

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

2023-162

Status Document

Muthukumarana M.W.A.N.C

IT20227890

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	Dataset.....	3
1.2	Age-Related Macular Degeneration Detection Model .....	4
2.	Project View .....	5
3.	Updated Gantt chart .....	7
4.	Screenshots of chats and calls of MS Teams .....	7

## 1. Project progress

### 1.1 Dataset

The dataset was obtained from Kaggle and comprises a total of 15,900 OCT images. The dataset is divided as follows:

- 7,950 OCT images with AMD (Age-Related Macular Degeneration)
- 7,950 Normal OCT images

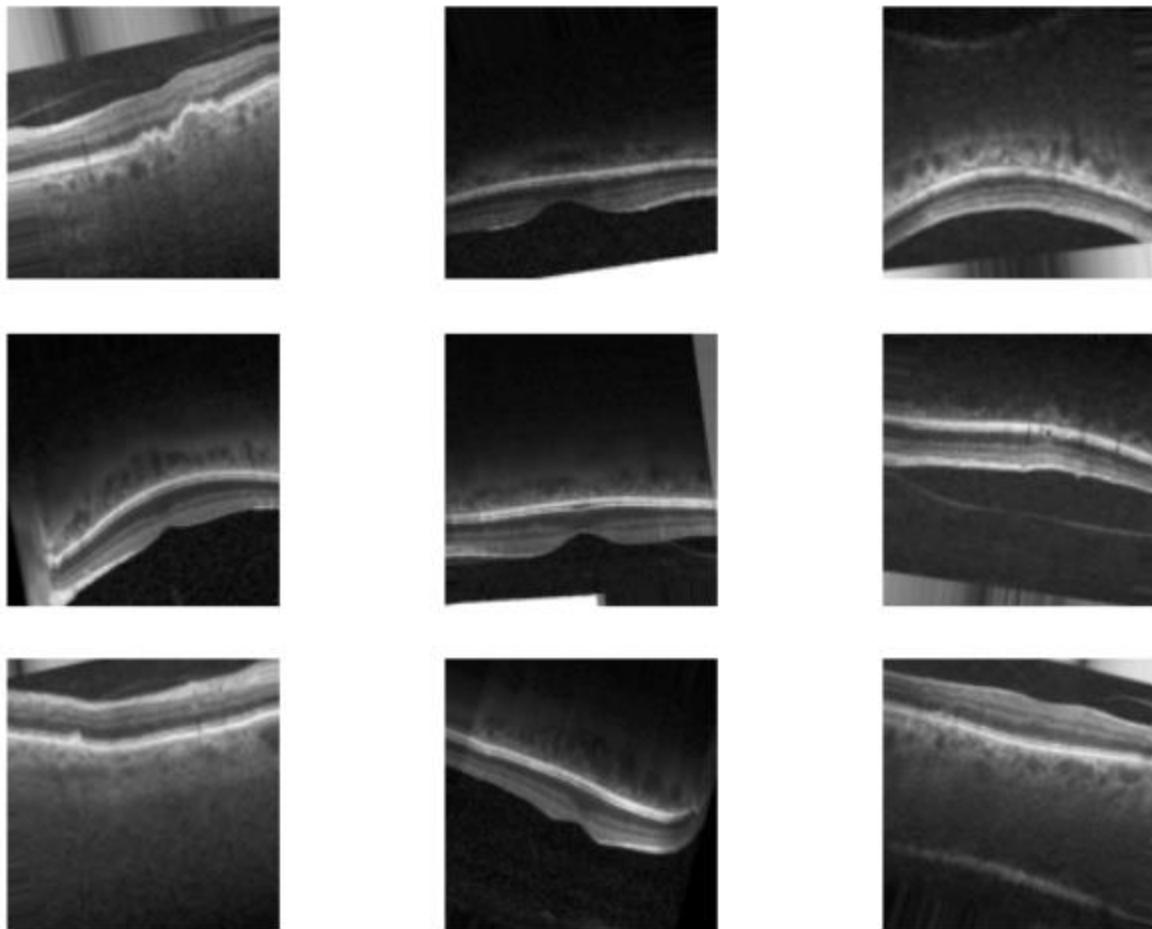


Figure 1 AMD and Normal OCT images

## 1.2 Age-Related Macular Degeneration Detection Model

Convolutional neural network (CNN) was utilized to detect symptoms of age-related macular degeneration (AMD) in retinal OCT images. The CNN architecture consisted of multiple layers: three 2D convolutional layers with increasing filter sizes (32, 64, and 128), each followed by max-pooling layers to reduce spatial dimensions. The feature maps were flattened and fed into a fully connected layer with 128 neurons and ReLU activation. To prevent overfitting, a dropout layer with a dropout rate of 0.5 was employed. The output layer had a single neuron with sigmoid activation for binary classification. The model was compiled with the Adam optimizer and binary cross-entropy loss, and accuracy was used as the evaluation metric. This model design aimed to effectively capture relevant patterns and features in retinal OCT images to accurately detect AMD symptoms.

```
▶ from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

# Define the model architecture
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 1)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model.summary()
```

Figure 2 Model structure 1

```
▶ num_epochs = 20 # define number of epochs

history = model.fit(train_generator,
                     steps_per_epoch=len(X_train) // batch_size,
                     epochs=num_epochs,
                     validation_data=val_generator,
                     validation_steps=len(X_val) // batch_size)
```

Figure 3 Model structure 2

Model: "sequential"		
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 32)	320
max_pooling2d (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 128)	0
flatten (Flatten)	(None, 86528)	0
dense (Dense)	(None, 128)	11075712
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
<hr/>		
Total params: 11,168,513		
Trainable params: 11,168,513		
Non-trainable params: 0		

Figure 4 Model summary

## 2. Project View

The screenshot shows the Microsoft Planner interface in 'Board' view. On the left, there's a sidebar with icons for Activity, Chat, Teams, Assignments, Calendar, Calls, Files, and Apps. The main area has a header with 'General', 'Posts', 'Files', 'Planner', and a '+' button. Below the header, there are three columns for team members: Praveen, Rasanga, and Chamod. Each column has a '+ Add task' button. Under each member's name, there's a list of tasks with due dates and assignees.

Team Member	Task Description	Due Date	Assignee
Praveen	Testing the Application	07/23	KI
	Integrate the Trained Model	07/22	KI
	Implement Functionality of Application using React Native	07/14	KI
	Implement UI using React Native	07/10	KI
	Design Mobile Application Interfaces	07/10	KI
	Design Mobile Application Interfaces	07/10	KI
Rasanga	Testing the Application	07/23	M
	Integrate the Trained Model	07/22	M
	Implement Functionality of Application using React Native	07/14	M
	Implement UI using React Native	07/10	M
	Design Mobile Application Interfaces	07/12	M
	Design Mobile Application Interfaces	07/12	M
Chamod	Testing the Application	07/23	M
	Integrate the Trained Model	07/22	M
	Implement Functionality of Application using React Native	07/14	M
	Implement UI using React Native	07/10	M
	Design Mobile Application Interfaces	07/12	M
	Design Mobile Application Interfaces	07/12	M

Figure 5 : Planner – Board View

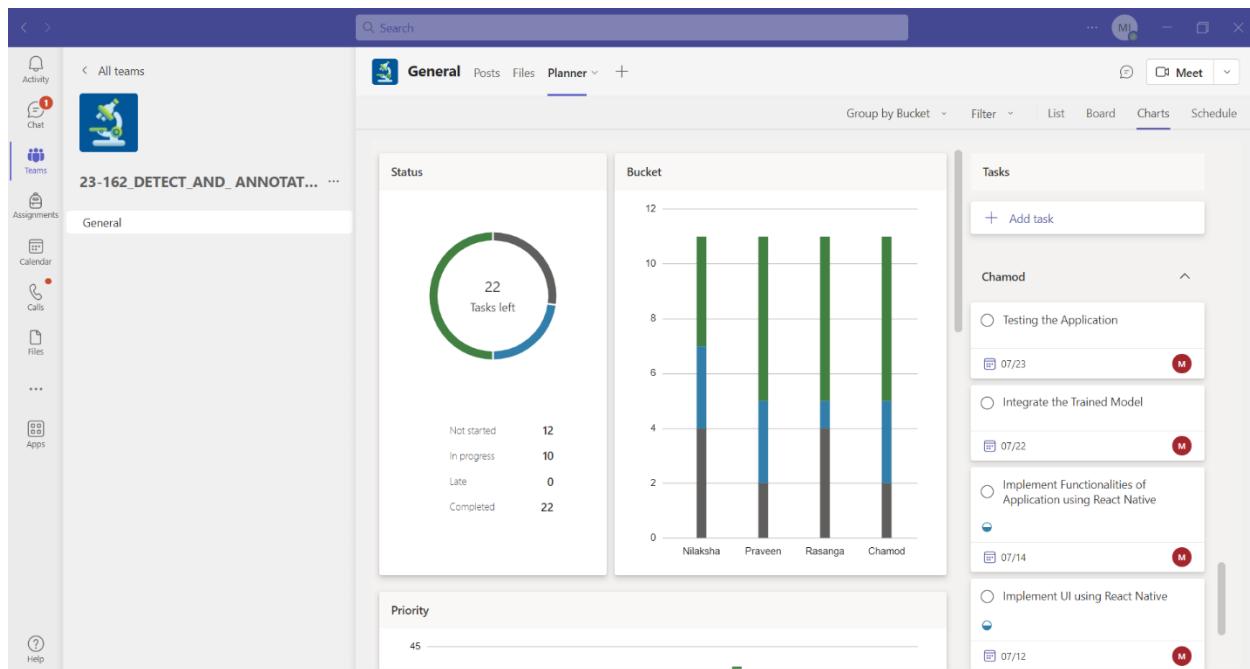


Figure 6: Planner – Chart View

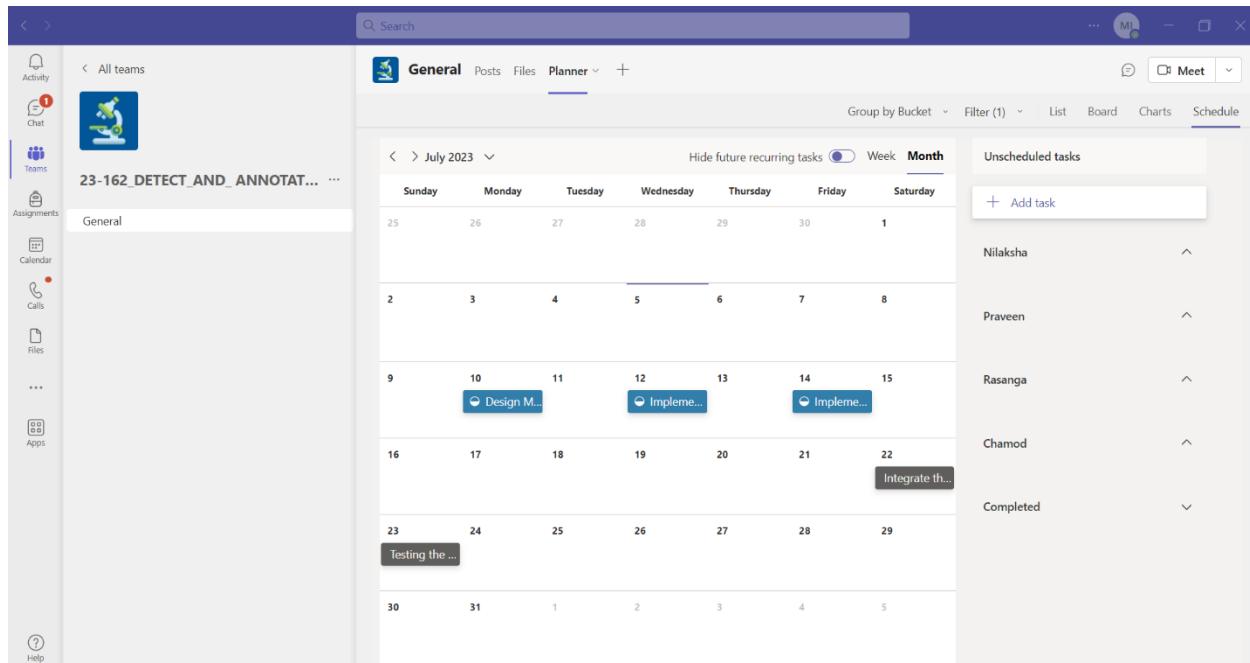


Figure 7: Planner – Schedule View

### 3. Updated Gantt chart

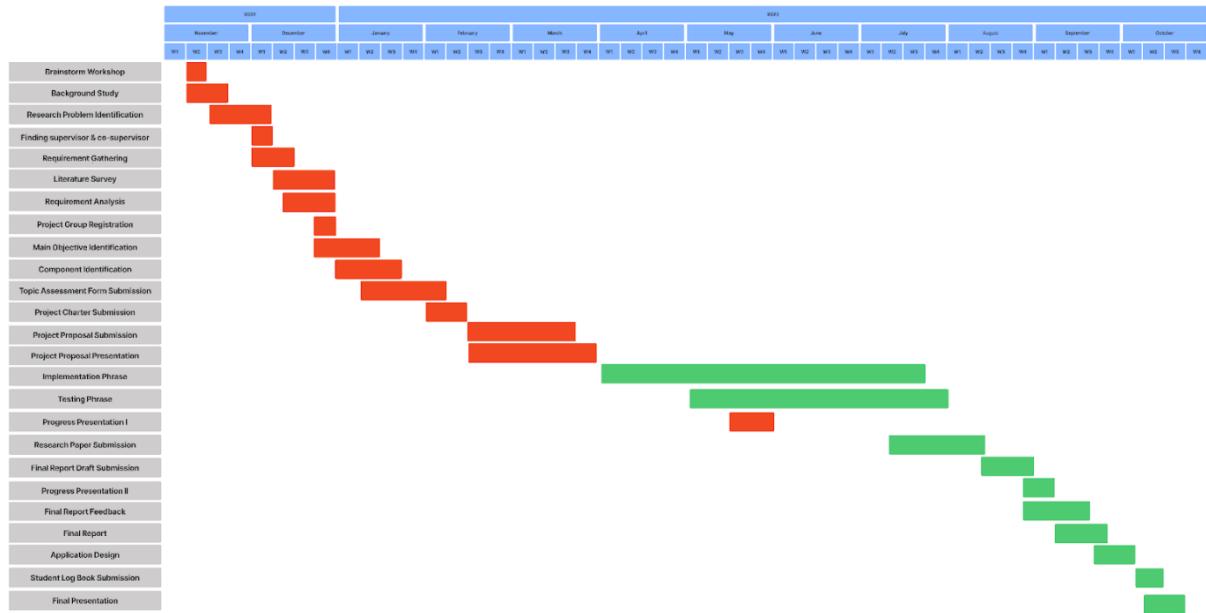


Figure 8: Gantt Chart

### 4. Screenshots of chats and calls of MS Teams

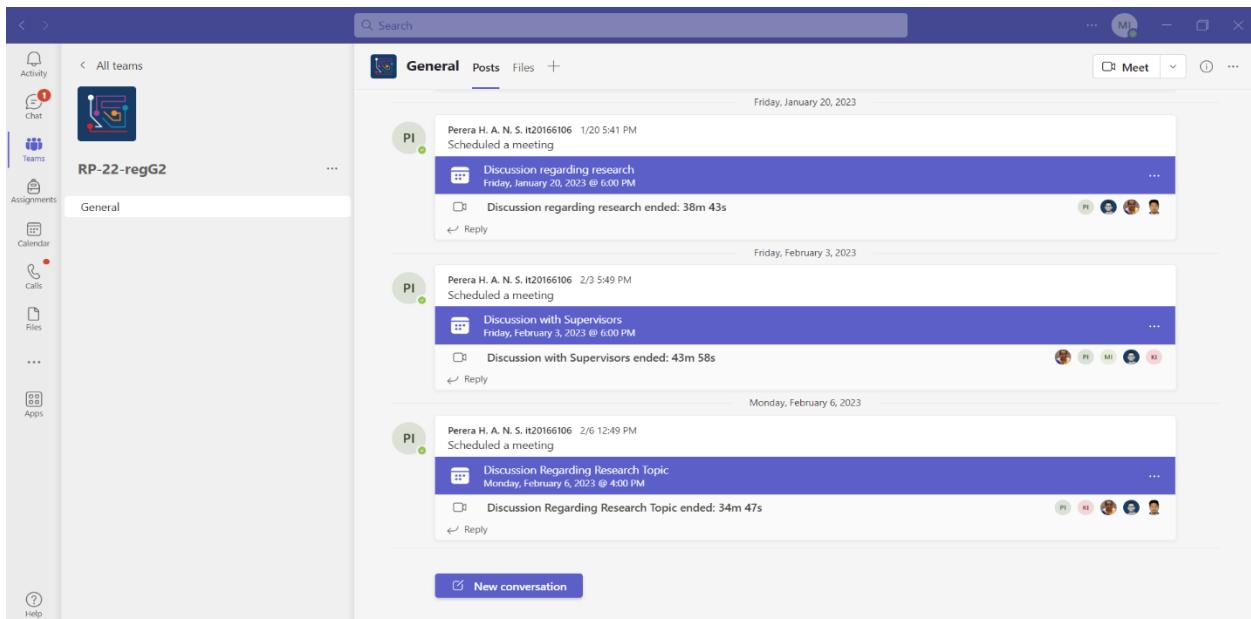


Figure 9: Screenshots of MS Teams Group

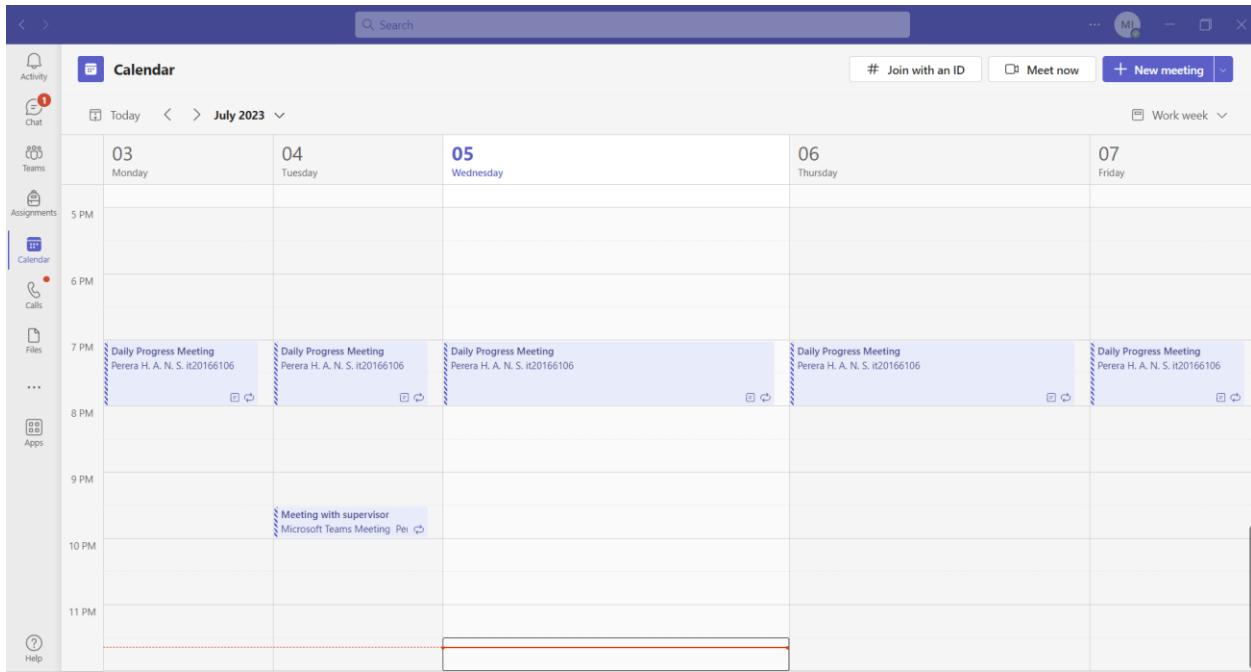


Figure 10: Screenshots of MS Teams Calendar

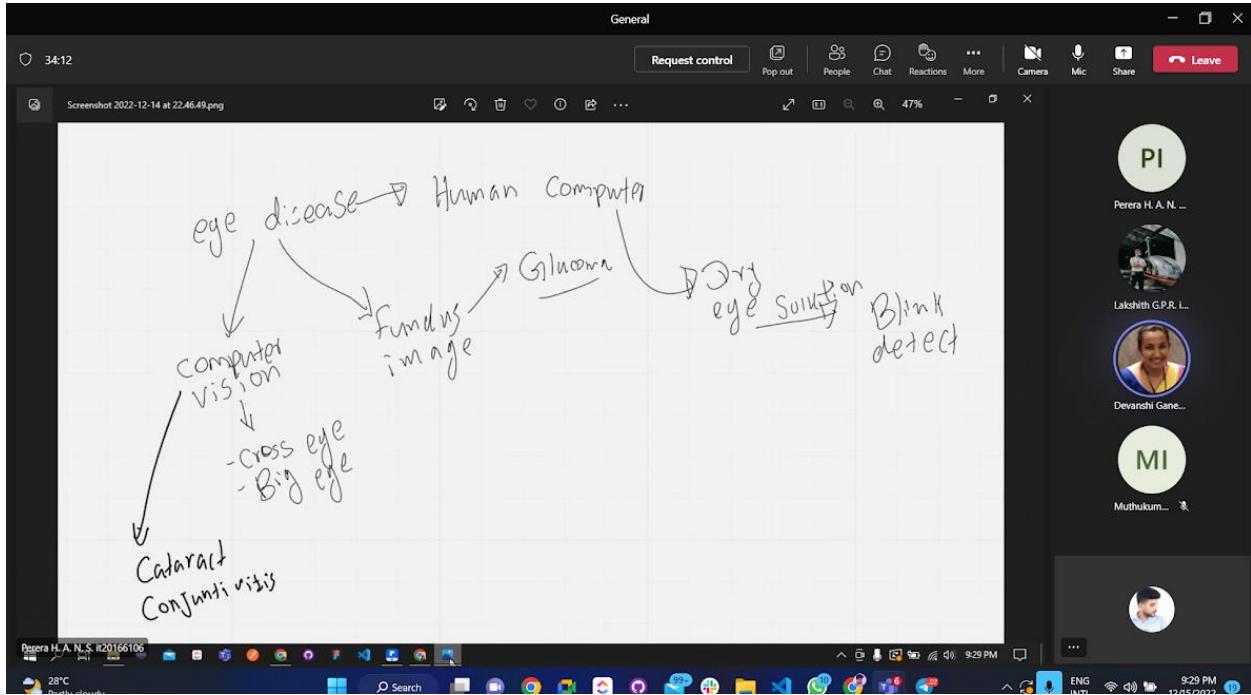


Figure 11: Screenshots of Group Meeting with Supervisor