

OCUAI – OCCULAR DISEASE DETECTION

PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology

(Deemed to be University)

Submitted in partial fulfilment of the requirements for the award
Of Bachelor of Technology Degree in Information Technology

By

ALINA MUSHTAQ

40120008



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119. TAMILNADU

April 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" Grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **ALINA MUSHTAQ (Reg. No: 40120008)** who carried out the project entitled "**OCUAI**" under the supervision from January 2023 to April 2023

A. Sivasangari
Internal Guide

Dr. A. Sivasangari M.E., Ph.D.,

R. Subhashini
Head of the Department

Dr. R. Subhashini M.E., Ph.D.,

Submitted for Viva voce Examination held on 29/04/2023

Internal Examiner

External Examiner

DECLARATION

I hereby declare that the Project entitled “OcuAI – Ocular disease detection” done by me under the guidance of **Dr.A.Sivasangari** at Sathyabama Institute of Science and Technology (Deemed to be University), Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai –600119 is submitted in partial fulfilment of the requirements for the award of **Bachelor of Technology degree in Information Technology**.

DATE : 29/04/2023

Place : Chennai

A handwritten signature in black ink, consisting of stylized letters and a flourish.

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to Board of management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey our thanks to **Dr. T. SASIKALA M.E., Ph.D., Dean, School of Computing** and **Dr. R. SUBHASHINI M.E., Ph.D., Head of the Department, Department of Information Technology** for providing us the necessary support and details at the right time during progressive reviews.

I would like to express our sincere and deep sense of gratitude to our Project **Dr.A .Sivasangari M.E., Ph.D.,** for her valuable guidance suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express our thanks to all Teaching and Non-teaching staff members of the Department of INFORMATION TECHNOLOGY who were helpful in many ways for the completion of the project.

ABSTRACT

OcuAI – Ocular Disease Detection

The main purpose of this project on OcuAi is to provide a diagnosing tool to detect eye diseases, from eye-fundus images using neural networks, which will be user friendly and easy to use for health care professionals and health care organisations.. The Application is programmed using Python with the concept of InceptionNet in Deep learning.

The Inception v3 model fetches the correct information of the object from the fundus retinal image training dataset to improve its disease detection proficiency. Due to the enhanced image classification capabilities of this model, such a system can finally be realized.

Once trained, the neural network can analyze new images and accurately detect and diagnose ocular diseases, often outperforming human experts. It also has the potential to improve access to healthcare in underserved communities by providing a cost-effective and scalable solution for early disease detection.

The diseases that can be detected:

- Cataract (C),
- Diabetes (D),
- Glaucoma (G),
- Age related Macular Degeneration (A),
- Hypertension (H),
- Pathological Myopia (M),

TABLE OF CONTENTS

Chapter	Title	Page No.
1	Introduction	1
2	Aim and Scope	2
	2.1. Aim	2
	2.2. Problem Statement	2
	2.3. Scope	2
3	Literarture Survey	3
4	Metholody and Implementation	6
	4.1. Architecture	6
	4.2. Neural Network	7
	4.2.1. Tensorflow	7
	4.2.2. Inception v3	7
	4.3. Dataset	9
	4.4. Diseases that can be detected	11
	4.4.1. Cataract	11
	4.4.2. Glaucoma	11
	4.4.3. Myopia	11
	4.4.4. Diabetes	12
	4.4.5. Age related Macular Diseease	12
	4.4.6. Hypertension	13
	4.5. Data Processing	13
	4.6. Flask	15
	4.7. Web Application UI	15

5	Experimental Result and Output	16
6	Conclusion and Future works	18
	Reference	19
	Appendix	20
	A.1. Screenshots	
	A.2. Codes	

LIST OF FIGURES

Figure no.	Figure name	Page.no
Fig. 4.1	Architecture	6
Fig. 4.2	Dataset – Fundus Images	10
Fig. 4.3	Dataset Information in .xls file	10
Fig. 4.4	Normal Eye	11
Fig. 4.5	Glaucoma	12
Fig. 4.6	Myopia	12
Fig. 4.7	Diabetes	13
Fig. 4.8	AMD	13
Fig. 4.9	Hypertension	14
A.1	Login Page	20
A.2	Patient Entry Page	21
A.3	Scanning page	22
A.4	Image Scanned	23
A.5	Final Report	24

CHAPTER - 1

INTRODUCTION

Various ocular diseases are capable of causing permanent and irreversible damage to the patient's vision, and in extreme cases, it can even lead to blindness. Although effective treatments are available for these ocular diseases, these treatment options can only be implemented if the disease is diagnosed as early as possible. Ocular diseases are primarily diagnosed using colour fundus photography or CFP. This technique is utilized in order to record the interior surface of the human eye so that various types of possible ocular diseases can be detected.

Although this method of diagnosis is effective, it's still quite difficult to detect certain ocular diseases using CFP. Some of the most prevalent ocular diseases, such as cataracts, myopia, and diabetic retinopathy are difficult to diagnose as they show very few initial symptoms. Moreover, the process of manually inspecting and detecting ocular diseases is a laborious task, and this process is not that accurate.

In recent years, neural networks have emerged as a powerful tool for ocular disease detection. Deep learning-based neural network models have shown promising results in medical image classification and object detection. Moreover, that is why convolutional neural network-based models have been extensively used for ocular disease detection.

CHAPTER – 2

AIM AND SCOPE

2.1.Aim

The Aim of this project is to develop a web application to detect Ocular disease from the fundus image using Inception(convolutional neural network model) of deep learning.

2.2. Problem Statement

Early ocular disease detection is an economic and effective way to prevent blindness caused by diabetes, glaucoma, cataract, age-related macular degeneration (AMD), and many other diseases. Rapid and automatic detection of diseases is critical and urgent in reducing the ophthalmologist's workload and prevents vision damage of patients. So there is a need of Computer vision to automatically detect the disease more accurately with the help of neural network.

2.3. Scope

The main purpose of this project is to provide health care professionals and health care organisations, a diagnosing software tool that detect the disease like cataract, glaucoma, diabetes, and etc., with more accuracy and efficiency.

CHAPTER – 3

LITERATURE SURVEY

This section reviews the existing diagnosis systems for ODs. We discuss the current limitations and highlight the main directions and remedies suggested in the proposed system to overcome the current shortcomings. For example, He et al. [8] presented a CAD system using a dense correlated network (DCNet) to classify color fundus images. They used a public dataset (ODIR 2019) based on seven types of ODs using ML-C. The authors utilized two fully connected layers with rectified linear unit (ReLU) activation in one of them. One dense layer is 512, and the other is 8 as the number of the output categories of the ODs. They employed an ML soft margin loss function. The main advantage of their method is that it can be used in multi-modal image analysis. However, they could not compare their model with other existing works because their method is patient-based, and the other studies are image-based. They shared the same backbone CNN to extract

features from the right and left eyes to reduce the computation complexity. Hence, they could not handle the unbalanced distribution of patient cases.

Wang et al. utilized transfer learning to extract features of the colour fundus images and then utilized ML-C based on problem transformation. The authors utilized a multi-label dataset with eight labels. They utilized histogram equalization on gray and coloured images. Then, they applied two classification models to the two images sets. Finally, they obtained the average of the sigmoid output probabilities from the two models. The main limitation of their work is the low network performance because of the variety of uncommon ODs categorized in the label “other diseases” in the utilized dataset. In addition, their system suffers from the data imbalance problem due to the limited data in some disease categories. Hence, some of the specific features learned are unknown.

Cheng et al. used a graphical convolution network (GCN) to detect eight DR lesions (laser scars, drusen, cup disc ratio, haemorrhages, retinal arteriosclerosis, microaneurysms, hard and soft exudates) from colour fundus images. They utilized ResNet-101,

then two convolutional (CONV) layers with a 3×3 kernel, stride 2, and adaptive max

pooling for feature extraction. Their model's accuracy (ACC) and receiver operating characteristic (ROC) values showed better detection results for laser scars, drusen, and haemorrhage lesions. In contrast, their system had poor detection ability for microaneurysms, soft exudates, and hard exudates. Thus, the model could not distinguish microaneurysms from the background of the fundus images. On the other hand, soft and hard exudate lesions often accompanied multiple other fundus lesions simultaneously, which made it difficult for the model to extract the features of all fundus lesions.

Dipu et al utilized transfer learning to detect eight ODs from the ODIR2019 dataset. They compared the performance of some state-of-the-art DL networks, such as

Resnet-34, EfficientNet, MobileNetV2, and VGG-16. The authors trained the cutting-edge on the utilized dataset and reported the results. They evaluated the models' performance by estimating the ACC. They ordered the models due to the resulting ACC into VGG-16, Resnet-34, MobileNetV2, and EfficientNet. However, the authors did not build a new model to detect the ODs. Moreover, calculating only ACC was not enough to estimate the model performance.

Choi et al. proposed a CAD system using random forest transfer learning based on VGG-19. They utilized a small dataset in order to detect ten OD categories. They observed that the ACC increases by decreasing the classes that need to be detected to three. On the contrary, when they increased the categories to ten, the ACC decreased with a difference of about 30%. The authors tried to use an ensemble classifier with transfer learning and found a slight increase in ACC of 5.5%. Although the authors made augmentation, they could not achieve good performance because of the data imbalance.

Diaz Pinto et al. utilized DL to detect glaucoma from colour fundus images. First, they cropped the images

around the optic disc. Then, other transformation processes were applied, such as random rotations, zooming by a range between 0 and 0.2, and horizontal and vertical flipping. They utilized VGG16, VGG19, InceptionV3, ResNet50, and Xception. Each architecture was followed by global average pooling. They used the soft max classifier to ML- or binary classification-based models because of overlapping labels.

Tan et al. classified AMD by using a CNN. Their proposed model consists of seven CONVs with a 3×3 kernel, four max-pooling (MP) (2×2), and three fully connected (FC) layers. The proposed model was fully automatic so that no hand-crafted feature extraction or selection was

From the previous review of the current studies conducted recently, we can conclude their main limitations in diagnosing multiple ODs based on the ML-C concept as follows:

- Increasing the number of classes decreases the model performance, mainly if the number of training samples is not sufficiently large;
- Some systems are conservative and cannot be applied in the real world because of the imbalanced and/or insufficient datasets;

required. Moreover, no classifier was required as they involved meticulous engineering in designing a feature extraction module that could extract highly distinctive features for classification. The proposed model can be installed in a cloud system. On the other hand, the authors highlighted the issues of their model, as the overall diagnostic performance of their CNN model was poor and would improve with more extensive data. Furthermore, the proposed model suffered from convergence and overfitting problems. In addition, the training of the CNN model was slow and intensive. A summary of the most recent studies (published from 2018 to 2021

- Some models suffer from overfitting;
- The overall performance of the ML-C based models is minor compared.

To overcome the previous limitations, I propose an Inception v3 model to accurately diagnose the different ODs from various ML colour fundus images. I apply some augmentation processes to enlarge the training dataset and avoid overfitting and data imbalance issues. On the other hand, data augmentation improves the model performance

CHAPTER - 4

METHODOLOGY AND IMPLEMENTATION

4.1.ARCHITECTURE

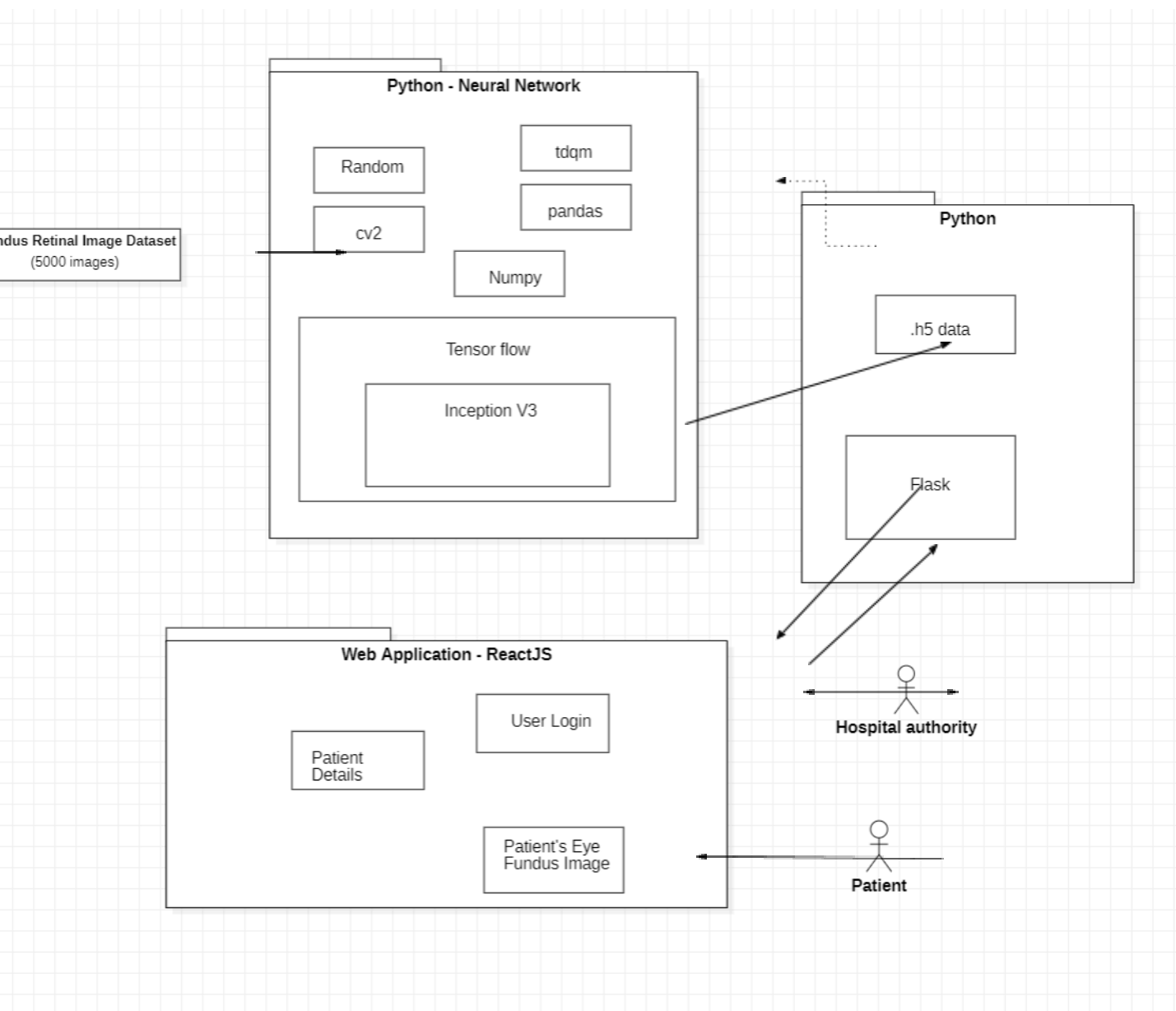


Fig.4.1 Architecture

4.2. NEURAL NETWORK

A neural network is a type of machine learning algorithm inspired by the structure and function of the human brain. It consists of interconnected nodes or neurons that are organized into layers.

The input layer receives data, which is then processed by one or more hidden layers. The output layer produces the final result. During training, the neural network adjusts the weights and biases of the connections between neurons in order to improve its accuracy.

Neural networks are used in a wide range of applications, such as image and speech recognition, natural language processing, and predictive analytics.

4.2.1. Tensorflow

TensorFlow is an open-source software library for dataflow and differentiable programming across a range of tasks. It is a popular choice for building neural networks and other machine learning models. In TensorFlow, a neural network is typically built by defining a computational graph that represents the operations and transformations applied to the input data to produce the desired output. This graph is then executed using a TensorFlow session, which performs the necessary computations and updates the network's parameters to improve its performance. TensorFlow also provides a range of tools and utilities for working with neural networks, including pre-built models, data pre-processing and augmentation tools, and visualization tools for inspecting and debugging network behaviour.

4.2.2. Inception V3

Inception-v3 is a convolutional neural network architecture that was developed by Google as part of the ImageNet Large-Scale Visual Recognition

$$\begin{aligned} A_X &= \begin{bmatrix} A_{1,1} & \dots & A_{1N} \\ A_{21} & \dots & A_{2N} \\ \vdots & \ddots & \vdots \\ A_{M1} & \dots & A_{MN} \end{bmatrix} * \begin{bmatrix} B_{1,1} & \dots & B_{1N} \\ B_{21} & \dots & B_{2N} \\ \vdots & \ddots & \vdots \\ B_{M1} & \dots & B_{MN} \end{bmatrix} \\ &= \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} A_{(M-1),(N-j)} B_{(i+1),(j+1)} \end{aligned}$$

Challenge in 2015. Inception-v3 is a 48-layer deep pre-trained convolutional neural network model, as shown in Eq. 1 and it is able to learn and recognize complex patterns and features in medical images. One of the key features of Inception V3 is its ability to scale to large datasets and to handle images of varying sizes and resolutions. This is important in the field of medical imaging, where images can vary greatly in terms of size, resolution, and quality. Typically, the Inception module includes one maximum pooling and three convolutions of various sizes (Ting et al., 2017). After the convolution operation, the channel is aggregated for the network output of the preceding layer, and the non-linear fusion is then carried out. In this model, over-fitting can be avoided while enhancing the network's expression and flexibility to various scales.

To avoid overfitting, we flattened the output layer and reduced its dimensions to one. We then added a sigmoid layer for classification, as well as a fully connected layer with 1,024 hidden units, a Relu activation function as shown in Eq. 2, and a dropout with the rate of 0.4. The weights of the classification layers were initialized using the algorithm described in (Beauxis-Aussalet and Hardman, 2014) as shown in Eq. 3. This approach allowed us to effectively use Inception V3 for our purposes.

Where $e \sim U(-a, b)$ is a uniform distribution in the interval $[-a, b]$, m is the size of the previous layer, and W_k stands for weight parameters in the CNN at iteration k . The complete model architecture and hyper parameter details are shown below

Hyper Parameters details used in the Inception V3 model according to our dataset.

$$f(x) = \max(0, x)$$

$$W_k \sim U\left(-\frac{1}{\sqrt{m}}, 1/\sqrt{m}\right)$$

Layer (type)	Output shape	Param
inception_v3 (Model)	(None, 8, 8, 2048)	21,802,784
flatten_1 (Flatten)	(None, 131072)	0

Layer (type)	Output shape	Param
activation_95 (Activation)	(None, 131072)	0
dropout_1 (Dropout)	(None, 131072)	0
dense_1 (Dense)	(None, 1,024)	1,34,218,752
activation_96 (Activation)	(None, 1,024)	0
dropout_2 (Dropout)	(None, 1,024)	0
dense_2 (Dense)	(None, 28)	28,700
activation_97 (Activation)	(None, 3)	0

Overall, Inception-v3 is a powerful and versatile neural network that has achieved state-of-the-art performance on a variety of image classification tasks.

4.3. DATASET

Ocular Disease Intelligent Recognition (ODIR) is a structured ophthalmic database of 5,000 patients with age, color fundus photographs from left and right eyes, and doctors' diagnostic keywords from doctors. This dataset is meant to represent the "real-life" set of patient information collected by Shanggong Medical Technology Co., Ltd. from different hospitals/medical centers in China. In these institutions, fundus images are captured by various cameras in the market, such as Canon, Zeiss, and Kowa, resulting in varied image resolutions. Annotations were labeled by trained human readers with quality control management. They classify patients into eight labels including

- Normal (N),
- Diabetes (D),
- Glaucoma (G),
- Cataract (C),
- AMD (A),
- Hypertension (H),
- Myopia (M), And
- Other Diseases/Abnormalities

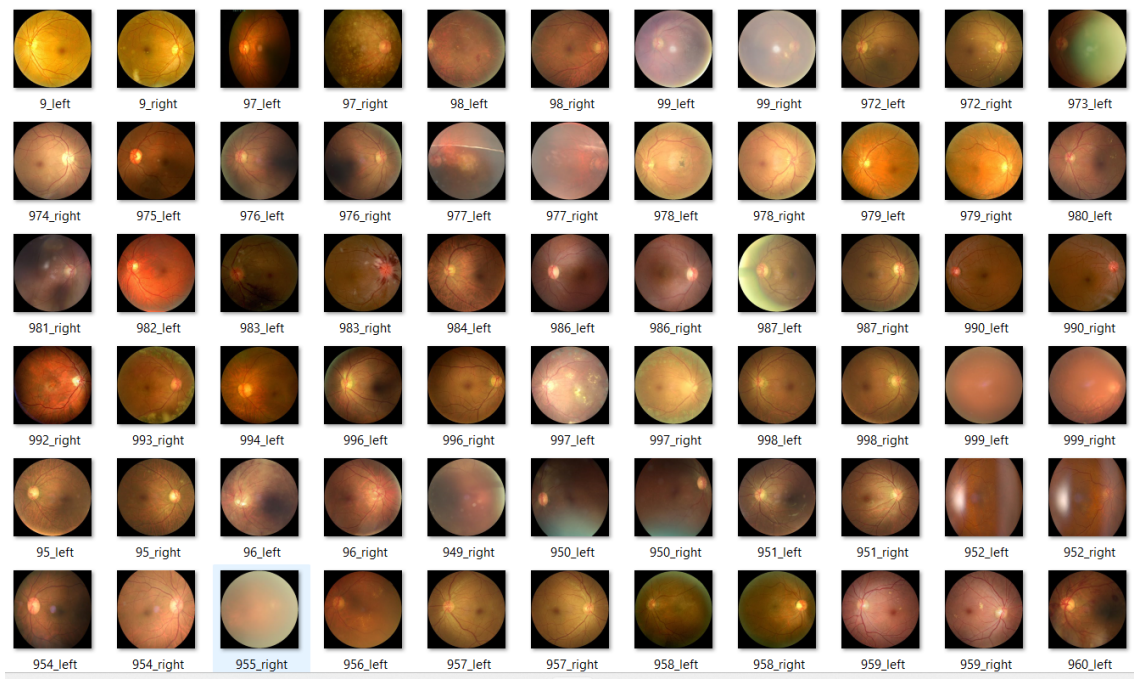


Fig. 4.2. Dataset – Fundus Images

	A	B	C	D	E	F	G
1	ID	Patient Age	Patient Sex	Left-Fundus	Right-Fundus	Left-Diagnostic Keywords	Right-Diagnostic Keywords
2	0	69	Female	0_left.jpg	0_right.jpg	cataract	normal fundus
3	1	57	Male	1_left.jpg	1_right.jpg	normal fundus	normal fundus
4	2	42	Male	2_left.jpg	2_right.jpg	laser spot, moderate non proliferative retinopathy	moderate non proliferative retinopathy
5	3	66	Male	3_left.jpg	3_right.jpg	normal fundus	branch retinal artery occlusion
6	4	53	Male	4_left.jpg	4_right.jpg	macular epiretinal membrane	mild nonproliferative retinopathy
7	5	50	Female	5_left.jpg	5_right.jpg	moderate non proliferative retinopathy	moderate non proliferative retinopathy
8	6	60	Male	6_left.jpg	6_right.jpg	macular epiretinal membrane	derate non proliferative retinopathy, epiretinal membr
9	7	60	Female	7_left.jpg	7_right.jpg	drusen	mild nonproliferative retinopathy
10	8	59	Male	8_left.jpg	8_right.jpg	normal fundus	normal fundus
11	9	54	Male	9_left.jpg	9_right.jpg	normal fundus	vitreous degeneration
12	10	70	Male	10_left.jpg	10_right.jpg	epiretinal membrane	normal fundus
13	11	60	Female	11_left.jpg	11_right.jpg	rate non proliferative retinopathy, hypertensive retinopathy	rate non proliferative retinopathy, hypertensive retino
14	12	65	Male	12_left.jpg	12_right.jpg	retinal pigmentation	retinal pigmentation
15	13	60	Female	13_left.jpg	13_right.jpg	pathological myopia	pathological myopia
16	14	55	Male	14_left.jpg	14_right.jpg	normal fundus	macular epiretinal membrane
17	15	50	Male	15_left.jpg	15_right.jpg	normal fundus	myelinated nerve fibers
18	16	54	Female	16_left.jpg	16_right.jpg	normal fundus	pathological myopia
19	17	57	Male	17_left.jpg	17_right.jpg	drusen	drusen
20	18	58	Male	18_left.jpg	18_right.jpg	pathological myopia	pathological myopia
21	19	45	Male	19_left.jpg	19_right.jpg	mild nonproliferative retinopathy	mild nonproliferative retinopathy
22	20	64	Female	20_left.jpg	20_right.jpg	rhegmatogenous retinal detachment	lens dust, normal fundus
23	21	76	Female	21_left.jpg	21_right.jpg	epiretinal membrane	epiretinal membrane
24	22	55	Female	22_left.jpg	22_right.jpg	moderate non proliferative retinopathy, laser spot	laser spot, moderate non proliferative retinopathy
25	23	47	Male	23_left.jpg	23_right.jpg	hypertensive retinopathy	hypertensive retinopathy
26	24	75	Female	24_left.jpg	24_right.jpg	normal fundus	cataract
27	25	63	Male	25_left.jpg	25_right.jpg	normal fundus	depigmentation of the retinal pigment epithelium
28	26	63	Female	26_left.jpg	26_right.jpg	moderate non proliferative retinopathy	oderate non proliferative retinopathy, abnormal pigme
29	27	33	Male	27_left.jpg	27_right.jpg	normal fundus	epiretinal membrane, moderate non proliferative reti
30	28	63	Female	28_left.jpg	28_right.jpg	hypertensive retinopathy	hypertensive retinopathy
31	29	59	Male	29_left.jpg	29_right.jpg	epiretinal membrane	normal fundus
32	30	66	Female	30_left.jpg	30_right.jpg	post laser photocoagulation, glaucoma	post laser photocoagulation
33	31	62	Male	31_left.jpg	31_right.jpg	epiretinal membrane	normal fundus

Fig. 4.3. Dataset Information in .xls file

4.4. DISEASES THAT CAN BE DETECTED

4.4.1. Cataract (C)

A cataract is a type of eye disease where the eyes look cloudy. A person with cataracts will have frosty or fogged-up vision.

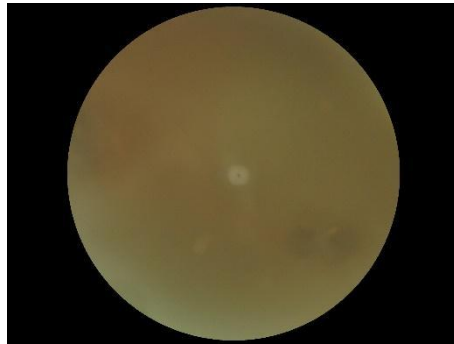


Fig.4.4 Cataract

4.4.2. Glaucoma (G)

Glaucoma is a complicated disease that damages optical nerve and causes irreversible blindness due to lack of proper diagnosis. This “sneak thief of sight” can affect anyone at any age. Even new born babies can be affected. According to doctors, it grows in such a manner that patient does not experience any complication. By 2040, the number of glaucoma affected people likely to increase 111.8 million. Glaucoma causes vision loss and blindness due to damage of optic nerve



Fig.4.5. Glaucoma

4.4.3. Myopia (M)

Near-sightedness (myopia) is a common vision condition in which near objects appear clear, but objects farther away look blurry. It occurs when the shape of the eye or the shape of certain parts of the eye causes light rays to bend (refract) inaccurately.

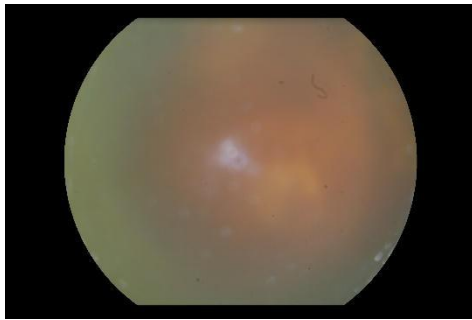


Fig.4.6.Myopia

4.4.4. Diabetes (D)

Diabetes is a chronic disease that occurs either when the pancreas does not produce enough insulin or when the body cannot effectively use the insulin it produces. Insulin is a hormone that regulates blood glucose

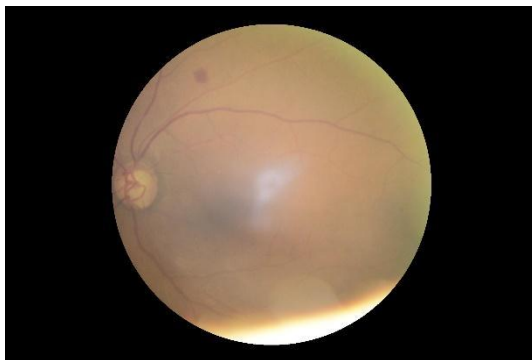


Fig.4.7.Diabetes

4.4.5. Age related Macular degeneration (A)

Age-related macular degeneration (AMD) is an eye disease that may get worse over time. It's the leading cause of severe, permanent vision loss in people over 60 years old. It happens when the small central portion of your retina, called the macula, wears down.



Fig.4.8. AMD

4.4.6. Hypertension (H)

High blood pressure is a common condition that affects the body's arteries. It's also called hypertension. If you have high blood pressure, the force of the blood pushing against the artery walls is consistently too high.

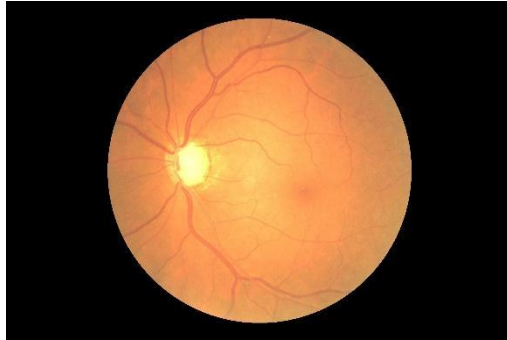


Fig.4.9. Hypertension

4.5. DATA PROCESSING

The images from the dataset are labelled. The labels relate to both eyes (left and right) at once whereas each eye can have a different disease. For example, if the left eye has a cataract and right eye has normal fundus, the label would be a cataract, not indicating a diagnosis of the right eye. Fortunately, the diagnostic keywords relate to a single eye. Dataset was created in a way to provide to the model as input both left and right eye images and return overall (for both eyes) cumulated diagnosis, neglecting the fact that one eye can be healthy. In my opinion, it does not make sense from a perspective of a final user of such a model, and it is better to get predictions separately for each eye, to know for example which eye should be treated. So, I enriched the dataset by creating a mapping between the diagnostic keywords to disease labels. This way, each eye is assigned to a proper label. Fragment of this mapping, in the form of a dictionary, is presented in the Fig. 1. Label information is added by renaming image names, and more specifically, by adding to the image file name one or more letters corresponding to the specific diseases. I applied this solution because this way I do not need to store any additional data frame with all labels.

Next, data augmentation on minority classes was applied on the training set to balance the dataset. Random zoom, random rotation, flip left-right, flip top-bottom

were applied. In the beginning, I used the TensorFlow dataset object for applying data augmentation “on the fly” while training the model[4] in order to keep my solution as scalable as possible. Unfortunately, it lacks many features like random rotation, therefore I performed data augmentation before creating the TensorFlow dataset object using other libraries for image processing like OpenCV. In the beginning, I also considered enhancing all images by applying contrast-limited adaptive histogram equalization (CLAHE) in order to increase the visibility of local details of an image, but since it was adding a lot of extra noise to the images (especially to the background, which originally is black) I decided not to follow that direction. Examples of data augmentation using my function written using PIL and OpenCV libraries.

Finally, the TensorFlow dataset object is created. It is developed very similarly to the one presented in official TensorFlow documentation for loading images. Since the library is complicated, and not easy to use for TensorFlow beginners, I would like to share here a summary of my findings on building scalable and fast input pipelines. The `tf.data` API enables you to build complex input pipelines from simple, reusable pieces. For example, the pipeline for an image model might aggregate data from files in a distributed file system. The `tf.data` API introduces a `tf.data.Dataset` abstraction that represents a sequence of elements, in which each element consists of one or more components. For example, in my image pipeline, an element is a single training example, with a pair of tensor components representing the image and its label. With the idea of creating mini-batches, TensorFlow introduces the so-called iterative learning process which is feeding to the model some portion of data (not entire dataset), training, and repeating with another portion, which are called batches. Batch size defines how many examples will be extracted at each training step. After each step, weights are updated. I selected batch size equal to 32, in order to avoid the overfitting problem. With small batch size, weights keep updating regularly and often. The downside of having a small batch size is that training takes much longer than with the bigger size. One important element of `tf.data` is the ability of the shuffling dataset. In shuffling, the dataset fills a buffer with elements, then randomly samples elements from this buffer, replacing the selected elements with new elements. It prevents situations when images of the same class will be repetitively filled to the batch, which is not beneficial for training the model.

4.6. FLASK

Flask is a web framework that is used to build RESTful APIs. It is a lightweight framework that is easy to use and can be extended with various third-party libraries. Flask API is built on top of the Flask microframework and provides a simple way to create web APIs.

I have created a Flask application and add Flask API to connect the backend processing with the web application. I then define a route for the /pred endpoint and return a JSON object that contains a trained data ad .h5 file.

4.7. WEB APPLICATION UI

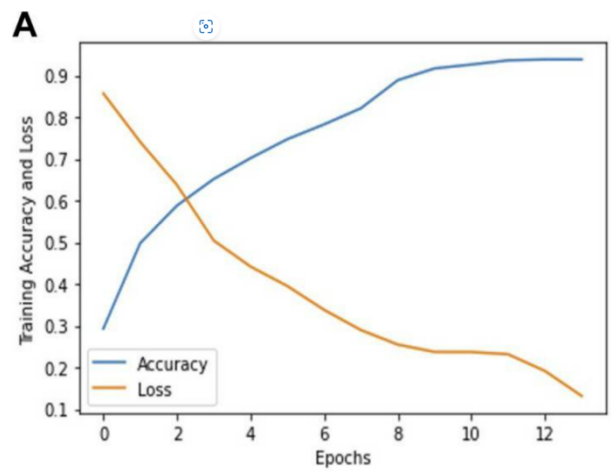
ReactJS is a popular JavaScript library used for building user interfaces. It is often used in web development to create fast and dynamic web applications.

The Web Application for collecting the patient's details and the fundus image was constructed with HTML, CSS, JavaScript and ReactJs.

CHAPTER – 5

EXPERIMENTAL RESULT AND OUTPUT

We used pre-trained deep learning models to improve the diagnosis and early treatment of various fundus diseases. After fine-tuning and adjusting hyper parameters in accordance with our classification problem, we used Inception V3 to reach **the greatest accuracy of 91.76%**, respectively, utilizing our proposed method. Figure 5.1 accuracy and loss graph was used to evaluate how well our models performed. In Figures 5.1 A, B show the accuracy and loss graphs for



classifying fundus images.

**Fig.5.1.
v3
and Loss**

Studies	Techniques	Accuracy (%)	Inception Accuracy
Lachure et al. (2015)	GLCM + SVM	82	
Priya and Aruna (2012)	SVM + NN	86.9	
Proposed approach	Inception v3 DL model	91.76	

Accuracy and loss graph according to our dataset. (A) represents the accuracy and loss graph using Inception V3.

Through this prototype of the Application , The disease is detected and displayed in the User Interface.

Diabetes	Negative
Glucoma	Positive
Cataract	Negative
Age related Macular Degeneration	Negative
Hypertension	Negative
Pathological Myopia	Negative
Abnormalities	Negative

The Patient has diagnosed with **GLUCOMA**.

Fig.5.2.Final Ouput

CHAPTER-6

CONCLUSION AND FUTURE WORKS

The results were very satisfactory, using a relatively simple network in 15 epochs my model got 91.7% on validation accuracy. This shows that using Convolution Neural Network, it is possible to automatically detect disease. For all experiments, the same neural network architecture was used. The only difference is the number of epochs each experiment needed to get to the presented results. This web application that is designed and developed using Inception V3 can successfully detects the disease using fundus image with more accuracy, which reduces the Ophthalmologist's workload and effectively prevents vision damage of patients.

In the future with more data points of eye diseases the model could be re-trained, and certainly will achieve much better results having more images to train on. I am sure that having a bigger dataset, would increase the accuracy of predictions and finally automate the process of detecting ocular diseases.

REFERENCES:

<https://www.kaggle.com/datasets/andrewmvd/ocular-disease-recognition-odir5k>

<https://www.who.int/publications/i/item/9789241516570>

<https://towardsdatascience.com/ocular-disease-recognition-using-convolutional-neural-networks-c04d63a7a2da>

<https://www.diva-portal.org/smash/get/diva2:1582350/FULLTEXT01.pdf>

<https://www.sciencedirect.com/science/article/abs/pii/S0010482522004401>

APPENDIX

SCREENSHOTS

Section 1

OcuAi

User Log In

Your User ID

Password

[Forgot Password ?](#)

Enter the below captcha

Contact Us
HealthSphere healthcare Solutuion Pvt.Ltd,
No.376, Anna Street, OMR, Chennai- 600 097.
Phone No. - 987654321. 9887766554

Copyright © 2023 HealthSphere Healthcare Solutions.
All Rights Reserved

This Content is owned and maintained by
HealthSphere Healthcare Solution. Pvt. Ltd
FOR ANY TECHNICAL ASSISTANCE :
healthspherehealthcares@gmail.com

A.1 Login page

OcuAi

REPORTSHELPLOG OUT

HOME

Welcome USER NAME

POSITION: NurseBlock: Out-PatientHospital:

PATIENT DETAILS

Patient Name:

Registration Application ID:

Date of Registration:

Doctor In-Charge:

Doctor's Email ID:

Patient's Age:

Gender:

Date Of Birth:

Address:

Aadhaar ID

Mobile Number:

Patient's Email ID:

CLEAR

NEXT

Contact Us

HealthSphere healthcare Solution Pvt.Ltd,
No.376, Anna Street, OMR, Chennai- 600 097,
Phone No. - 987654321 9887766554

Copyright © 2023 HealthSphere Healthcare Solutions.

All Rights Reserved

This Content is owned and maintained by
HealthSphere Healthcare Solution, Pvt. Ltd

FOR ANY TECHNICAL ASSISTANCE :
healthspherehealthcares@gmail.com

A.2. Patient Entry Page

OcuAi

REPORTS

HELP

LOG OUT

Scan

UPLOAD IMAGES

Left Eye

Right Eye

SCAN

IMPORTANT INFO :

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ut nisl nec purus molestie egestas et sed neque. Vestibulum sit amet mi et erat porttitor viverra. Interdum et malesuada fames ac ante ipsum primis in faucibus. Nunc condimentum rutrum hendrerit. Vestibulum vitae augue vitae orci maximus porttitor dapibus et magna. Quisque bibendum nulla sit amet fringilla condimentum. Suspendisse non semper mi. Sed rutrum lorem vel lobortis rhoncus. Aliquam ac diam ac neque fringilla mattis. Suspendisse congue arcu neque, nec imperdiet libero dapibus id. Nam sed justo turpis. Curabitur a feugiat nulla.

Contact Us

HealthSphere healthcare Solution Pvt.Ltd,
 No.376, Anna Street, OMR, Chennai- 600 097,
 Phone No. - 987654321, 9887766554

Copyright © 2023 HealthSphere Healthcare Solutions.

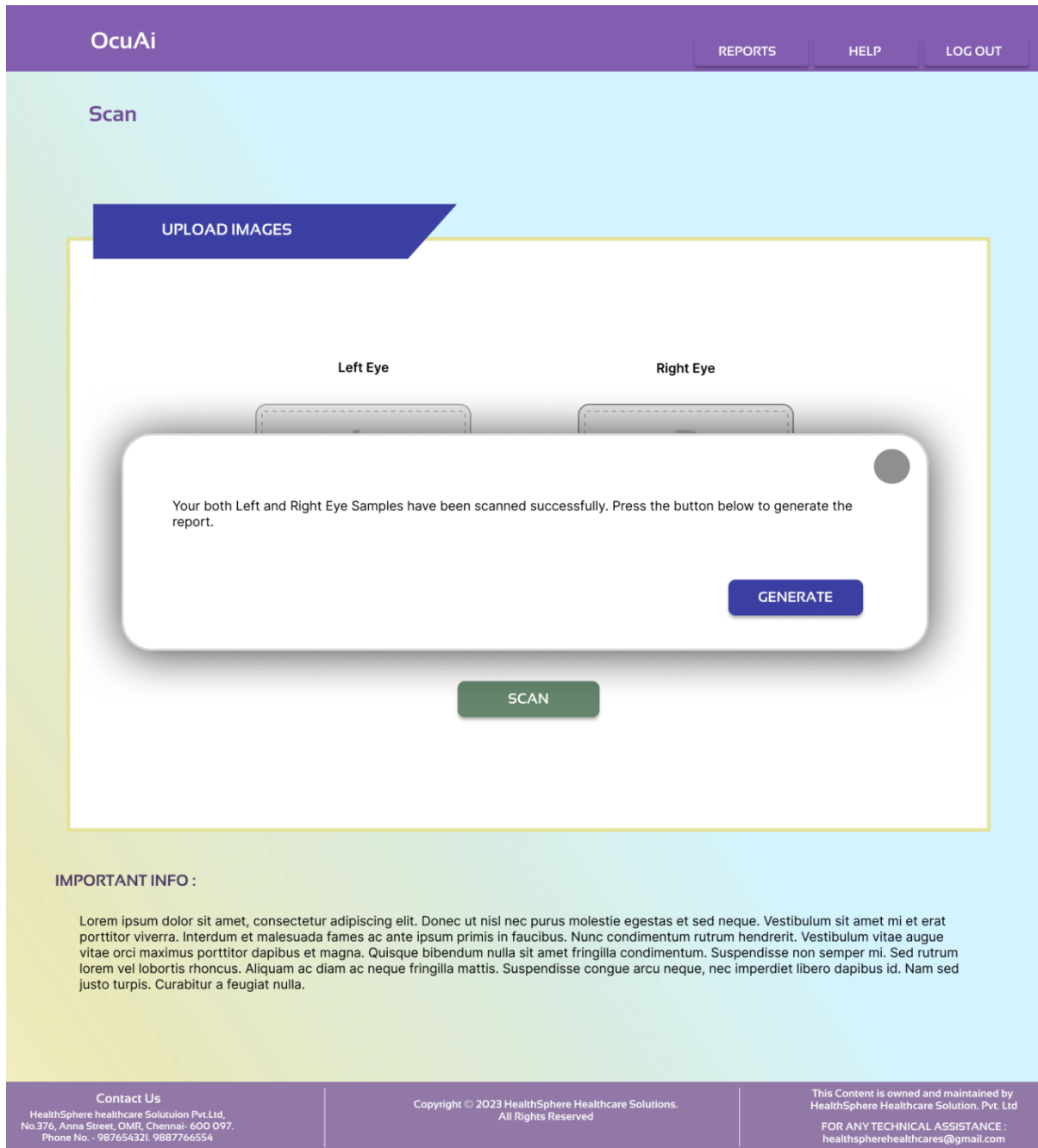
All Rights Reserved

This Content is owned and maintained by
 HealthSphere Healthcare Solution. Pvt. Ltd

FOR ANY TECHNICAL ASSISTANCE :
 healthspherehealthcares@gmail.com

A.3. Scanning Page

22



A.4. Image Scannes

OcuAi

REPORTSHELPLOG OUT

Scan

REPORTS

Name: Patient name

Email: username123@gmail.com

Application ID: USHDN0908108

Doctor In-charge: doctername

Age: 27

Sex: M

Phone Number: 9876543210

RESULT

Diabetes

Negative

Glucoma

Positive

Cataract

Negative

Age related Macular Degeneration

Negative

Hypertension

Negative

Pathological Myopia

Negative

Abnormalities

Negative

The Patient has diagnosed with **GLUCOMA**.

SEND

Contact Us

HealthSphere healthcare Solution Pvt.Ltd,
No.376, Anna Street, OMR, Chennai- 600 097.
Phone No. - 987654321. 9887766554

Copyright © 2023 HealthSphere Healthcare Solutions.
All Rights Reserved

This Content is owned and maintained by
HealthSphere Healthcare Solution, Pvt. Ltd

FOR ANY TECHNICAL ASSISTANCE :
healthspherehealthcares@gmail.com

A.5. Final Report

ML Inception Net for cataract detection

Cataract.py

```
import numpy as np
import pandas as pd
import cv2
import random
from tqdm import tqdm
import matplotlib.pyplot as plt
import tensorflow
from tensorflow.keras.applications.inception_v3 import preprocess_input
from tensorflow.keras.preprocessing import image
import os
from sklearn.model_selection import train_test_split
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Flatten,Dense
df = pd.read_csv('archive (6)/full_df.csv')
def has_cataract(text):
    if "cataract" in text:
        return 1
    else:
        return 0
df["left_cataract"] = df["Left-Diagnostic Keywords"].apply(lambda x: has_cataract(x))
df["right_cataract"] = df["Right-Diagnostic Keywords"].apply(lambda x:
has_cataract(x))
left_cataract = df.loc[(df.C ==1) & (df.left_cataract == 1)]["Left-Fundus"].values
right_cataract = df.loc[(df.C ==1) & (df.right_cataract == 1)]["Right-Fundus"].values
left_normal = df.loc[(df.C ==0) & (df["Left-Diagnostic Keywords"] == "normal
fundus")]["Left-Fundus"].sample().values
right_normal = df.loc[(df.C ==0) & (df["Right-Diagnostic Keywords"] == "normal
fundus")]["Right-Fundus"].sample().values
cataract = np.concatenate((left_cataract,right_cataract),axis=0)
normal = np.concatenate((left_normal,right_normal),axis=0)
dataset_dir = "archive (6)/preprocessed_images/"
image_size=224
labels = []
dataset = []
def cc(c,ll):
    for img in tqdm(c):
        image_path = os.path.join(dataset_dir,img)
        try:
            image = cv2.imread(image_path,cv2.IMREAD_COLOR)
            image = cv2.resize(image,(image_size,image_size))
        except:
            continue
```

```

        dataset.append([np.array(image),np.array(l)])
    random.shuffle(dataset)
    return dataset
dataset=cc(cataract,1)
dataset=cc(normal,0)
x = np.array([i[0] for i in dataset]).reshape(-1,image_size,image_size,3)
y = np.array([i[1] for i in dataset])
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
InceptionV3 =
tensorflow.keras.applications.inception_v3.InceptionV3(weights="imagenet",include_
top = False,input_shape=(image_size,image_size,3))
for layer in InceptionV3.layers:
    layer.trainable = False
def ff():
    model = Sequential()
    model.add(InceptionV3)
    model.add(tensorflow.keras.layers.Flatten())
    model.add(tensorflow.keras.layers.Dense(1,activation="sigmoid"))
    model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
    return model
checkpoint = tensorflow.keras.callbacks.ModelCheckpoint("")
earlystop =
tensorflow.keras.callbacks.EarlyStopping(monitor="val_acc",patience=5,verbose=1)
model=ff()
model.fit(x_train,y_train,batch_size=32,epochs=15,validation_data=(x_test,y_test),
        verbose=1,callbacks=[checkpoint,earlystop])
model.save('model1')

```

ML Inception Net for Glaucoma detection

Glaucoma.py

```

def has_retinopathy(text):
    if "glaucoma" in text:
        return 1
    else:
        return 0
df["left_glaucoma"] = df["Left-Diagnostic Keywords"].apply(lambda x:
has_glaucoma(x))
df["right_glaucoma"] = df["Right-Diagnostic Keywords"].apply(lambda x:
has_glaucoma(x))
left_glaucoma = df.loc[(df.G ==1) & (df.left_glaucoma == 1)]["Left-Fundus"].values
right_glaucoma = df.loc[(df.G ==1) & (df.right_glaucoma ==
1)]["Right-Fundus"].values

```

```

left_normal = df.loc[(df.G ==0) & (df["Left-Diagnostic Keywords"] == "normal
fundus")]["Left-Fundus"].sample(250,random_state=42).values
right_normal = df.loc[(df.G ==0) & (df["Right-Diagnostic Keywords"] == "normal
fundus")]["Right-Fundus"].sample(250,random_state=42).values
glaucoma = np.concatenate((left_glaucoma,right_glaucoma),axis=0)
normal = np.concatenate((left_normal,right_normal),axis=0)
dataset_dir = "archive (6)/preprocessed_images/"
image_size=224
labels = []
dataset = []
def cc(c,ll):
    for img in tqdm(c):
        image_path = os.path.join(dataset_dir,img)
        try:
            image = cv2.imread(image_path,cv2.IMREAD_COLOR)
            image = cv2.resize(image,(image_size,image_size))
        except:
            continue
        dataset.append([np.array(image),np.array(ll)])
    random.shuffle(dataset)
    return dataset
dataset=cc(glaucoma,1)
dataset=cc(normal,0)
x = np.array([i[0] for i in dataset]).reshape(-1,image_size,image_size,3)
y = np.array([i[1] for i in dataset])
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
InceptionV3 =
tensorflow.keras.applications.inception_v3.InceptionV3(weights="imagenet",include_
top = False,input_shape=(image_size,image_size,3))
for layer in InceptionV3.layers:
    layer.trainable = False
def ff():
    model = Sequential()
    model.add(InceptionV3)
    model.add(tensorflow.keras.layers.Flatten())
    model.add(tensorflow.keras.layers.Dense(1,activation="sigmoid"))
    model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
    return model
earlystop =
tensorflow.keras.callbacks.EarlyStopping(monitor="val_acc",patience=5,verbose=1)
model=ff()
model.fit(x_train,y_train,batch_size=32,epochs=15,validation_data=(x_test,y_test),
        verbose=1,callbacks=[checkpoint,earlystop])

```

ML Inception Net for Glaucoma detection

Myopia.py

```
def has_myopia(text):
    if "pathological myopia" in text:
        return 1
    else:
        return 0

df["left_myopia"] = df["Left-Diagnostic Keywords"].apply(lambda x: has_myopia(x))
df["right_myopia"] = df["Right-Diagnostic Keywords"].apply(lambda x: has_myopia(x))
left_myopia = df.loc[(df.M == 1) & (df.left_myopia == 1)]["Left-Fundus"].values
right_myopia = df.loc[(df.M == 1) & (df.right_myopia == 1)]["Right-Fundus"].values
left_normal = df.loc[(df.M == 0) & (df["Left-Diagnostic Keywords"] == "normal fundus")]["Left-Fundus"].sample(250,random_state=42).values
right_normal = df.loc[(df.M == 0) & (df["Right-Diagnostic Keywords"] == "normal fundus")]["Right-Fundus"].sample(250,random_state=42).values
myopia = np.concatenate((left_myopia,right_myopia),axis=0)
normal = np.concatenate((left_normal,right_normal),axis=0)
dataset_dir = "archive (6)/preprocessed_images/"
image_size=224
labels = []
dataset = []
def cc(c,ll):
    for img in tqdm(c):
        image_path = os.path.join(dataset_dir,img)
        try:
            image = cv2.imread(image_path,cv2.IMREAD_COLOR)
            image = cv2.resize(image,(image_size,image_size))
        except:
            continue
        dataset.append([np.array(image),np.array(ll)])
    random.shuffle(dataset)
    return dataset
dataset=cc(myopia,1)
dataset=cc(normal,0)
x = np.array([i[0] for i in dataset]).reshape(-1,image_size,image_size,3)
y = np.array([i[1] for i in dataset])
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
InceptionV3 =
tensorflow.keras.applications.inception_v3.InceptionV3(weights="imagenet",include_top = False,input_shape=(image_size,image_size,3))
for layer in InceptionV3.layers:
    layer.trainable = False
```

```

def ff():
    model = Sequential()
    model.add(InceptionV3)
    model.add(tensorflow.keras.layers.Flatten())
    model.add(tensorflow.keras.layers.Dense(1,activation="sigmoid"))
    model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
    return model
earlystop = tensorflow.keras.callbacks.EarlyStopping(monitor="val_acc",patience=5,verbose=1)
model=ff()
model.fit(x_train,y_train,batch_size=32,epochs=15,validation_data=(x_test,y_test),
        verbose=1,callbacks=[checkpoint,earlystop])

```

ML Inception Net for Hypertension detection

Hypertension.py

```

def has_hypertension(text):
    if "hypertensive retinopathy" in text:
        return 1
    else:
        return 0
df["left_hypertension"] = df["Left-Diagnostic Keywords"].apply(lambda x:
has_hypertension(x))
df["right_hypertension"] = df["Right-Diagnostic Keywords"].apply(lambda x:
has_hypertension(x))
left_hypertension = df.loc[(df.H ==1) & (df.left_hypertension ==
1)][["Left-Fundus"].values
right_hypertension = df.loc[(df.H ==1) & (df.right_hypertension ==
1)][["Right-Fundus"].values
left_normal = df.loc[(df.H ==0) & (df["Left-Diagnostic Keywords"] == "normal
fundus")][["Left-Fundus"].sample(250,random_state=42).values
right_normal = df.loc[(df.H ==0) & (df["Right-Diagnostic Keywords"] == "normal
fundus")][["Right-Fundus"].sample(250,random_state=42).values
hypertension = np.concatenate((left_hypertension,right_hypertension),axis=0)
normal = np.concatenate((left_normal,right_normal),axis=0)
dataset_dir = "archive (6)/preprocessed_images/"
image_size=224
labels = []
dataset = []
def cc(c,ll):
    for img in tqdm(c):
        image_path = os.path.join(dataset_dir,img)

```

```

try:
    image = cv2.imread(image_path,cv2.IMREAD_COLOR)
    image = cv2.resize(image,(image_size,image_size))
except:
    continue
dataset.append([np.array(image),np.array(ll)])
random.shuffle(dataset)
return dataset
dataset=cc(hypertension,1)
dataset=cc(normal,0)
x = np.array([i[0] for i in dataset]).reshape(-1,image_size,image_size,3)
y = np.array([i[1] for i in dataset])
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
InceptionV3 =
tensorflow.keras.applications.inception_v3.InceptionV3(weights="imagenet",include_
top = False,input_shape=(image_size,image_size,3))
for layer in InceptionV3.layers:
    layer.trainable = False
def ff():
    model = Sequential()
    model.add(InceptionV3)
    model.add(tensorflow.keras.layers.Flatten())
    model.add(tensorflow.keras.layers.Dense(1,activation="sigmoid"))
    model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
    return model
checkpoint =
tensorflow.keras.callbacks.ModelCheckpoint("vgg19.h5",monitor="val_acc",verbose=
1,save_best_only=True,save_weights_only=False,period=1)
earlystop =
tensorflow.keras.callbacks.EarlyStopping(monitor="val_acc",patience=5,verbose=1)
model=ff()
model.fit(x_train,y_train,batch_size=32,epochs=15,validation_data=(x_test,y_test),
        verbose=1,callbacks=[checkpoint,earlystop])

```

ML Inception Net for AMD detection

Agerelatedmaculardisease.py

```

def has_AgerelatedMacularDegeneration(text):
    if "age-related macular degeneration" in text:

```

```

        return 1
    else:
        return 0
df["left_AgerelatedMacularDegeneration"] = df["Left-Diagnostic
Keywords"].apply(lambda x: has_AgerelatedMacularDegeneration(x))
df["right_AgerelatedMacularDegeneration"] = df["Right-Diagnostic
Keywords"].apply(lambda x: has_AgerelatedMacularDegeneration(x))
left_AgerelatedMacularDegeneration = df.loc[(df.A ==1) &
(df.left_AgerelatedMacularDegeneration == 1)]["Left-Fundus"].values
right_AgerelatedMacularDegeneration = df.loc[(df.A ==1) &
(df.right_AgerelatedMacularDegeneration == 1)]["Right-Fundus"].values
left_normal = df.loc[(df.A ==0) & (df["Left-Diagnostic Keywords"] == "normal
fundus")]["Left-Fundus"].sample(250,random_state=42).values
right_normal = df.loc[(df.A ==0) & (df["Right-Diagnostic Keywords"] == "normal
fundus")]["Right-Fundus"].sample(250,random_state=42).values
AgerelatedMacularDegeneration =
np.concatenate((left_AgerelatedMacularDegeneration,right_AgerelatedMacularDege
neration),axis=0)
normal = np.concatenate((left_normal,right_normal),axis=0)
dataset_dir = "archive (6)/preprocessed_images/"
image_size=224
labels = []
dataset = []
def cc(c,ll):
    for img in tqdm(c):
        image_path = os.path.join(dataset_dir,img)
        try:
            image = cv2.imread(image_path,cv2.IMREAD_COLOR)
            image = cv2.resize(image,(image_size,image_size))
        except:
            continue
        dataset.append([np.array(image),np.array(ll)])
    random.shuffle(dataset)
    return dataset
dataset=cc(AgerelatedMacularDegeneration,1)
dataset=cc(normal,0)
x = np.array([i[0] for i in dataset]).reshape(-1,image_size,image_size,3)
y = np.array([i[1] for i in dataset])
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
InceptionV3 =
tensorflow.keras.applications.inception_v3.InceptionV3(weights="imagenet",include_
top = False,input_shape=(image_size,image_size,3))
for layer in InceptionV3.layers:
    layer.trainable = False

```

```

def ff():
    model = Sequential()
    model.add(InceptionV3)
    model.add(tensorflow.keras.layers.Flatten())
    model.add(tensorflow.keras.layers.Dense(1,activation="sigmoid"))
    model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
    return model
earlystop = tensorflow.keras.callbacks.EarlyStopping(monitor="val_acc",patience=5,verbose=1)
model=ff()
model.fit(x_train,y_train,batch_size=32,epochs=15,validation_data=(x_test,y_test),
        verbose=1,callbacks=[checkpoint,earlystop])

```

ML Inception Net for Diabetes detection

Diabetes.py

```

def has_retinopathy(text):
    if "retinopathy" in text:
        return 1
    else:
        return 0
df["left_retinopathy"] = df["Left-Diagnostic Keywords"].apply(lambda x:
has_retinopathy(x))
df["right_retinopathy"] = df["Right-Diagnostic Keywords"].apply(lambda x:
has_retinopathy(x))
left_retinopathy = df.loc[(df.D ==1) & (df.left_retinopathy == 1)]["Left-Fundus"].values
right_retinopathy = df.loc[(df.D ==1) & (df.right_retinopathy ==
1)]["Right-Fundus"].values
left_normal = df.loc[(df.D ==0) & (df["Left-Diagnostic Keywords"] == "normal
fundus")]["Left-Fundus"].sample(250,random_state=42).values
right_normal = df.loc[(df.D ==0) & (df["Right-Diagnostic Keywords"] == "normal
fundus")]["Right-Fundus"].sample(250,random_state=42).values
retinopathy = np.concatenate((left_retinopathy,right_retinopathy),axis=0)
normal = np.concatenate((left_normal,right_normal),axis=0)
dataset_dir = "archive (6)/preprocessed_images/"
image_size=224
labels = []
dataset = []
def cc(c,ll):
    for img in tqdm(c):
        image_path = os.path.join(dataset_dir,img)
        try:
            image = cv2.imread(image_path,cv2.IMREAD_COLOR)
            image = cv2.resize(image,(image_size,image_size))

```

```

        except:
            continue
        dataset.append([np.array(image),np.array(ll)])
    random.shuffle(dataset)
    return dataset
dataset=cc(retinopathy,1)
dataset=cc(normal,0)
x = np.array([i[0] for i in dataset]).reshape(-1,image_size,image_size,3)
y = np.array([i[1] for i in dataset])
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
InceptionV3 = tensorflow.keras.applications.inception_v3.InceptionV3(weights="imagenet",include_top = False,input_shape=(image_size,image_size,3))
for layer in InceptionV3.layers:
    layer.trainable = False
def ff():
    model = Sequential()
    model.add(InceptionV3)
    model.add(tensorflow.keras.layers.Flatten())
    model.add(tensorflow.keras.layers.Dense(1,activation="sigmoid"))
    model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
    return model
earlystop = tensorflow.keras.callbacks.EarlyStopping(monitor="val_acc",patience=5,verbose=1)
model=ff()
model.fit(x_train,y_train,batch_size=32,epochs=15,validation_data=(x_test,y_test),
        verbose=1,callbacks=[checkpoint,earlystop])

```

ReactJs Code – Frontend

```

import './App.css';
import {useRef} from 'react';
import {Img} from './pages/Img.js';
import { useState,useEffect} from 'react';
import { useNavigate,Route,Routes,BrowserRouter } from 'react-router-dom';
import {Hh} from './pages/Hh.js'
function App() {
  const [data,setData]=useState(()=>{
    return "Scan the Images";
  });
  return (
    <div className='Parentdiv'>

```

```

<div className='div1'>
  <h2 className='h2'>{data}</h2>
</div>
<div className='div2'>
  <Routes>
    <Route path="/" element={<Hh/>}></Route>
    <Route path="/data" element={<Img/>}></Route>
  </Routes>
</div>
</div>
)
}
export default App;

```

Flask Code – Backend

```

import './App.css';
import {useRef} from 'react';
import {Img} from './pages/Img.js';
import { useState,useEffect} from 'react';
import { useNavigate,Route,Routes,BrowserRouter } from 'react-router-dom';
import {Hh} from './pages/Hh.js'
function App() {
  const [data,setData]=useState(()=>{
    return "Scan the Images";
  });
  return (
    <div className='Parentdiv'>
      <div className='div1'>
        <h2 className='h2'>{data}</h2>
      </div>
      <div className='div2'>
        <Routes>
          <Route path="/" element={<Hh/>}></Route>
          <Route path="/data" element={<Img/>}></Route>
        </Routes>
      </div>
    </div>
  )
}
export default App;

```

Login.html

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="css/bootstrap.min.css" />
    <link
      rel="stylesheet"

href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.4.1/font/bootstrap-icons.css"
    />
    <link rel="stylesheet" href="css/dataTables.bootstrap5.min.css" />
    <link rel="stylesheet" href="css/style.css" />
    <title>OcuAI</title>
  </head>
  <body>
<div class=e70_50>
  <div class=e1_2>
    <div class="e3_4"></div>
    <div class="e3_48"></div>
      <div class="e3_47"></div><span class="e3_5">OcuAi</span><span
class="e3_6">User Log In</span>
    <div class=e3_13>
      <div class="e3_11"></div>
      <div class="e3_12"></div>
      <div class="e3_9"></div>
    </div>
    <div class=e3_14>
      <div class="e3_15"></div>
      <div class="e3_16"></div>
      <div class="e3_17"></div>
        </div><span class="e3_19">Your User ID </span><span
class="e3_20">Password</span>
    <div class=e3_27>
      <div class="e3_28"></div>
      <div class="e3_29"></div>
      <div class="e3_30"></div>
    </div><span class="e3_31">Enter the below captcha</span>
    <div class="e3_32"></div><span class="e3_57">Forgot Password ?</span>
    <div class=e3_38>
      <div class="e3_34"></div>
      <div class="e3_35"></div>
      <div class="e3_36"></div>

```

```

        <div class="e3_37"></div>
    </div>
    <div class=e26_159>
        <div class="e3_41"></div><span class="e3_42">Contact Us</span><span
class="e3_43">HealthSphere healthcare Solutuion Pvt.Ltd,
No.376, Anna Street, OMR, Chennai- 600 097.
Phone No. - 987654321. 9887766554
    </span><span class="e3_44">Copyright © 2023 HealthSphere Healthcare
Solutions. All Rights Reserved</span><span class="e3_45">This Content is owned
and maintained by
HealthSphere Healthcare Solution. Pvt. Ltd</span><span class="e3_46">FOR ANY
TECHNICAL ASSISTANCE :
healthspherehealthcares@gmail.com</span>
        <div class="e26_157"></div>
        <div class="e26_158"></div>
    </div>
</div>
<div class=e38_42>
    <div class="e38_41"></div>
</div>
</div>
    <script src="./js/script.js"></script>
</body>
</html>

```

Entry.html

```

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <link rel="stylesheet" href="css/bootstrap.min.css" />
        <link
            rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.4.1/font/bootstrap-icons.css"
        />
        <link rel="stylesheet" href="css/dataTables.bootstrap5.min.css" />
        <link rel="stylesheet" href="css/style.css" />
        <title>OcuAI</title>
    </head>
    <body>

```

```

<div class=e3_2>
  <div class="e3_3"></div><span class="e18_3">OcuAi</span>
  <div class=e18_19>
    <div class="ei18_19_18_4"></div><span class="ei18_19_18_5">HELP</span>
  </div>
  <div class=e18_31>
    <div class="ei18_31_18_26"></div><span class="ei18_31_18_27">LOG
OUT</span>
  </div>
  <div class=e18_41>
    <div class="ei18_41_18_36"></div><span
class="ei18_41_18_37">REPORTS</span>
  </div>
  <div class=e18_49>
    <div class="e18_44"></div>
    <div class="e22_54"></div>
    <div class=e18_48>
      <div class="e18_45"></div><span class="e18_47">Welcome USER
NAME</span>
    </div>
    <div class=e22_55>
      <div class="e22_56"></div><span class="e22_57">PATIENT DETAILS</span>
      </div><span class="e18_51">POSITION: Nurse</span><span
class="e18_52">Block: Out-Patient</span><span class="e18_53">Hospital:
</span>
    </div>
    <div class=e22_68><span class="e22_66">Patient Name: </span>
      <div class="e22_67"></div>
    </div>
    <div class=e22_69><span class="e22_70">Registration Application ID:</span>
      <div class="e22_71"></div>
    </div>
    <div class=e22_96><span class="e22_97">Patient's Age:</span>
      <div class="e22_98"></div>
    </div>
    <div class=e25_133><span class="e25_134">Mobile Number:</span>
      <div class="e25_135"></div>
    </div>
    <div class=e25_142><span class="e25_143">Patient's Email ID:</span>
      <div class="e25_144"></div>
    </div>
    <div class=e22_119><span class="e22_120">Date Of Birth:</span>
      <div class="e22_121"></div>

```

```

</div>
<div class=e22_116><span class="e22_117">Aadhaar ID</span>
  <div class="e22_118"></div>
</div>
<div class=e22_113><span class="e22_114">Address:</span>
  <div class="e22_115"></div>
</div>
<div class=e22_100><span class="e22_101">Gender:</span>
  <div class="e22_102"></div>
</div>
<div class=e22_93><span class="e22_94">Doctor In-Charge:</span>
  <div class="e22_95"></div>
</div>
<div class=e25_139><span class="e25_140">Doctor's Email ID:</span>
  <div class="e25_141"></div>
</div>
<div class=e22_72><span class="e22_73">Date of Registration:</span>
  <div class="e22_74"></div>
</div>
<div class=e26_160>
  <div class="e26_161"></div><span class="e26_162">Contact Us</span><span
class="e26_163">HealthSphere healthcare Solutuion Pvt.Ltd,
No.376, Anna Street, OMR, Chennai- 600 097.
Phone No. - 987654321. 9887766554
  </span><span class="e26_164">Copyright © 2023 HealthSphere Healthcare
Solutions. All Rights Reserved</span><span class="e26_165">This Content is
owned and maintained by
HealthSphere Healthcare Solution. Pvt. Ltd</span><span class="e26_166">FOR
ANY TECHNICAL ASSISTANCE :
healthspherehealthcares@gmail.com</span>
  <div class="e26_167"></div>
  <div class="e26_168"></div>
</div>
<div class=e26_189>
  <div class="ei26_189_26_181"></div><span
class="ei26_189_26_182">CLEAR</span>
</div>
<div class=e26_209>
  <div class="ei26_209_26_169"></div><span
class="ei26_209_26_171">NEXT</span>
  <div class=ei26_209_26_197>
    <div class="ei26_209_26_198"></div>
    <div class="ei26_209_26_199"></div>
  </div>

```

```
</div><span class="e28_86">HOME</span>
</div>
</body>
</html>
```