

# IMAGE CLASSIFICATION

Eye Disease Classification



Streamlit



PyTorch

<https://github.com/hardiantots/EyeDiseaseClassification>



# CONTENT

1

About Image Classification

2

Workflow In Pytorch

3

Deploying in Streamlit

4

Challenge & Conclusion



# ABOUT IMAGE CLASSIFICATION

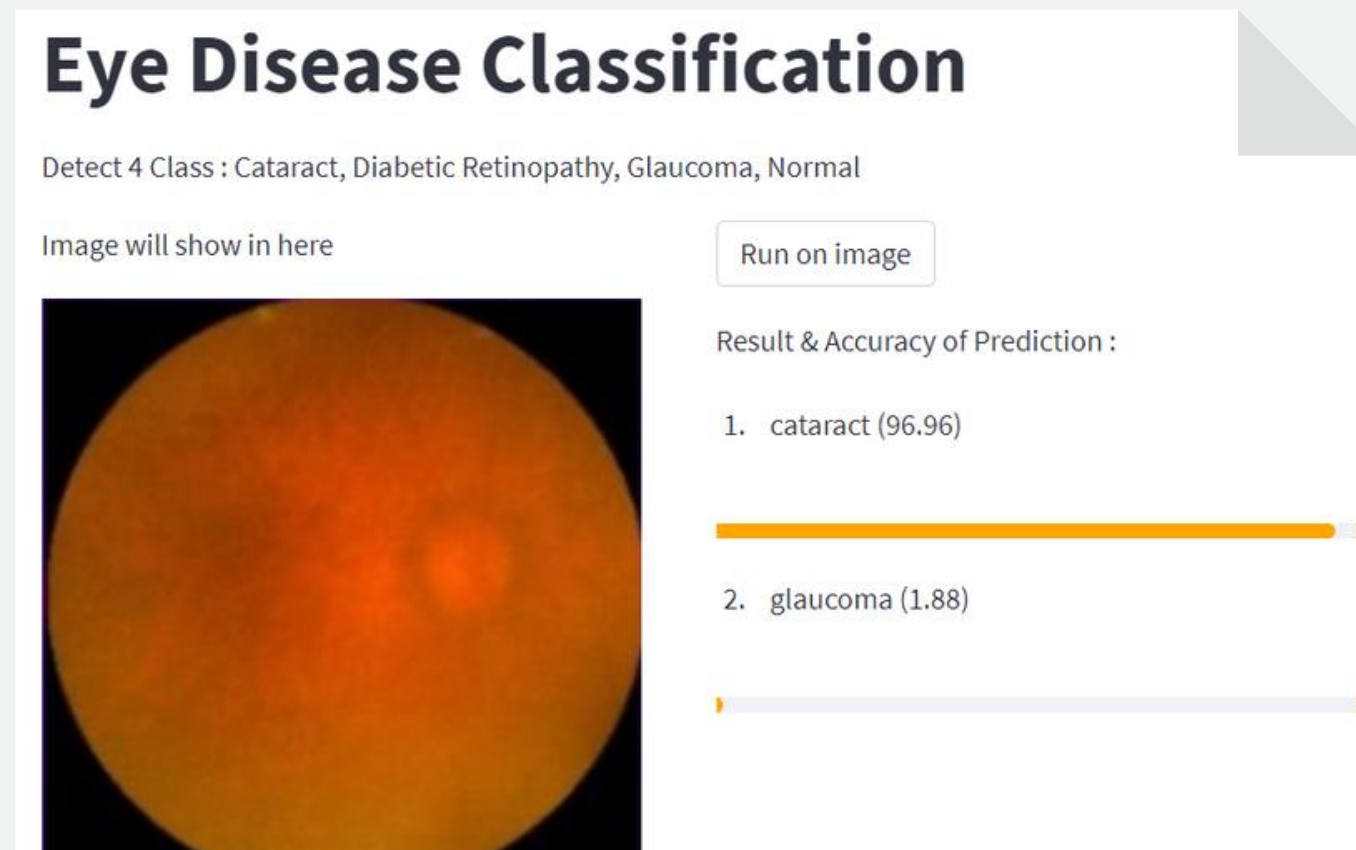


Image classification is a supervised learning problem, in which define a set of target classes (objects to identify in an image), and train a model to recognize them using labeled sample photos.

# ABOUT IMAGE CLASSIFICATION

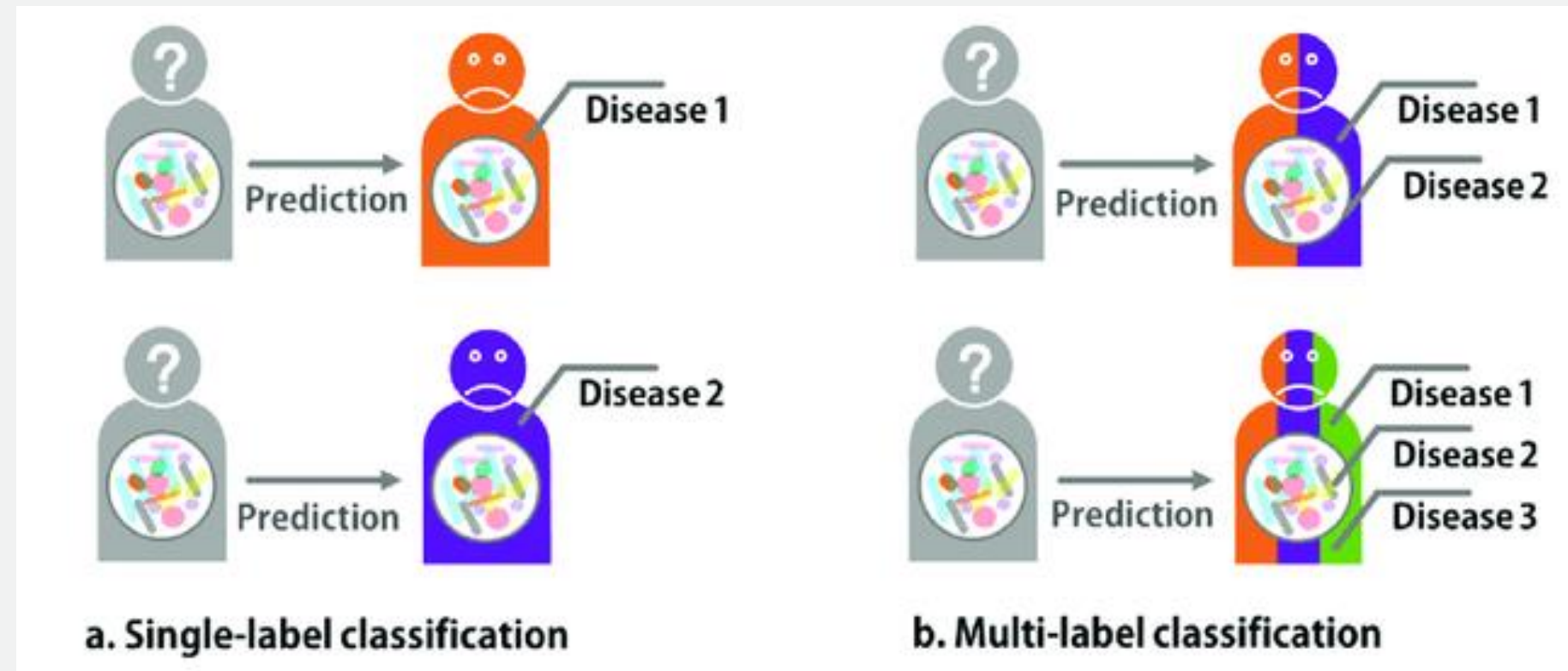


Image classification is divided into two types, namely **Single-label Classification** (only detects one object in the image, that using in this project) & **Multi-label Classification** (can detect more than one object in the image)

# WORKFLOW IN PYTORCH

For Complete Process :

<https://github.com/hardiantots/EyeDiseaseClassification/blob/main/ClassificationEyeDisease.ipynb>

- Prepare the Datasets
- Prepare model to be used
- Train & Evaluation model
- Plot the Accuracy, Train & Validation Loss
- Test result of training model
- Save the model

# Prepare the datasets



kaggle



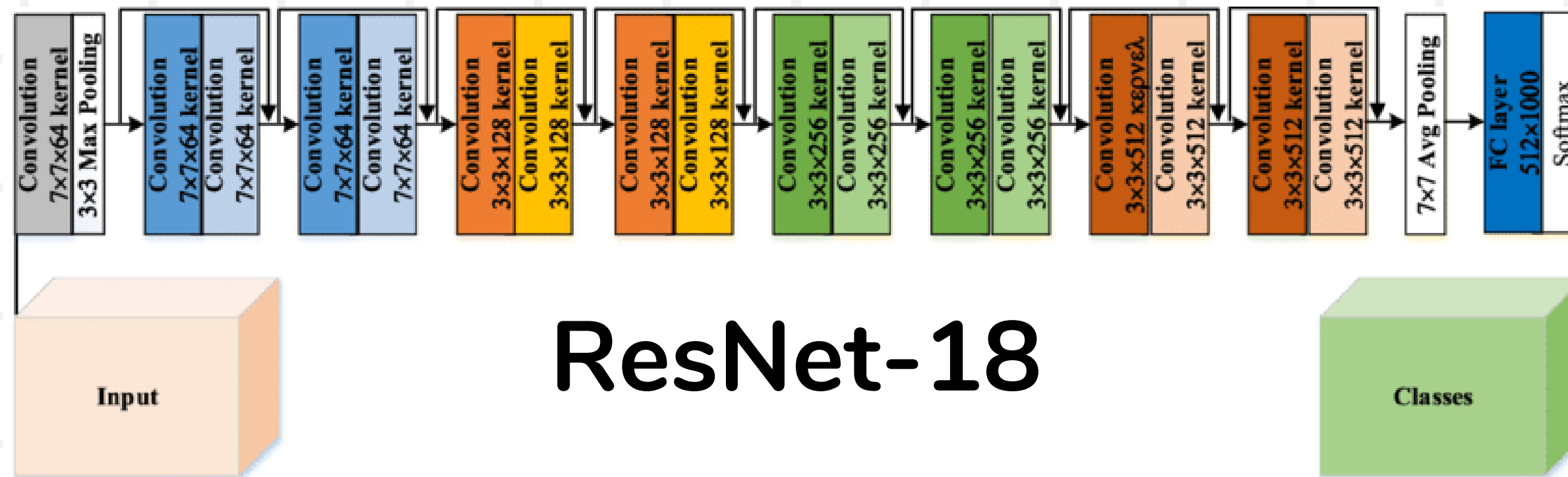
**Fetch dataset from kaggle**

<https://www.kaggle.com/datasets/gunavenkatdoddi/eye-diseases-classification>

**Labelling, processing, divide the Image Dataset into train, val, & loss in Roboflow**

<https://universe.roboflow.com/eye-disease/eyedisease-1/model/1>

# Prepare model to be used



ResNet-18 is a Convolutional Neural Network that is trained on more than a million images from the ImageNet database. There are 18 layers present in its architecture. It is very useful and efficient in image classification. Modifications were made to the output layer which was changed to 4 because there were only 4 classes in the dataset






# Train & Evaluation Model




In the process of train & evaluation model, there are several important things that need to be known :

- **Loss Function** : Cross Entropy Loss
- **Optimizer** : SGD (Stochastic Gradient Descent)
- **Epoch (Iteration)** : 60
- **Learning Rate** : 0.0004





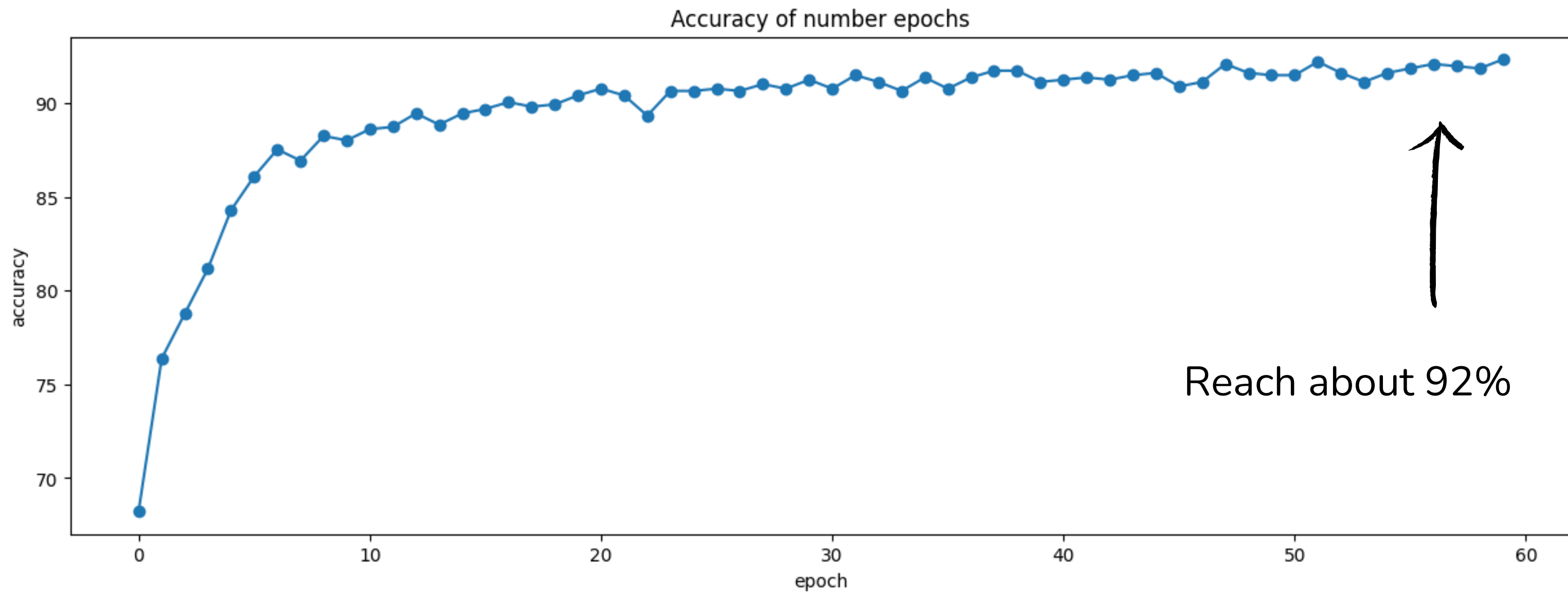
# Plot the Accuracy, Train & Validation Loss



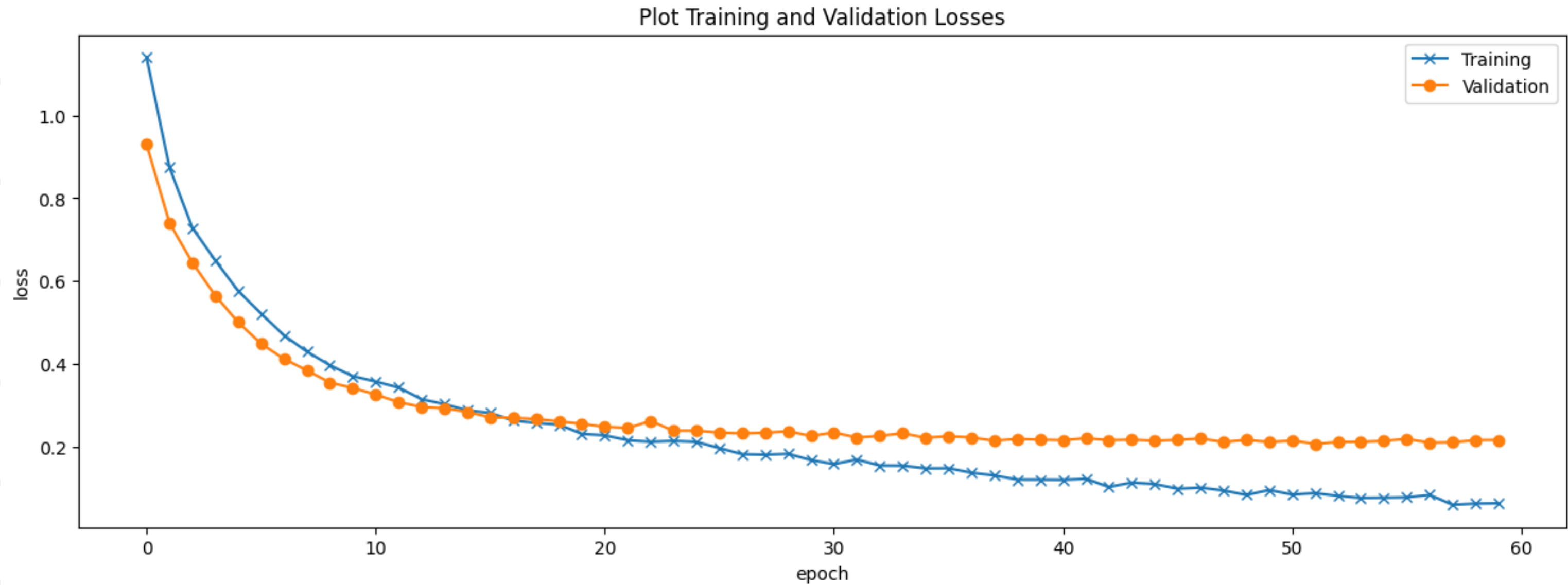
**Final result from training the model :**

- Train Loss = 0.064
- Train Accuracy = 98.562%
- Validation Loss = 0.217
- Validation Accuracy = 92.326%

```
Train Loss : 0.064 | Acc : 98.562%  
Val Loss : 0.217 | Acc : 92.326%
```



**Graph of Accuracy from training model**  
(Using Validation Accuracy value for Accuracy Model)



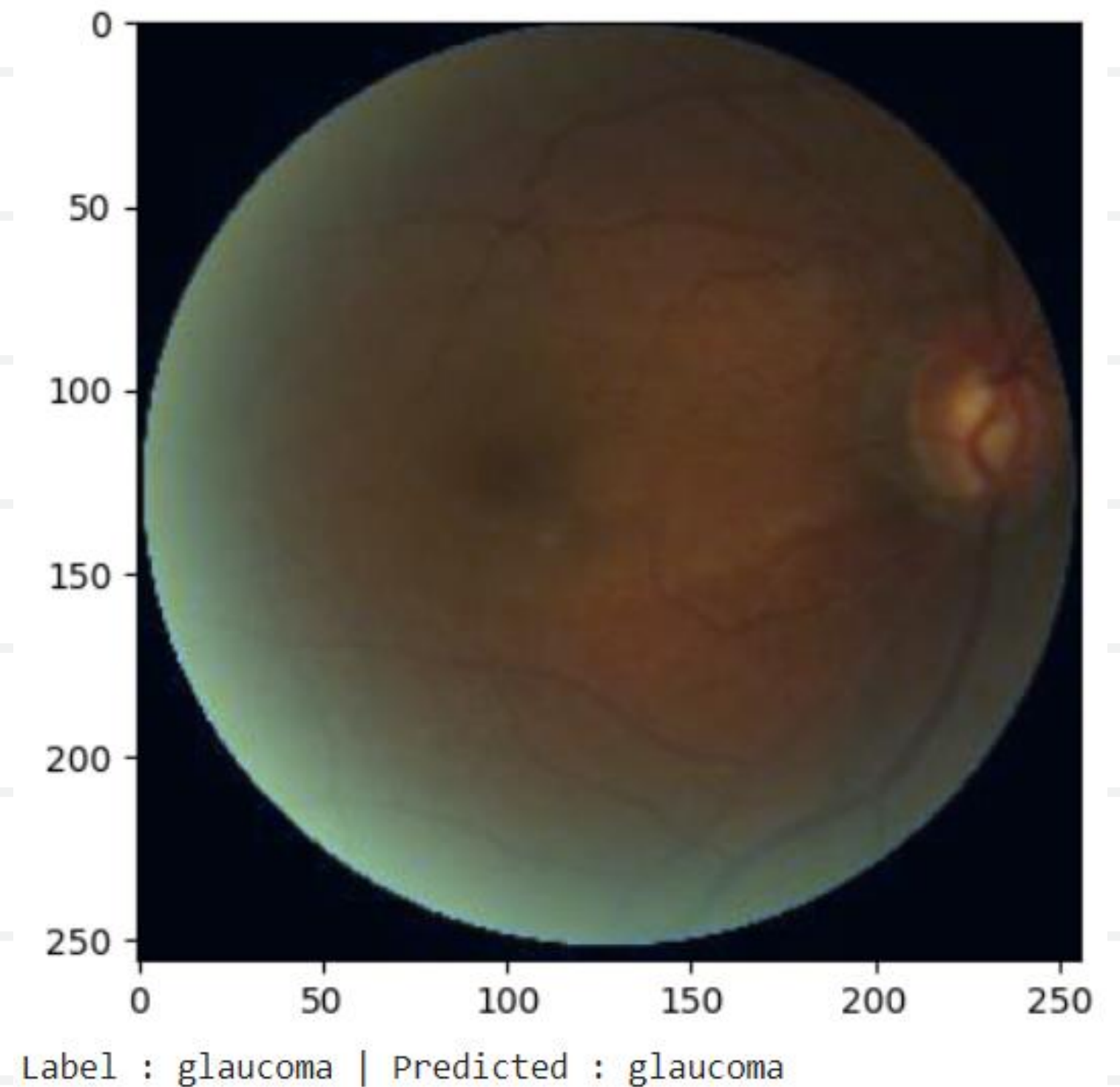
**Graph of Training & Validation Loss from training model**  
(From epoch 20, validation loss did not experience a significant decrease, even tended to stabilize around 0.2)

# Test result of training model



**The model has been able to predict the object in the image correctly**

\*although the accuracy is already high, it does not rule out the possibility of errors in the classification



# Save the model



The **model parameters** that have been **trained**, which **include learnable weights and biases**, are **saved to be loaded** when entering the **deploy process** with **streamlit**

```
from pathlib import Path

MODEL_PATH = Path("models")
MODEL_PATH.mkdir(parents=True, exist_ok=True)

MODEL_NAME = "ClassificationEyeDisease.pth"
MODEL_SAVE_PATH = MODEL_PATH / MODEL_NAME

print(f"Saving model to: {MODEL_SAVE_PATH}")
torch.save(obj=model.state_dict(),
            f=MODEL_SAVE_PATH)
```



# DEPLOYING IN STREAMLIT



Streamlit



## **LOAD MODEL**

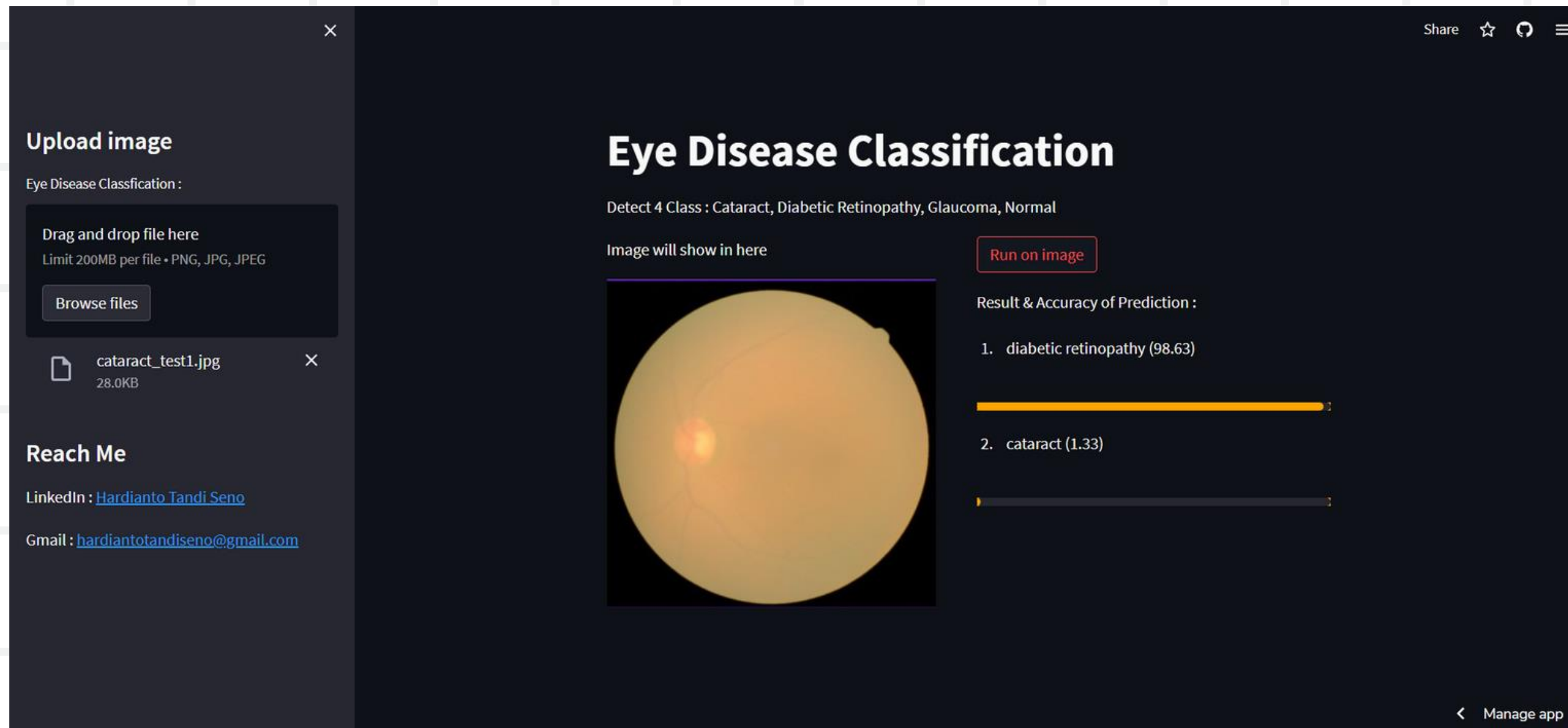
Load the model parameter for make sure that trained model is used to classify Eye Disease on Streamlit

## **DEPLOY ON STREAMLIT**

After load the model, we can code for make classification app via streamlit and then deploy it to streamlit cloud



# DEPLOYMENT RESULT ON STREAMLIT



Link Deployment : <https://eyediseaseclassificationhts.streamlit.app>

When testing the model on Streamlit, it turns out that 1 of the 6 images in the testing folder is misclassified. This can happen maybe because the image can be seen as a model that is more similar to another class than the original class



**CHALLENGES**

**CONCLUSION**



# CHALLENGES

- Looking for the right dataset for the project :
  - **Dataset** is crucial for the Image Classification process. This is because the **diversity of the dataset** obtained will **affect the accuracy of the model** itself and the **ability** to **classify images outside of the dataset provided**
  - **Choosing an inaccurate dataset** from existing sources (e.g. roboflow) will in fact **affect the model training process**. I'm **having a validation loss case that doesn't go down**. I've been **trying to tweak the model structure that I designed myself**, up to **replacing the model with ResNet-18**. Besides that, I **tried to find a good optimizer, epoch and learning rate** for the modeling process, but **the accuracy obtained was difficult to reach 80%** and the **validation loss did not decrease either**.
  - Finally, after searching the dataset on **kaggle** and processing the dataset on **roboflow**, my problem was solved.




# CHALLENGES

- **Deployment Process on Streamlit :**
  - When the modeling problem has been completed, another problem arises, namely an **error** that occurs when **deploying the project to streamlit cloud**.
  - The problem that arises is that **there are several modules that are not available** in Streamlit, so I have to find a way to be able to provide these modules to the Streamlit cloud.
  - Also, there are **some things that are different** when we **run streamlit on local device** and **run streamlit on the cloud provided**, so we need to adjust our **existing code** according to where streamlit is run.



# CONCLUSION

From the various stages that I have done, there are several conclusions that I can get :

- Image classification is **very dependent on the diversity and amount in the image dataset**, the less diversity & number of datasets, it **will be difficult** to provide **accurate classification predictions** if **images are used outside of the dataset provided**.
  - The **model used** will also **affect the accuracy and loss results**.
  - Even though **the accuracy** during the model training process is **high**, it is **not certain** that it can **classify all the images given correctly** even though the **chance of error** is also small.
- 

# THANK YOU



Hardianto Tandi Seno



hardiantotandiseno@gmail.com

Hardianto Tandi Seno

Future ML Engineer