Machine Learning Approach to Detect & Annotate Eye Diseases using Retinal Images

2023-162

Our Team

Team Leader
IT20166106

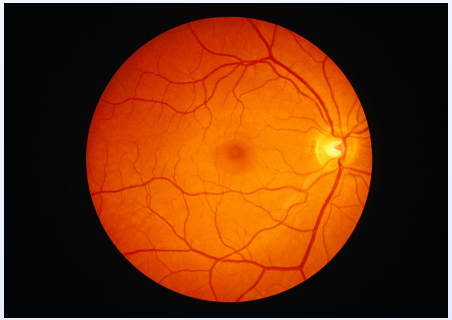
Member 01
IT20165666

Member 02
IT20227890

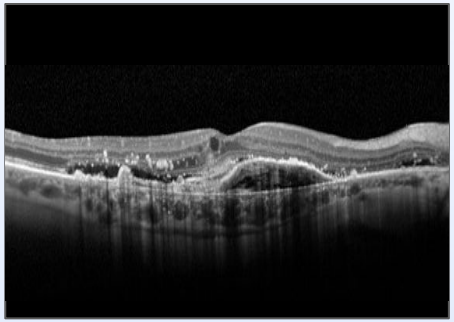
Member 03
IT20172978

Introduction

- Development of an mobile application for the identification of Diabetic Retinopathy and Age-Related Macular Degeneration to aid eye specialists with accurate diagnoses.



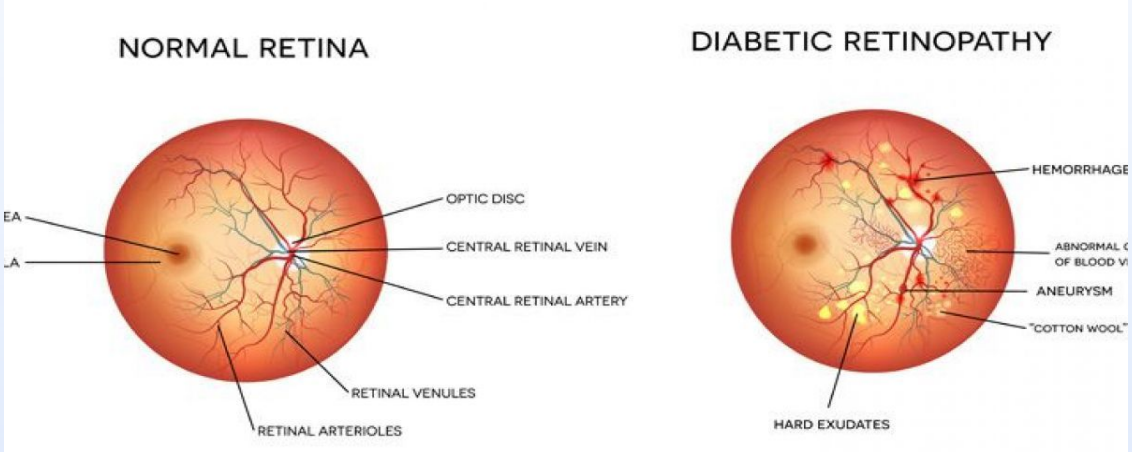
Fundus Image



OCT Image

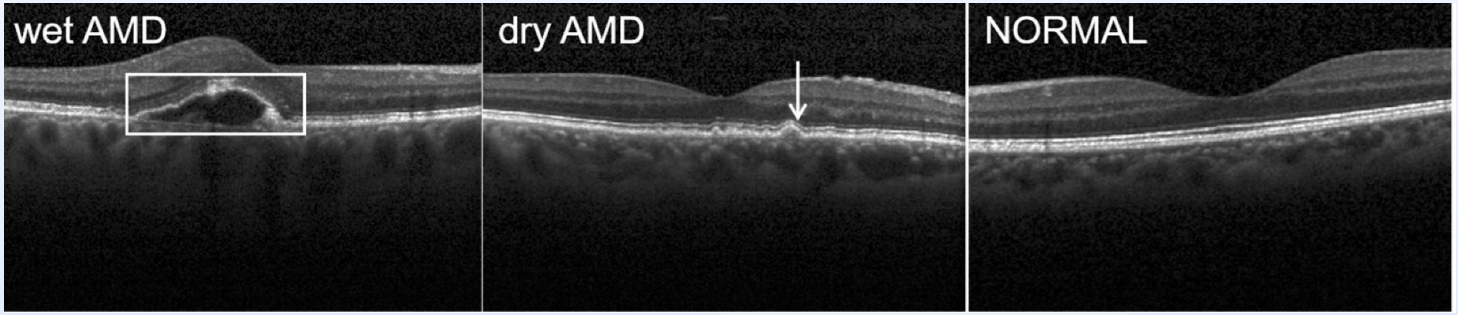
What is Diabetic Retinopathy

- Diabetic retinopathy is a complication of diabetes
- Diabetic retinopathy is the 4th leading cause of blindness and 5th common cause of visual impairment



What is Age-related Macular Degeneration

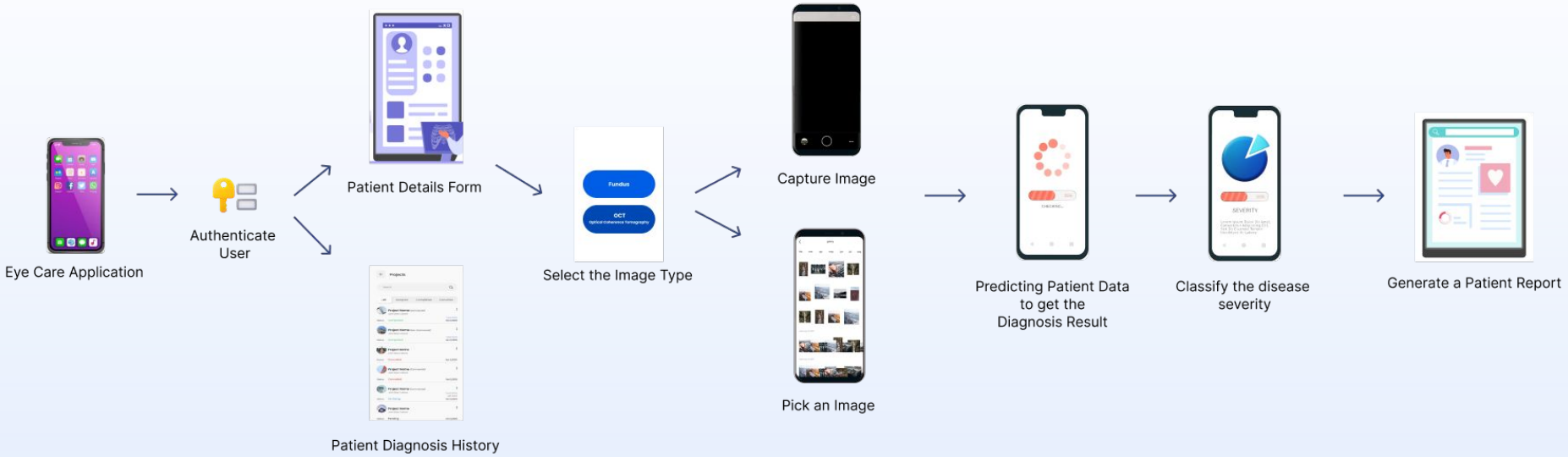
- Age-Related Macular Degeneration (AMD) causes progressive vision loss and affects millions worldwide.
- There are two types of AMD:
  - Dry AMD (Drusen)
  - Wet AMD (CNV)



Research Problem

- Unavailability of a verified mobile application to automate the diagnose process.
- Diagnosis of eye diseases is time-consuming and requires expertise and training.
- Accurate detection of eye diseases are crucial to avoid visual impairments

User Flow Diagram



Commercialization

- Target market for our mobile application includes healthcare institutions, clinics, hospitals and ophthalmologists
- Unique Selling Proposition
  - Our mobile app offers a fast and accurate solution for diabetic retinopathy detection
  - Unavailability of verified mobile applications for the purpose

IT20166106 | Perera H. A. N. S

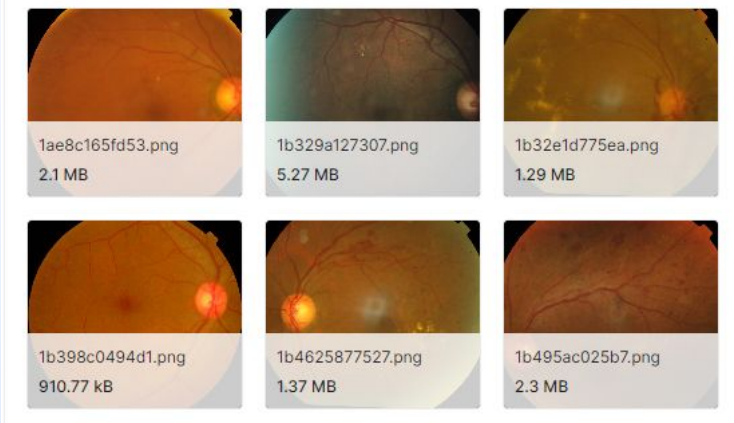
Detect Symptoms of Diabetic Retinopathy using Retinal Fundus Images

Introduction

- A deep learning algorithm to accurately detect the presence of Diabetic Retinopathy in a retinal fundus image to aid the screening process of diagnosis

Datasets

- Asia Pacific Tele-Ophthalmology Society 2019 Blindness Detection (APTOS 2019 BD) dataset
- 3662 samples of retinal fundus images
- https://www.kaggle.com/c/aptos2019-blindness-detection/overview



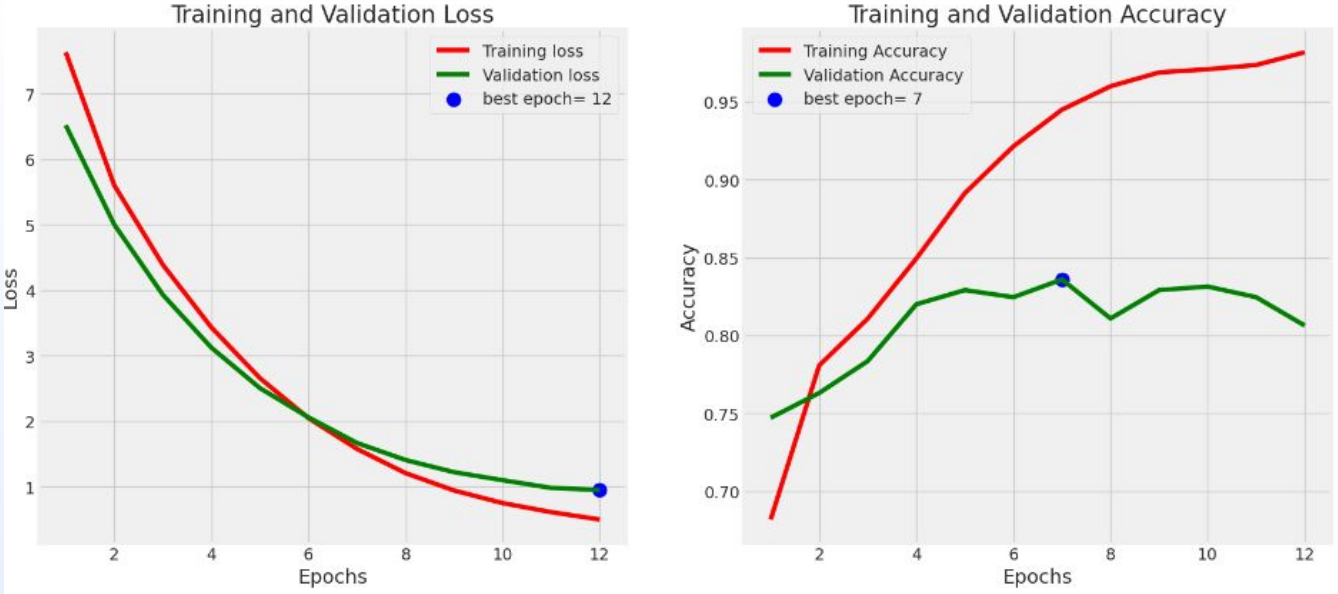
Implementation of Component

- Frameworks and Libraries
  - Tensorflow
  - keras
- Model Architecture Summary
  - EfficientNetB3
  - BatchNormalization
  - Dense
  - Dropout
- Loss Function and Optimization
  - Adam

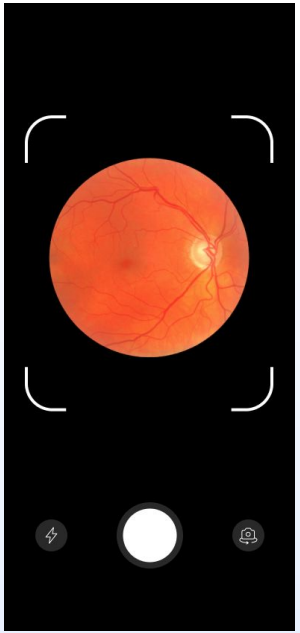
Result Comparison

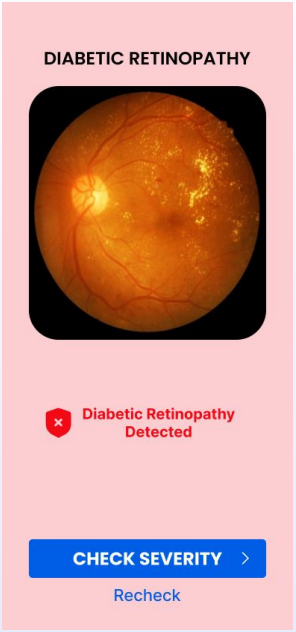
Epoch	Loss	Accuracy	V_loss	V_acc	LR	Next LR	Monitor	% Improv	Duration
1 /40	7.634	68.180	6.51997	74.715	0.00100	0.00100	accuracy	0.00	129.90
2 /40	5.602	78.081	5.00284	76.310	0.00100	0.00100	accuracy	14.52	45.96
3 /40	4.389	81.086	3.93151	78.360	0.00100	0.00100	accuracy	3.85	46.50
4 /40	3.433	84.944	3.12381	82.005	0.00100	0.00100	accuracy	4.76	47.67
5 /40	2.658	89.143	2.50608	82.916	0.00100	0.00100	accuracy	4.94	48.10
6 /40	2.048	92.147	2.05986	82.460	0.00100	0.00100	val_loss	17.81	46.43
7 /40	1.578	94.503	1.66799	83.599	0.00100	0.00100	val_loss	19.02	46.82
8 /40	1.208	96.005	1.40822	81.093	0.00100	0.00100	val_loss	15.57	46.24
9 /40	0.943	96.893	1.22289	82.916	0.00100	0.00100	val_loss	13.16	46.80
10 /40	0.751	97.098	1.10002	83.144	0.00100	0.00100	val_loss	10.05	47.11
11 /40	0.613	97.371	0.98263	82.460	0.00100	0.00100	val_loss	10.67	47.17
12 /40	0.500	98.191	0.95089	80.638	0.00100	0.00100	val_loss	3.23	47.27

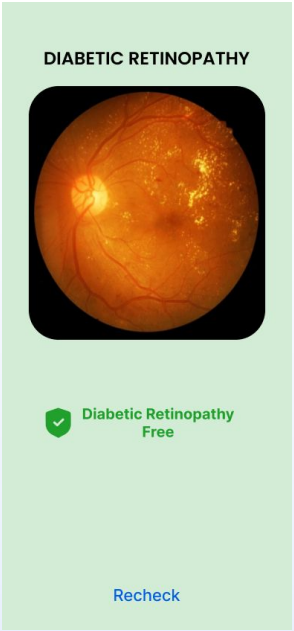
Result Comparison



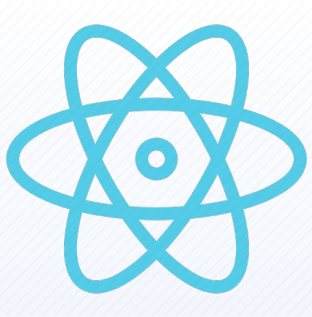
User Interfaces



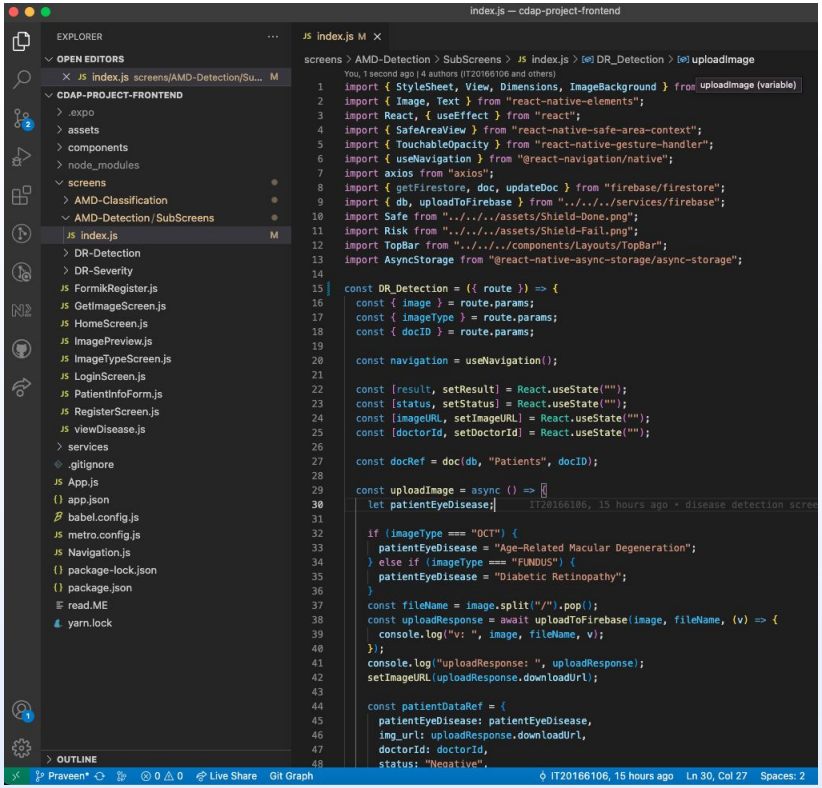


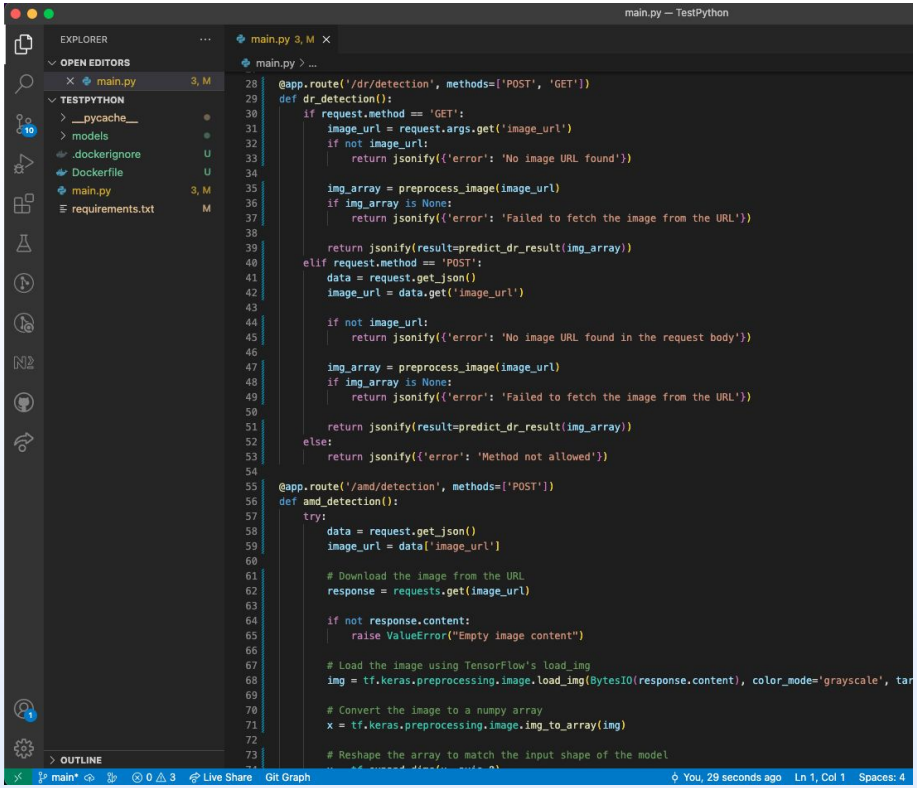


Frontend & Backend - Technologies



Frontend & Backend - Code





IT20165666 | Lakshith G. P. R

Grade Severity of Diabetic Retinopathy using Retinal Fundus Images

Introduction

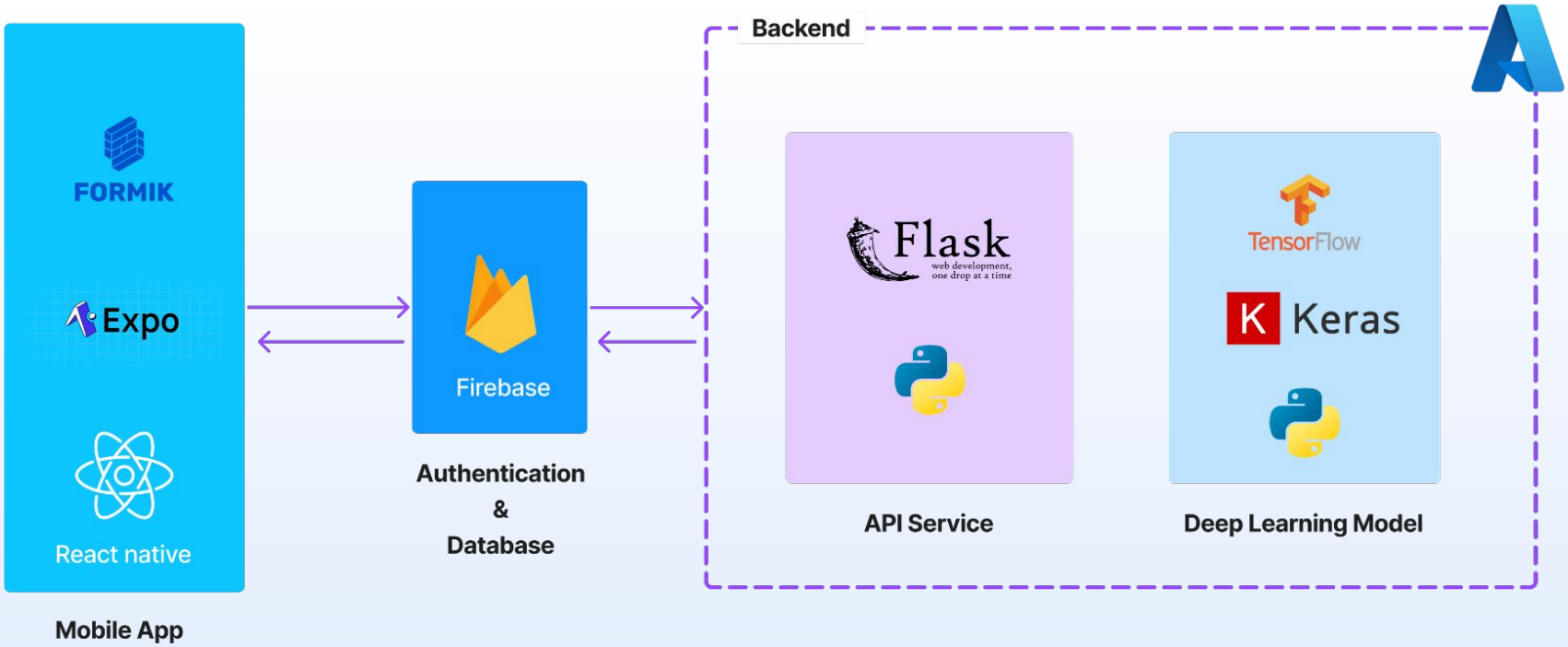
“To ensure effective treatments for DR it is crucial to determine the patient’s current stage of DR.”

- National Eye Institute -

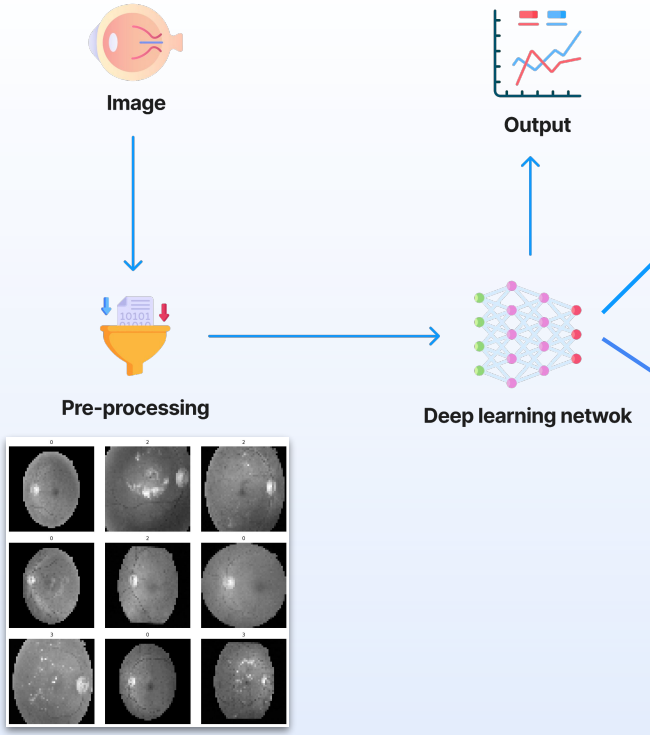
Objectives

- Develop a mobile optimized deep learning model
- Minimize the training time
- Deploy the model and backend into a highly available cloud infrastructure
- Develop a cross-platform support mobile app

Tools & Technologies

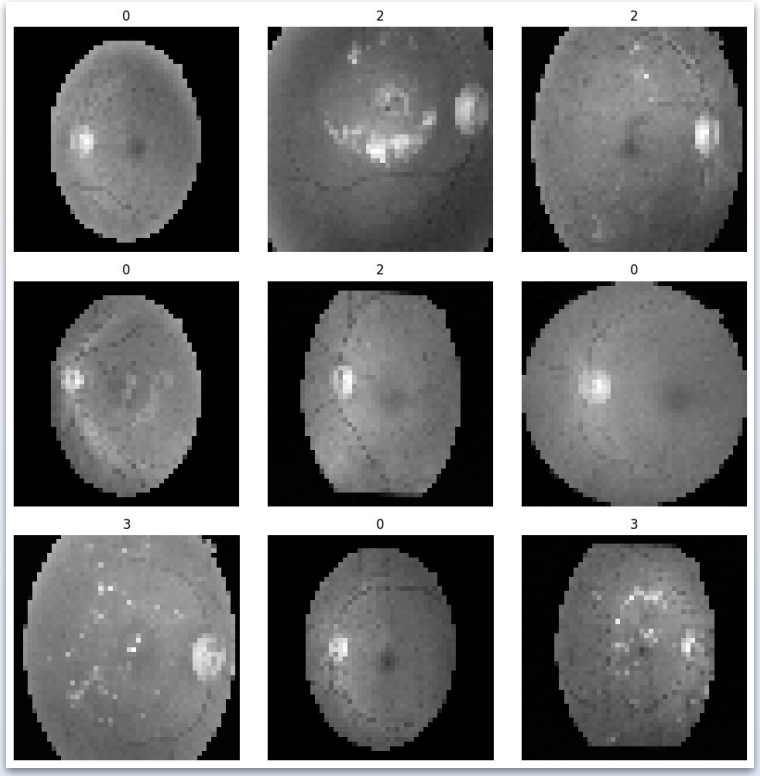


Component Implementation




```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 48, 48, 64)        640

 max_pooling2d (MaxPooling2D  (None, 24, 24, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 24, 24, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 12, 12, 64)       0
 2D)

 dropout (Dropout)           (None, 12, 12, 64)        0

 flatten (Flatten)           (None, 9216)              0

 dense (Dense)               (None, 5)                 46085

=================================================================
Total params: 83,653
Trainable params: 83,653
Non-trainable params: 0
_________________________________________________________________
```

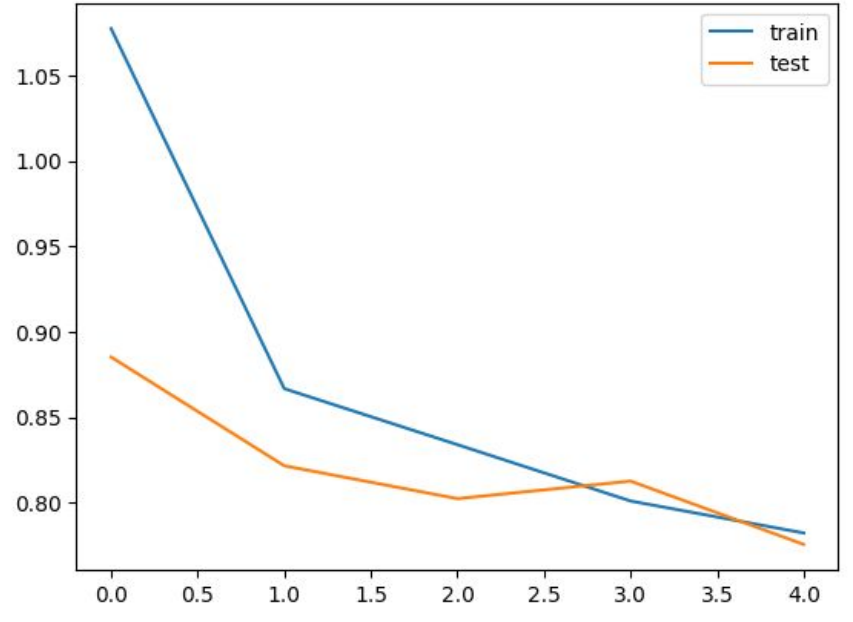
Component Imp. - Pre Processed Image



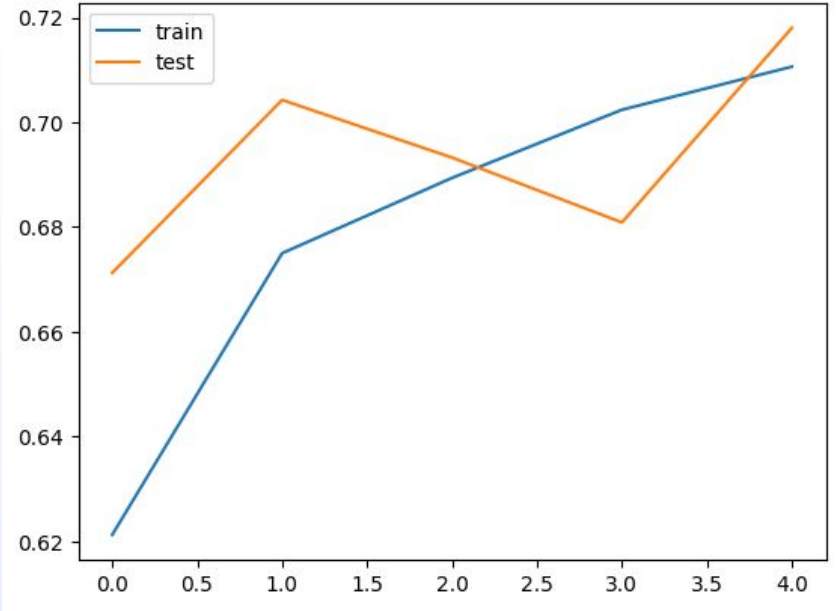
Component Imp. - Lightweight Model

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 48, 48, 64)        640

 max_pooling2d (MaxPooling2D  (None, 24, 24, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 24, 24, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 12, 12, 64)       0
 2D)

 dropout (Dropout)           (None, 12, 12, 64)        0

 flatten (Flatten)           (None, 9216)              0

 dense (Dense)               (None, 5)                 46085

=================================================================
Total params: 83,653
Trainable params: 83,653
Non-trainable params: 0
_________________________________________________________________
```

Result Comparison

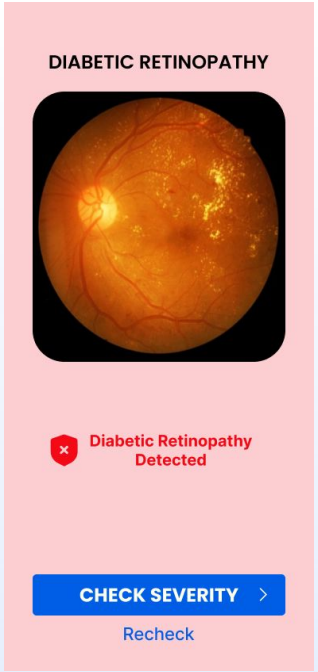


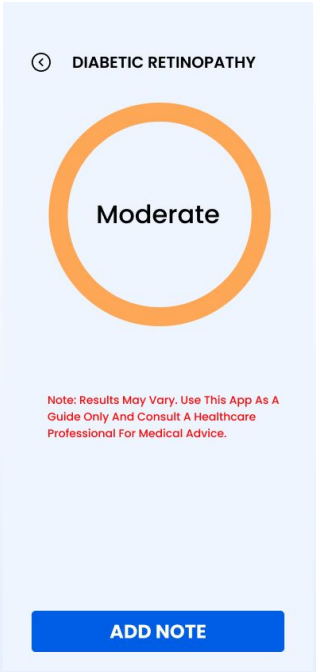
Loss variation

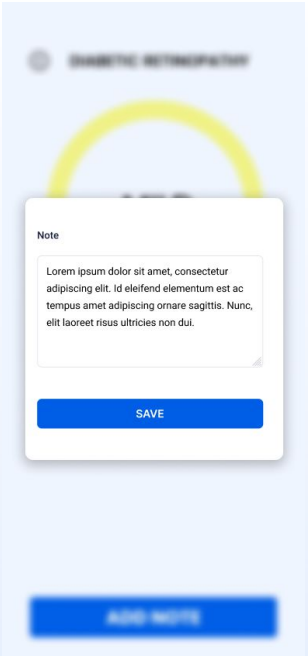


Accuracy variation

User Interface







Research Findings

- Data fragmentation techniques lead to model over-fitting
- Object identification algorithm to enhance results

IT20227890 | Muthukumarana M. W. A. N. C.

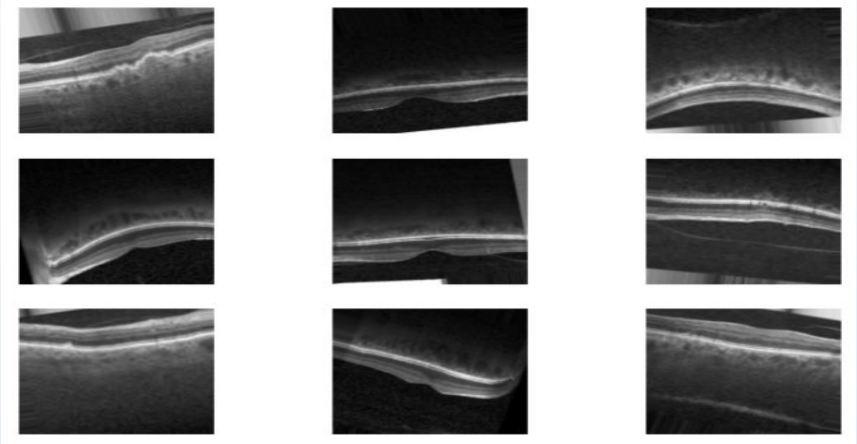
Detect Symptoms of Age-related Macular Degeneration using Retinal OCT Images

01. Introduction

- A machine learning model for the detection of age-related macular degeneration (AMD) from optical coherence tomography (OCT) images.

04. Datasets

- There are Total Samples 15,900 of OCT images
  - 7950 AMD OCT images
  - 7950 Normal OCT Images
- https://www.kaggle.com/datasets/obulisainaren/retinal-oct-c8



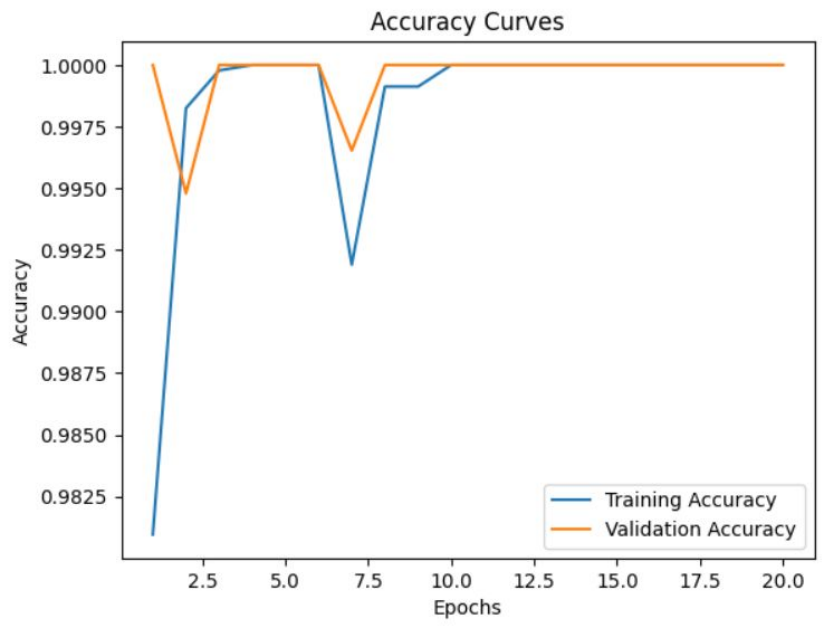
05. Implementation of Component

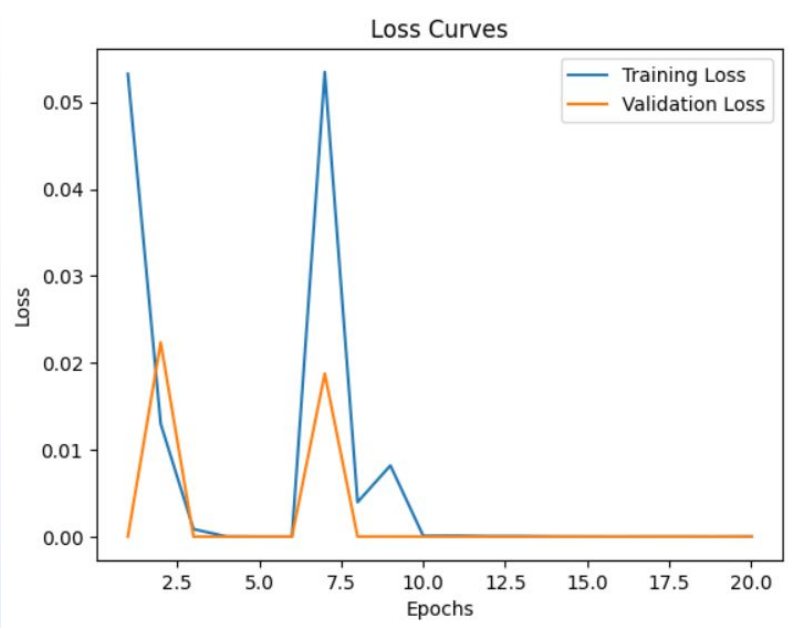
- Frameworks and Libraries
  - Tensorflow
  - keras
- Model Architecture Summary
  - Conv2D
  - MaxPooling2D
  - Flatten
  - Dense
  - Dropout
- Loss Function and Optimization
  - Adam

06. Result Comparison

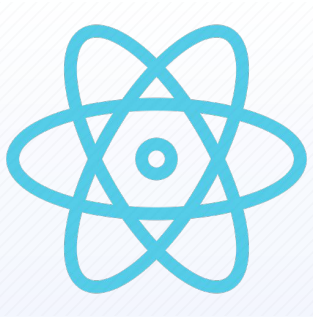
```
Epoch 1/20
143/143 [==============================] - 25s 164ms/step - loss: 0.0533 - accuracy: 0.9810 - val_loss: 8.8125e-06 - val_accuracy: 1.0000
Epoch 2/20
143/143 [==============================] - 25s 175ms/step - loss: 0.0130 - accuracy: 0.9982 - val_loss: 0.0224 - val_accuracy: 0.9948
Epoch 3/20
143/143 [==============================] - 22s 151ms/step - loss: 8.6288e-04 - accuracy: 0.9998 - val_loss: 6.4991e-08 - val_accuracy: 1.0000
Epoch 4/20
143/143 [==============================] - 26s 185ms/step - loss: 2.1652e-05 - accuracy: 1.0000 - val_loss: 1.1854e-08 - val_accuracy: 1.0000
Epoch 5/20
143/143 [==============================] - 22s 155ms/step - loss: 4.1566e-06 - accuracy: 1.0000 - val_loss: 5.7801e-09 - val_accuracy: 1.0000
Epoch 6/20
143/143 [==============================] - 25s 174ms/step - loss: 2.2885e-06 - accuracy: 1.0000 - val_loss: 4.2507e-09 - val_accuracy: 1.0000
Epoch 7/20
143/143 [==============================] - 22s 151ms/step - loss: 0.0535 - accuracy: 0.9919 - val_loss: 0.0188 - val_accuracy: 0.9965
Epoch 8/20
143/143 [==============================] - 25s 172ms/step - loss: 0.0040 - accuracy: 0.9991 - val_loss: 4.1885e-06 - val_accuracy: 1.0000
Epoch 9/20
143/143 [==============================] - 25s 176ms/step - loss: 0.0082 - accuracy: 0.9991 - val_loss: 1.3567e-05 - val_accuracy: 1.0000
Epoch 10/20
143/143 [==============================] - 23s 161ms/step - loss: 9.2370e-05 - accuracy: 1.0000 - val_loss: 2.8335e-06 - val_accuracy: 1.0000
Epoch 11/20
143/143 [==============================] - 27s 186ms/step - loss: 1.0697e-04 - accuracy: 1.0000 - val_loss: 2.4121e-06 - val_accuracy: 1.0000
Epoch 12/20
143/143 [==============================] - 25s 173ms/step - loss: 5.1908e-05 - accuracy: 1.0000 - val_loss: 8.3272e-07 - val_accuracy: 1.0000
```

06. Result Comparison

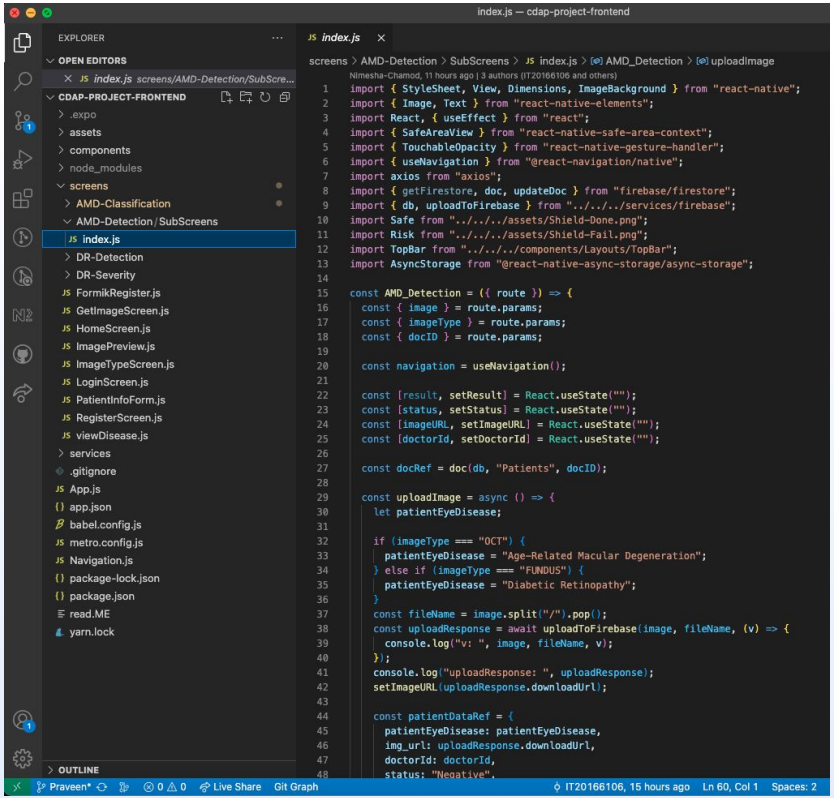


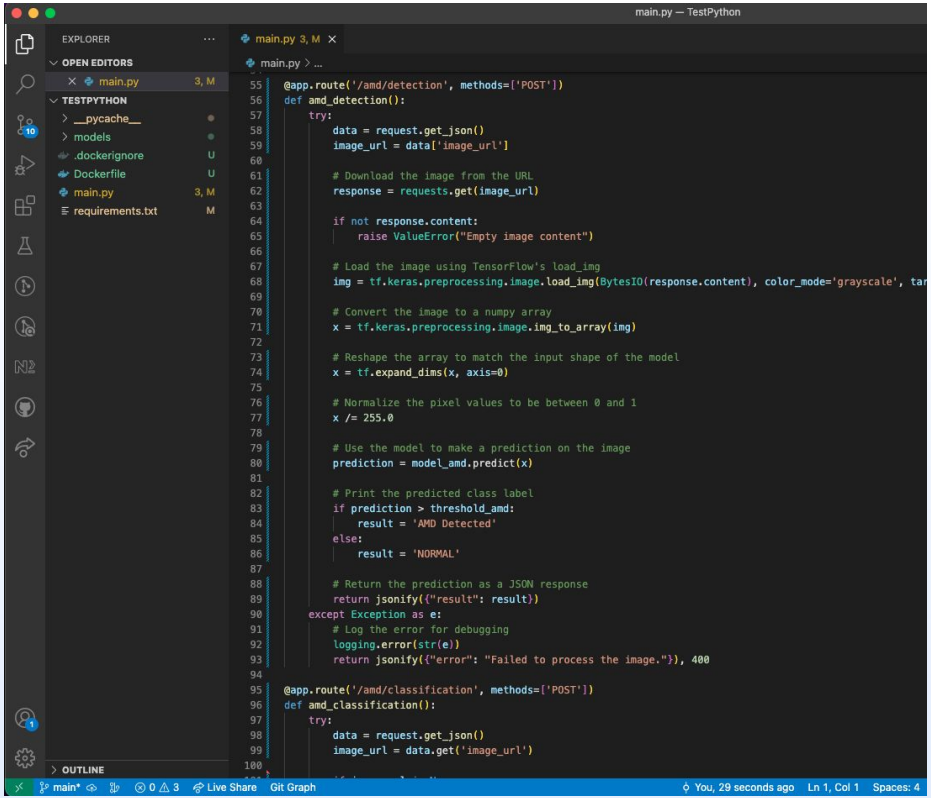


Frontend & Backend - Technologies

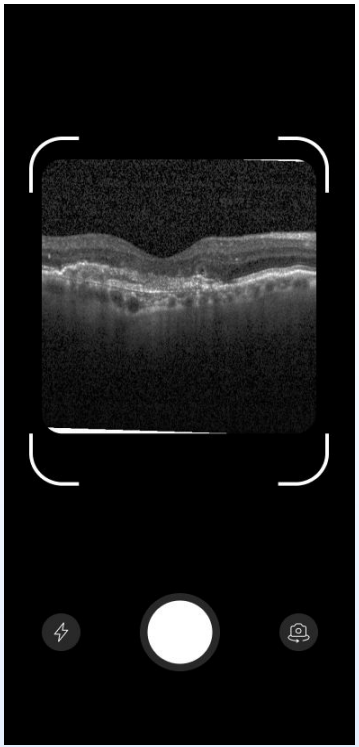


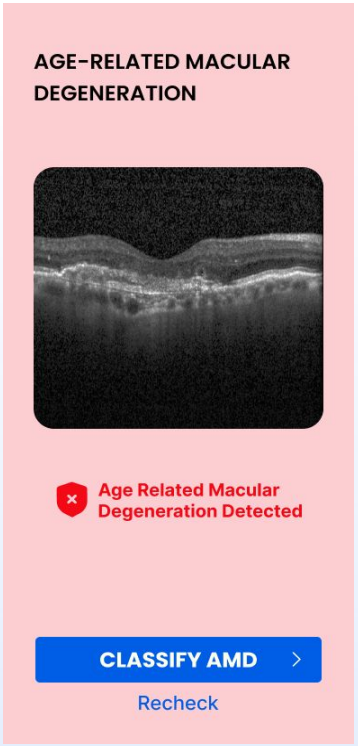
Frontend & Backend - Code

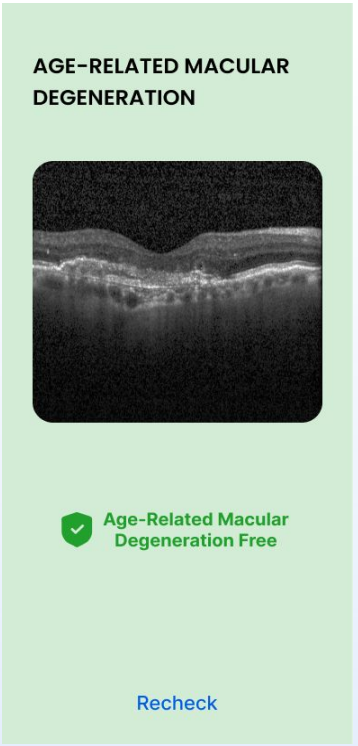




07. User Interfaces





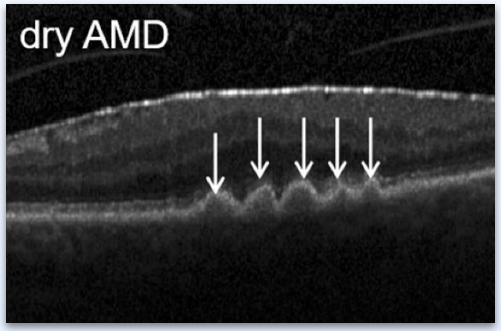


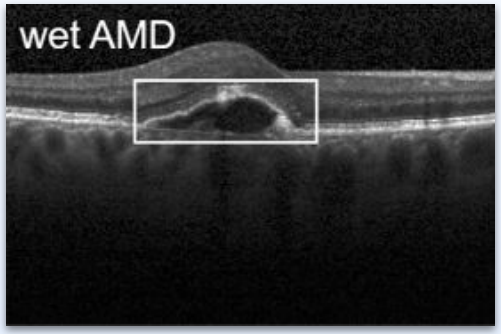
IT20172978 | Kariyawasam K. G. P. C.

Classification of Age-related Macular Degeneration using Retinal OCT Images

Introduction

- Develop a novel DL model to accurately detect and classify wet and dry AMD based on a single OCT image while providing annotations for the affected regions.

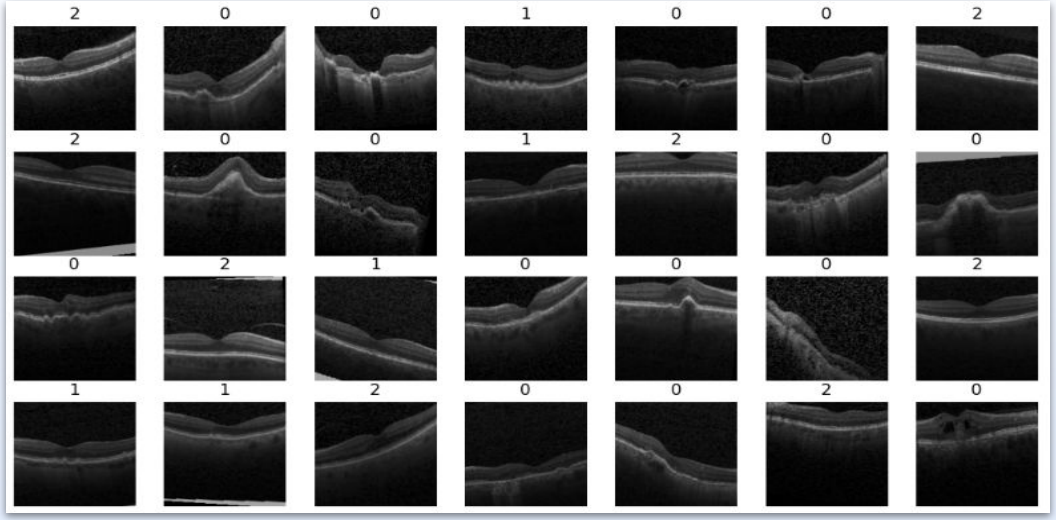




Datasets

- Drusen - 8600 OCT Images
- CNV - 13000 OCT Images
- Normal - 20000 OCT Images

https://www.kaggle.com/datasets/paultimothymooney/kermany2018



Model Implementation - Technologies

- Frameworks and Libraries
  - Tensorflow
  - keras
- Model Architecture Summary
  - Conv2D
  - MaxPooling2D
  - Dense
  - Dropout
- Loss Function and Optimization
  - Categorical Cross-entropy
  - Adam

Model Implementation - Code

AMD_Classification.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Preprocessing

```
#Calculate class weight for the imbalanced data
total = 72136

count_cnv = 37205
count_drusen = 8616
count_normal = 26315

cnv_weight = (1/count_cnv) * (total/3)
drusen_weight = (1/count_drusen) * (total/3)
norm_weight = (1/count_normal) * (total/3)

class_weight = {0 : cnv_weight, 1 : drusen_weight, 2: norm_weight}
print(class_weight)


labels = ['CNV', 'DRUSEN', 'NORMAL']

def load(path):
    listed_paths = []
    for i in labels:
        base_path = os.path.join(path, i)
        imgs = os.listdir(base_path)
        for img in imgs:
            img_path = base_path + '/' + img
            listed_paths.append(img_path)
    return listed_paths

train_paths = load(train_path)
val_paths = load(val_path)
test_paths = load(test_path)
```

```
{0: 0.6462930609685078, 1: 2.790776849272671, 2: 0.9137500791690417}
```

AMD_Classification.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

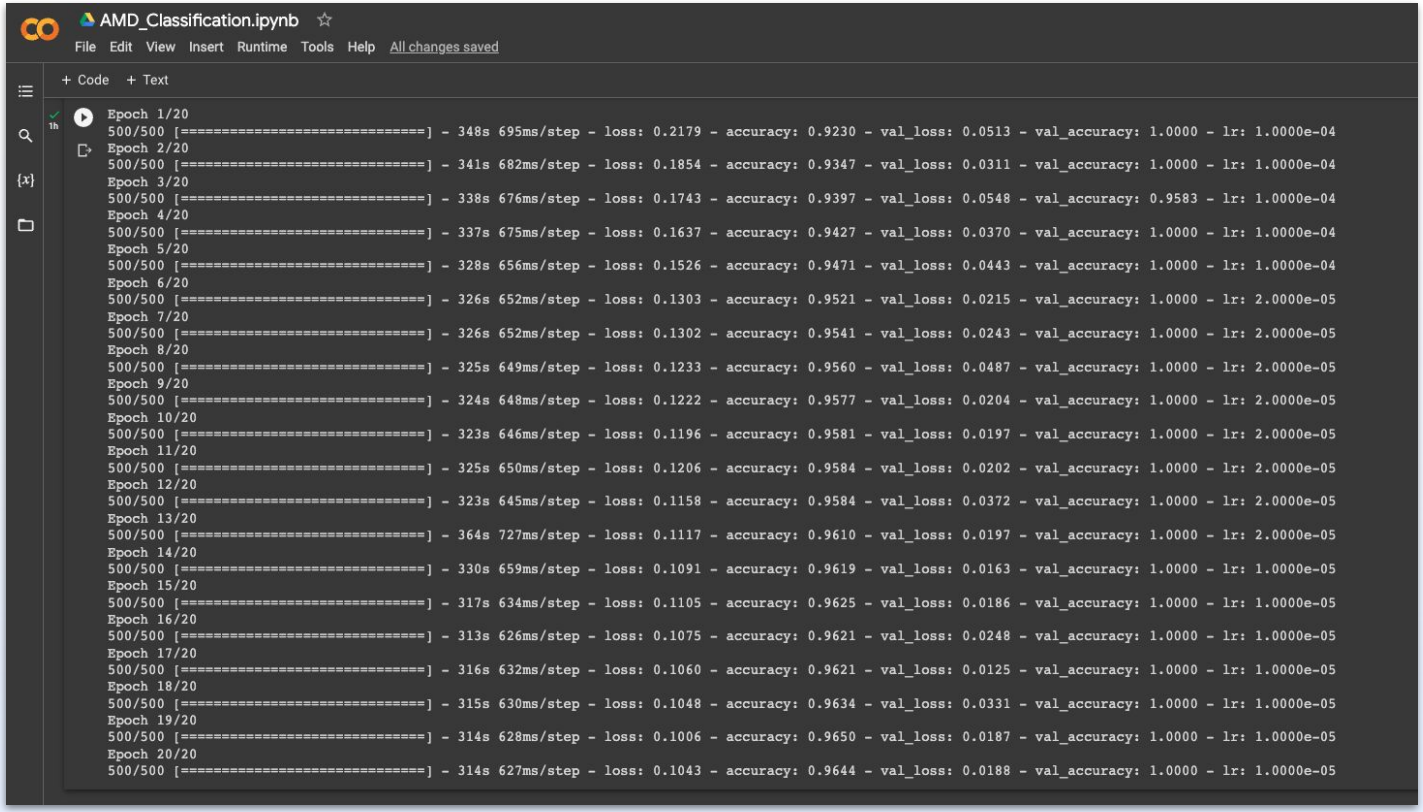
Build the Model VGG16

```
# pre-trained VGG16 model

base_model = VGG16(weights='imagenet', include_top=False, input_shape=(150, 150, 3))
trainable_layers = len(base_model.layers[:-5])
for layer in base_model.layers[:trainable_layers]:
    layer.trainable = False # Freeze the weights of all layers except for the last 5
for layer in base_model.layers[trainable_layers:]:
    layer.trainable = True
```

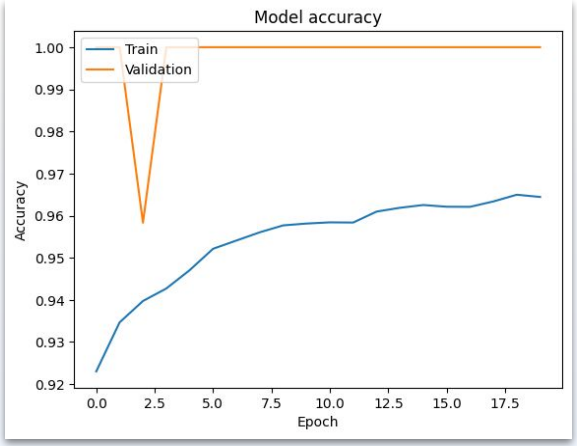
```
[ ] # Add new classification layers on top of the pre-trained layers
    num_classes = 3  # cnv, drusen, and normal.
    x = Flatten()(base_model.output)
    x = Dense(256, activation='relu')(x)
    x = Dropout(0.5)(x) # Add a dropout layer with a rate of 0.5
    predictions = Dense(num_classes, activation='softmax')(x)
```

```
[ ] # Create the final model
    model_3 = Model(inputs=base_model.input, outputs=predictions)
    # Compile the model with Adam optimizer
    opt = Adam(lr=0.0001)
    model_3.compile(loss='categorical_crossentropy', optimizer=opt, metrics=['accuracy'])
    # Define the callbacks
    reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=0.00001)
    early_stop = EarlyStopping(monitor='val_loss', patience=5)
    # Train the model
    num_epochs = 20
    history_3 = model_3.fit(train_generator,
                            epochs = num_epochs,
                            steps_per_epoch = 500,
                            validation_data = val_generator,
                            validation_steps = len(val_generator),
                            callbacks = [reduce_lr, early_stop])
```

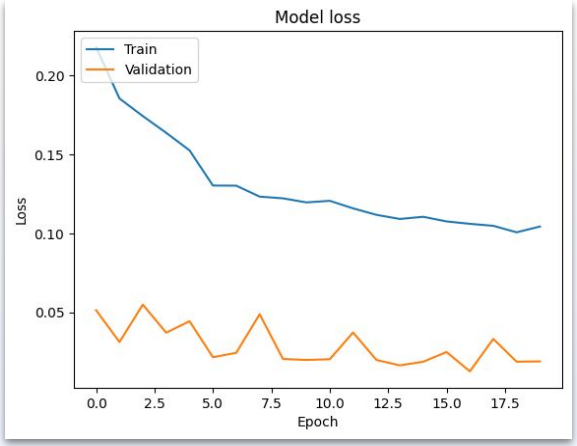
Result Comparison



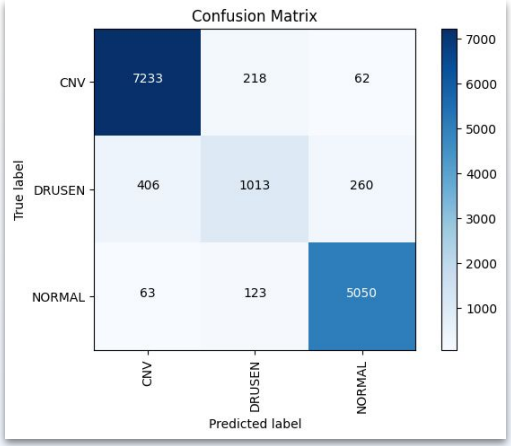
Evidence



Plot training & validation accuracy values



Plot training & validation loss values

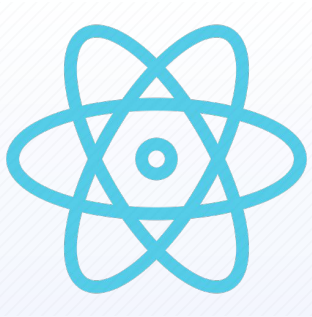


Confusion Matrix

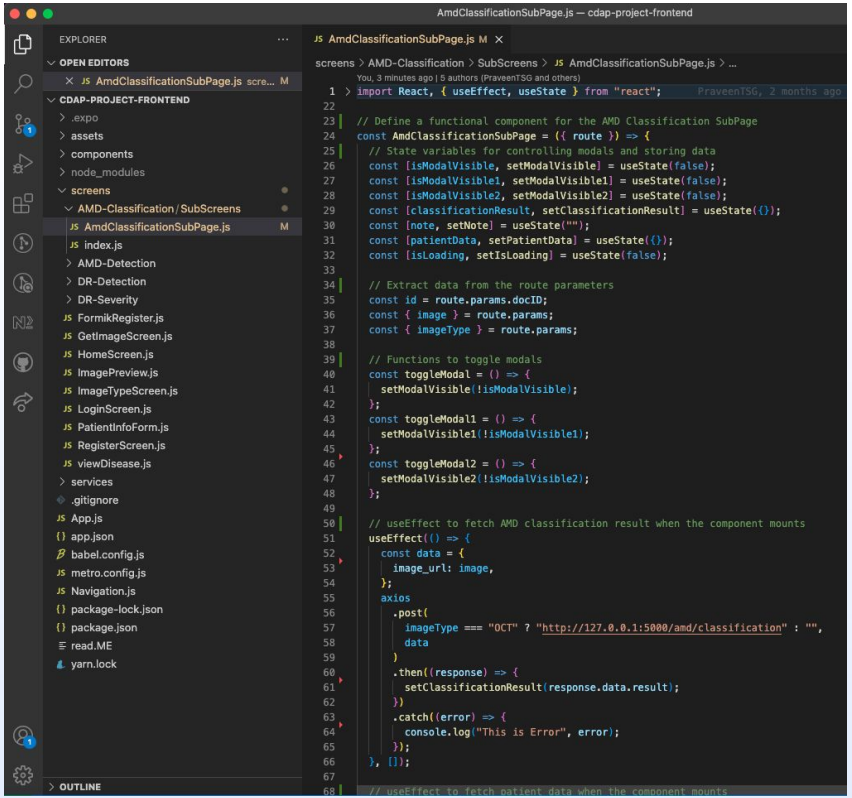
```
Train accuracy: 0.9542208164930344
Train loss: 0.13051368705928326

Validation accuracy: 1.0
Validation loss: 0.024401552975177765

Test accuracy: 0.9820936918258667
Test loss: 0.05077371001243591
```

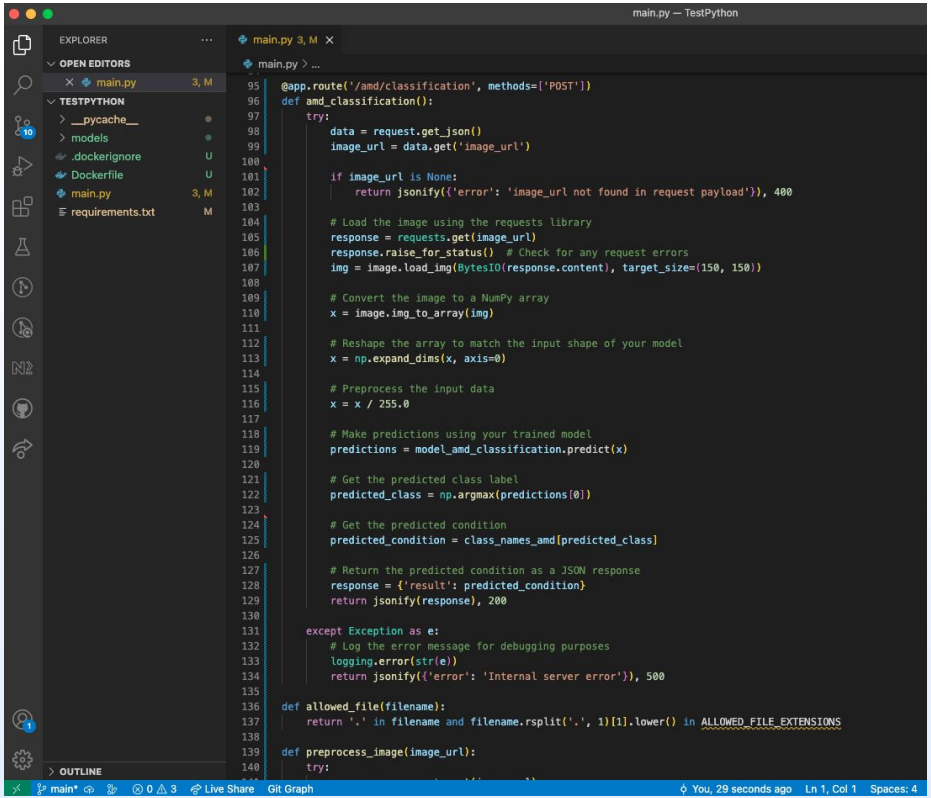
Train, Validation, and Test Accuracies

Frontend & Backend - Technologies

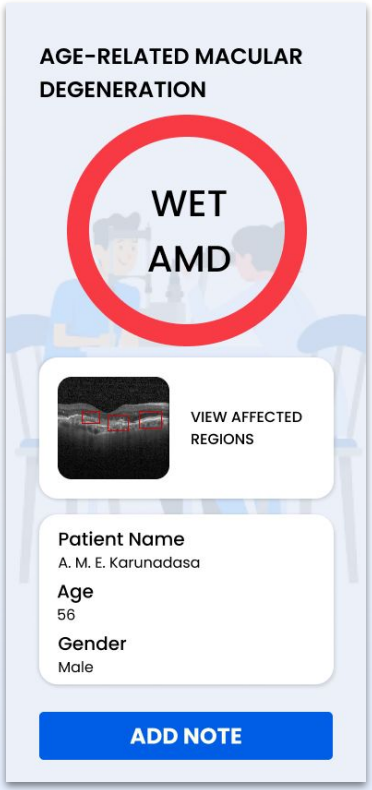


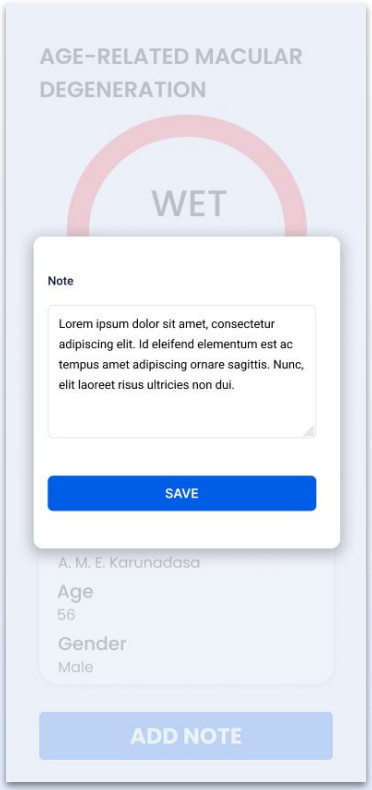
Frontend & Backend - Code

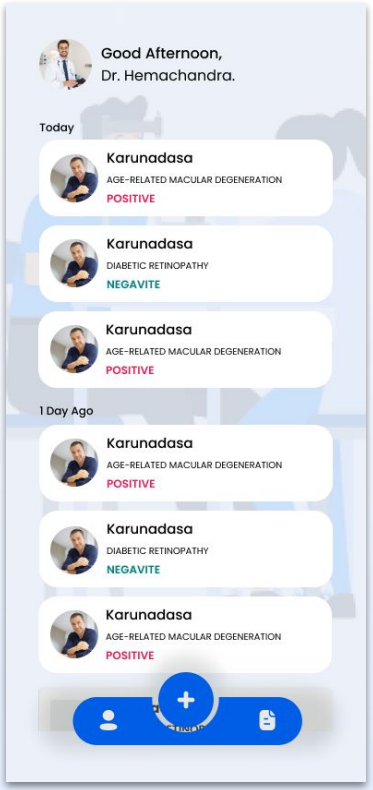




User Interfaces







Demonstration

Eye Care - Mobile Application for Eye Disease Diagnosis

Features and Benefits

- User-Friendly Interface
- Upload or capture Retinal Image
- Detect and Classify using ML algorithms
- Real-Time Results
- Secure Data Handling
- Cross-platform Application

Non-functional Requirements

- Security & Authentication
- Minimized Response Time
- Improved User-experience
- Database Scalability
- Device Compatibility

Thank You !