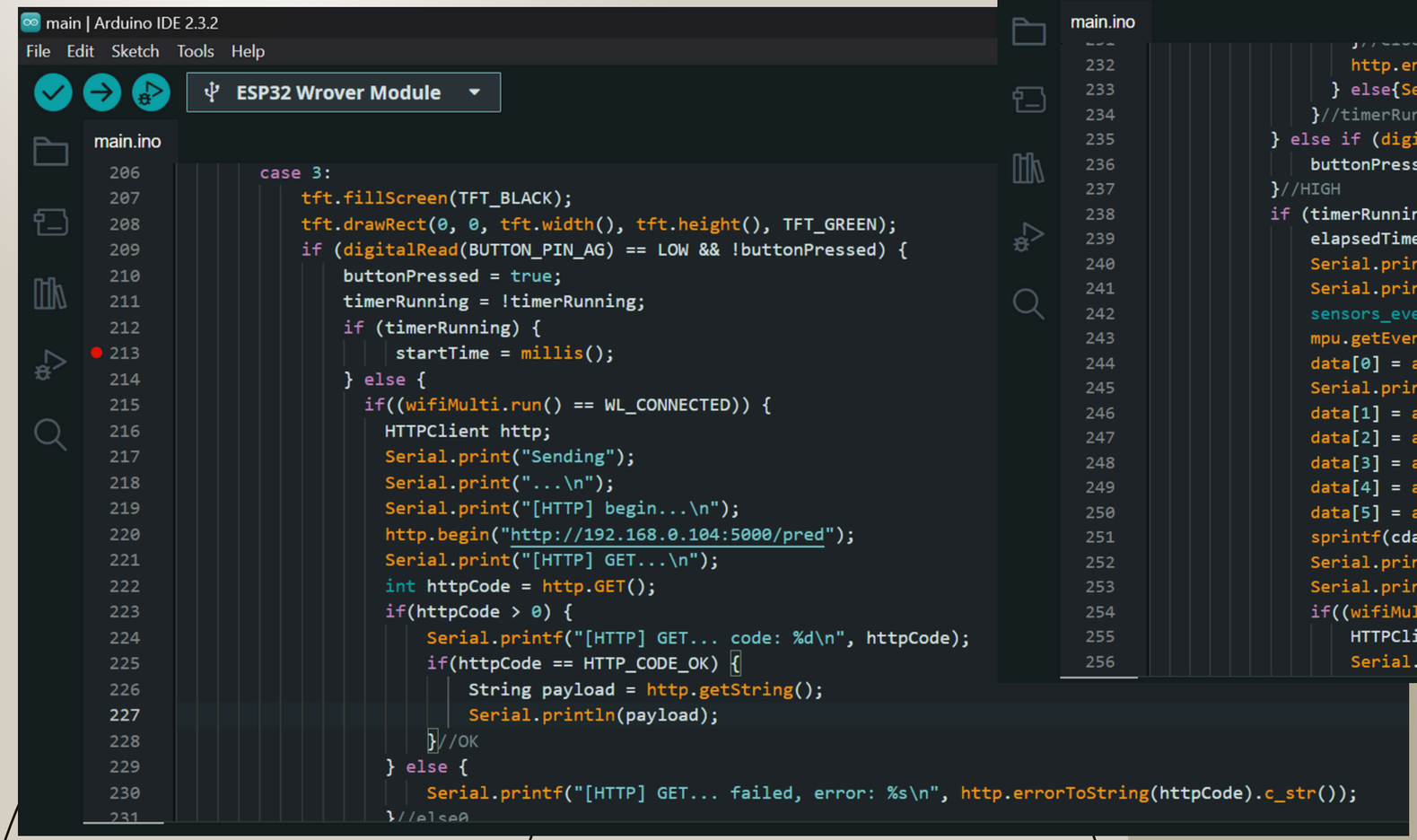
```ino
    }
    Serial.println("");
    tft.init();
    tft.fillScreen(TFT_BLACK);
    delay(100);
}

void nextState()  {
    currentState++; // move to the next state
    if (currentState > 5) { // if the current state exceeds the maximum state, reset to 1
        currentState = 1;
    }
    if (currentState == 3){
        if((wifiMulti.run() == WL_CONNECTED)) {
            HTTPClient http;
            Serial.print("[HTTP] begin...\n");
            String urlString = "http://192.168.1.6:5000/cworkout";
            http.begin(urlString.c_str()); //HTTP
            Serial.print("[HTTP] GET...\n");
            int httpCode = http.GET();
            if(httpCode > 0) {
                Serial.printf("[HTTP] GET... code: %d\n", httpCode);
                if(httpCode == HTTP_CODE_OK) {
                    String payload = http.getString();
                    Serial.println(payload.substring(2,5));
```

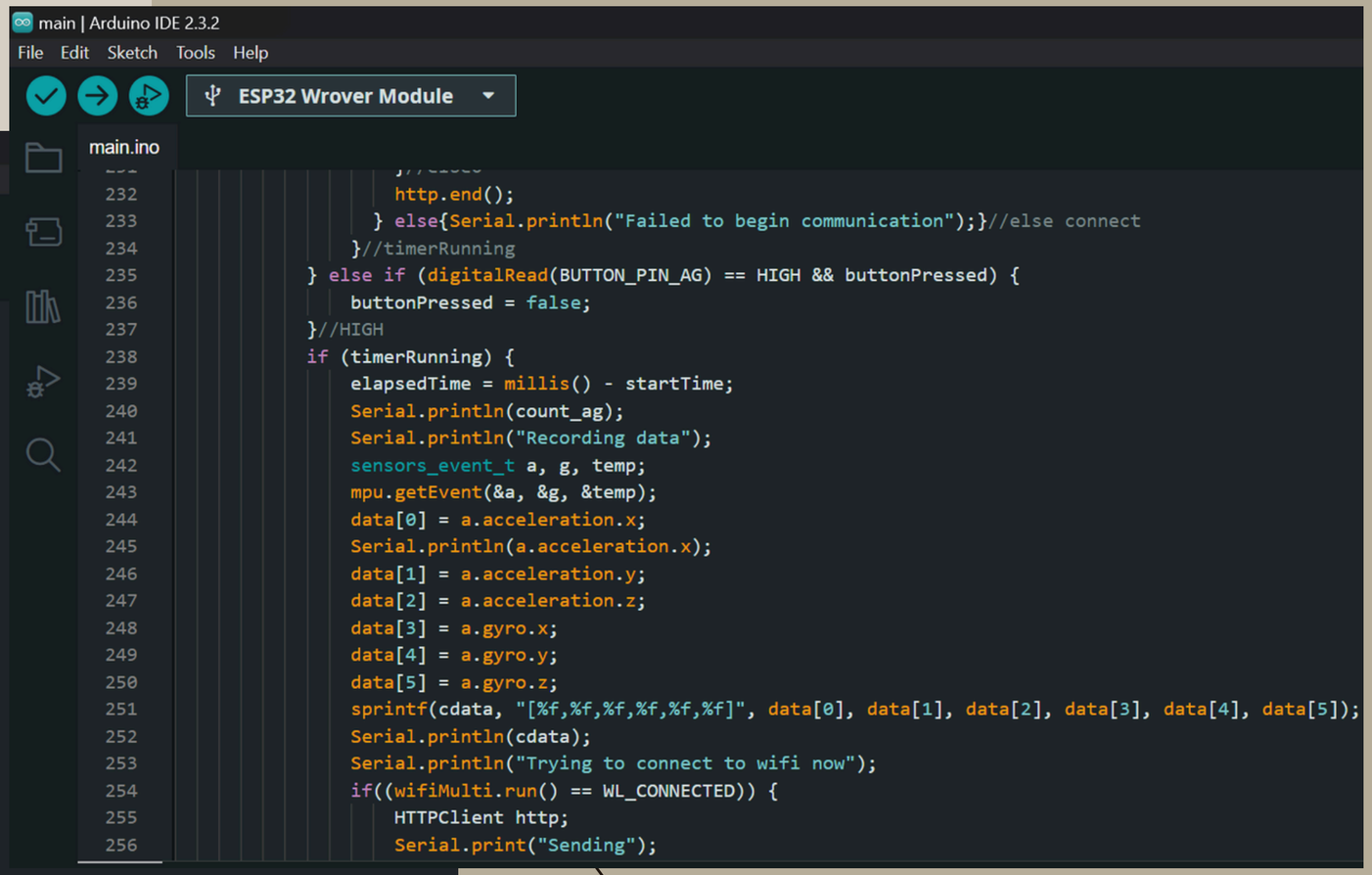
IMPLEMENTATION: WATCH





IMPLEMENTATION: WATCH





IMPLEMENTATION: WATCH

main | Arduino IDE 2.3.2

File Edit Sketch Tools Help

ESP32 Wrover Module

main.ino

```
sprintf(cdata, "[%f,%f,%f,%f,%f,%f]", data[0], data[1], data[2], data[3], data[4], data[5]);
Serial.println(cdata);
Serial.println("Trying to connect to wifi now");
if((wifiMulti.run() == WL_CONNECTED)) {
    HTTPClient http;
    Serial.print("Sending");
    Serial.print("...\n");
    Serial.print("[HTTP] begin...\n");
    String urlString = "http://192.168.0.104:5000/sendArgs?ag_data=" + String(cdata);
    http.begin(urlString.c_str());
    Serial.print("[HTTP] GET...\n");
    int httpCode = http.GET();
    if(httpCode > 0) {
        Serial.printf("[HTTP] GET... code: %d\n", httpCode);
        if(httpCode == HTTP_CODE_OK) {
            String payload = http.getString();
            Serial.println(payload);
        }//OK
    } else {
        Serial.printf("[HTTP] GET... failed, error: %s\n", http.errorToString(httpCode).c_str());
    }//else0
    http.end();
} else{Serial.println("Failed to begin communication");}//else connect
char timeString[16];
sprintf(timeString, "%02d:%02d", minutes, seconds);
```

Help

ESP32 Wrover Module

```
case 4:
    tft.fillScreen(TFT_BLACK);
    tft.drawRect(0, 0, tft.width(), tft.height(), TFT_GREEN);
    if (digitalRead(BUTTON_PIN_AG) == LOW && !buttonPressed) {
        buttonPressed = true;
        timerRunning = !timerRunning;
        if (timerRunning) {
            startTime = millis();
        } else {}
    } else if (digitalRead(BUTTON_PIN_AG) == HIGH && buttonPressed) {
        buttonPressed = false;
    }//elseHIGH
    if (timerRunning) {
        elapsedTime = millis() - startTime;
        Serial.println(count_ag);
        Serial.println("Recording data");
        sensors_event_t a, g, temp;
        mpu.getEvent(&a, &g, &temp);
        data[0] = a.acceleration.x;
        Serial.println(a.acceleration.x);
        data[1] = a.acceleration.y;
        data[2] = a.acceleration.z;
        data[3] = a.gyro.x;
        data[4] = a.gyro.y;
        data[5] = a.gyro.z;
```

DEMO VIDEO

https://drive.google.com/file/d/12qZoK2mq690vt8tCjXizODcbNkykJB8g/view?usp=drive_link

FUTURE PROSPECTS

This project opens up opportunities for further enhancements and refinements. We can continuously improve the system's functionality and user experience by leveraging data analytics and user feedback. Additionally, exploring partnerships and expanding the system's capabilities to support diverse workout routines and fitness goals will contribute to its long-term success.

5 main future prospects we would like to highlight are:

- Advancements in Hardware
- Influencer Workout led content
- UI/UX
- MedPALM Integration
- Compact PCB

THANK YOU

The development and implementation of the iFit project have resulted in a comprehensive IoT-based fitness user management system that integrates a smartwatch and a companion mobile application.

REFERENCES

1. Z. Liu, X. Liu and K. Li (2020). "Deeper Exercise Monitoring for Smart Gym using Fused RFID and CV Data". Available at: https://ieeexplore.ieee.org/document/9155360

2. Galetsi, P., Katsaliaki, K. and Kumar, S. (2022). “Exploring benefits and ethical challenges in the rise of mHealth (mobile healthcare) technology for the common good: An analysis of mobile applications for health specialists”. Available at: https://doi.org/10.1016/j.technovation.2022.102598

3. Dinesh Kumar, A., Bhargav, K., Rayal, R. and Saraswathi, M. (2020). “Smart Gym Management System. International Journal of Scientific Research & Engineering Trends”. Available at: https://ijsret.com/wp-content/uploads/2020/05/IJSRET_V6_issue3_493.pdf

4. Gubbi J, Buyya R, Marusic S, Palaniswami M (2013). “Internet of things (IoT): A Aision, Architectural Elements, and Future Directions". Available at:https://www.sciencedirect.com/science/article/pii/S0167739X13000241

5. Bhatia M, Sood SK (2018). “An intelligent framework for workouts in gymnasium: M-health perspective”. Available at: https://www.sciencedirect.com/science/article/pii/S004579061732236X

6. Thompson, Walter R. Ph.D., FACSM - “Worldwide Survey of Fitness Trends for 2022”. Available at: https://journals.lww.com/acsm-healthfitness/fulltext/2022/01000/worldwide_survey_of_fitness_trends_for_2022.6.aspx

7. Binbin Yong, Zijian Xu, Xin Wang, Libin Cheng, Xue Li, Xiang Wu, Qingguo Zhou - “IoT-based intelligent fitness system”. Available at: https://www.sciencedirect.com/science/article/pii/S0743731517301570

8. Lyons, Elizabeth J. Ph.D., M.P.H.; Swartz, Maria C. Ph.D., M.P.H., RD. (2017). “Motivational Dynamics of Wearable Activity Monitors”. Available at: https://journals.lww.com/acsm-healthfitness/fulltext/2017/09000/motivational_dynamics_of_wearable_activity.8.aspx+ more