

**MACHINE LEARNING APPROACH TO DETECT &
ANNOTATE EYE DISEASES USING RETINAL IMAGES**

2023-162

Status Document II

Kariyawasam K.G.P.C

IT20172978

B.Sc. (Hons) Degree in Information Technology
Specializing in Software Engineering

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

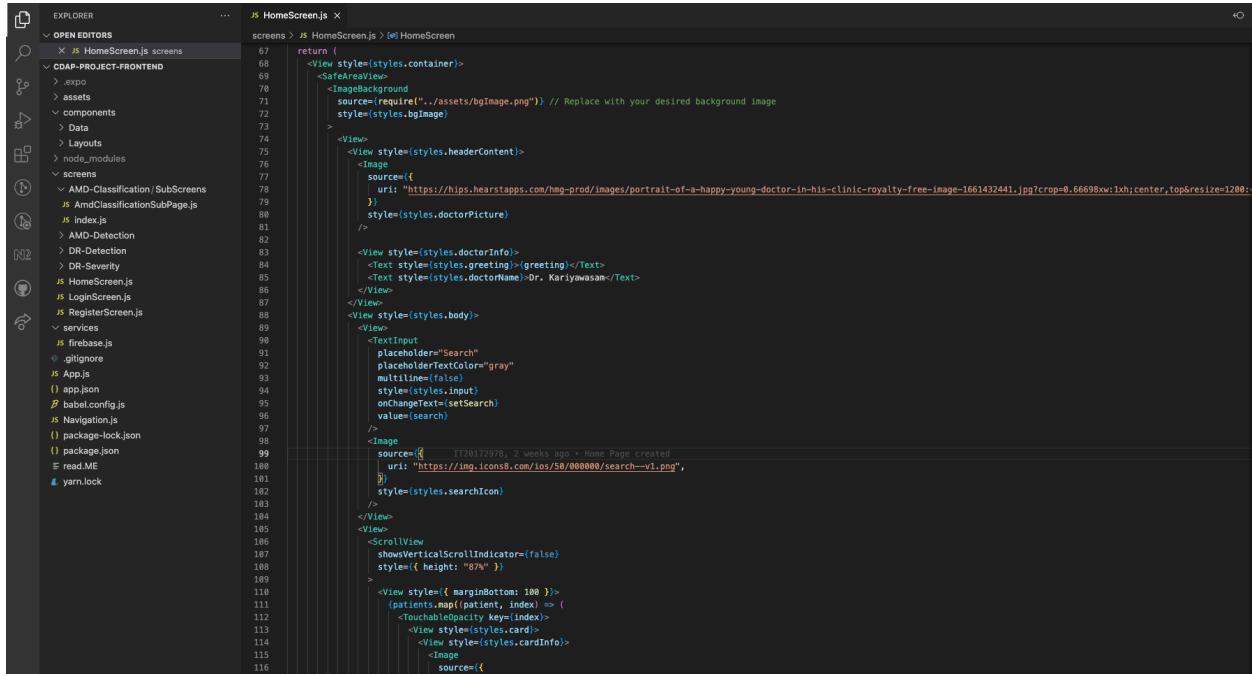
May 2023

Table of Contents

1. Project progress.....	3
1.1 Project Frontend.....	3
1.2 Project Backend.....	4
1.3 Mobile Application UIs.....	5
2. Project View.....	6
3. Updated Gantt chart.....	8
4. Screenshots of chats and calls of MS Teams.....	9

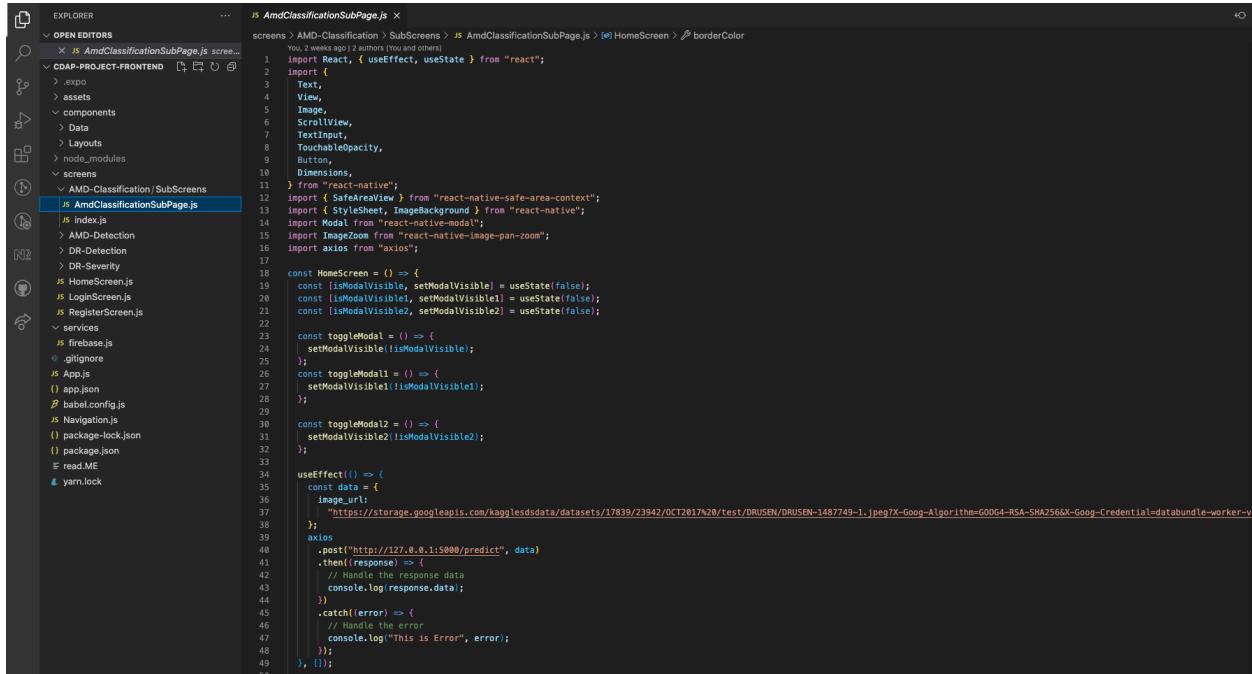
1. Project progress

1.1 Project Frontend



```
JS HomeScreen.js
screens > HomeScreen.js > (e) HomeScreen
67   return (
68     <SafeAreaView>
69       <ImageBackground
70         source={require("../assets/bgImage.png")} // Replace with your desired background image
71         style={styles.bgImage}
72       >
73         <View>
74           <View style={styles.headerContent}>
75             <Image
76               source={
77                 {
78                   uri: "https://hips.hearstapps.com/hmg-prod/images/portrait-of-a-happy-young-doctor-in-his-clinic-royalty-free-image-1661432441.jpg?crop=0.66698x:1x;center,top&resize=1200:1600"
79                 }
80               }
81             style={styles.doctorPicture}
82           />
83           <View style={styles.doctorInfo}>
84             <Text style={styles.greeting}>{greeting}</Text>
85             <Text style={styles.doctorName}>Dr. Kariyawasam</Text>
86           </View>
87         </View>
88         <View style={styles.body}>
89           <View>
90             <TextInput
91               placeholder="Search"
92               placeholderTextColor="gray"
93               multiline={false}
94               style={styles.input}
95               onChangeText={setSearch}
96               value={search}
97             />
98             <Image
99               source={(
100                 {
101                   uri: "https://img.icons8.com/ios/50/000000/search-v1.png"
102                 }
103               )
104             }
105             style={styles.searchIcon}
106           />
107         </View>
108         <ScrollView
109           showsVerticalScrollIndicator={false}
110           style={{
111             height: "87%"
112           }}
113         >
114           <View style={{ marginBottom: 100 }}>
115             {patients.map((patient, index) => (
116               <Touchableopacity key={index}>
117                 <View style={styles.card}>
118                   <View style={styles.cardInfo}>
119                     <Image
120                       source={(
121                         {
122                           uri: "https://storage.googleapis.com/kaggledsdata/datasets/17839/23942/OCT2017%20/test/DRUSEN/DRUSEN-1487749-1.jpeg?X-Google-Algorithm=G00G4-RSA-SHA256&X-Google-Credential=databundle-worker-v1"
123                         }
124                       )
125                     }
126                   </View>
127                 </View>
128               </Touchableopacity>
129             );
130           );
131         </View>
132       </SafeAreaView>
133     );
134   );
135   import Modal from "react-native-modal";
136   import ImageZoom from "react-native-image-pan-zoom";
137   import axios from "axios";
138
139   const HomeScreen = () => {
140     const [isModalVisible, setModalVisible] = useState(false);
141     const [isModalVisible2, setModalVisible2] = useState(false);
142     const [isModalVisible3, setModalVisible3] = useState(false);
143
144     const toggleModal = () => {
145       setModalVisible(!isModalVisible);
146     };
147     const toggleModal2 = () => {
148       setModalVisible2(!isModalVisible2);
149     };
150     const toggleModal3 = () => {
151       setModalVisible3(!isModalVisible3);
152     };
153
154     useEffect(() => {
155       const data = {
156         image_url:
157           "https://storage.googleapis.com/kaggledsdata/datasets/17839/23942/OCT2017%20/test/DRUSEN/DRUSEN-1487749-1.jpeg?X-Google-Algorithm=G00G4-RSA-SHA256&X-Google-Credential=databundle-worker-v1"
158       };
159       axios
160         .post("http://127.0.0.1:5000/predict", data)
161         .then((response) => {
162           // Handle the response data
163           console.log(response.data);
164         })
165         .catch((error) => {
166           // Handle the error
167           console.log("This is Error", error);
168         });
169     }, []);
170   
```

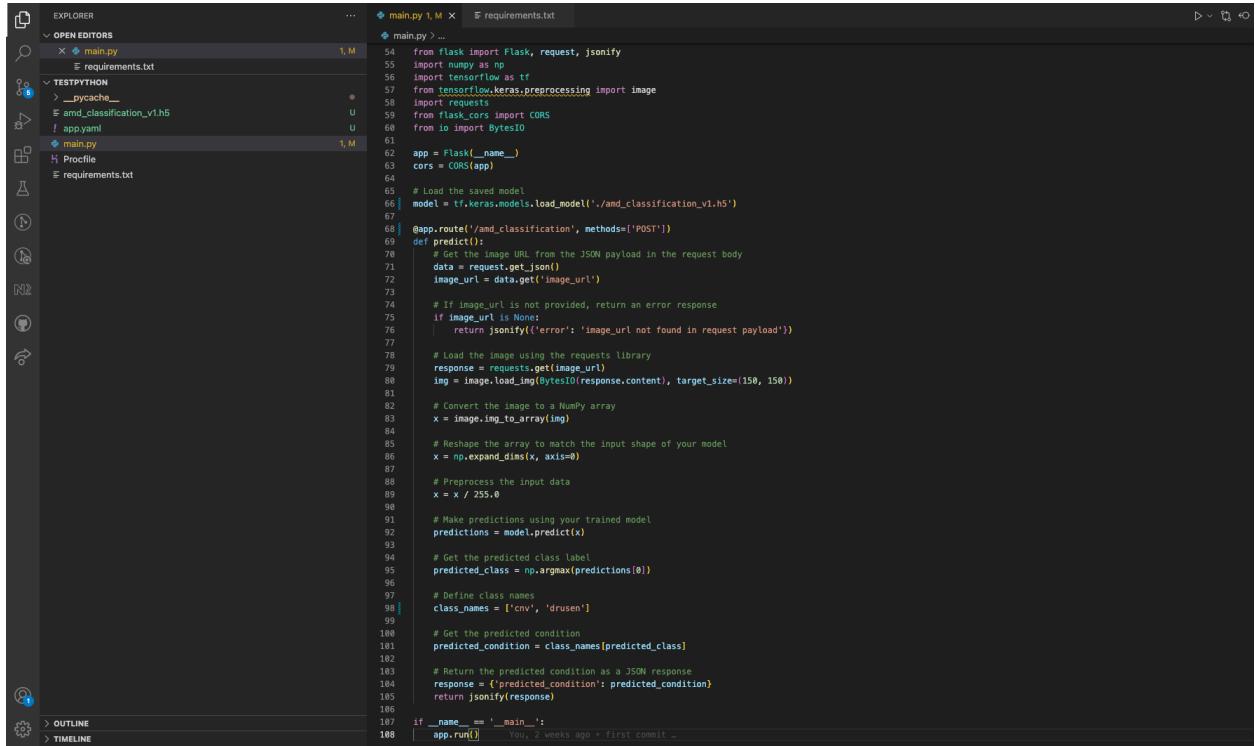
Figure 01: Mobile Application Home Frontend code



```
JS AmdClassificationSubPage.js
screens > AMD-Classification > SubScreens > JS AmdClassificationSubPage.js > (e) HomeScreen > borderColor
You, 2 weeks ago | 2 authors (You and others)
1   import React, { useEffect, useState } from "react";
2   import {
3     Text,
4     View,
5     Image,
6     ScrollView,
7     TextInput,
8     Touchableopacity,
9     Button,
10    Dimensions,
11  } from "react-native";
12  import { SafeAreaView } from "react-native-safe-area-context";
13  import { ImageBackground } from "react-native";
14  import Modal from "react-native-modal";
15  import ImageZoom from "react-native-image-pan-zoom";
16  import axios from "axios";
17
18  const HomeScreen = () => {
19    const [isModalVisible, setModalVisible] = useState(false);
20    const [isModalVisible2, setModalVisible2] = useState(false);
21    const [isModalVisible3, setModalVisible3] = useState(false);
22
23    const toggleModal = () => {
24      setModalVisible(!isModalVisible);
25    };
26    const toggleModal2 = () => {
27      setModalVisible2(!isModalVisible2);
28    };
29    const toggleModal3 = () => {
30      setModalVisible3(!isModalVisible3);
31    };
32
33    useEffect(() => {
34      const data = {
35        image_url:
36          "https://storage.googleapis.com/kaggledsdata/datasets/17839/23942/OCT2017%20/test/DRUSEN/DRUSEN-1487749-1.jpeg?X-Google-Algorithm=G00G4-RSA-SHA256&X-Google-Credential=databundle-worker-v1"
37      };
38      axios
39        .post("http://127.0.0.1:5000/predict", data)
40        .then((response) => {
41          // Handle the response data
42          console.log(response.data);
43        })
44        .catch((error) => {
45          // Handle the error
46          console.log("This is Error", error);
47        });
48      }, []);
49    
```

Figure 02: Mobile Application AMD Classification Frontend code

1.2 Project Backend



The screenshot shows a code editor interface with the following details:

- EXPLORER**: Shows files: requirements.txt, main.py, amd_classification_v1.h5, app.yaml, main.py, and requirements.txt.
- OPEN EDITORS**: Displays two tabs: main.py (active) and requirements.txt.
- TESTPYTHON**: Shows a single file: __pycache__.
- Profile**: Shows a single file: requirements.txt.
- Code Editor Content (main.py):**

```
54 from flask import Flask, request, jsonify
55 import numpy as np
56 import tensorflow as tf
57 from tensorflow.keras.preprocessing import image
58 import requests
59 from flask_cors import CORS
60 from io import BytesIO
61
62 app = Flask(__name__)
63 cors = CORS(app)
64
65 # Load the saved model
66 model = tf.keras.models.load_model('./amd_classification_v1.h5')
67
68 @app.route('/amd_classification', methods=['POST'])
69 def predict():
70     # Get the image URL from the JSON payload in the request body
71     data = request.get_json()
72     image_url = data.get('image_url')
73
74     # If image_url is not provided, return an error response
75     if image_url is None:
76         return jsonify({'error': 'image_url not found in request payload'})
77
78     # Load the image using the requests library
79     response = requests.get(image_url)
80     img = image.load_img(BytesIO(response.content), target_size=(150, 150))
81
82     # Convert the image to a NumPy array
83     x = image.img_to_array(img)
84
85     # Reshape the array to match the input shape of your model
86     x = np.expand_dims(x, axis=0)
87
88     # Preprocess the input data
89     x = x / 255.0
90
91     # Make predictions using your trained model
92     predictions = model.predict(x)
93
94     # Get the predicted class label
95     predicted_class = np.argmax(predictions[0])
96
97     # Define class names
98     class_names = ['cnv', 'drusen']
99
100    # Get the predicted condition
101    predicted_condition = class_names[predicted_class]
102
103    # Return the predicted condition as a JSON response
104    response = {'predicted_condition': predicted_condition}
105    return jsonify(response)
106
107 if __name__ == '__main__':
108     app.run()
```

Figure 03: Mobile Application AMD Classification Backend code

1.3 Mobile Application UIs

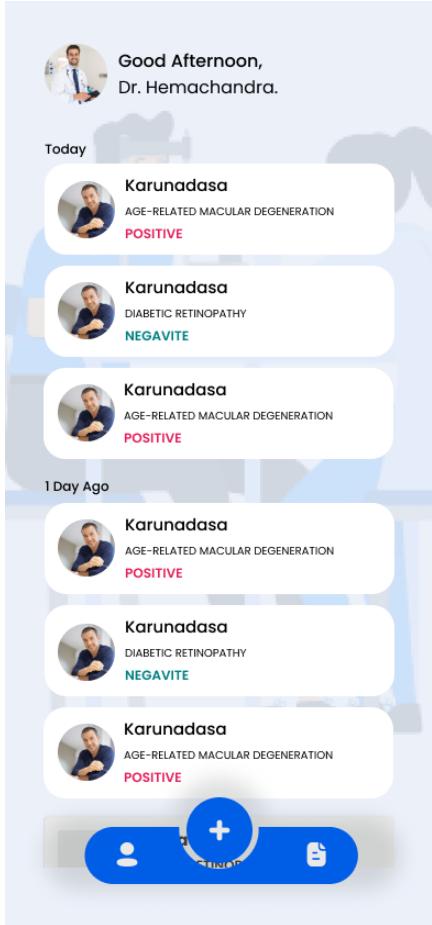


Figure 04: Mobile Application Home Page

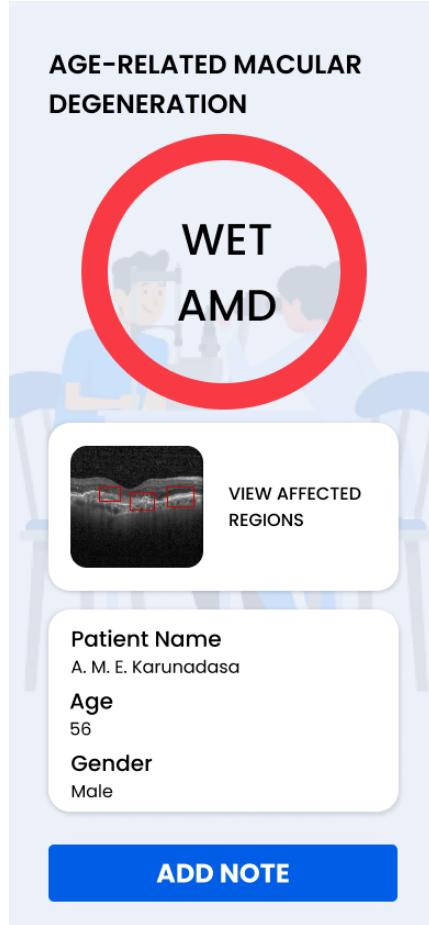


Figure 05: Mobile Application AMD Classification Page

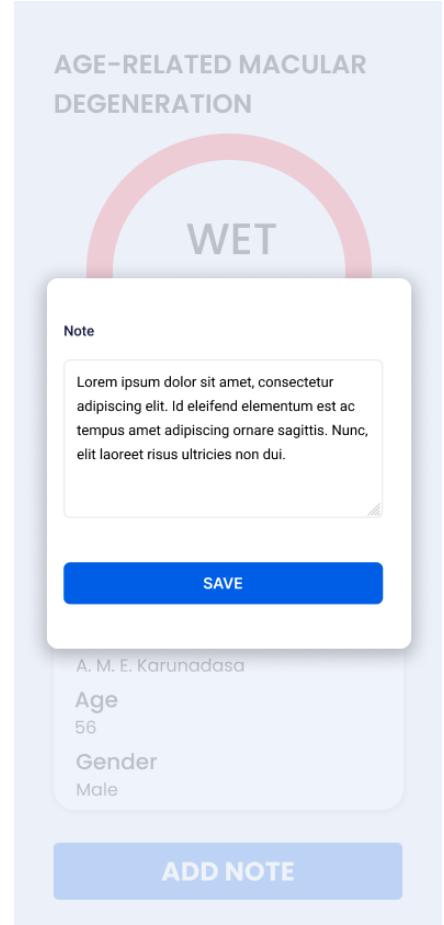


Figure 06: Mobile Application AMD Classification Add Note Popup

2. Project View

The screenshot shows the Planner - Board View for a project titled "23-162_DETECT_AND_ANNOТА...". The interface is divided into four columns, each representing a team member: Nilaksha, Praveen, Rasanga, and Chamod. Each column contains a list of tasks with their due dates and status indicators (green checkmark for completed, red X for late). The tasks are categorized under General.

Team Member	Task Description	Due Date	Status	
Nilaksha	Testing the Application	07/23	Completed	
	Integrate the Trained Model	07/22	Completed	
	Implement Functionality of Application using React Native	07/14	Completed	
	Implement UI using React Native	07/10	Completed	
	Design Mobile Application Interfaces	07/09	Completed	
	Train the Model	07/17	Late	
	Design and Implement the Model	07/16	Completed	
Praveen	Testing the Application	07/23	Late	
	Integrate the Trained Model	07/22	Late	
	Implement Functionality of Application using React Native	07/14	Late	
	Implement UI using React Native	07/10	Late	
	Design Mobile Application Interfaces	07/10	Late	
	Rasanga	Testing the Application	07/23	Late
		Integrate the Trained Model	07/22	Late
Implement Functionality of Application using React Native		07/14	Late	
Implement UI using React Native		07/10	Late	
Design Mobile Application Interfaces		07/10	Late	
Chamod		Testing the Application	07/23	Late
		Integrate the Trained Model	07/22	Late
	Implement Functionality of Application using React Native	07/14	Late	
	Implement UI using React Native	07/10	Late	
	Design Mobile Application Interfaces	07/12	Late	

Figure 07: Planner – Board View

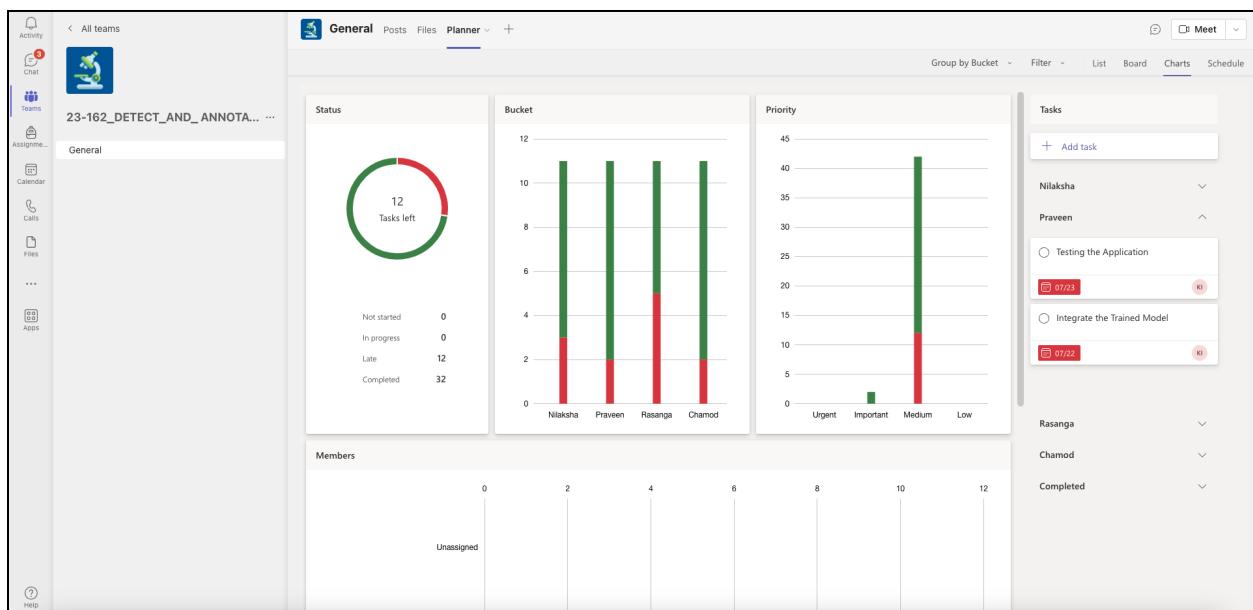


Figure 08: Planner – Chart View

The screenshot shows a Microsoft Planner interface. At the top, there's a navigation bar with tabs for General, Posts, Files, and Planner. Below the navigation is a toolbar with Group by Bucket, Filter, List, Board, Charts, and Schedule buttons. The main area is a calendar for July 2023, showing days from Sunday to Saturday. Tasks are represented as colored boxes: green for General, blue for Nilaksha, red for Praveen, orange for Rasanga, purple for Chamod, and pink for Completed. Some tasks have '+' or '+3 more' labels indicating more items. On the left, a sidebar lists navigation links: Activity, CMR, Teams, Assignments, Calendar, Calls, and Files. A bottom navigation bar includes Help and Meet buttons.

Figure 09: Planner – Schedule View

3. Updated Gantt chart

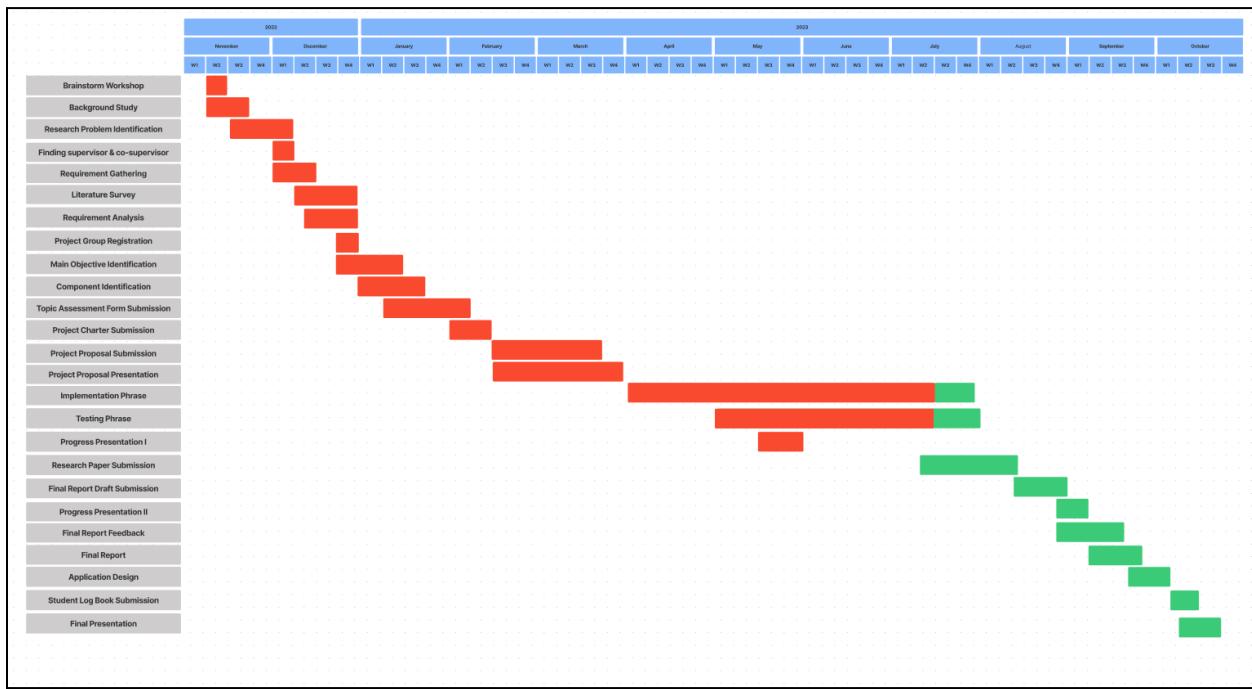


Figure 10: Gantt Chart

4. Screenshots of chats and calls of MS Teams

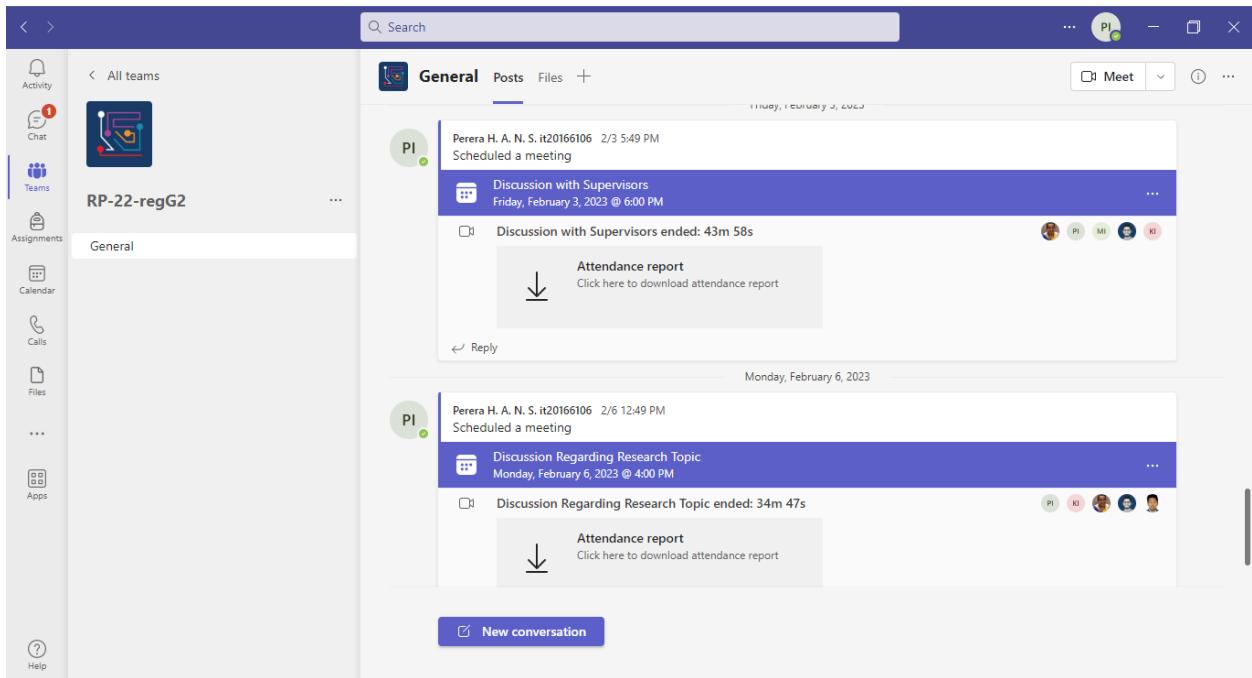


Figure 11: Screenshots of MS Teams Chats

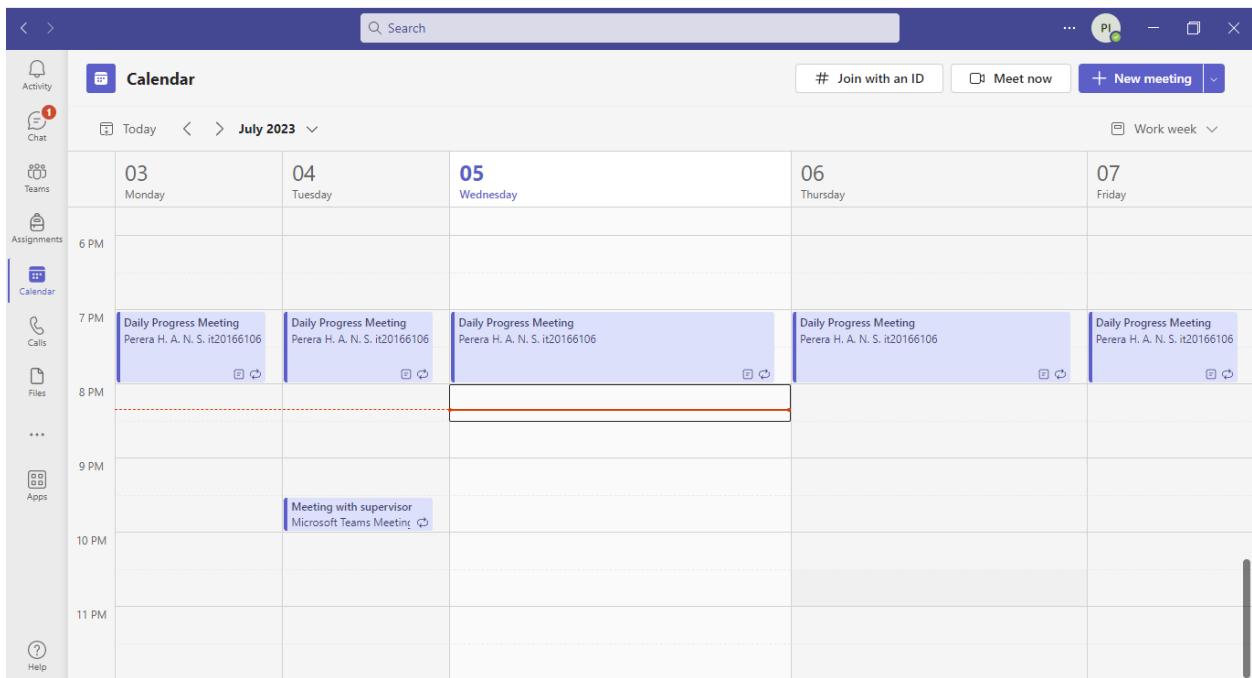


Figure 12: Screenshots of MS Teams Calendar Schedule

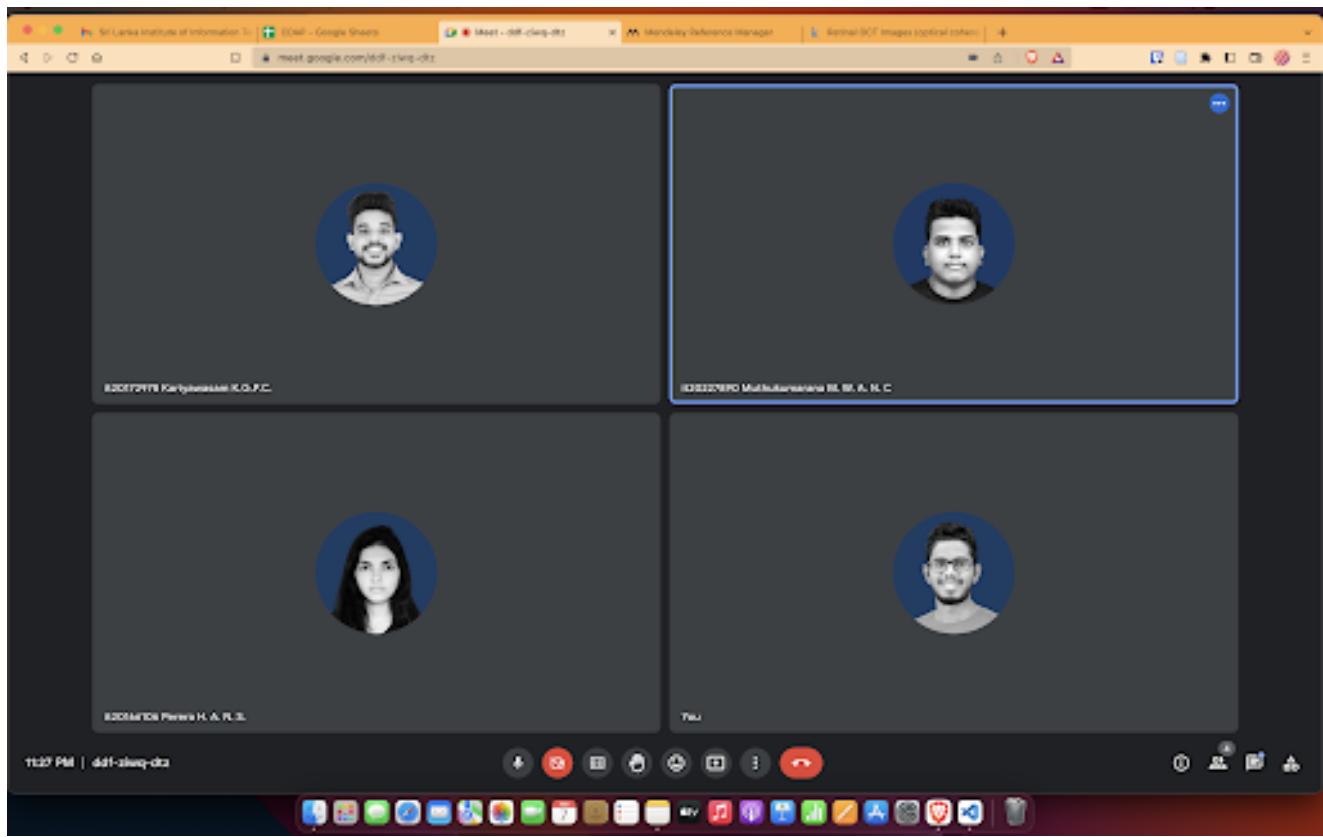


Figure 12: Screenshots of Google Meet Meetings