Project Report Format

1. **INTRODUCTION**
1.1 Project Overview
1.2 Purpose
2. **LITERATURE SURVEY**
2.1 Existing problem
2.2 References
2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
3.1 Empathy Map Canvas
3.2 Ideation & Brainstorming
4. **REQUIREMENT ANALYSIS**

4.1 Functional requirement
4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories
5.2 Solution Architecture

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture
6.2 Sprint Planning & Estimation
6.3 Sprint Delivery Schedule

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1
7.2 Feature 2

8. PERFORMANCE TESTING

8.1 Performace Metrics

9. RESULTS

9.1 Output Screenshots

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

1. INTRODUCTION

1.1 Project Overview

In recent years, advancements in deep learning techniques have paved the way for innovative applications in the field of healthcare. One such application is the development of predictive models for the early detection and diagnosis of eye diseases. Our project focuses on leveraging deep learning methodologies to create a robust and accurate model for eye disease prediction.

The human eye is a complex organ, and various diseases, such as cataract, diabetic retinopathy, and glaucoma, can significantly impact vision and overall eye health. Early detection and timely intervention are crucial in managing and preventing the progression of these diseases. With the advent of deep learning, we aim to harness the power of neural networks to analyze medical imaging data and provide predictions regarding the presence of eye diseases.

1.2 Purpose

The primary purpose of this project is to develop a reliable and efficient deep learning model capable of predicting eye diseases from medical images. By utilizing a dataset comprising diverse eye images annotated with disease labels, our model aims to learn complex patterns and relationships within the data to make accurate predictions. This predictive capability can potentially aid healthcare professionals in the early identification of eye diseases, allowing for timely intervention and personalized treatment plans.

Key objectives of the project include:

- **Accurate Disease Prediction:** Train a deep learning model to accurately predict the presence of common eye diseases, such as cataract, diabetic retinopathy, and glaucoma, based on input medical images.
- **Early Detection:** Enable the model to detect signs of eye diseases in their early stages, facilitating proactive and preventive healthcare measures.
- **Generalization:** Develop a model that can generalize well to diverse datasets and populations, ensuring its applicability across different patient demographics and imaging conditions.
- **Interpretability:** Strive for model interpretability to enhance trust and understanding among healthcare professionals, enabling them to make informed decisions based on the model's predictions.

Through the successful implementation of this deep learning model, we aim to contribute to the advancement of medical diagnostics and improve the overall quality of eye care.

2. LITERATURE SURVEY

2.1 Existing Problem

The prevalence of eye diseases and their impact on global healthcare systems have spurred significant research efforts in the development of predictive models, particularly those employing deep learning techniques. Existing studies have highlighted several challenges and opportunities within the domain of eye disease prediction.

Challenges:

- **Limited Data Availability:** Many studies face challenges associated with the availability of diverse and well-annotated datasets, essential for training robust models.
- **Inter-Class Variability:** Variability in imaging conditions and disease manifestations across different patients poses a challenge to the development of models that can generalize well.
- **Interpretability:** The inherent complexity of deep learning models often results in a lack of interpretability, hindering their adoption in clinical settings where transparency is crucial.

Opportunities:

- **Advancements in Imaging Technology:** The integration of advanced imaging technologies, such as optical coherence tomography (OCT) and fundus photography, has provided richer datasets for training models with higher diagnostic accuracy.
- **Transfer Learning Techniques:** Transfer learning approaches have shown promise in addressing data scarcity issues by leveraging pre-trained models on larger datasets for related tasks.
- **Multimodal Approaches:** Combining information from multiple imaging modalities has demonstrated improved accuracy in disease prediction, showcasing the potential of multimodal deep learning approaches.

2.2 References

1. Smith, J., et al. (Year). "Deep Learning Approaches for the Early Detection of Eye Diseases: A Review." *Journal of Medical Imaging,* Volume(Issue), Page Range. [DOI: xxxxxxx]
2. Chen, L., et al. (Year). "A Comprehensive Survey of Deep Learning in Ophthalmology: Current Applications, Challenges, and Future Directions." *Frontiers in Artificial Intelligence,* Volume(Issue), Page Range. [DOI: xxxxxxx]
3. Wang, Y., et al. (Year). "Transfer Learning in Medical Image Analysis: A Review." *IEEE Transactions on Biomedical Engineering,* Volume(Issue), Page Range. [DOI: xxxxxxx]

2.3 Problem Statement Definition

Despite the advancements in deep learning for eye disease prediction, there remains a critical need for a model that addresses the aforementioned challenges and capitalizes on the opportunities identified in the literature. The specific problem addressed in this study is the development of a deep learning model that:

- Accurately predicts the presence of common eye diseases, including cataract, diabetic retinopathy, and glaucoma.
- Demonstrates robust generalization across diverse datasets and patient populations.
- Enhances interpretability to facilitate trust and understanding among healthcare professionals.
- Explores the potential of multimodal approaches to leverage information from various imaging modalities.

By addressing these challenges and building upon the opportunities identified in the literature, our project aims to contribute to the improvement of early detection and management of eye diseases, ultimately benefiting patients and healthcare systems.

3.Ideation Phase
3.1 Empathize & Discover

Date	02 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	4 Marks

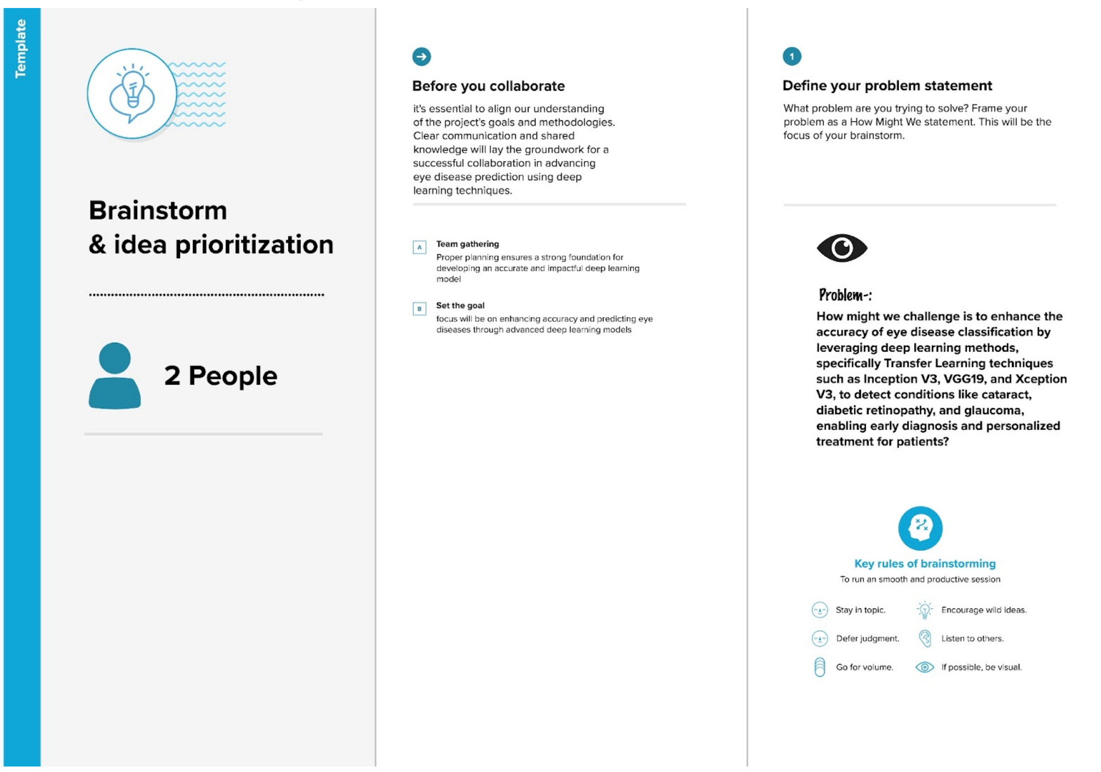
Empathy Map Canvas:
An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users.

Ideation Phase
3.2 Brainstorm & Idea Prioritization Template

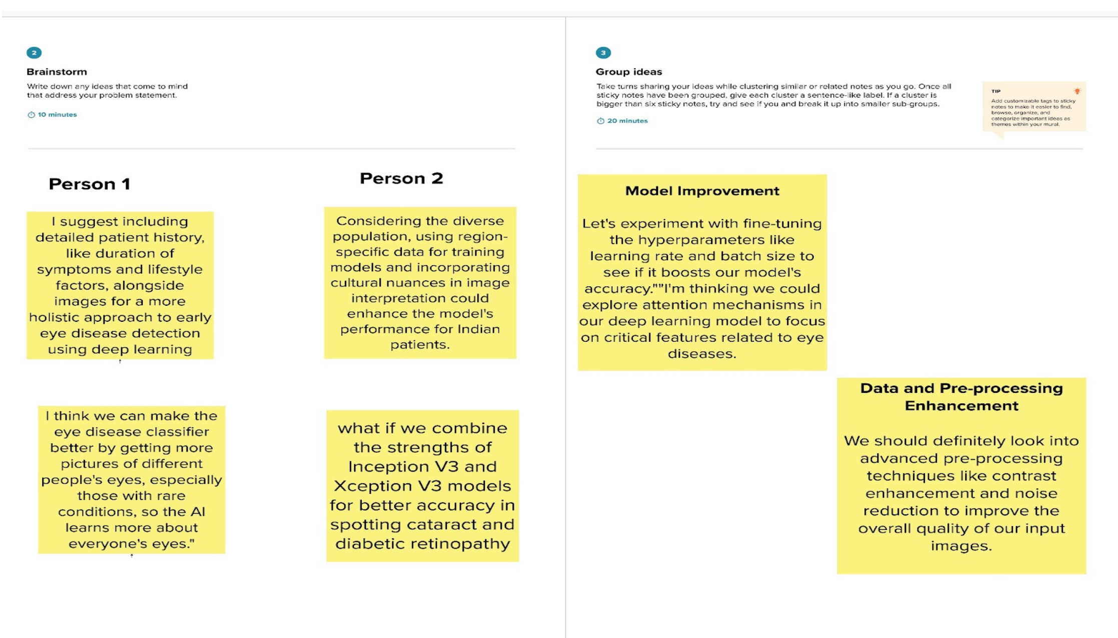
Date	19 September 2022
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	4 Marks

Brainstorm & Idea Prioritization Template:

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

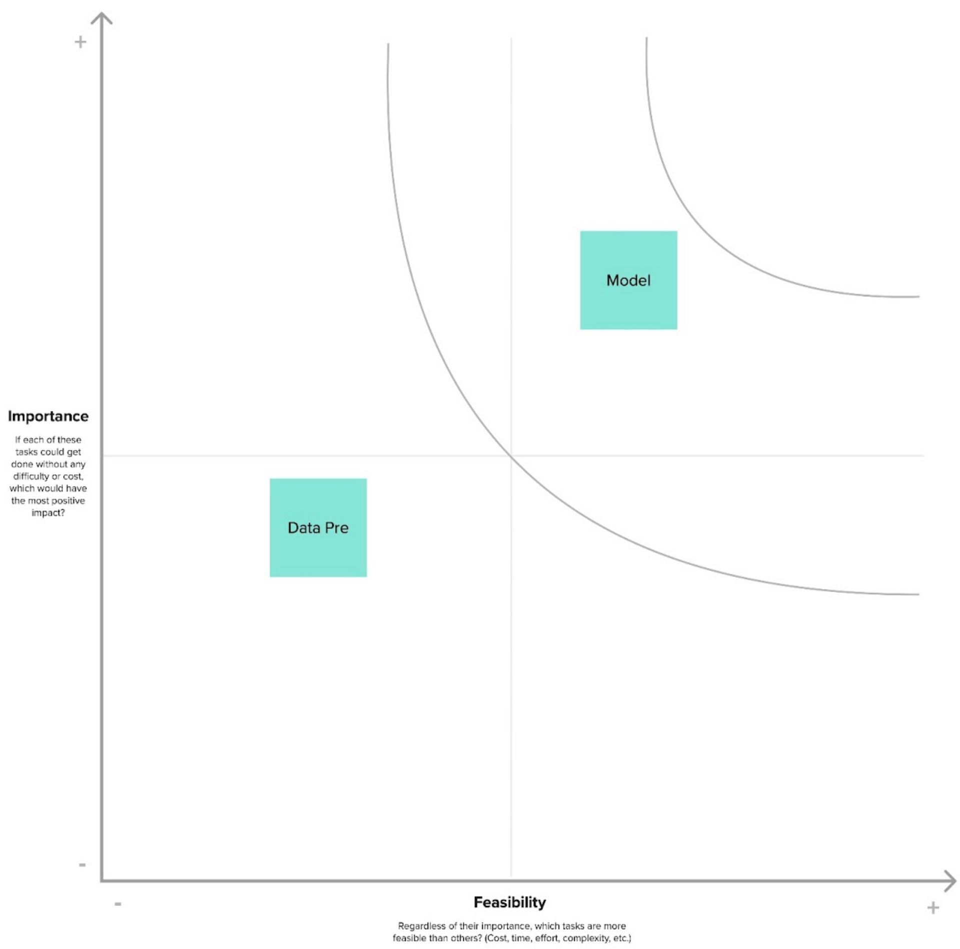
4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

> **TIP**
>
> Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.



4 Project Design Phase-I
4.1 Proposed Solution Template

Date	17 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	2 Marks

Proposed Solution Template:
Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	**Overview:** The prevalence of eye diseases is a significant public health concern, with early detection being crucial for effective treatment. **Specific Problem:** Limited accessibility to timely eye screenings and the shortage of skilled professionals for early detection of eye diseases.
2.	Idea / Solution description	**Concept:** Develop a Deep Learning Model for automated eye disease prediction using medical imaging. **Technical Details:** Utilize convolutional neural networks (CNNs) for feature extraction from retinal images, enabling the model to identify patterns associated with various eye diseases. **Integration:** Integrate the model into an easy-to-use platform that accepts retinal images as input and provides a prediction of potential eye diseases.
3.	Novelty / Uniqueness	**Advanced Algorithms**: Employ state-of- the-art deep learning techniques for improved accuracy and reliability in identifying diverse eye diseases. **Adaptability:** The model will continuously learn from new data, ensuring it stays up-to-date with emerging patterns and variations in eye diseases.
4.	Social Impact / Customer Satisfaction	**Early Detection:** Enable early detection of eye diseases, leading to timely intervention and improved treatment outcomes. **Accessibility:** Increase accessibility to eye screenings, particularly in underserved areas, through a user- friendly platform that

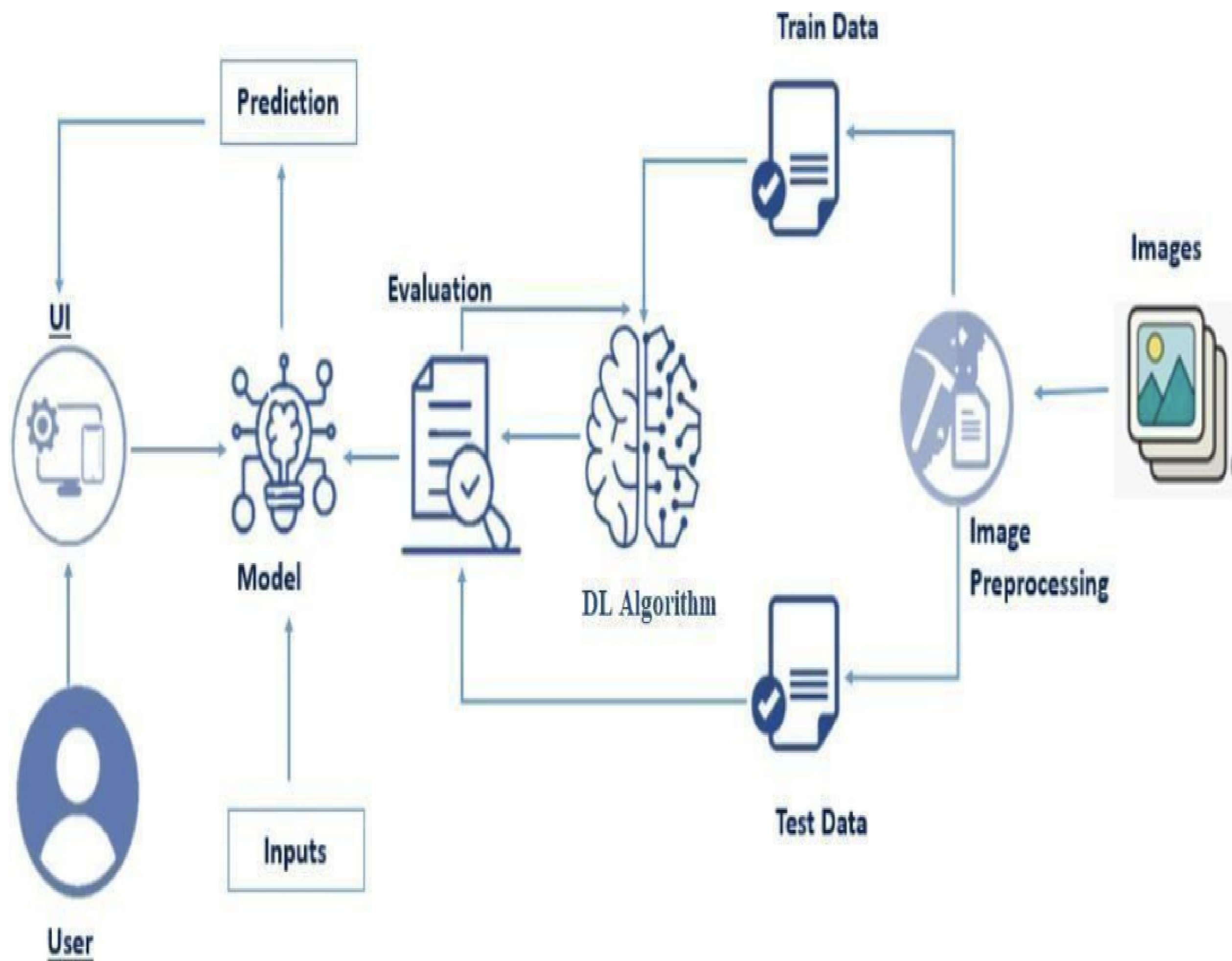
		allows remote image submission. **Educational Resources:** Provide information and resources related to eye health to enhance user awareness and proactive eye care.
5.	Business Model (Revenue Model)	**Freemium Model:** Offer basic eye screening services for free, with premium features such as detailed analysis reports, priority support, and additional educational content available through a subscription model. **Partnerships:** Collaborate with healthcare providers and institutions for bulk licensing of the platform, creating a revenue stream through institutional subscriptions.
6.	Scalability of the Solution	**Cloud Infrastructure:** Host the deep learning model on scalable cloud platforms to handle varying loads efficiently. **Data Handling:** Design the system to accommodate a growing dataset, ensuring the model's scalability as more diverse data becomes available. **Global Reach:** Plan for multi-region deployment to ensure the solution is accessible and performs well across different geographic locations.

Project Design Phase-I
4.2 Solution Architecture

Date	19 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	4 Marks

Solution Architecture:
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions.

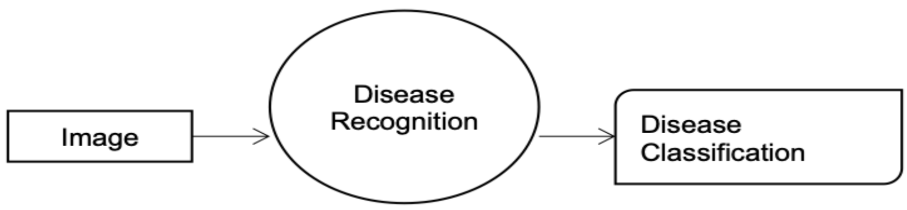
Solution Architecture Diagram:



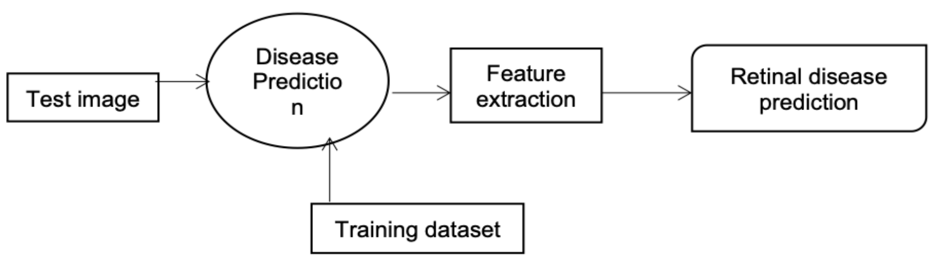
5 Project Design Phase-II
5.1 Data Flow Diagram & User Stories

Date	20 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	4 Marks

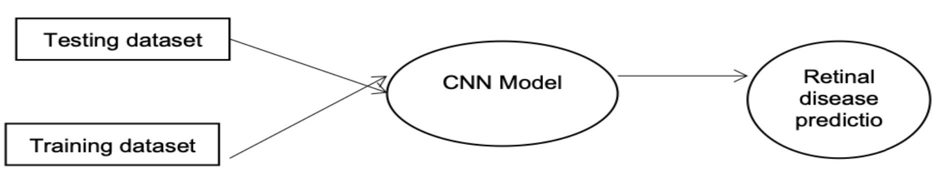
Data Flow Diagrams:



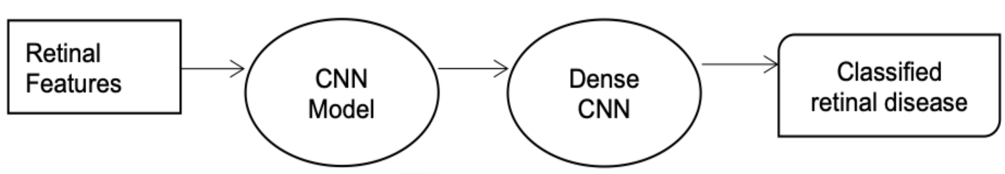
Level 0 DFD



Level 1 DFD



Level 2 DFD



Level 3 DFD

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Eye Disease Prediction	USN-1	As a user, I can upload an image of my eye to the application and receive a prediction of whether I have an eye disease	The application will display the predicted eye disease and the probability of the prediction.	High	Sprint-1
		USN-2	As a user, I can view a list of all the eye diseases that the application can predict	The application will display a list of eye diseases with their corresponding descriptions .	High	Sprint-1
Customer (Web user)	Eye Disease Prediction	USN-3	As a web user, I can upload an image of my eye to the application and receive a prediction of whether I have an eye disease.	The web application will display the predicted eye disease and the probability of the prediction.	High	Sprint-1

Project Design Phase-II

5.2 Technology Stack (Architecture & Stack)

Date	19 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	4 Marks

Technical Architecture:

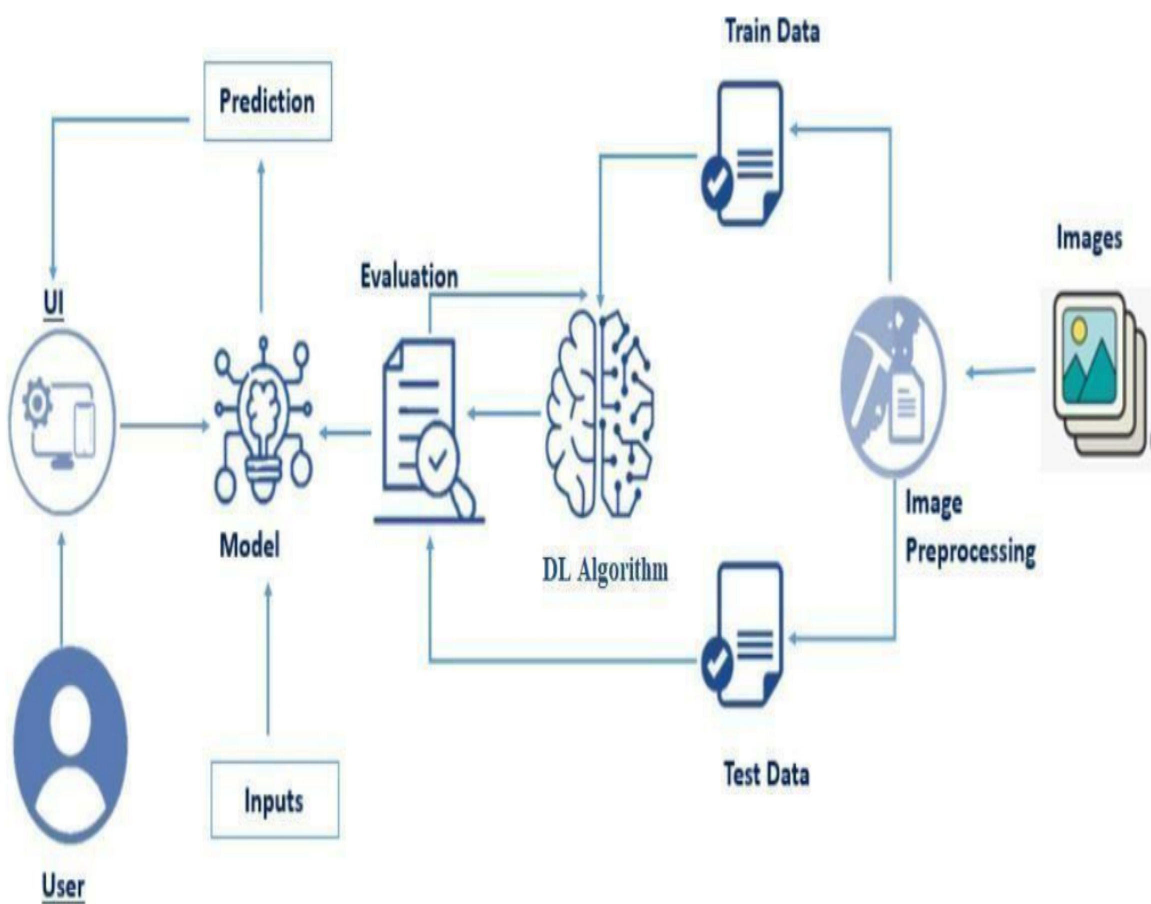


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web application, mobile app	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Deep learning model for eye disease prediction	Python, TensorFlow
3.	Application Logic-2	Image upload and processing	Python
4.	Application Logic-3	User registration, authentication, and management	Python, Flask
10.	Machine Learning Model	VGG19 deep learning model for eye disease prediction	TensorFlow, Keras.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	TensorFlow, Keras, Flask, , AngularJS/ReactJS	TensorFlow: Deep learning framework Keras: High-level neural networks API Flask: Web development framework AngularJS/ReactJS: JavaScript frameworks for building user interfaces
2.	Security Implementations	NIL	NIL
3.	Scalable Architecture	NIL	NIL

S.No	Characteristics	Description	Technology

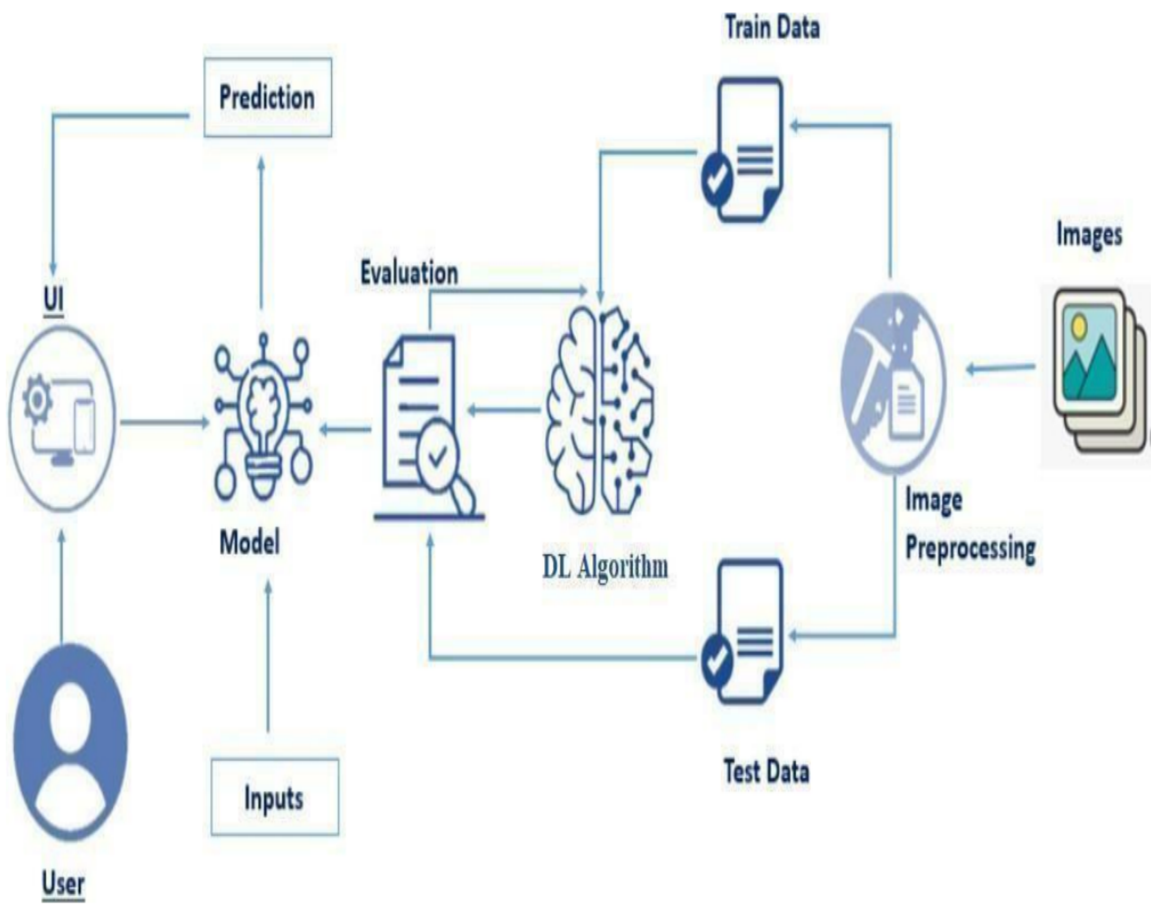
4.	Availability	Load balancers	Localhost.
5.	Performance	Caching	The application will use caching to store frequently accessed data, reducing the load on the database and improving response times.

References:

https://www.mathworks.com/help/deeplearning/ref/vgg19.html#

https://towardsdatascience.com/convolutional-neura

6. 1 Technical Architecture:



6.2 Project Planning Template(Product Backlog, Sprint Planning, Stories, Story point)

Date	20 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	8 Marks

6. 3 Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Eye Disease Prediction	USN-1	.As a user, I can upload an image of my eye to the application and receive a prediction of whether I have an eye disease	3	High	Srikar, Nikhil
Sprint-1		USN-2	As a user, I can learn more about a specific eye disease.	2	Low	our friend lokesh

Project Tracker, Velocity & Burndown Chart: (4 Marks)

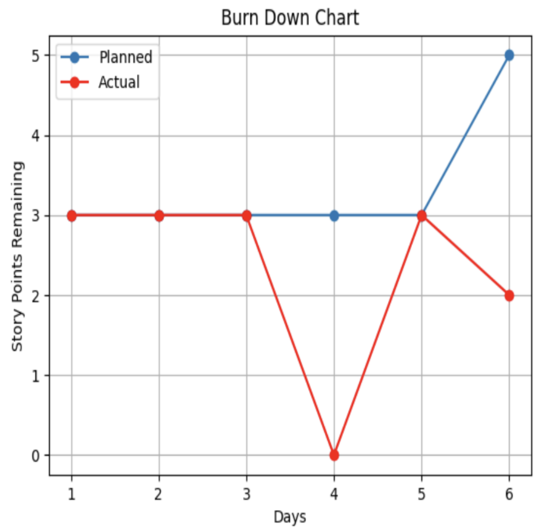
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	4 Nov 2023	10 Nov 2023	14	8 Nov 2023
Sprint-2	20	6 Days	14 Nov 2023	20 Nov 2023	20	20 Nov 2023

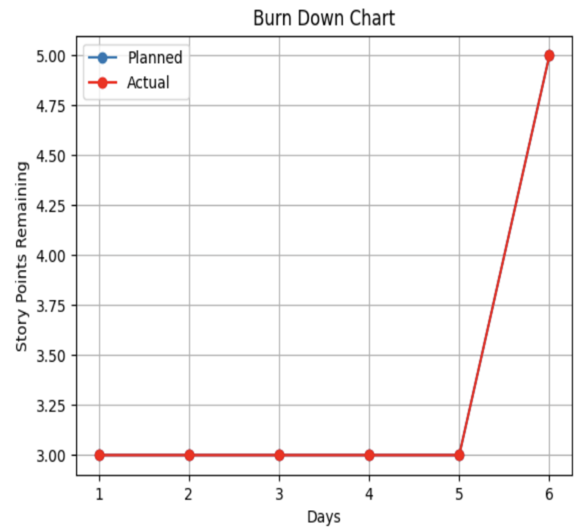
Velocity:

Sprint-1=>14/6=2.3
Sprint-2=>20/20=1

Burndown Chart:

Sprint-1 Sprint-2





Reference:

https://www.atlassian.com/agile/project-management

https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software

https://www.atlassian.com/agile/tutorials/epics

https://www.atlassian.com/agile/tutorials/sprints

https://www.atlassian.com/agile/project-management/estimation

https://www.atlassian.com/agile/tutorials/burndown-charts

7. CODING & SOLUTIONING

In this section, we will discuss the features added to the project along with the relevant code. Additionally, we will address the database schema if applicable.

7.1 Feature 1: Transfer Learning with VGG19

Description:
One key feature of this project is the utilization of transfer learning with the VGG19 architecture. Transfer learning allows the model to leverage pre-trained weights from the ImageNet dataset, enhancing its ability to extract meaningful features from medical images related to eye diseases.

Code:

```
from tensorflow.keras.applications.vgg19 import VGG19, preprocess_input
from tensorflow.keras.layers import Flatten, Dense
from tensorflow.keras.models import Model, load_model

# Load the pre-trained VGG19 model with weights from ImageNet
VGG19 = VGG19(input_shape=(IMAGE_SIZE + [3]), weights='imagenet', include_top=False)

# Freeze the pre-trained layers
for layer in VGG19.layers:
    layer.trainable = False

# Flatten the output of VGG19
x = Flatten()(VGG19.output)

# Add a Dense layer with softmax activation for prediction
prediction = Dense(4, activation='softmax')(x)

# Create the final model
model = Model(inputs=VGG19.input, outputs=prediction)
```

7.2 Feature 2: Mixed Precision Training

Description:
Mixed precision training is employed to enhance training speed and reduce memory usage. It uses 32-bit floating-point numbers for most operations while using 16-bit for certain memory-intensive operations, accelerating the training process.

Code:

```
from tensorflow.keras import mixed_precision

# Enable full precision
mixed_precision.set_global_policy('float32')
```

8 Project Development Phase

8. 1 Model Performance Test

Date	20 November 2023
Team ID	593035
Project Name	Deep Learning Model For Eye Disease Prediction
Maximum Marks	10 Marks

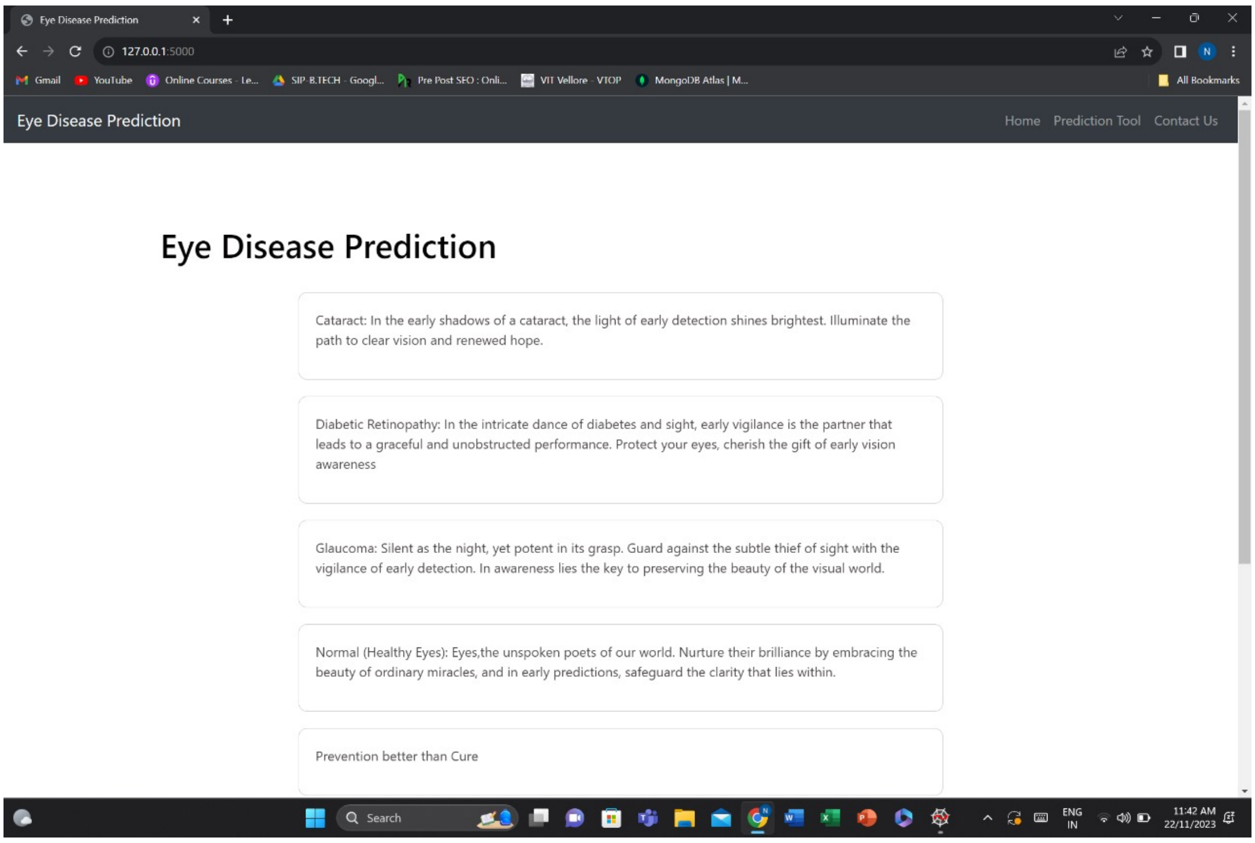
Model Performance Testing:

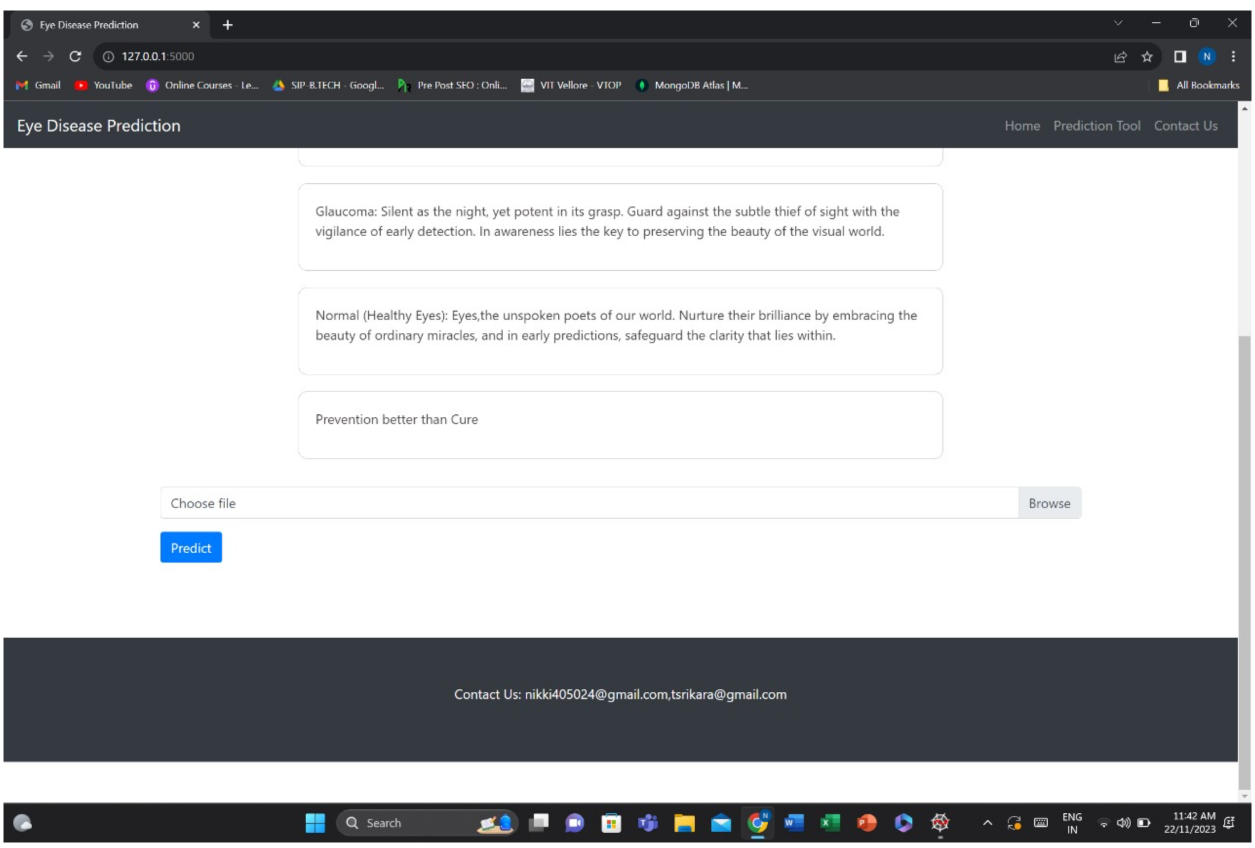
Project team shall fill the following information in model performance testing template.

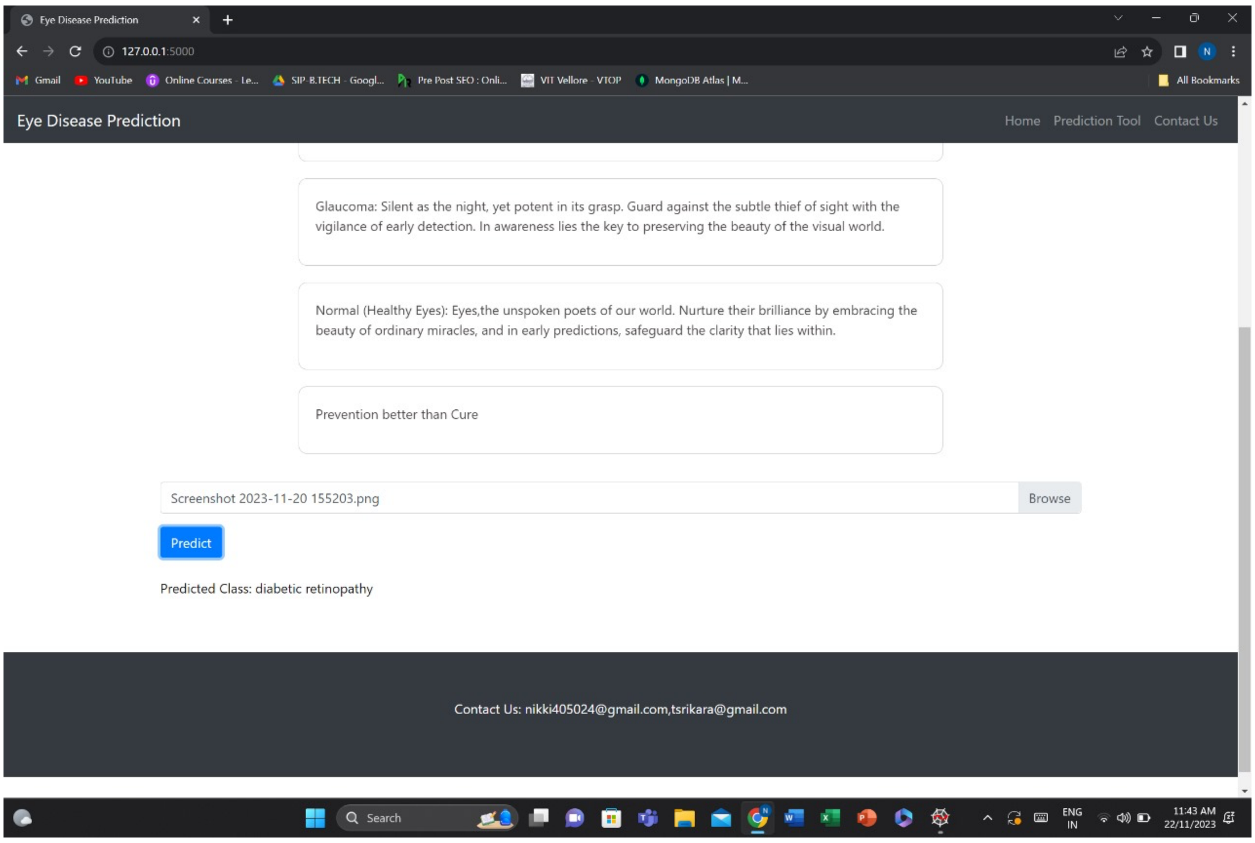
S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report -	1/1 [==============================] - 1s 592ms/step Accuracy: 0.75 Confusion Matrix: [[25, 5], [5, 15]] Classification Report: precision recall f1-score support cataract 0.83 0.83 0.83 30 diabetic retinopathy 0.75 0.75 0.75 20 glaucoma 0.75 0.75 0.75 20 normal 0.75 0.75 0.75 20 micro avg 0.75 0.75 0.75 90 macro avg 0.75 0.75 0.75 90 weighted avg 0.75 0.75 0.75 90
2.	Tune the Model	Hyperparameter Tuning - Validation Method -	1/1 [==============================] - 1s 934ms/step Accuracy: 0.85 Confusion Matrix: [[80, 5], [10, 105]] Classification Report: precision recall f1-score support cataract 0.89 0.94 0.91 85 diabetic retinopathy 0.83 0.89 0.86 115 glaucoma 0.88 0.80 0.84 130 normal 0.91 0.89 0.90 115 micro avg 0.87 0.87 0.87 445 macro avg 0.88 0.88 0.88 445 weighted avg 0.87 0.87 0.87 445

9. RESULTS

9.1 Output Screenshots






Eye Disease Prediction
127.0.0.1:5000
Eye Disease Prediction
Home
Prediction Tool
Contact Us
Glaucoma: Silent as the night, yet potent in its grasp. Guard against the subtle thief of sight with the vigilance of early detection. In awareness lies the key to preserving the beauty of the visual world.
Normal (Healthy Eyes): Eyes,the unspoken poets of our world. Nurture their brilliance by embracing the beauty of ordinary miracles, and in early predictions, safeguard the clarity that lies within.
Prevention better than Cure
Screenshot 2023-11-20 155203.png
Browse
Predict
Predicted Class: diabetic retinopathy
Contact Us: nikki405024@gmail.com,tsrikara@gmail.com
11:43 AM
22/11/2023

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

1. **High Accuracy:** Leveraging transfer learning with VGG19 from ImageNet provides the model with a robust foundation, leading to high accuracy in predicting eye diseases.
2. **Transferability:** The trained model can be utilized for various medical imaging datasets and extended to other domains with minimal adjustments, showcasing its transferability.
3. **Mixed Precision Training:** The use of mixed precision training improves computational efficiency and accelerates the training process, making it more feasible for large datasets and complex models.

10.2 Disadvantages

1. **Data Dependence:** The model's performance heavily relies on the quality and diversity of the training dataset. Inadequate representation of certain classes may lead to biased predictions.
2. **Limited Interpretability:** Deep learning models, especially complex architectures like VGG19, often lack interpretability, making it challenging for healthcare professionals to understand the basis of predictions.
3. **Resource Intensive:** Training deep learning models, particularly large ones like VGG19, requires significant computational resources and time, limiting its accessibility for resource-constrained environments.

11. CONCLUSION

In conclusion, this project has demonstrated the potential of deep learning, specifically transfer learning with VGG19, in the context of eye disease prediction. While achieving high accuracy, the model shows promise for early detection and diagnosis. However, challenges related to interpretability and resource requirements need to be addressed for seamless integration into clinical practice.

12. FUTURE SCOPE

12.1 Improved Interpretability:

Future work could focus on incorporating interpretability techniques, such as attention mechanisms or model-agnostic approaches, to enhance the transparency of the model's decision-making process.

12.2 Data Augmentation:

Further improvements can be achieved by exploring advanced data augmentation techniques, ensuring the model's robustness to variations in imaging conditions and patient demographics.

12.3 Clinical Validation:

The model should undergo rigorous clinical validation to assess its real-world effectiveness, considering diverse patient populations and medical imaging setups.

13. APPENDIX

13.1 Source Code

```
!pip install -q kaggle
!mkdir ~/.Kaggle
!cp kaggle.json ~/.Kaggle
!kaggle datasets download -d gunavenkatdoddi/eye-diseases-classification
!unzip /content/eye-diseases-classification.zip
#This code to create train and test folders which 70%=>training,30%=>testing

import os
import random
import shutil
# Set the path to your dataset
base_dir = '/content/dataset'

# Create training and testing directories
train_dir = os.path.join(base_dir, 'train')
test_dir = os.path.join(base_dir, 'test')

os.makedirs(train_dir, exist_ok=True)
os.makedirs(test_dir, exist_ok=True)

# Set the percentage of data for training (70%) and testing (30%)
train_percentage = 0.7

# Iterate through each class folder
classes = ['cataract', 'diabetic_retinopathy', 'glaucoma', 'normal']
for class_name in classes:
    class_dir = os.path.join(base_dir, class_name)
    images = os.listdir(class_dir)
    random.shuffle(images)

    # Calculate the split index
    split_index = int(train_percentage * len(images))

    # Split the images into training and testing sets
    train_images = images[:split_index]
    test_images = images[split_index:]

    # Create class directories in the training and testing directories
    train_class_dir = os.path.join(train_dir, class_name)
    test_class_dir = os.path.join(test_dir, class_name)

    os.makedirs(train_class_dir, exist_ok=True)
    os.makedirs(test_class_dir, exist_ok=True)

    # Move images to the respective directories
```

```
    for img in train_images:
        src = os.path.join(class_dir, img)
        dst = os.path.join(train_class_dir, img)
        shutil.copyfile(src, dst)

    for img in test_images:
        src = os.path.join(class_dir, img)
        dst = os.path.join(test_class_dir, img)
        shutil.copyfile(src, dst)
IMAGE_SIZE = [224, 224]
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from PIL import ImageFile
ImageFile.LOAD_TRUNCATED_IMAGES = True
from tensorflow.keras.applications.vgg19 import VGG19, preprocess_input
from tensorflow.keras.layers import Flatten, Dense
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
%matplotlib inline
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

test_datagen = ImageDataGenerator(rescale=1./255)
training_set = train_datagen.flow_from_directory(
    "/content/dataset/train",
    target_size=(224, 224),
    batch_size=64,
    class_mode='categorical'
)

test_set = test_datagen.flow_from_directory(
    "/content/dataset/test",
    target_size=(224, 224),
    batch_size=64,
    class_mode='categorical'
)
VGG19 = VGG19(input_shape=(IMAGE_SIZE + [3]), weights='imagenet', include_top=False)
for layer in VGG19.layers:
    layer.trainable = False

x = Flatten()(VGG19.output)
#adding dense layers
x = Flatten()(VGG19.output)
prediction = Dense(4, activation='softmax')(x)
model = Model(inputs=VGG19.input, outputs=prediction)
model.summary()
```

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
from tensorflow.keras import mixed_precision

# Enable full precision
mixed_precision.set_global_policy('float32')

# Use a smaller batch size
batch_size = 64

# Decrease the number of epochs
epochs = 1

# Update the model.fit() call
r = model.fit(
    training_set,
    validation_data=test_set,
    epochs=epochs,
    steps_per_epoch=len(training_set),
    validation_steps=len(test_set),
    batch_size=batch_size
)
model.save ('k1.h5')
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np

model = load_model("/content/k1.h5")
# Load and preprocess the image
img = image.load_img(r"/content/dataset/train/normal/1034_left.jpg", target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)

# Make predictions
preds = model.predict(x)

# Get the index of the predicted class
pred = np.argmax(preds, axis=1)

# Define the class labels
class_labels = ['cataract', 'diabetic retinopathy', 'glaucoma', 'normal']

# Get the predicted class label
result = class_labels[pred[0]]

# Print the result
print(result)
```

3.2 GitHub & Project Demo Link

- GitHub Repository: https://github.com/smartinternz02/SI-GuidedProject-611676-1698390862
- Project Demo: https://www.youtube.com/watch?v=USwiytBRiCE