



**INFYM**

# **DIABETES DETECTION THROUGH RETINOPATHY HACKATHON REPORT**

**TEAM MASAKALI**

Hamza Ali Khan  
Farzam Nisar  
Aqiba Abdul Qadir

# TABLE OF CONTENTS

<b>INTRODUCTION.....</b>	<b>3</b>
<b>PROBLEM STATEMENT.....</b>	<b>3</b>
<b>TECHNOLOGY STACK.....</b>	<b>3</b>
<b>DATASET DETAILS.....</b>	<b>3</b>
<b>IMPLEMENTATION STEPS.....</b>	<b>4</b>
Step 1: Data Preprocessing.....	4
Step 2: Model Development.....	4
Step 3: Model Training & Evaluation.....	4
Step 4: Front-end Development.....	4
<b>METHODOLOGY &amp; MODEL TRAINING.....</b>	<b>4</b>
Model 1: Pre-trained EfficientNet-B3.....	4
Model 2: Final Fine-tuned EfficientNet-B3.....	5
Model 3: Fine-tuned EfficientNet-B3.....	5
Model 4: Pre-trained Inception-V3.....	5
<b>MODEL COMPARISONS &amp; PERFORMANCE.....</b>	<b>6</b>
<b>CHALLENGES FACED.....</b>	<b>6</b>
1. Limited Computation Power in Colab.....	6
2. Overfitting While Training EfficientNet-B3 Model.....	6
<b>TESTING SNAPSHOTS.....</b>	<b>7</b>
1. Class 0.....	7
2. Class 1.....	8
3. Class 2.....	9
4. Class 3.....	10
5. Class 4.....	11
<b>CONCLUSION.....</b>	<b>12</b>

# INTRODUCTION

---

This report outlines our project on *Diabetes Detection Through Retinopathy*, developed during the hackathon. Our goal was to utilise deep learning to detect diabetic retinopathy from retinal images, enabling early diagnosis and reducing the risk of blindness.

## PROBLEM STATEMENT

---

Diabetic retinopathy is a major cause of vision impairment worldwide. The manual diagnosis process is time-consuming and requires expert ophthalmologists. Our aim was to automate and enhance the detection process using AI-driven image classification models.

## TECHNOLOGY STACK

---

- **Programming Language:** Python, HTML, CSS
- **Frameworks & Libraries:** PyTorch, Django, Matplotlib, Numpy, scikit-learn
- **Model Architectures:** EfficientNet-B3, InceptionV3, ResNet
- **Dataset:** kushagrandon12/diabetic-retinopathy-balanced
- **Development Tools:** Google Colab, Visual Studio Code

## DATASET DETAILS

---

- The dataset can be downloaded from <https://www.kaggle.com/datasets/kushagrandon12/diabetic-retinopathy-balanced/data>
- The dataset consists of labeled retinal fundus images.
- Images are provided in JPEG format.
- There are 5 Types Of Diabetic Retinopathy Stages:
  1. No\_Dr
  2. Mild
  3. Moderate
  4. Severe
  5. Proliferative DR

# IMPLEMENTATION STEPS

---

## Step 1: Data Preprocessing

- Downloaded the dataset of retinal images.
- Applied resizing and normalization.

## Step 2: Model Development

- Implemented and trained multiple deep learning models.
- Fine-tuned pre-trained architectures, including ResNet, EfficientNet, and Inception.

## Step 3: Model Training & Evaluation

- Trained models using different hyperparameters.
- Evaluated models using accuracy, precision, recall, and F1-score.

## Step 4: Front-end Development

- Developed a Django-based web application for real-time diabetic retinopathy detection.
- Integrated the trained model into Django views to process uploaded images.
- Hosted the web application for accessibility.

# METHODOLOGY & MODEL TRAINING

---

To optimize performance, we employed the following training strategies for three different models before finalizing our most efficient model:

## Model 1: Pre-trained EfficientNet-B3

- **Pretrained Model:** Used EfficientNet-B3 with ImageNet weights.
- **Hyperparameters:**
  - Learning Rate: 0.001 (Adam optimizer)
  - Batch Size: 32
  - Epochs: 10
- **Loss Function:** Cross-Entropy Loss
- **Evaluation:** Achieved the best training accuracy of 98.89% and validation accuracy of 77.94% after fine-tuning.
- **Notebook:** initialModel.ipynb

	precision	recall	f1-score	support
0	0.65	0.66	0.66	1000
1	0.68	0.67	0.67	971
2	0.67	0.67	0.67	1000
3	0.93	0.93	0.93	1000
4	0.97	0.96	0.97	1000
accuracy			0.78	4971
macro avg	0.78	0.78	0.78	4971
weighted avg	0.78	0.78	0.78	4971

## Model 2: Final Fine-tuned EfficientNet-B3

- **Pretrained Model:** Used EfficientNet-B3 with pre-trained weights from previous training. Trained model on validation set to improve accuracy.
- **Hyperparameters:**
  - Learning Rate: 0.0001 (Adam optimizer)
  - Batch Size: 16
  - Epochs: 14
- **Loss Function:** Cross-Entropy Loss
- **Evaluation:** Achieved the best validation accuracy of 79.75% after fine-tuning.
- **Notebook:** modelFinetuned.ipynb
- **Model:** fineTunedEfficientnet\_b3.pt

```
Epoch 9/20, Loss: 0.6910, LR: 1e-05
Validation Accuracy: 79.75%
Epoch 10/20, Loss: 0.6886, LR: 1.0000000000000002e-06
Validation Accuracy: 79.68%
Epoch 11/20, Loss: 0.6876, LR: 1.0000000000000002e-06
Validation Accuracy: 79.73%
Epoch 12/20, Loss: 0.6876, LR: 1.0000000000000002e-06
Validation Accuracy: 79.64%
Epoch 13/20, Loss: 0.6897, LR: 1.0000000000000002e-06
Validation Accuracy: 79.39%
Epoch 14/20, Loss: 0.6905, LR: 1.0000000000000002e-06
Validation Accuracy: 79.71%
```

## Model 3: Fine-tuned EfficientNet-B3

- **Pretrained Model:** Used EfficientNet-B3 with ImageNet weights.
- **Hyperparameters:**
  - Learning Rate: 0.001 (Adam optimizer)
  - Batch Size: 32
  - Epochs: 10
- **Loss Function:** Cross-Entropy Loss
- **Evaluation:** Achieved the best validation accuracy of 78.26% after fine-tuning.
- **Notebook:** tuningHyperparams.ipynb
- **Model:** efficientModel.pt

## Model 4: Pre-trained Inception-V3

- **Pretrained Model:** Used Inception-V3 with ImageNet weights.
- **Hyperparameters:**
  - Learning Rate: 0.0001 (Adam optimizer)
  - Batch Size: 32
  - Epochs: 13
- **Loss Function:** Cross-Entropy Loss
- **Evaluation:** Achieved the best validation accuracy of 83.21% after fine-tuning. Due to a bug in the code the training process stopped also computation

```
Epoch 9 | Train Loss: 0.2282 | Val Loss: 0.4867 | Val Accuracy: 82.25%
No improvement for 1 epochs...
Epoch 10 | Train Loss: 0.1980 | Val Loss: 0.4777 | Val Accuracy: 83.25%
No improvement for 2 epochs...
Epoch 11 | Train Loss: 0.1732 | Val Loss: 0.4546 | Val Accuracy: 83.63%
No improvement for 3 epochs...
Epoch 12 | Train Loss: 0.1647 | Val Loss: 0.5345 | Val Accuracy: 83.07%
No improvement for 4 epochs...
Epoch 13 | Train Loss: 0.1466 | Val Loss: 0.5183 | Val Accuracy: 83.21%
No improvement for 5 epochs...
due to constrain we stop here
Training is completed! Team Masakali
```

limit on google colab disconnected the current state. We believe that the training process can lead to a validation score of above 85% accuracy.

- **Notebook:** inceptionFineTuned.ipynb
- **Model:** fineTunedInception.pt

## MODEL COMPARISONS & PERFORMANCE

---

Model	Testing Accuracy	Precision	Recall	F1-Score
EfficientNet-B3	98.89%(slight overfitting)	78%(avg)	0.66,0.67,0.67 , 0.93,0.96	78%(avg)
Fine-Tuned EFnet-B3	98.89%(slight overfitting)	Approximately same as above	Same as above	Same as above
Inception-V3	97%	Not tested	Not tested	Not tested

We finalised the fine-tuned EFnet-B3 model as we already created an interface for testing the EFnet-B3 and only found out about inceptionv3 when it was already too late :((. The best we could do is train and provide the model file.

## CHALLENGES FACED

---

### 1. Limited Computation Power in Colab

Due to hardware constraints, such as limited GPU memory and processing power, training times were prolonged, and batch sizes had to be reduced.

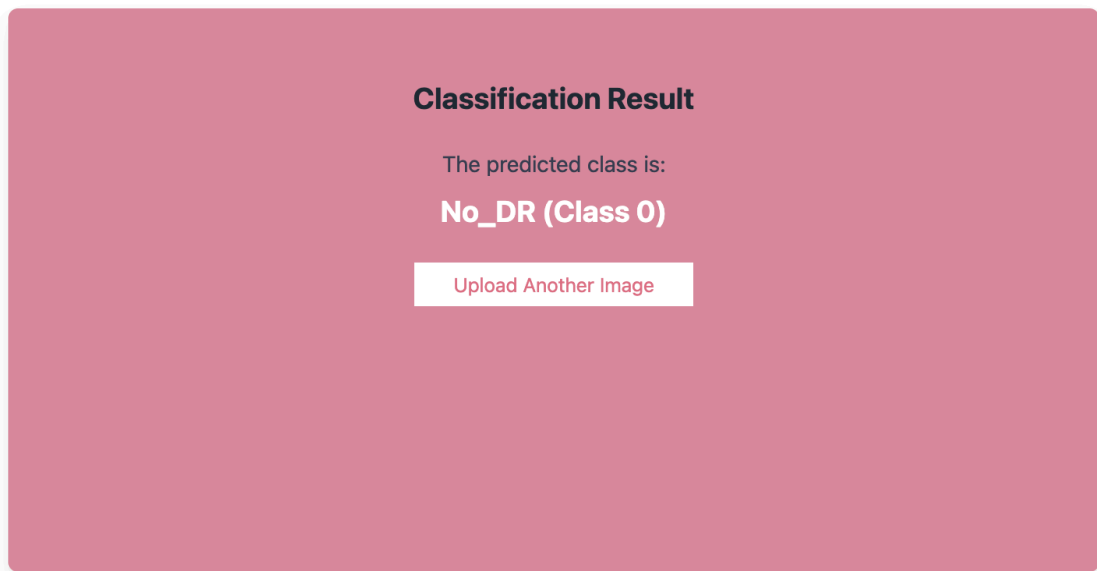
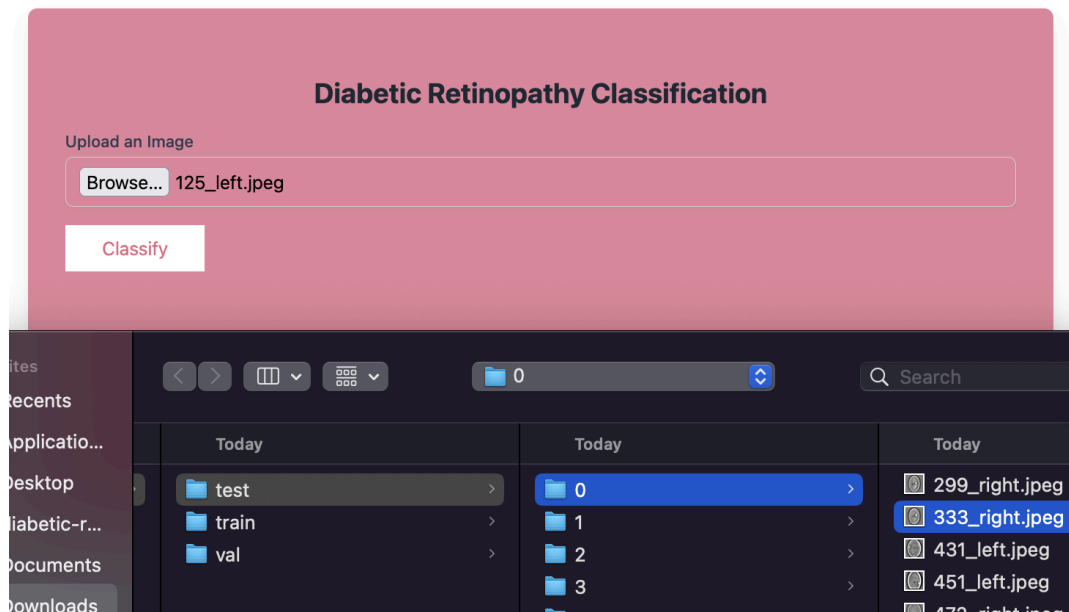
### 2. Overfitting While Training EfficientNet-B3 Model

EfficientNet-B3, being a powerful yet complex model, showed signs of overfitting during training. The model performed exceptionally well on the training dataset but struggled to generalize on validation data.

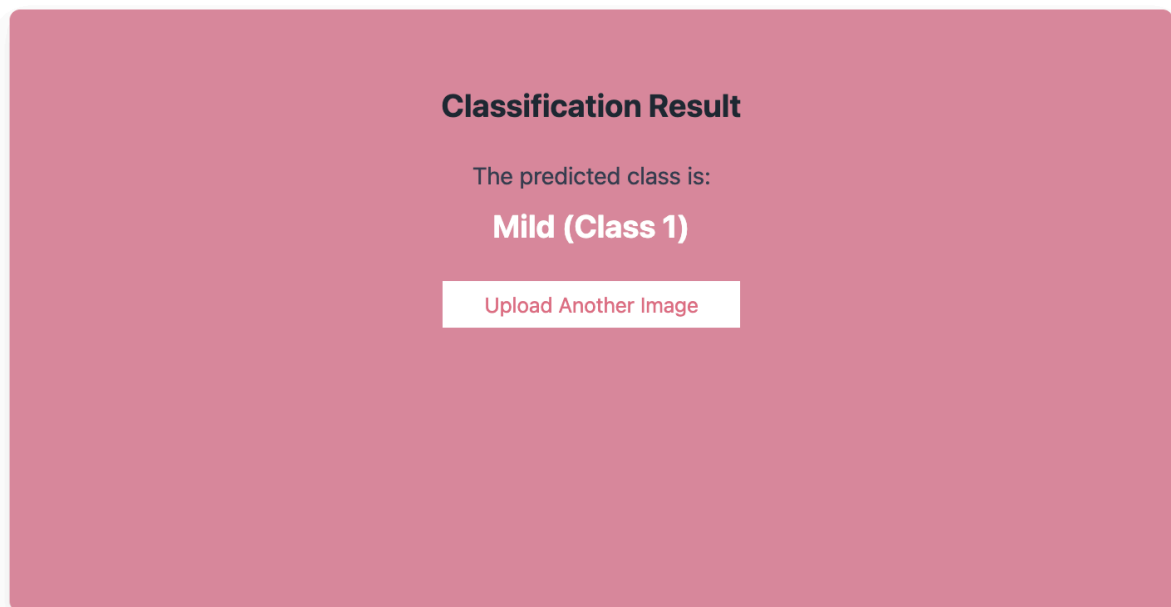
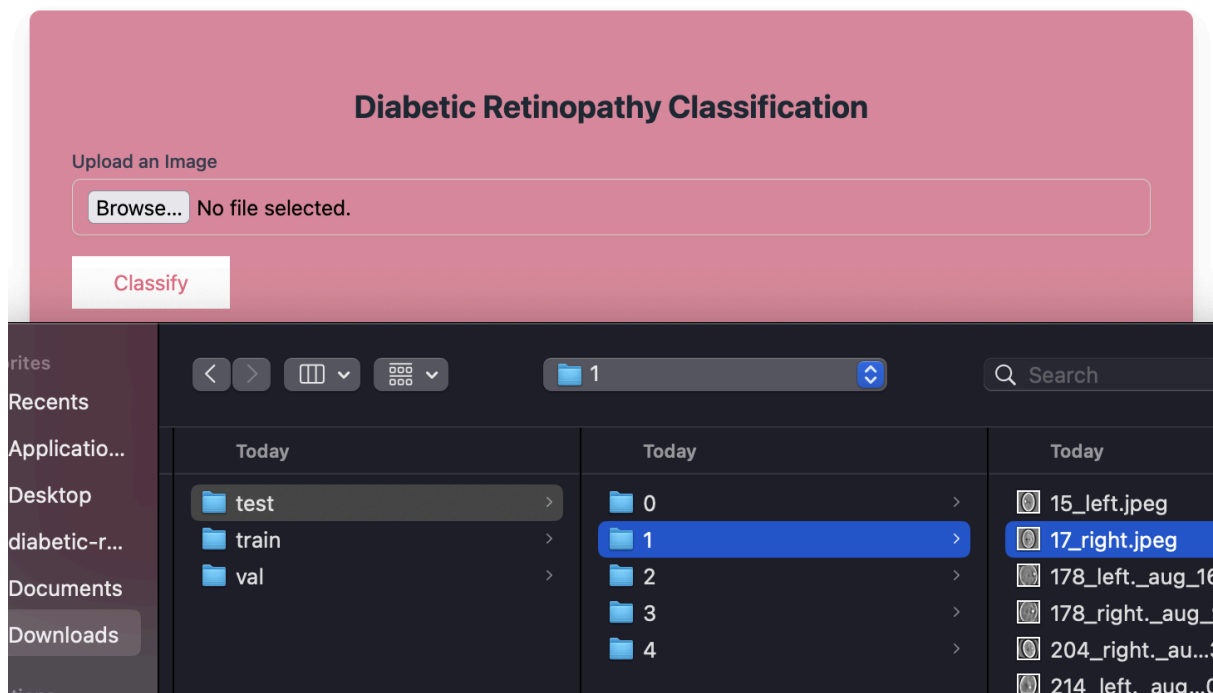
# TESTING SNAPSHOTS

---

## 1. Class 0

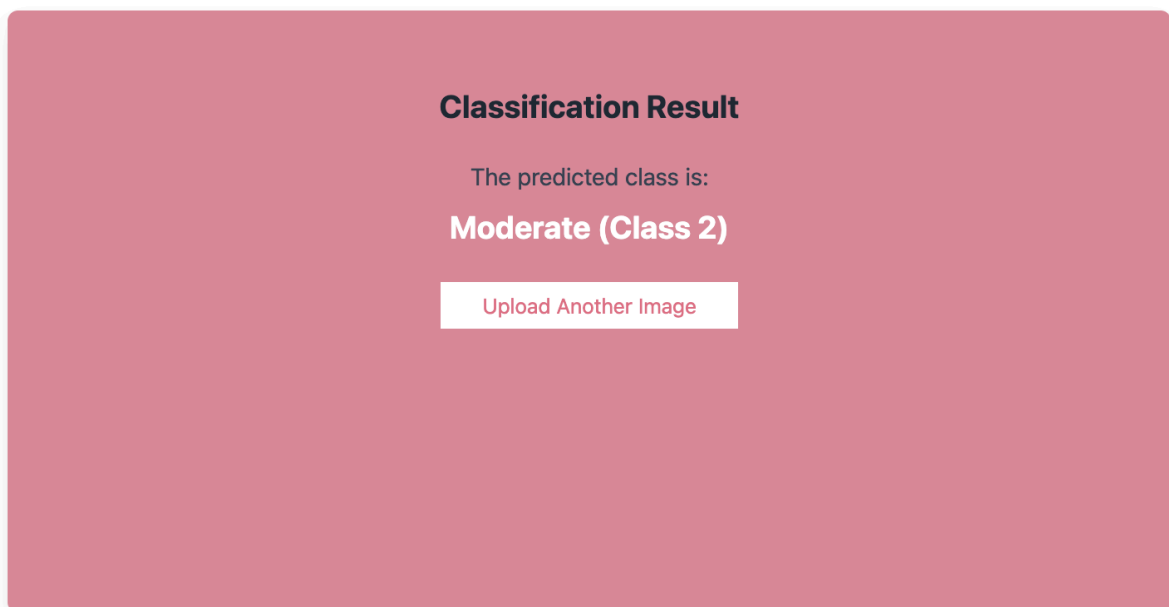
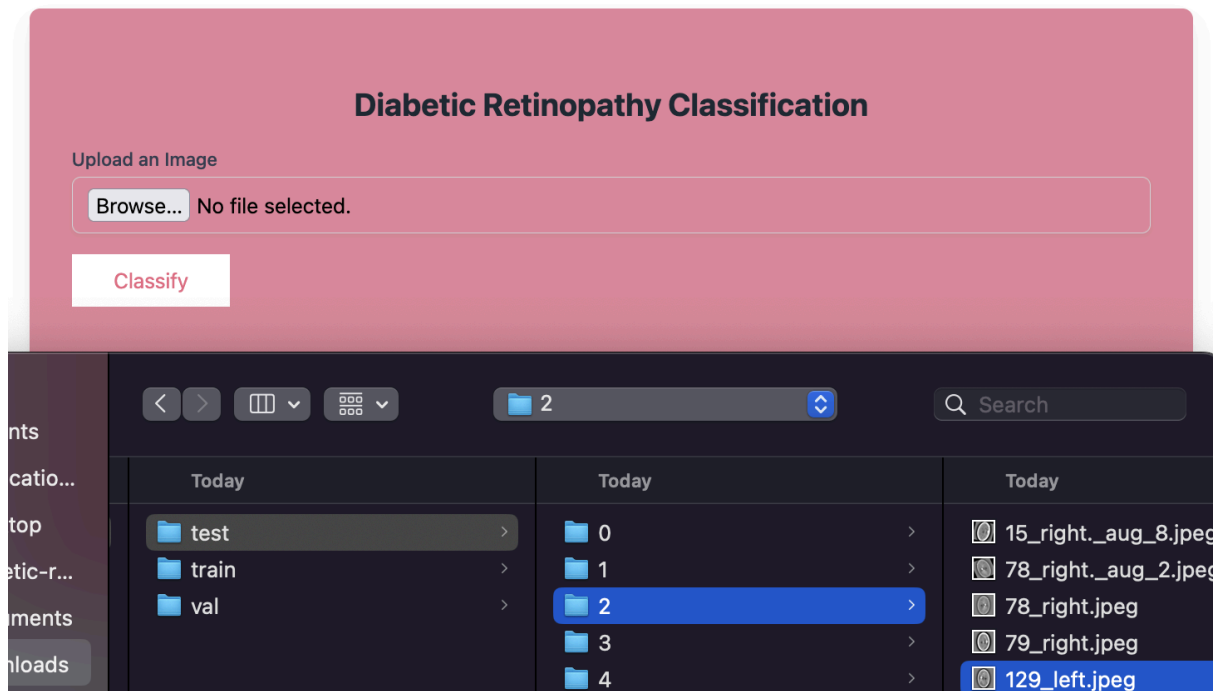


## 2. Class 1

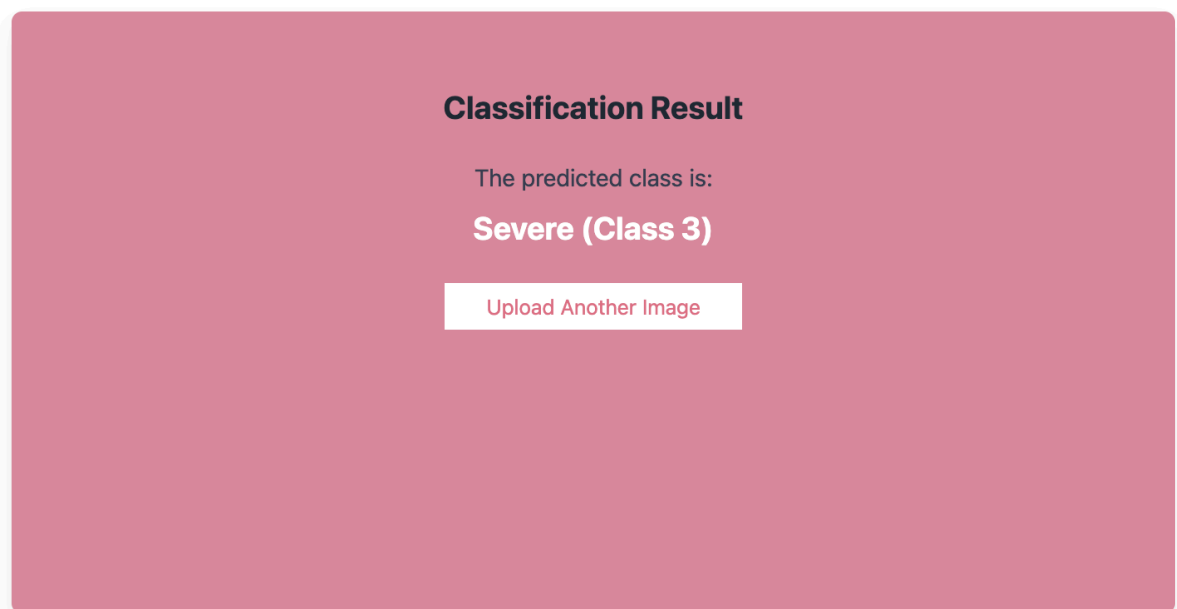
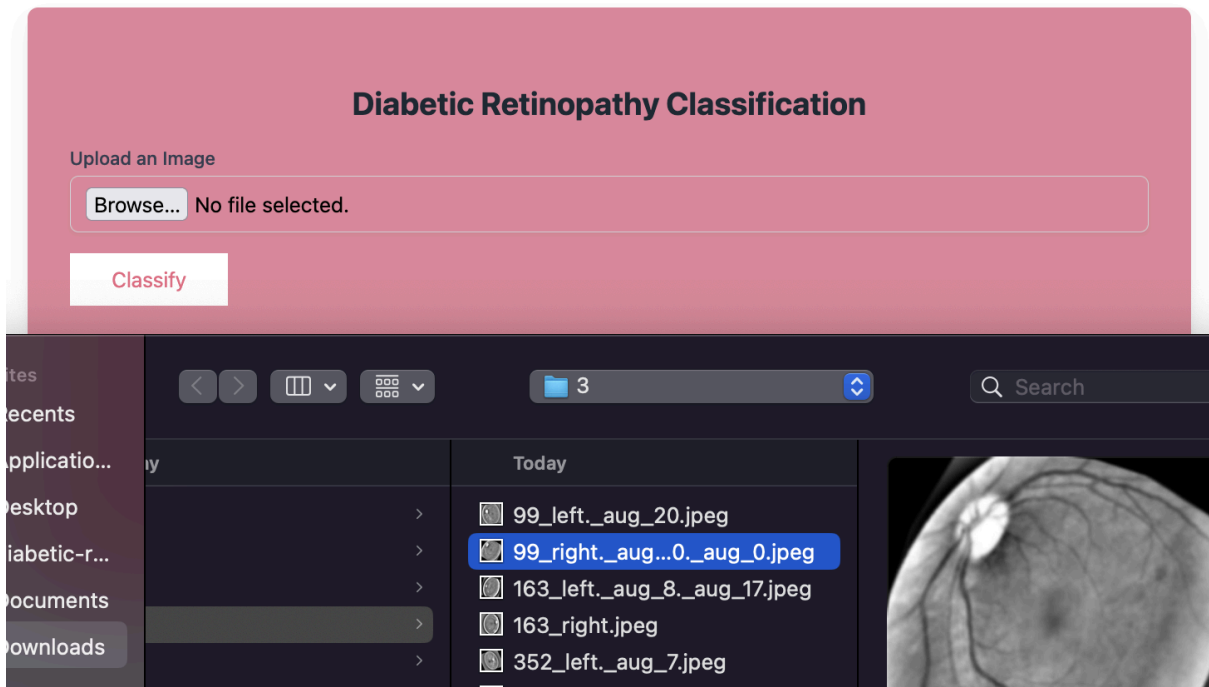




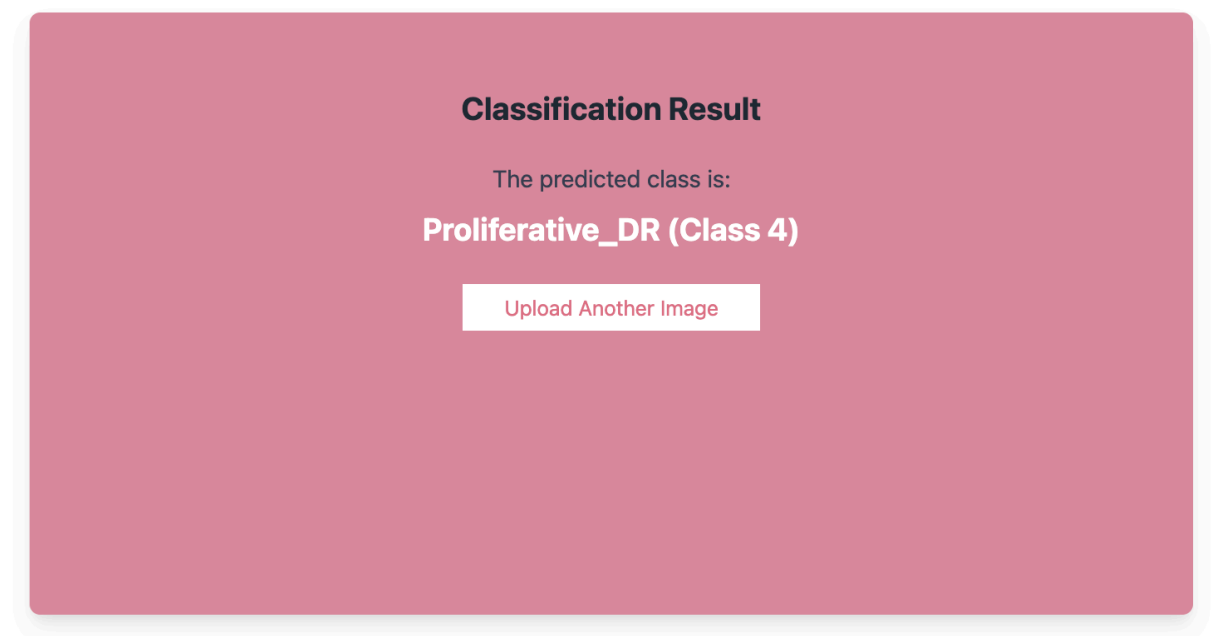
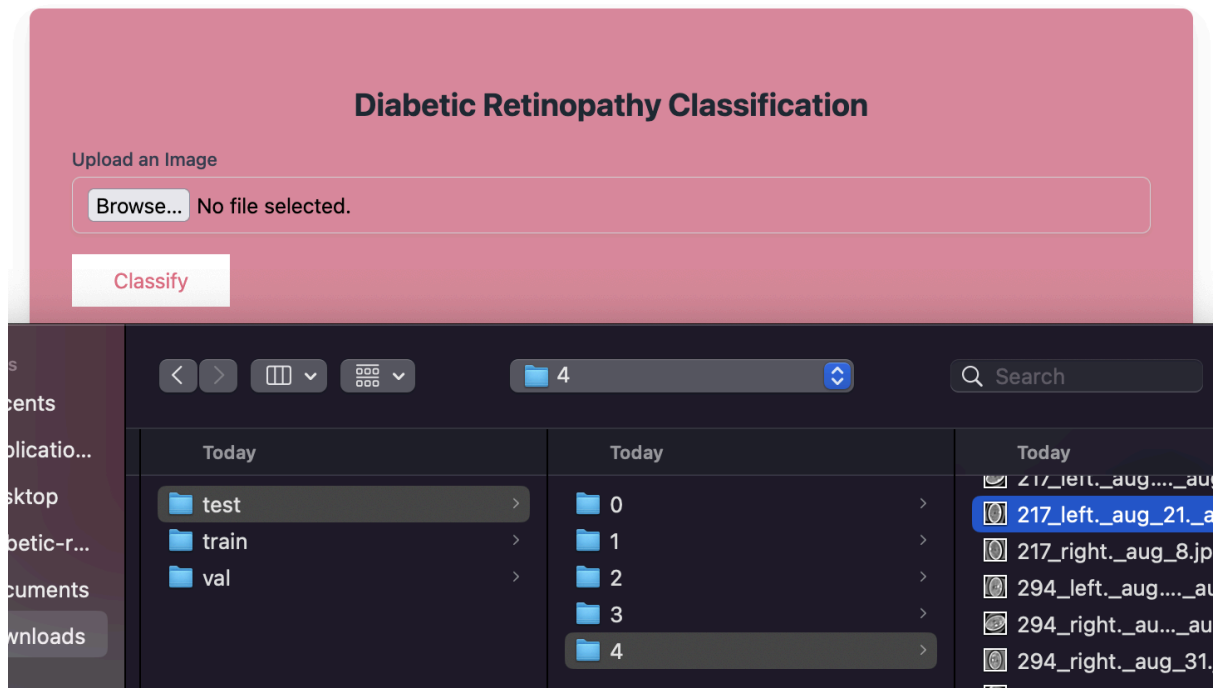
### 3. Class 2



## 4. Class 3



## 5. Class 4



# CONCLUSION

---

Our deep learning-based approach successfully demonstrated the potential of AI in early diabetic retinopathy detection. With further improvements, this system can significantly assist medical professionals in diagnosing the disease efficiently.