CHAPTER 4

DESIGN

System Design is the process of defining the architecture, components, modules, interfaces, and data for a to satisfy specified requirements. System design could be seen as the specification of the systems theory to product development. The system design process builds up general framework building design. Programming outline includes speaking to the product framework works in a shape that may be changed into one or more projects. The prerequisite indicated by the end client must be put in a systematically manner. Outline is an inventive procedure; a great configuration is the way to viable framework. The framework "Outline" is characterized as "The procedure of applying different systems and standards with the end goal of characterizing a procedure or a framework in adequate point of interest to allow its physical acknowledgment". Different configuration components are taken after to add to the framework. The configuration detail portrays the components of the framework. The segments or components of the framework and their appearance, to end-users. System designing in terms of software engineering has its own value and importance in the system development process. To mention it may though seem as simple us anything or simply the design of systems, but in a broader sense it implies a systematic and rigorous approach to design such as system which fulfills all the practical aspects including flexibility, efficiency, and security.

4.1 System Architecture

A system architecture diagram would be used to show the relationship between different components. Usually, they are created for systems which include hardware and software, and these are represented in the diagram to show the interaction between them. The architectural configuration procedure is concerned with building up a fundamental basic system for a framework. It includes recognizing the real parts of the framework and interchanges between these segments. The beginning configuration procedure of recognizing these subsystems and building up a structure for subsystem control and correspondence is called construction modeling outline and the yield of this outline. Procedure is a portrayal of the product structural planning. The proposed architecture for this system is given below. It shows the way this system is designed and brief working of the system. The reason for the design is to arrange the arrangement of the issue determined by the necessities report.

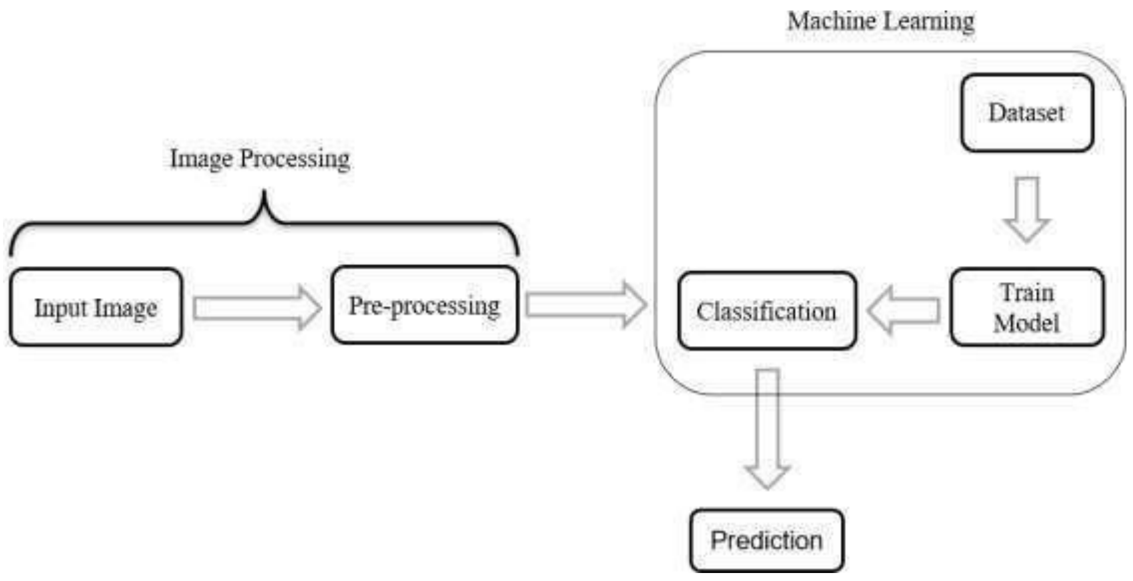


Fig 4.1 System Architecture

A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system. on the dataset to build the model and is then saved for later deployment. The pre-processed image is then provided to this model which classifies the image, and predicts it as keratoconus or not keratoconus.

4.2 Use Case Model

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

A use case is a functionality provided by a system. Use cases can include both successful and unsuccessful scenarios of user interactions with the system. A use case is depicted by an ellipse and name of the use case is written inside it.

The use case diagram for our project is as follows: The Use case diagram in Figure 4.2 shows the actor-to-actor interaction between the Ophthalmologist and a Model. The patient consults an Ophthalmologist, the Ophthalmologist then obtains the images of the corneal topography and then the Ophthalmologist will login by giving his/her credentials. After Successful validation he/she input the images to the model for knowing the result.Once after uploading the image the model does the preprocessing of the given input and then image resizing is done and then in later stages feature extraction is done and finally classified result is sent back to the Ophthalmologist where he/she can assist and treat the patient by knowing whether it is infected eye or not.

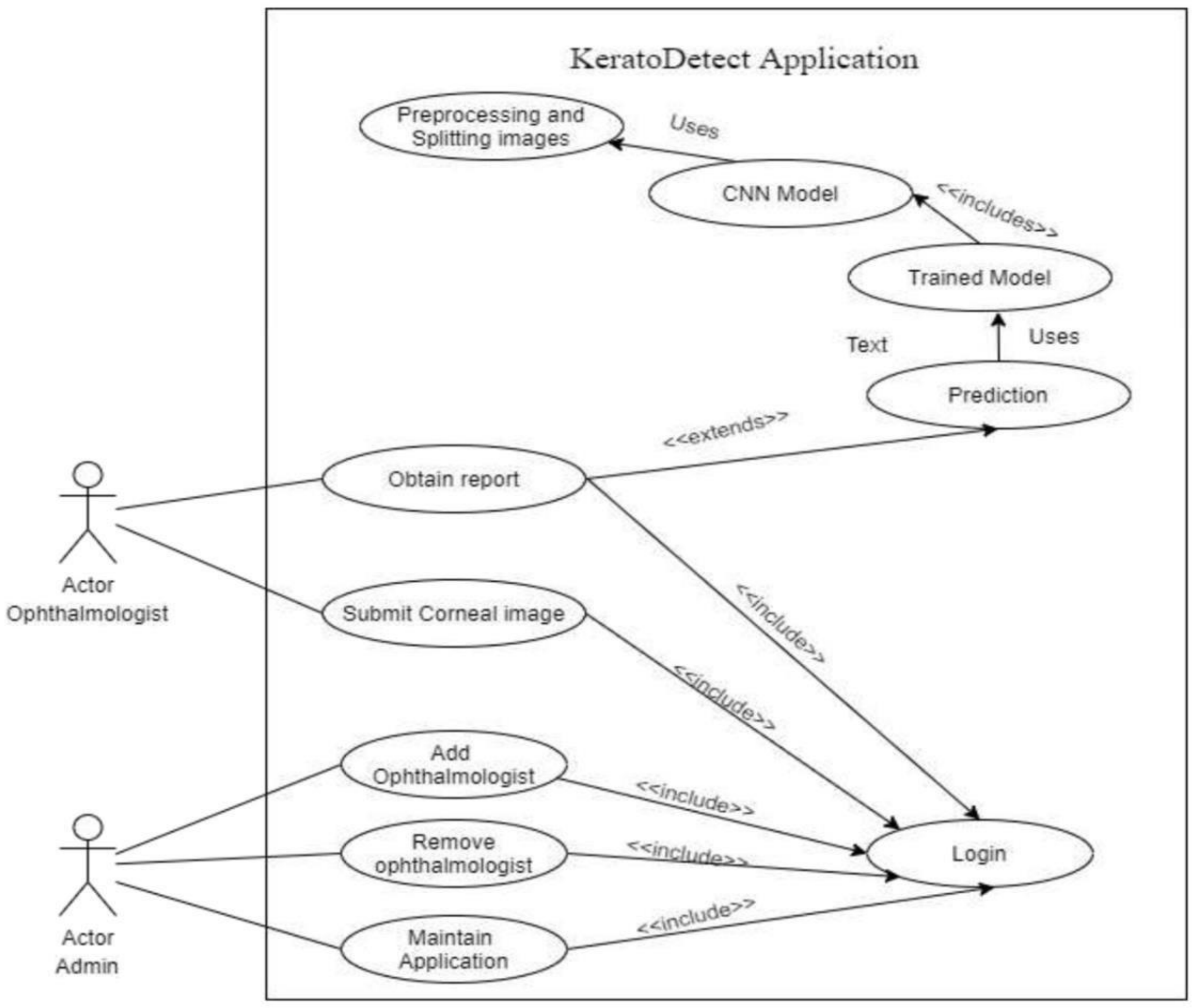


Fig 4.2 Use Case Diagram

4.3 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objectsand classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

4.3.1 Sequence Diagram for Preprocessing

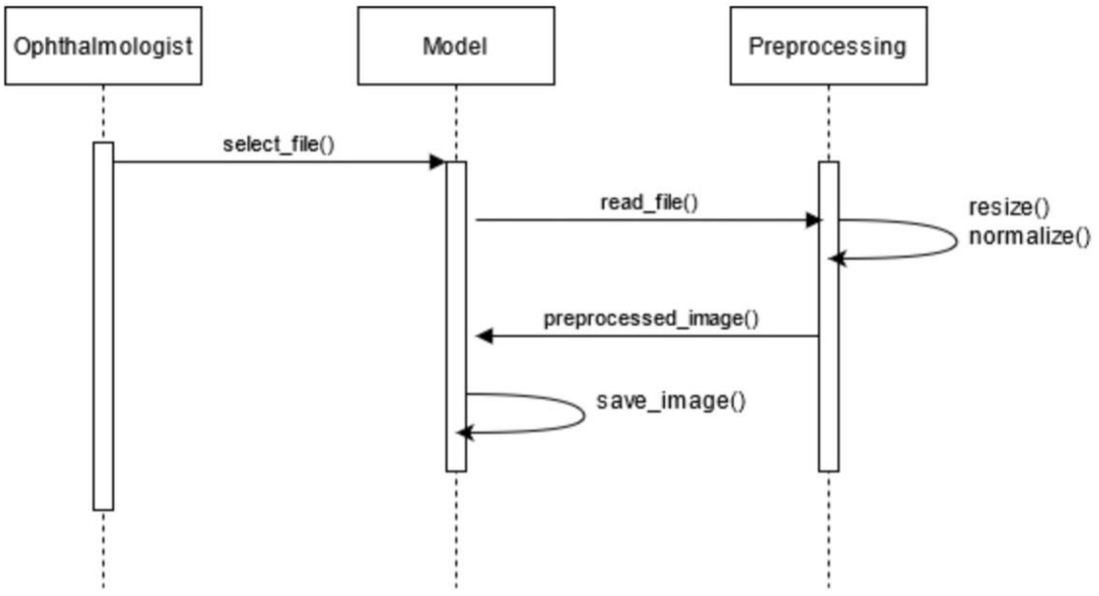


Fig 4.3.1: Sequence diagram for Preprocessing

The ophthalmologist will select the file which contain the images for preprocessing. The modelwill then read the images by using data loader function, these loaded images will be fed as inputto preprocessor. The first step in preprocessing stage is to resize the image, every image dimension will be transformed to 180 x 240 pixels. After resizing, the images willbe normalized along all three RGB (red, green, blue) color channels by calculating mean and standard deviation for the images loaded. Finally, the preprocessed images will be saved by model in separate folder.

4.3.2 Sequence Diagram for Splitting Dataset

Initially, the model will select path of folder where preprocessed images are saved. The data loader function will these images into the model. The dataset will be split into two set train setand test set for model training and evaluation. For splitting, random split function is used whichdivide the whole image set in 80:20 ratio i.e., train set will have 80% of images and test set willhave 20% of images. These train set and test set are returned to model after splitting.

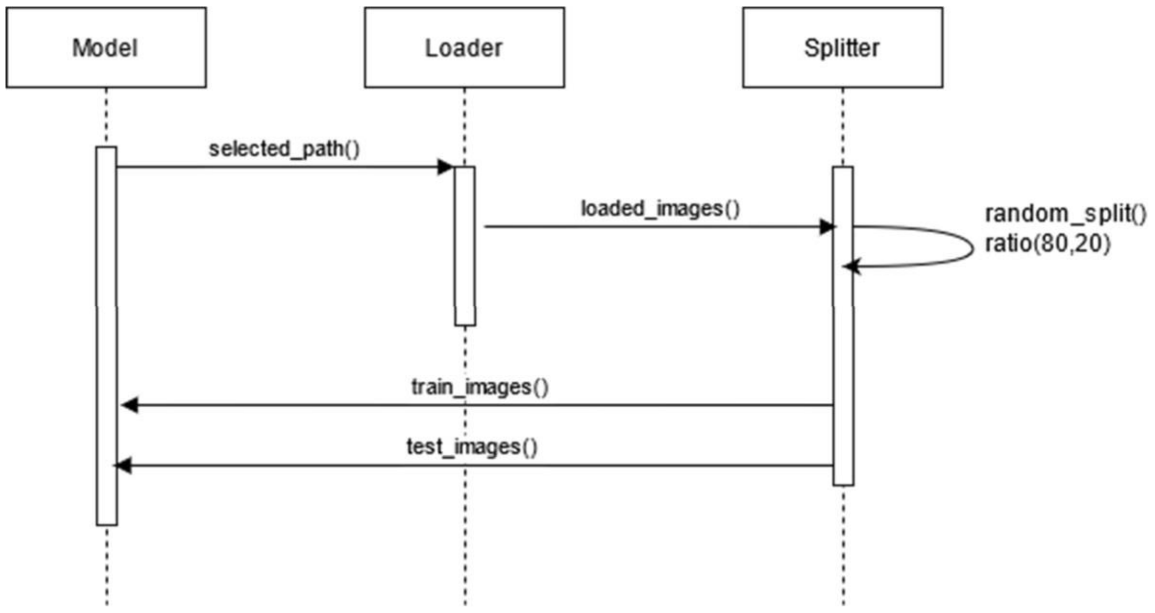


Fig 4.3.2 Sequence Diagram for Splitting Dataset

4.3.3 Sequence Diagram for Model Training

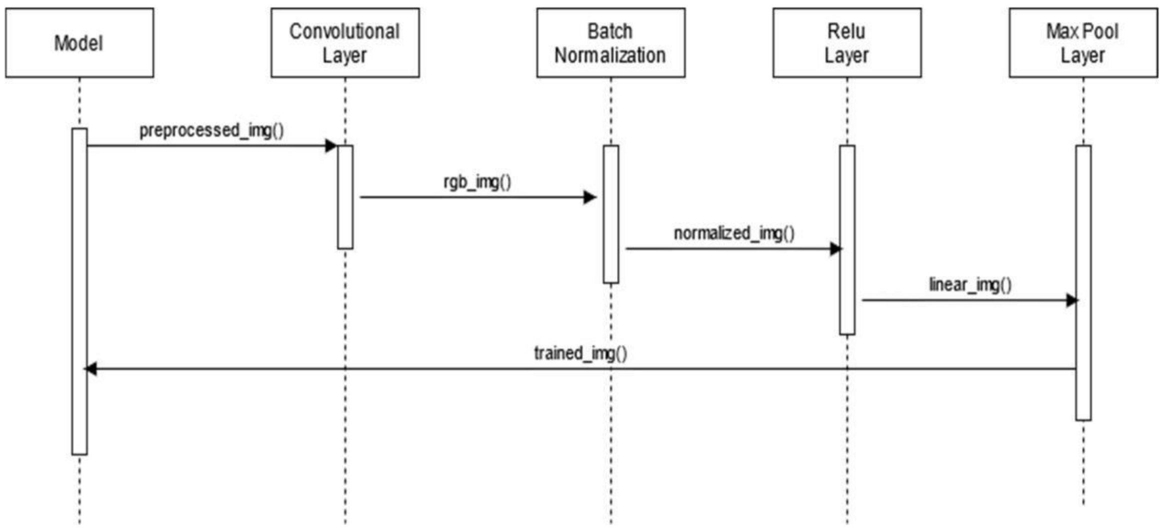


Fig 4.3.3 Sequence Diagram for Model Training

The ophthalmologist will select image that need to be tested after logging in the Application. The application will load this image using loader function and for preprocessing stage. The image will be preprocessed by resizing and normalizing. This image will be input for the trainedmodel, the model will output probabilities of target classes for the given image.

4.4 Activity Diagram

An activity diagram is a behavioral diagram i.e., it depicts the behavior of a system. An activitydiagram the control flow from a start point to a finish point showing the various decision pathsthat exist while the activity is being executed. We can depict both sequential processing and concurrent processing of activities using a activity diagram.

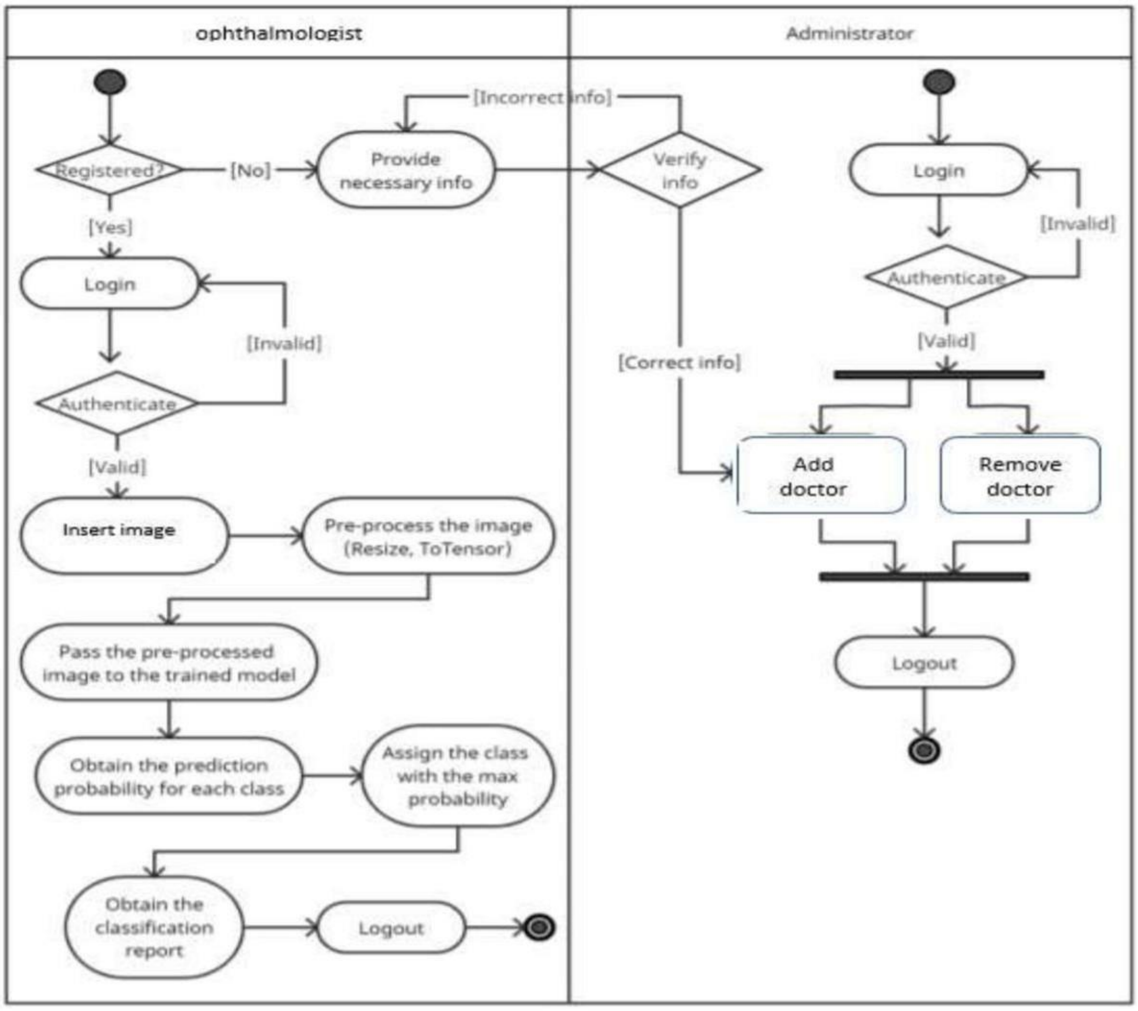


Fig 4.4 Activity Diagram

On the start of the application, the admin has to login by using their credentials like user ID and password. If they enter incorrect credentials, the login fails and they'll be asked toretry by providing valid credentials. Upon entering the valid credentials, the admin is considered as an authenticated user resulting in successful login. After logging in, the admin can add an ophthalmologist, remove an ophthalmologist and he's also responsible for maintaining the application. The admin collects all the information and then adds the ophthalmologist by entering the details in the form. They can also remove a ophthalmologist using the unique user ID provided to them initially.

The admin can also make changes to the application. After performing the necessary action, the admin logs out of the application thereby ending the session. The activity diagram of admin is illustrated as shown in Figure 4.4.

If the ophthalmologist is a new user, they should get registered by providing their details to the admin. The admin on adding the ophthalmologist provides them with the login credentials (user ID and password) for future use. The ophthalmologist then logs in to the application by entering their credentials. If they enter incorrect credentials, they'll be asked to retry with valid credentials. After the login is successful, the ophthalmologist inputs the corneal topography images as well as other details collected from the patient. The data is then fed to the application as test data. The CNN uses the pre-processed image and the data to perform analysis and gives out the output. After performing the necessary action, the ophthalmologist logs out of the application thereby ending the session. The activity diagram of the ophthalmologist is illustrated as shown in Figure 4.7.

4.5 State Chart Diagram

State Chart diagram describes different states of a component in a system. The states are specific to a component/object of a system. A State chart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events. They define different states of an object during its lifetime and these states are changed by events.

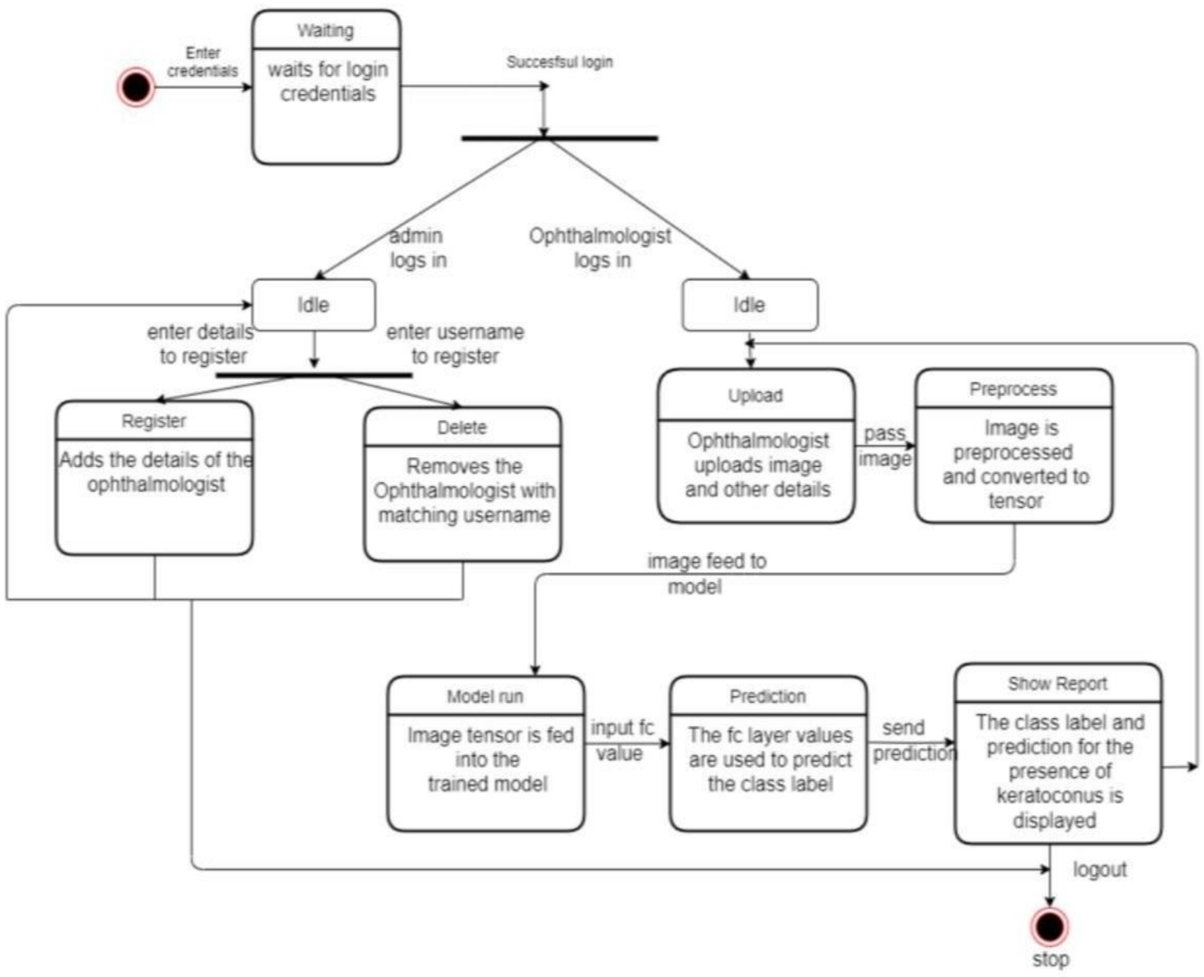

Enter credentials
Waiting
waits for login credentials
Succesfsul login
admin logs in
Ophthalmologist logs in
Idle
Idle
enter details to register
enter username to register
Register
Adds the details of the ophthalmologist
Delete
Removes the Ophthalmologist with matching username
Upload
Ophthalmologist uploads image and other details
pass image
Preprocess
Image is preprocessed and converted to tensor
image feed to model
Model run
Image tensor is fed into the trained model
input fc value
Prediction
The fc layer values are used to predict the class label
send prediction
Show Report
The class label and prediction for the presence of keratoconus is displayed
logout
stop

Fig 4.5 State Chart Diagram

when user tries to login using his credentials the database checks for his credentials in the database so it is called as prompt login and it will be in waiting state. Once after login he inputs the images and it takes some time to upload to the model and that phase will be in the waiting state and that stage is called as prompt for input images and then the preprocessing of the input images needs to take place and this stage is called perform input pre-process and halts there. Now the process inside the model starts and it acquires images from the dataset and sends preprocessed image set to the CNN model for training and then the halted process resumes and send the test input into CNN model for classification and then the result is evaluated and then the user logs out from the application.

4.6 Class Diagram

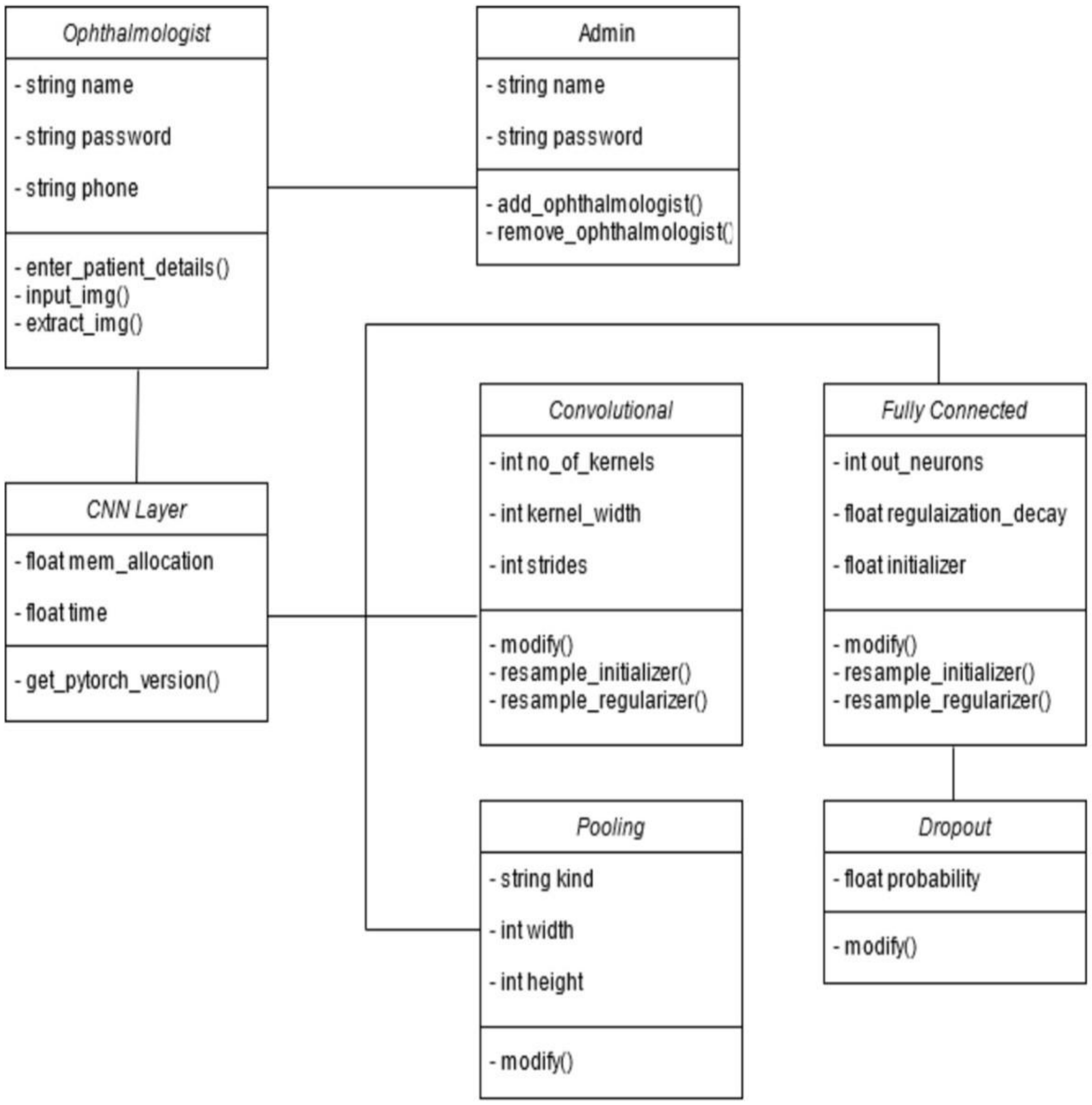


Fig 4.6 Class Diagram

The class diagram for this application is as shown in Figure 4.6. It consists of three classes namely Admin, Ophthalmologist and CNN Model. These three classes have a relationship of simple association that can be represented by a straight line between the classes Ophthalmologist- Admin and Ophthalmologist- CNN Model.

An object of Ophthalmologist has the properties such as Ophthalmologist ID, password, license number, name, phone, age, sex, address and qualification out of which the ID, password and license number should be made private. The Ophthalmologist can perform the operations such as entering and storing the patient data, inputting the topographic images and extracting the patient’s report. These operations can be performed by the Ophthalmologist only and hence are considered private.

The admin has the properties such as admin ID, name, email, phone and password. The sensitive data - ID and password are kept private. The admin maintains the application and hence can get the statistics. In addition to this, the admin can perform two major operations like Registering and Removing doctor (Ophthalmologist). Only the admin can perform these operations and thus they are considered private.

CNN model has its own properties like image pixels, number of topography images in the dataset, number of layers and learning rate which are private and are referred to as hyper-parameters in the context of Deep Learning. The CNN model can pre-process the lesion image and then classify the image into two classes. CNN model can be used by the dermatologist and the relation is represented by a simple association.

CHAPTER 5

IMPLEMENTATION

5.1 Import Libraries

```
import os
from flask import Flask
from flask import render_template, redirect
from flask import request
import torch
import torch.nn as nn
from PIL import Image
import torch.nn.functional as F
import torchvision.transforms as transforms
import mysql.connector
```

5.2 Image Preprocessing and its Parameters

```
class Net(nn.Module):
def __init__(self):
 super(Net, self).__init__()
 # convolutional layer (sees 180x240x3 image tensor)
 self.conv1 = nn.Conv2d(3, 16, 3, padding=1)
 self.bn1 = nn.BatchNorm2d(16)
 # convolutional layer (sees 90x120x16 tensor)
 self.conv2 = nn.Conv2d(16, 32, 3, padding=1)
 self.bn2 = nn.BatchNorm2d(32)
 # convolutional layer (sees 45x60x32 tensor)
 self.conv3 = nn.Conv2d(32, 64, 3, padding=1)
 self.bn3 = nn.BatchNorm2d(64)
 # max pooling layer
 self.pool = nn.MaxPool2d(2, 2)

 # linear layer (64*45*60 -> 10)

 self.fc1 = nn.Linear(64*45*60, 10)

  # linear layer (10 -> 2)

  self.fc2 = nn.Linear(10, 2)

  def forward(self, x):

   # Define forward behavior

   x = self.pool(F.relu(self.bn1(self.conv1(x))))

   #x = self.pool(F.relu(self.bn2(self.conv2(x))))

   x = self.pool(F.relu(self.bn2(self.conv2(x))))

   x = F.relu(self.bn3(self.conv3(x)))

   # flatten image input

   x = x.view(-1, 64 * 45 * 60)

   # linear layer, with relu activation function

   x = F.relu(self.fc1(x))

   # linear output layer

   x = self.fc2(x)

   return x
```

5.3 Convolutional Model and Filter Details

```
class Net(nn.Module):

def__init__ (self):

super(Net, self). init ()

# convolutional layer (sees 180x240x3 image tensor)

self.conv1 = nn.Conv2d(3, 16, 3, padding=1)
self.bn1 = nn.BatchNorm2d(16)
# convolutional layer (sees 90x120x16 tensor)
self.conv2 = nn.Conv2d(16, 32, 3, padding=1)
self.bn2 = nn.BatchNorm2d(32)
# convolutional layer (sees 45x60x32 tensor)
self.conv3 = nn.Conv2d(32, 64, 3, padding=1)
self.bn3 = nn.BatchNorm2d(64)
# max pooling layer
self.pool = nn.MaxPool2d(2,2)
# linear layer (64*45*60 -> 10)
self.fc1 = nn.Linear(64*45*60, 10)
# linear layer (10 -> 2)
self.fc2 = nn.Linear(10,2)
def forward(self, x):
## Define forward behavior
x = self.pool(F.relu(self.bn1(self.conv1(x))))
#x = self.pool(F.relu(self.bn2(self.conv2(x))))
x = self.pool(F.relu(self.bn2(self.conv2(x))))
x = F.relu(self.bn3(self.conv3(x)))
# flatten image input
x = x.view(-1, 64 * 45 * 60)
# linear layer, with relu activation function
x = F.relu(self.fc1(x))
# linear output
```

```
layer x = self.fc2(x)

return x

# instantiate the CNN and send it to cuda if available or to cpu

model = Net().to(device)

print(model)
```

5.4 Training the Model

```
losses = {'train':[], 'validation':[]}

def accuracy(outputs, labels,correct):

pred = outputs.data.max(1, keepdim=True)[1]

correct = np.sum(np.squeeze(pred.eq(labels.data.view_as(pred))).cpu().numpy())

return correct

def train(n_epochs, loaders, model, optimizer, criterion, gpu_available, save_path):

# initialize tracker for minimum validation loss

valid_loss_min = np.Inf # history =[]

#result =[[]]

for epoch in range(1, n_epochs+1):

# initialize variables to monitor training and validation loss train_loss = 0.0

valid_loss = 0.0

acc = 0.0 model.train()

for batch_idx, (data, target) in enumerate(loaders['train']): # move to GPU

if gpu_available:

data, target = data.cuda(), target.cuda() optimizer.zero_grad()

# forward pass: compute predicted outputs by passing inputs to the

model output = model(data)
```

```
# calculate the batch loss

loss = criterion(output, target)

# backward pass: compute gradient of the loss with respect to model parameters
loss.backward()

# perform a single optimization step (parameter update) optimizer.step()

# update average training loss

train_loss = train_loss + ((1 / (batch_idx + 1)) * (loss.data - train_loss)) model.eval()

for batch_idx, (data, target) in enumerate(loaders['valid']): # move to GPU

if gpu_available:

data, target = data.cuda(), target.cuda() # update the average validation loss

# forward pass: compute predicted outputs by passing inputs to the model output =
model(data)

# calculate the batch loss

loss = criterion(output, target)

# update average validation loss

valid_loss = valid_loss + ((1 / (batch_idx + 1)) * (loss.data - valid_loss))
losses['train'].append(train_loss)

losses['validation'].append(valid_loss)

# calculate validation accuracy for each epoch acc += accuracy(output, target,0.)

# print training/validation statistics

print('Epoch: {} \tTraining Loss: {:.6f} \tValidation Loss: {:.6f} \tValidation Accuracy:
{:.6f}'.format( epoch, train_loss, valid_loss, acc * 100 / 492 ))

#result[train_loss] = train_loss

#result[valid_loss] = valid_loss #result[acc] = acc #history.append(result)

# save the model if validation loss has decreased
```

```
if valid_loss <= valid_loss_min:

print('Validation loss decreased ({:.6f} --> {:.6f}). Saving model ...'.format( valid_loss_min,

valid_loss))

torch.save(model.state_dict(), 'model.pt')

valid_loss_min = valid_loss

# return trained model return model

# train the model

model = train(10, loaders, model, optimizer, criterion, gpu_available, 'model.pt')
```

5.5 Flask Implementation

```
from flask import Flask, render_template, request import mysql.connector

app = Flask(   name    )

conn = mysql.connector.connect(

host="localhost", user="root", password="", database="ktc_project") cursor =
conn.cursor(buffered=True)

#load saved model model = Net()

model.load_state_dict(torch.load("model.pt")) model.eval()

# image -> transform

def transform_image(image): transform_test = transforms.Compose([
transforms.Resize((180, 240)), transforms.ToTensor(),

transforms.Normalize((0.8629, 0.8765, 0.63575),(0.2921, 0.2525, 0.4618)) ])

return transform_test(image)

# prediction

def get_prediction(img, model):

# Convert to a batch of 1 xb = img.unsqueeze(0)

# Get predictions from model yb = model(xb)
# Pick index with highest probability_,

preds = torch.max(yb, dim=1)

# Retrieve the class label

return preds.item()

@app.route('/', methods=["GET", "POST"]) def home():

return render_template("index.html")
```

5.6 KeratoDetect Homepage Template

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta content="width=device-width, initial-scale=1.0" name="viewport">

<title>KeratoDetect</title>

<meta content="" name="description">

<meta content="" name="keywords">

<header id="header" class="fixed-top">

<div class="container d-flex align-items-center">

<h1 class="logo mr-auto"><a href="index.html">KeratoDetect</a></h1>

<a href="index.html" class="logo mr-auto"><img src="../static/img/logo.png" alt=""
class="img-fluid"></a>

<nav class="nav-menu d-none d-lg-block">

<ul>

<li class="active"><a href="/">Home</a></li>

<li><a href="#about">About</a></li>
```

```
</ul>

</nav>

<a href="#login" class="appointment-btn scrollto">Login</a>

</div>

</header>

<section id="hero" class="d-flex align-items-center">

<div class="container">

<h1>Welcome to KeratoDetect Application</h1>

<h2>A personalized web-app to predict the existence of Keratoconus disease</h2>

</div>

</section>

<h3>About the Application</h3>

<p>Keratoconus (KTC) is a non-inflammatory disorder characterized by progressive
thinning, corneal deformation and scarring of the cornea.<p>

<p>The application analyzes these corneal topographic images of the eye using a
convolutional neural network (CNN) to extract the learning features and classifying images
as normal eye or keratoconus eye providing maximum accuracy.</p>

<p>KeratoDetect can assist the ophthalmologist in rapid screening of its patients, thus
reducing diagnostic errors and facilitating treatment.</p>

<section id="login" class="appointment section-bg">

<div class="container" data-aos="fade-up">

<div class="section-title">

<h2>Enter Login Details</h2>

<p>Only Ophthalmologist and Admins can login. If you're a new ophthalmologist, then
kindly contact the admin.</p>

</div>
```

```
<form action="/login" method="post" data-aos="fade-up" data-aosdelay="100">
<div class="row">
<div class="col-md-4 form-group">
<input type="email" name="email" class="form-control" id="email" placeholder="E-mail"
required>
</div>
<div class="col-md-4 form-group mt-3 mt-md-0">
<input type="password" class="form-control" name="password" id="pass"
placeholder="Password" required>
</div>
<div class="col-md-2 form-group mt-3 mt-md-0">
<input type="checkbox" name="admin" id="admin" />&nbsp; Admin
</div>
<div class="col-md-2 form-group mt-3 mt-md-0">
<input type="checkbox" name="ophthalmologist" id="ophthalmologist"
/>&nbsp; Ophthalmologist
</div>
</div><br/>
<div class="text-center"><button class="btn btn-primary"
type="submit">Submit</button></div>
<br/><br/><br/><br/></form></div>
```

CHAPTER 6

RESULTS

The purpose of the discussion of results is to interpret and describe the significance of one's findings considering what was already known about the problem being investigated and explain any new understanding or fresh insights about the situation after one has considered the results. The discussion will always connect to the introduction through the questions or hypotheses posed and the literature reviewed.

Web Performance Analysis is the practice of analysing and monitoring websites to make sure they display the essential content as quickly as possible, load their pages at consistent speeds, and don't suffer unexpected downtimes. Performance Analysis concerning machine learning is concerned with measuring the correctness of a developed model. It involves the process of comparing the results generated by the model with that of the ground truth. Classification matrix, Accuracy, Precision, F1 score, and ROC-AUC are some of the evaluation metrics used to analyze the model's performance.

6.1 User Documentation

User documentation refers to the documentation for a product or service provided to the end-users. The user documentation is designed to assist end-users in using the product or service. This is often referred to as user assistance. The user documentation is a part of the overall product delivered to the customer. In the case of our project, end users can be categorized into two groups:

- Admin
- Ophthalmologist

So, it is essential to have the user documentation for both the end-users separately since they have their functionalities. Our web application allows the admin to use the below mentioned functionalities:

- Login: The admin login is necessary to use the admin-specific functionalities like registering and removing the ophthalmologist.

Clicking the Admin Login button allows the user to log in as the admin.

- **Add Ophthalmologist:** This allows the admin to register the ophthalmologist by entering the details of the ophthalmologist such as name, email-id, password, date of birth, and address andthen clicking the Create button.

- **Remove Ophthalmologist:** The admin can remove the ophthalmologist by entering their username/email-id and then clicking the Remove button.

- **Logout:** The admin can end the session by logging out of the application by clicking the Logout button.

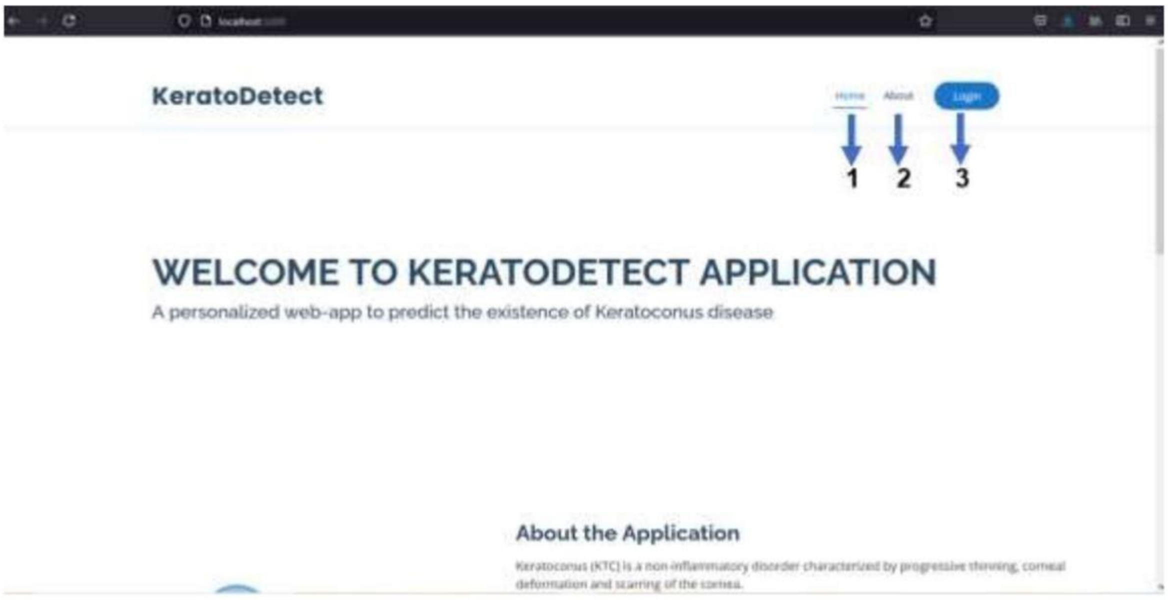
- The focus of our web application is on the prediction of the status of the eye i.e., whether it is infected eye (Keratoconus) or normal eye by uploading the Corneal Topography image. Thiscan only be done by using the ophthalmologist login. The functionalities provided for the ophthalmologist are listed below:

- **Login:** The ophthalmologist login is necessary to use the KeratoDetect application to predictthe eye status. Clicking the Login button allows an ophthalmologist to enter his credentials andlog in successfully.

- **Diagnose:** This functionality allows the ophthalmologist to enter the patient details such as patient name, age, gender, email-id, phone number and to attach an image of Corneal Topography that must be uploaded by clicking the Upload button. The ophthalmologist gets the eye status of that patient.

- **Logout:** The ophthalmologist can end their session by logging out of the application by clicking the Logout button.

The buttons mentioned above are placed in specific web pages based on their requirements. The user can get a detailed understanding of navigating throughout the web application by referring to the figures below.



WELCOME TO KERATODETECT APPLICATION

A personalized web-app to predict the existence of Keratoconus disease

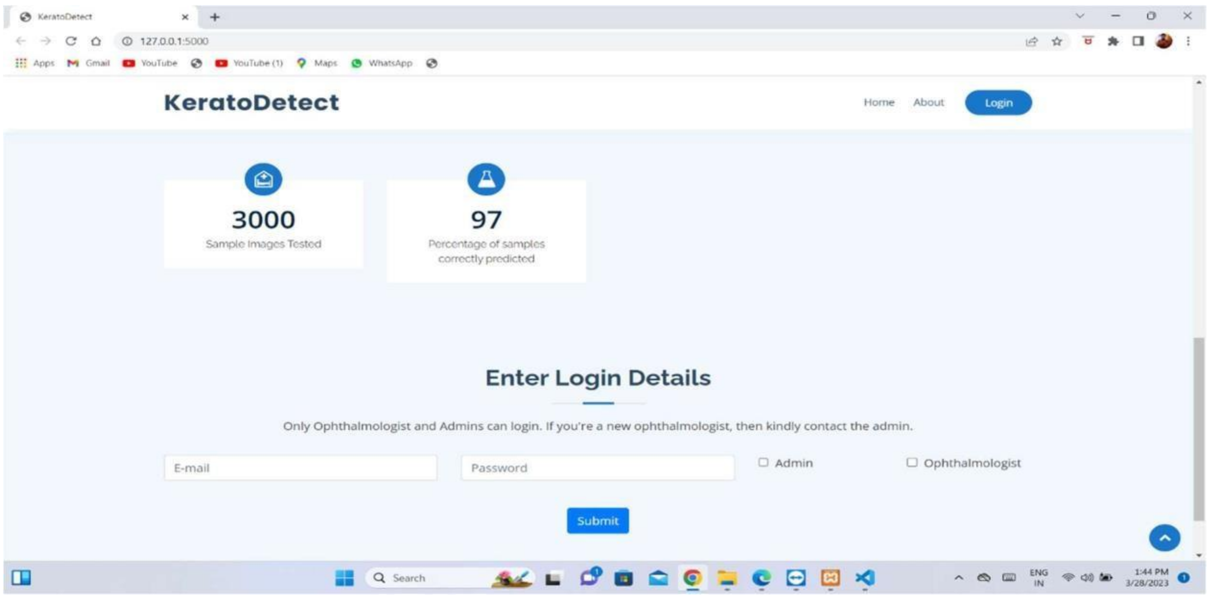
About the Application

Keratoconus (KTC) is a non-inflammatory disorder characterized by progressive thinning, corneal deformation and scarring of the cornea.

Fig 6.1 Home Page

The Web application called "Kerato Detect" home page is shown in above figure 3.8.The button reloads to same home page of the application. The button takes to the aboutsection of the application which provides the information about the Keratoconus disease. The button takes you to the login section of the application.

The Web application can be accessed by registered ophthalmologist as well as admin of the website. The admin will have a defined email-id and password to log in to theapplication. The admin can only log in only through the admin login checkbox present on the login form.

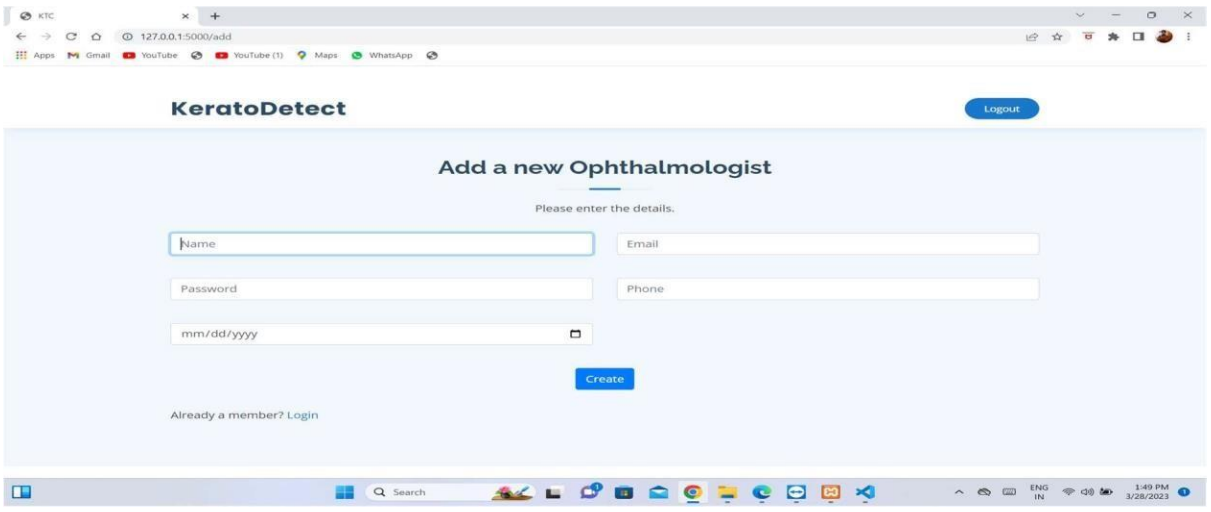


Enter Login Details

Only Ophthalmologist and Admins can login. If you're a new ophthalmologist, then kindly contact the admin.

Fig 6.2 Login Form

If an ophthalmologist is already registered with the application, he/she can log in by checking the ophthalmologist checkbox . If the ophthalmologist is not registered, they can be registered by contacting the admin. To perform the prediction, the ophthalmologist who is logged in to the system can use the application for disease prediction.

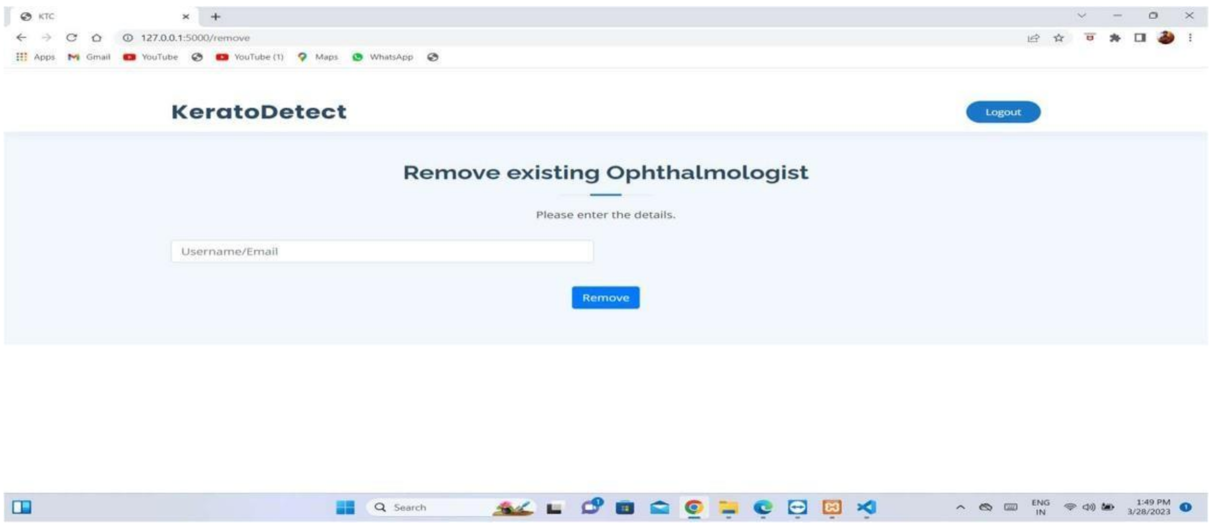


Add a new Ophthalmologist

Please enter the details.

Already a member? Login

Fig 6.3 Admin Home Page



Remove existing Ophthalmologist

Please enter the details.

Fig 6.4 Ophthalmologist Remove page

The admin has privileges to make changes to the application, such as adding a new ophthalmologist by providing the details of the ophthalmologist to the application, and deleting the ophthalmologist profile from the application. If the admin deletes the pulmonologist profile, then they cannot access the application using their credentials.

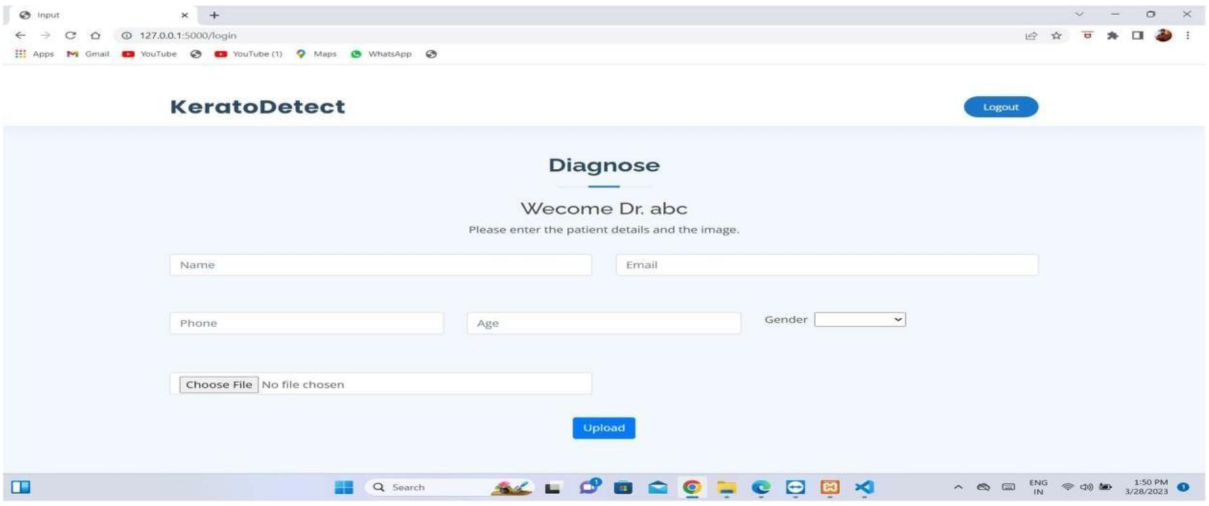
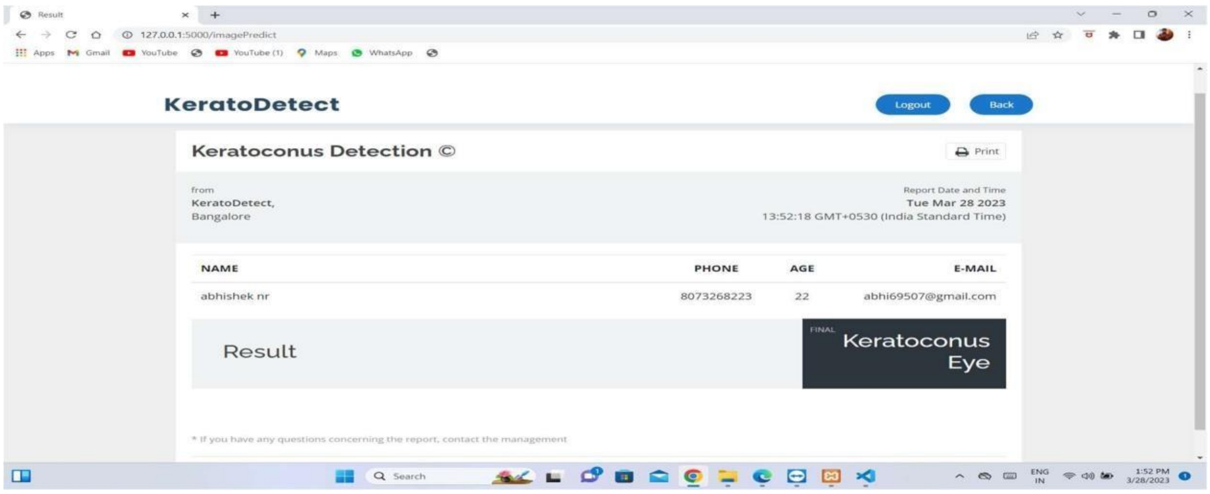


Fig 6.5 Ophthalmologist Home Page

The primary purpose of the ophthalmologist in this application is to predict the eye status of the patient. For the ophthalmologist to perform prediction, they must enter the patient's details in the form patients' details include patient name, email-id, phone number, age, gender. The ophthalmologist also needs to upload a Corneal Topography image of the patient .



Keratoconus Detection ©

from
KeratoDetect,
Bangalore

Report Date and Time
Tue Mar 28 2023
13:52:18 GMT+0530 (India Standard Time)

NAME	PHONE	AGE	E-MAIL
abhishek nr	8073268223	22	abhi69507@gmail.com

Result — FINAL Keratoconus Eye

* If you have any questions concerning the report, contact the management

Fig 6.6 Result Prediction Page

After entering the patient details in the form and uploading the Corneal Topography image , the output prediction can be seen in prediction page and the predicted status of the eye is indicated in that page.

CONCLUSION

The main contribution of this paper is represented by the: implementation and testing of an algorithm that facilitates the diagnosis of advanced as well as early keratoconus. This instrument has to come in the help of the ophthalmologist and to facilitate the detection of keratoconus by recognizing specific corneal patterns which cannot be seen by the untrained eye. Thus, adequate treatment can be applied at the correct time, contributing to the long-term management of the illness by slowing down or even stopping the progression of keratoconus, thus greatly improving the patient’s quality of life. Oftentimes, keratoconus sets in during the years prior to puberty, thus affecting children the most. Thus, it is crucial to develop and implement new methods which can lead to easier detection and diagnosis processes, offering the patients the chance of a normal life. Another contribution of this work includes the analysis of the keratoconus illness and the evaluation of the different methods and algorithms used in the diagnostic process. The aspect of analyzing different parameters that can be integrated in an automated diagnosis system is an open one because what is needed is the continuous improvement of relevant data that can come in handy when it comes to the detection of keratoconus at an early stage.

REFERENCES

[1] V. Galvis, T. Sherwin, A. Tello, J. Merayo, R. Barrera, and A. Acera, "Keratoconus: an inflammatory disorder?," Nature Eye, vol. 29, no. 7, pp. 843–859, 2015.

[2] A. Salmon, D. Chalk, K. Stein, and N. A. Frost, "Cost effectiveness of collagen crosslinking for progressive keratoconus in the UK NHS," Nature Eye, vol. 29, no. 11, pp. 1504–1511, 2015

[3] Ahmad R. Dhaini, Manal Chokr, Sara Maria El-Oud, Maamoun Abdul Fattah, and Shady Awwad, "Automated Detection and Measurement of Corneal Haze and Demarcation Line in Spectral-Domain Optical Coherence Tomography Images" , 2018 (IEEE)

[4] Sahar Jorjandi, Hossein Rabbani, RahelehKafieh, Zahra Amini, "Statisticalmodelling of Optical Coherence Tomography images by asymmetric Normal Laplace mixture model", 2.17(IEEE), DOI: 10.1109/EMBC.2017.8037831

[5] AlexandruLavric, Valentin Popa, Hidenori Takahashi and SiamakYousefi, "Detecting Keratoconus from Corneal Imaging Data using Machine Learning", 2020 (IEEE), DOI: 10.1109/ACCESS.2020.3016060

[6] Francisco Cavas-Martnez1, Laurent Bataille, Daniel G "A new approach to keratoconus detection based on corneal morphogeometric analysis", 2017, DOI: 10.1371/journal.pone.0184569

[7] Hitesh Sonar, Ankit Kadam, Prasad Bhoirand Dr. Bharti Joshi, "Detection of KeratoconusDisease", 2020, DOI: 10.1051/itmconf/20203203019

[8] Roberta Lopes, João Marcelo Lyra, Guilherme Ribeiro, and Renato Ambrósio Jr, "An application of algorithm based on abstract data types to the keratoconus diagnosis", 2015,DOI: 10.1109/ANTHOLOGY.2013.6784907

[9] Roberta Lopes, João Marcelo Lyra, Guilherme Ribeiro, and Renato Ambrósio Jr, "An application of algorithm based on abstract data types to the keratoconus diagnosis", 2015, DOI:10.1109/ANTHOLOGY.2013.6784907