# PROJECT REPORT

## INTRODUCTION

* 1. **Project Overview**

Diabetic Retinopathy (DR) is a common complication of diabetes mellitus, which

causes lesions on the retina that affect vision. If it is not detected early, it can lead to blindness. Unfortunately, DR is not a reversible process, and treatment only sustains vision. DR early detection and treatment can signiﬁcantly reduce the risk of vision loss. The manual diagnosis process of DR retina fundus images by ophthalmologists is time, effort and cost-consuming and prone to misdiagnosis unlike computer-aided diagnosis systems.

Transfer learning has become one of the most common techniques that has achieved better performance in many areas, especially in medical image analysis and classiﬁcation. We used Transfer Learning techniques like Inception V3,Resnet50,Xception V3 that are more widely used as a transfer learning method in medical image analysis and they are highly effective.

* 1. **Purpose**

Early detection (‘screening’) and timely treatment have been shown to prevent visual loss and blindness in patients with retinal complications of diabetes.In the next decade, projections for the United States are that the average age will increase, the number of people with diabetes in each age category will increase, and there will be an undersupply of qualiﬁed eye care providers, at least in the near-term. This “perfect storm” of healthcare trends will challenge the public health capacity to care for both patients with DR and people with diabetes at risk for this complication. If the previous scenario plays out, it will be necessary to either screen (perform early detection on) large numbers of people with diabetes for DR, ration access to eyecare, or both.

1. **LITERATURE SURVEY**
   1. **Existing problem**

Diabetic retinopathy can cause abnormal blood vessels to grow out of the retina and block ﬂuid from draining out of the eye. This causes a type of glaucoma (a group of eye diseases that can cause vision loss and blindness).

* 1. **References**

[1]. M. Chetoui, M. A. Akhlouﬁ and M. Kardouchi, "Diabetic Retinopathy Detection Using Machine Learning and Texture Features", 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), pp. 1-4, 2018**.**

[2]. Kangrok Oh, Hae Min Kang, Dawoon Leem, Hyungyu Lee, Kyoung Yul Seo, Sangchul Yoon, "Early detection of diabetic retinopathy based on deep learning and ultra-wide-ﬁeld fundus images", Scientiﬁc Reports 11, Article No:1897 1-9,2021.

[3]. Sheikh Muhammad Saiful Islam, Md Mahedi Hasan, Sohaib Abdullah, "Deep learning based early detection and grading of diabetic retinopathy using retinal fundus images" , arXiv preprint arXiv:1812.10595, 2018.

[4]. E. V. Carrera, A. González and R. Carrera, "Automated detection of diabetic retinopathy using SVM", 2017 IEEE XXIV International Conference on Electronics Electrical Engineering and Computing (INTERCON), pp. 1-4, 2017.

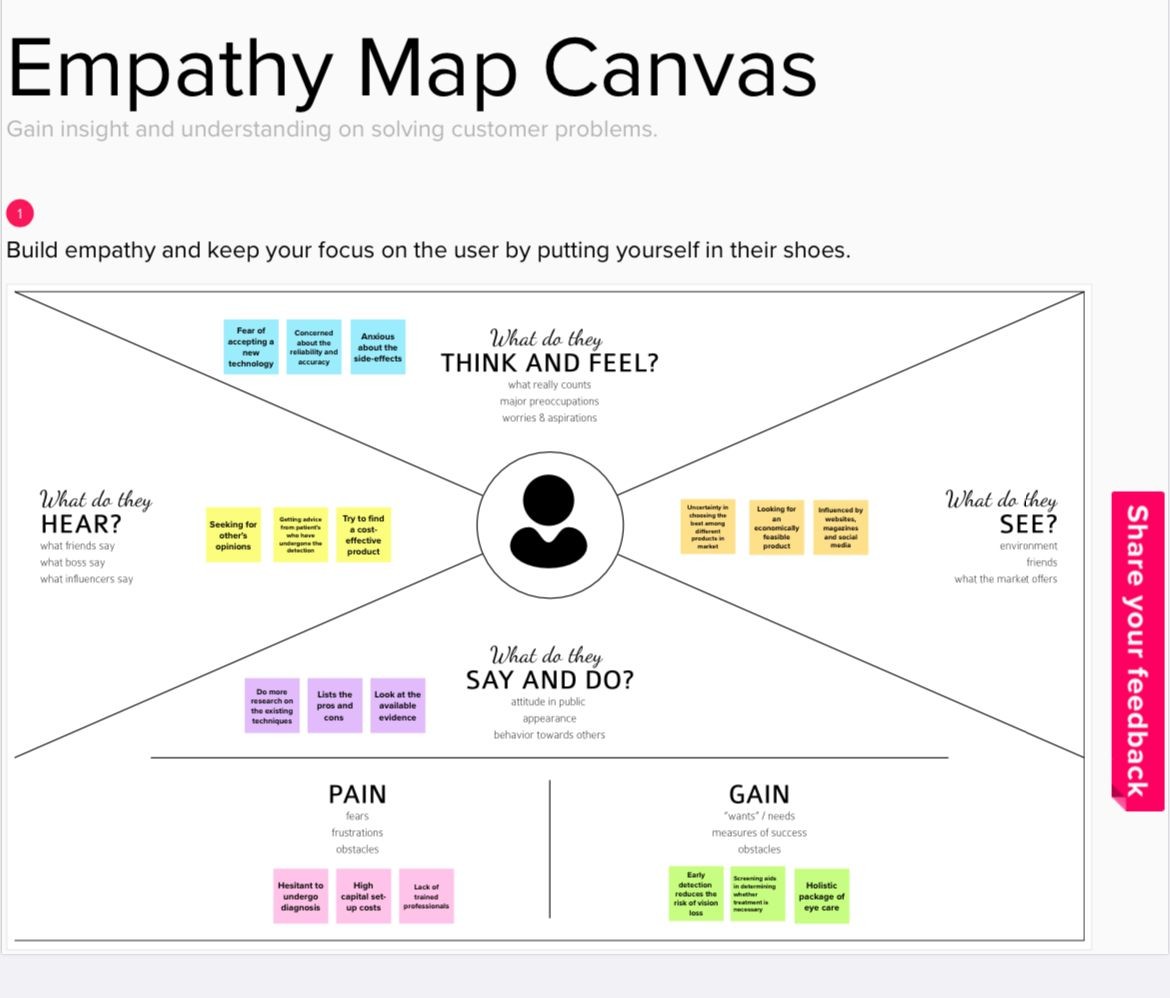
* 1. **Problem Statement Deﬁnition**

Diabetic retinopathy is caused by damage to the blood vessels in the tissue at the back

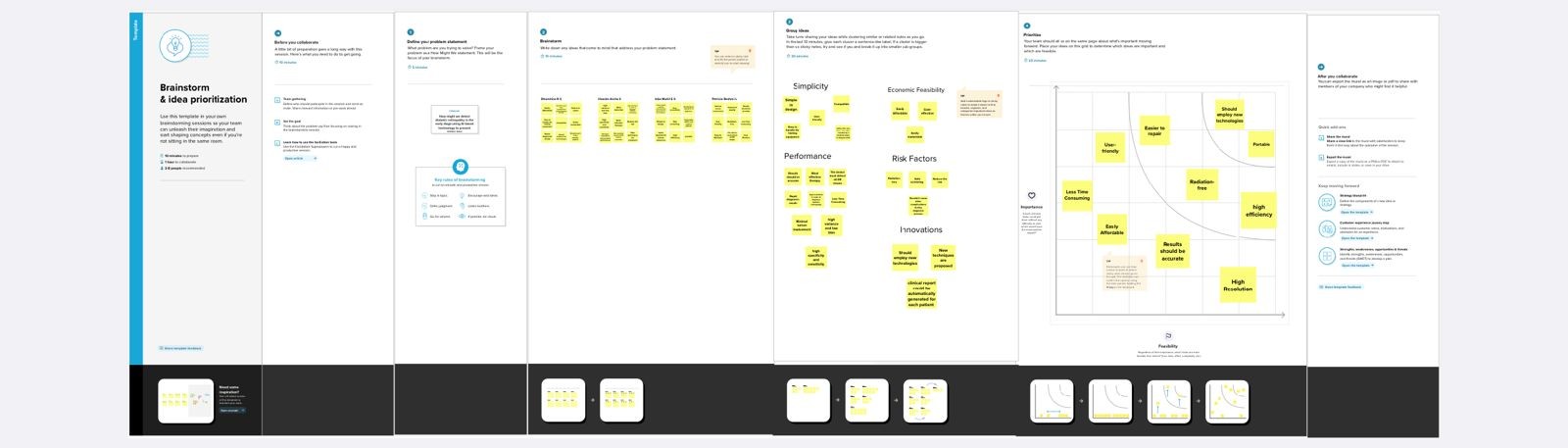
of the eye (retina). Poorly controlled blood sugar is a risk factor. Early symptoms include ﬂoaters, blurriness, dark areas of vision and diﬃculty perceiving colours.If not detected and treated properly, it can lead to severe complications and loss of vision.Our project aims to

develop a better screening using techniques like Inception V3, Resnet50, Xception V3 that are more widely used as a transfer learning method in medical image analysis and they are highly effective.

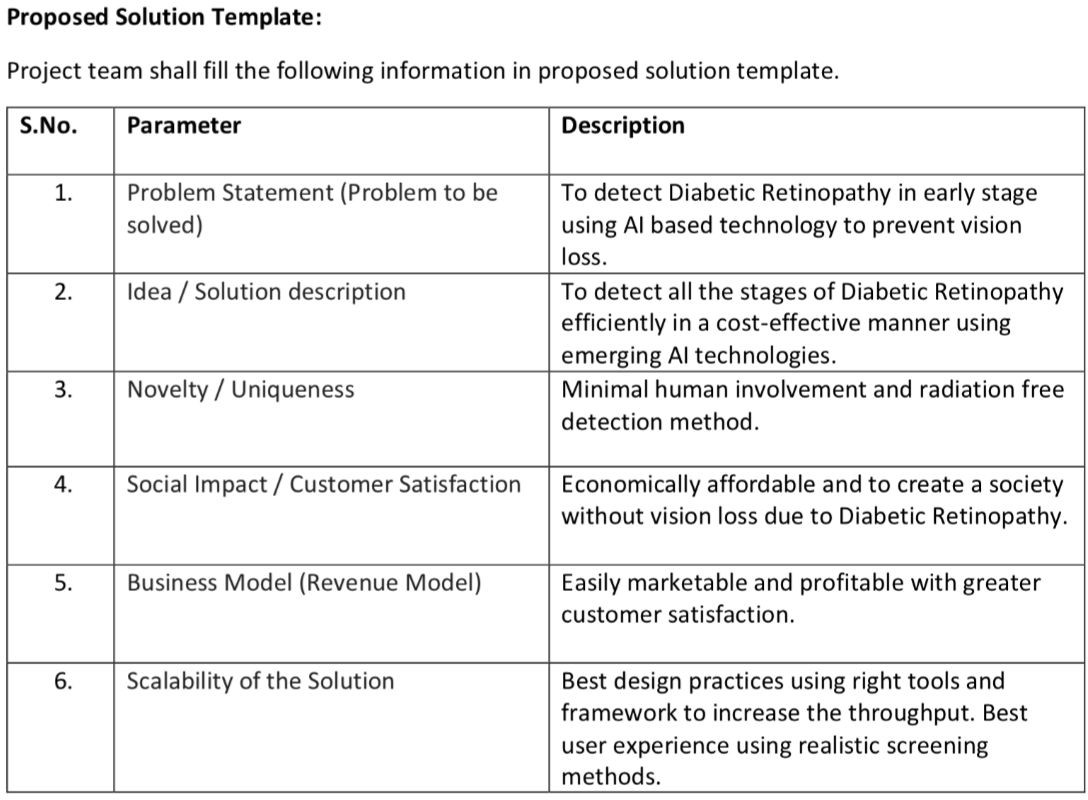
1. **IDEATION & PROPOSED SOLUTION**
   1. **Empathy Map Canvas**



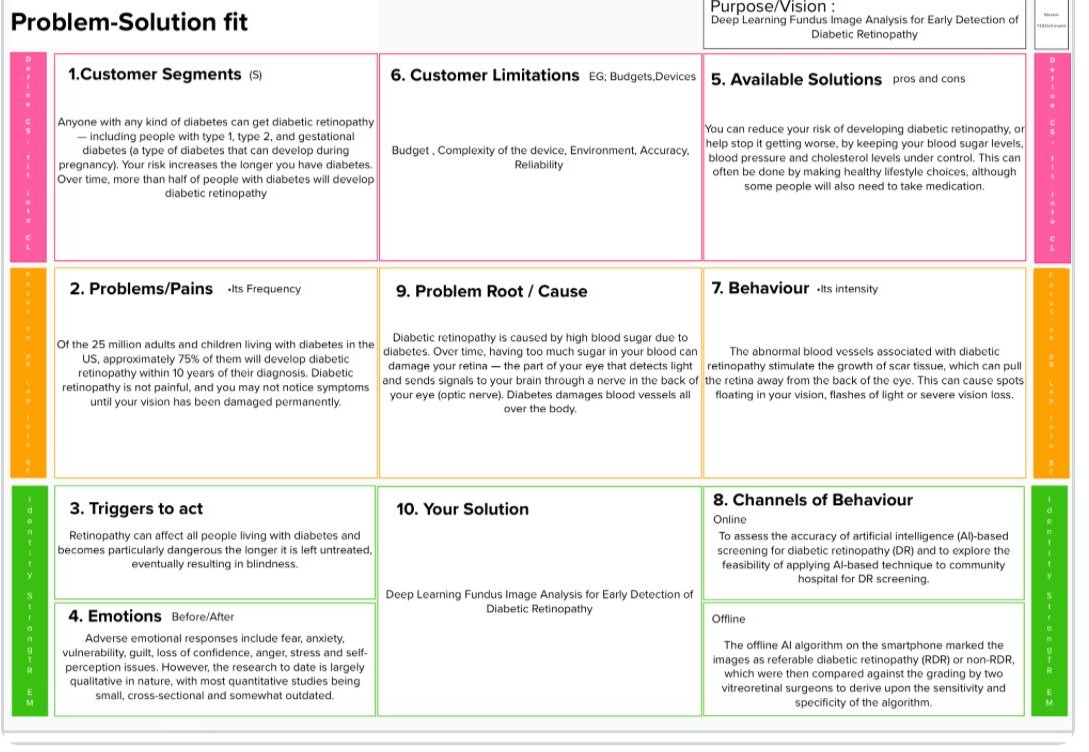
* 1. **Ideation & Brainstorming**



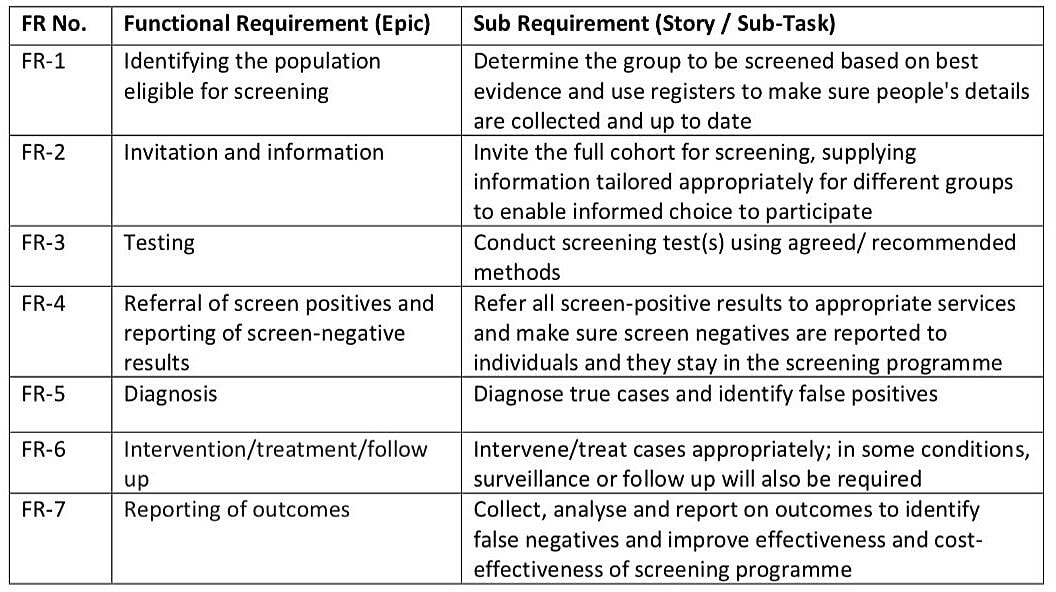
* 1. **Proposed Solution**



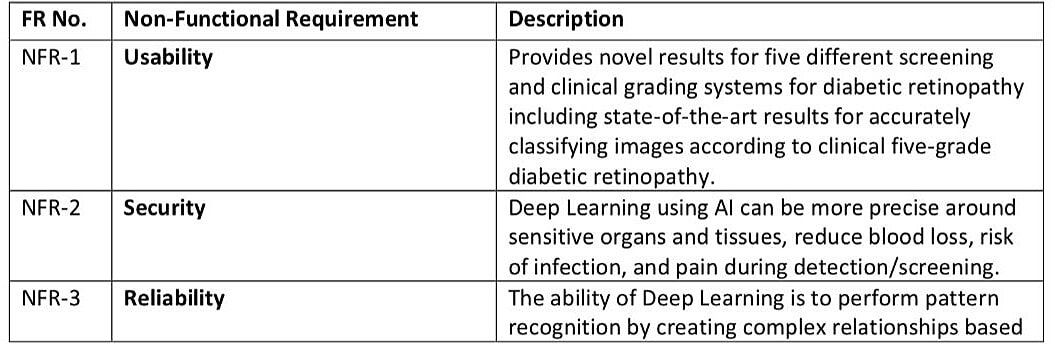
* 1. **Problem Solution ﬁt**

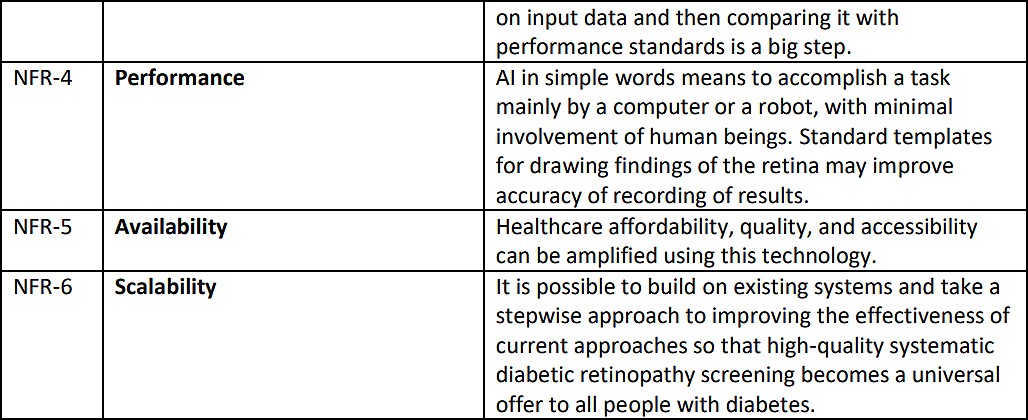


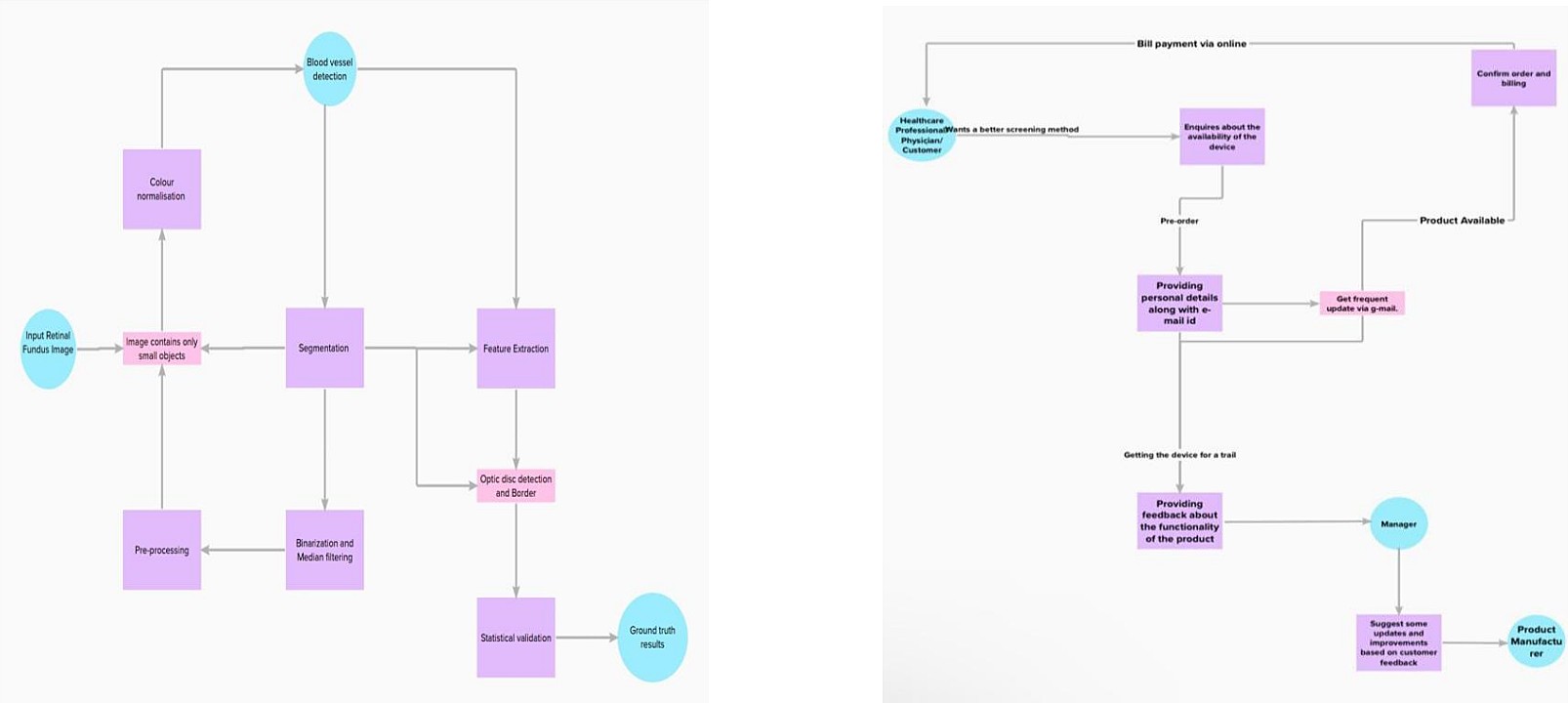
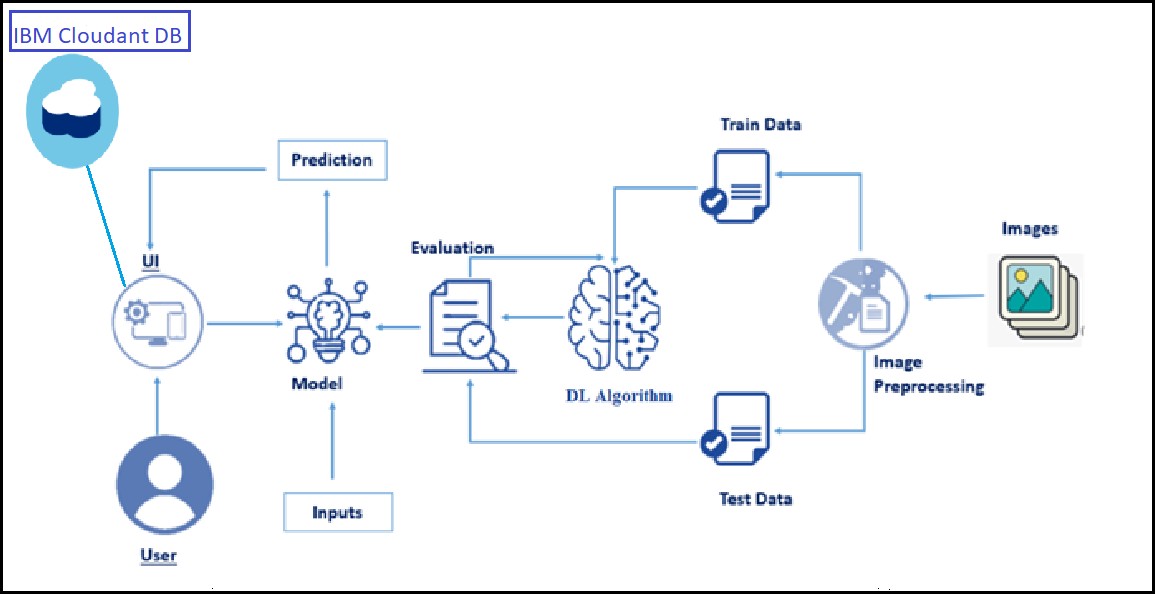
1. **REQUIREMENT ANALYSIS**
   1. **Functional requirement**



* 1. **Non-Functional requirements**

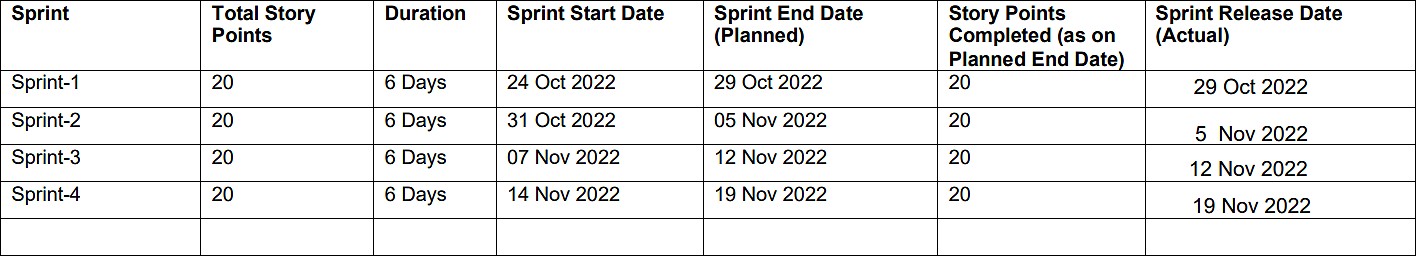




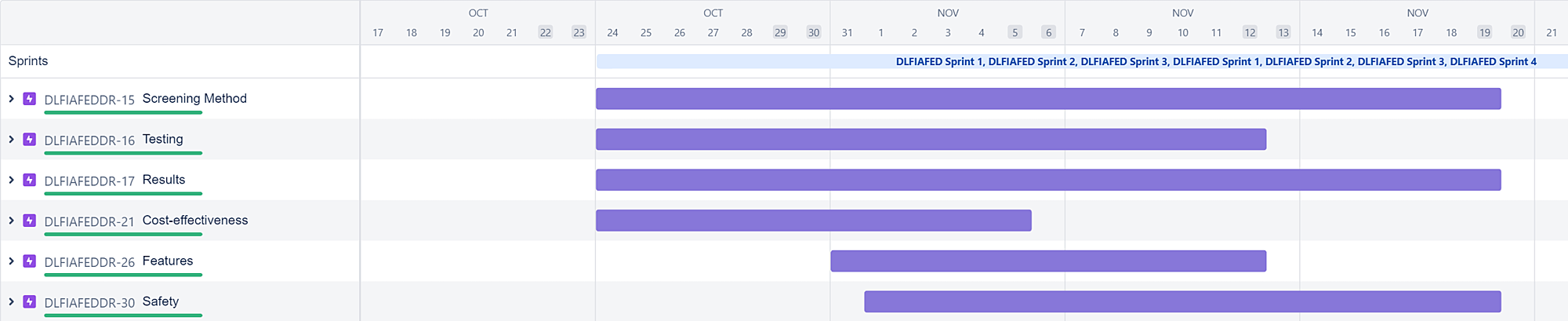
1. **PROJECT DESIGN**
   1. **Data Flow Diagrams**
   2. **Solution & Technical Architecture**
2. **PROJECT PLANNING & SCHEDULING**
   1. **Sprint Planning & Estimation**

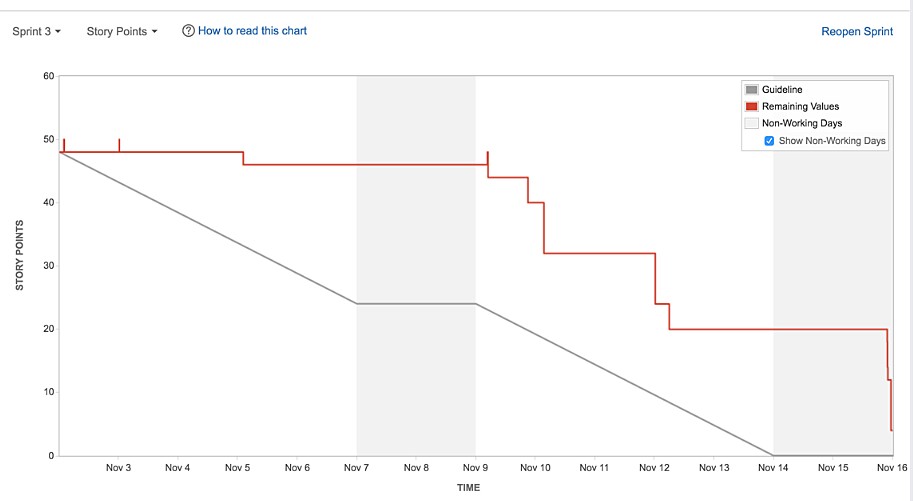
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Data Collection | Task-1 | In Deep Learning Model, It can be spilt into Testing and Training set. | 4 | Medium | M. Keerthna  Chiraj Gupta |
| Sprint-1 | Data Pre-processing | Task-2 | Import the required data for pre-processing.  Application of the image data generator to the train and test set. | 7 | Low | Varun Bhardwaj  Johanna Smriti  Chirag Gupta |
| Sprint-1 | Build Homepage | USN-1 | Homepage give the brief description to the user. | 4 | Medium | Johanna Smriti  Varun Bhardwaj |
| Sprint-2 | Create Registration page | USN-2 | In this page, User will able to register for the application. | 2 | Low | M. Keerthna  Johanna Smriti |
| Sprint-2 | Develop the model | Task-3 | Build a model | 3 | High | M. Keerthna  Chirag Gupta |
| Sprint-3 | Create Service Instance | Task-4 | Configure the location of resource such as web server and cloud storage for an application. | 7 | High | Varun Bhardwaj  Johanna Smriti  Chirag Gupta |
| Sprint-3 | Creating Database | Task-5 | IBM Cloud, offered the required credentials to access the services and the database accessed by the users. | 6 | High | Varun Bhardwaj  M. Keerthna |
| Sprint-3 | Creating Tables in Database | Task-6 | Structure the required tables with necessary attributes in cloud DB. | 4 | Medium | Chirag Gupta  Johanna Smriti |
| Sprint-4 | Building login page | USN-3 | User will be able to login by using the credentials. | 3 | Low | M. Keerthna  Johanna Smriti |
| Sprint-4 | Create Image uploading page | Task-7 | Upload the test image | 2 | Low | Varun Bharadwaj  Chirag Gupta |
| Sprint-4 | Building Prediction page | USN-4 | User able to receive the diagnosis on their diabetic retinopathy. | 2 | Medium | M. Keerthna  Johanna Smriti |
| Sprint-4 | Building logout page | USN-5 | User will be able to logout their account in this  Page. | 2 | Medium | Varun Bharadwaj  Chirag Gupta |
| Sprint-4 | Build Python code | Task-8 | The Necessary modules should be initialize and the libraries should be imported. | 1 | Medium | M. Keerthna  Johanna Smriti |
| Sprint-4 |  | Task-9 | Use the database using initiating client and rendering HTML page. | 2 | Medium | Varun Bharadwaj  Chirag Gupta |
| Sprint-4 |  | Task-10 | Configuring the registration, login pages and evaluating the credentials. | 2 | Medium | Johanna Smriti  Varun Bhardwaj |
| Sprint-4 |  | Task-11 | The model prediction will be showcased on UI. | 1 | High | M. Keerthna  Johanna Smriti  Chirag Gupta |
| Sprint-4 | Run the Application | Task-12 | Run to check the application. | 2 | High | Varun Bharadwaj  Chirag Gupta |
| Sprint-4 |  | Task-13 | Upload image in the homepage to predict the diabetic retinopathy. | 5 | High | M. Keerthna  Johanna Srmiti |
| Sprint-4 | Train model on IBM | Task-14 | Train the model on IBM and integrate it with the flask application. | 3 | High | M. Keerthna  Johanna Smriti  Chirag Gupta  Varun Bharadwaj |

* 1. **Sprint Delivery Schedule**

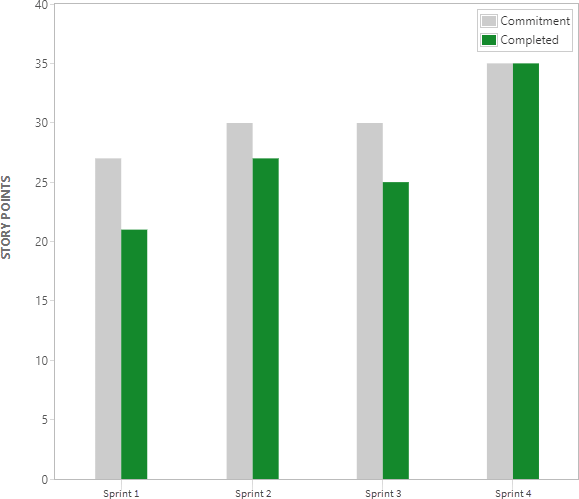


* 1. **Reports from JIRA Roadmap**



**Sprint Burndown chart**

**Velocity report**



1. **CODING & SOLUTIONING**
   1. **Feature 1**

**DATA COLLECTION**

! pip install kaggle

! mkdir **~/.**kaggle

! cp kaggle**.**json **~/.**kaggle**/**

! chmod 600 **~/.**kaggle**/**kaggle**.**json

**Downloading the dataset**

! kaggle datasets download arbethi**/**diabetic**-**retinopathy**-**level**-**detection

### Unzipping the dataset

! unzip diabetic**-**retinopathy**-**level**-**detection**.**zip

### Creating Training And Testing Path

imageSize **=** [299,299]

trainPath **=** r"/content/preprocessed dataset/preprocessed dataset/training" testPath **=** r"/content/preprocessed dataset/preprocessed dataset/testing"

### DATA PRE-PROCESSING

**Importing The Libraries**

**from** tensorﬂow.keras.layers **import** Dense,Flatten,Input

**from** tensorﬂow.keras.models **import** Model

**from** tensorﬂow.keras.preprocessing **import** image

**from** tensorﬂow.keras.preprocessing.image **import** ImageDataGenerator,load\_img **from** tensorﬂow.keras.applications.xception **import** Xception,preprocess\_input **from** glob **import** glob

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

### Conﬁguring ImageDataGenerator Class

train\_datagen **=** ImageDataGenerator (rescale**=**1.**/**255,shear\_range**=** 0.2, zoom\_range

**=** 0.2,horizontal\_ﬂip **= True**)

test\_datagen **=** ImageDataGenerator (rescale **=** 1.**/**255)

### Applying ImageDataGenerator Functionality To Train Set And Test Set

training\_set **=** train\_datagen**.**ﬂow\_from\_directory('/content/preprocessed dataset/preprocessed dataset/training',target\_size **=** (299,299),batch\_size **=**32, class\_mode **=** 'categorical')

test\_set **=** test\_datagen**.**ﬂow\_from\_directory('/content/preprocessed dataset/preprocessed dataset/testing',target\_size **=**(299,299),batch\_size **=** 32,class\_mode **=** 'categorical')

### MODEL BUILDING

**Pre-Training CNN Model As A Feature Extractor**

xception **=** Xception(input\_shape **=** imageSize **+**

[3],weights**=**'imagenet',include\_top = **False**)

**for** layer **in** xception**.**layers: layer**.**trainable **= False**

x **=** Flatten()(xception**.**output)

### Adding Dense Layers

prediction **=** Dense( 5,activation **=**'softmax')(x)

model **=** Model(inputs**=**xception**.**input,outputs**=**prediction) model**.**summary()

Model : "model"

**Conﬁguring The Learning Process**

model**.**compile(

loss **=** 'categorical\_crossentropy', optimizer **=** 'adam',

metrics **=**['accuracy']

)

### Training The Model

*# ﬁt the model*

r **=** model**.**ﬁt\_generator( training\_set,

validation\_data**=**test\_set, epochs**=**30,

steps\_per\_epoch**=**len (training\_set)**//**32, validation\_steps**=**len(test\_set)**//**32

)

### Saving The Model

model**.**save("diabetic.h5")

## Output:

Found 3662 images belonging to 5 classes. Found 734 images belonging to 5 classes.

Downloading data from https://storage.googleapis.com/tensorﬂow/keras- applications/xception/xception\_weights\_tf\_dim\_ordering\_tf\_kernels\_notop.h5 83683744/83683744 [==============================] - 1s 0us/step

Model: "model"

Layer (type) Output Shape Param # Connected to

==============================================================================

====================

|  |  |  |  |
| --- | --- | --- | --- |
| input\_1 (InputLayer) | [(None, 299, 299, 3  )] | 0 | [] |
| block1\_conv1 (Conv2D) ['input\_1[0][0]'] | (None, 149, 149, 32  ) | 864 |  |
| block1\_conv1\_bn (BatchNormaliz ['block1\_conv1[0][0]'] | (None, 149, 149, 32 | 128 |  |
| ation) | ) |  | |
| block1\_conv1\_act (Activation) ['block1\_conv1\_bn[0][0]'] | (None, 149, 149, 32  ) | 0 | |
| block1\_conv2 (Conv2D) ['block1\_conv1\_act[0][0]'] | (None, 147, 147, 64  ) | 18432 | |
| block1\_conv2\_bn (BatchNormaliz | (None, 147, 147, 64 | 256 | |
| ['block1\_conv2[0][0]'] |  |  |  |
| ation) | ) |  | |
| block1\_conv2\_act (Activation) ['block1\_conv2\_bn[0][0]'] | (None, 147, 147, 64  ) | 0 | |
| block2\_sepconv1 (SeparableConv | (None, 147, 147, 12 | 8768 | |
| ['block1\_conv2\_act[0][0]'] |  |  |  |
| 2D) | 8) | | |

block2\_sepconv1\_bn (BatchNorma (None, 147, 147, 12 512 ['block2\_sepconv1[0][0]']

lization) 8)

block2\_sepconv2\_act (Activatio (None, 147, 147, 12 0 ['block2\_sepconv1\_bn[0][0]']

n) 8)

block2\_sepconv2 (SeparableConv (None, 147, 147, 12 17536 ['block2\_sepconv2\_act[0][0]']

2D) 8)

block2\_sepconv2\_bn (BatchNorma (None, 147, 147, 12 512 ['block2\_sepconv2[0][0]']

lization) 8)

conv2d (Conv2D) (None, 74, 74, 128) 8192 ['block1\_conv2\_act[0][0]']

block2\_pool (MaxPooling2D) (None, 74, 74, 128) 0 ['block2\_sepconv2\_bn[0][0]']

batch\_normalization (BatchNorm (None, 74, 74, 128) 512 ['conv2d[0][0]']

alization)

add (Add) (None, 74, 74, 128) 0 ['block2\_pool[0][0]',

'batch\_normalization[0][0]']

['add[0][0]']

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| block3\_sepconv1\_act (Activatio | (None, | 74, | 74, | 128) | 0 |
| n)  block3\_sepconv1 (SeparableConv | (None, | 74, | 74, | 256) | 33920 |

['block3\_sepconv1\_act[0][0]'] 2D)

block3\_sepconv1\_bn (BatchNorma (None, 74, 74, 256) 1024 ['block3\_sepconv1[0][0]']

lization)

block3\_sepconv2\_act (Activatio (None, 74, 74, 256) 0 ['block3\_sepconv1\_bn[0][0]']

n)

block3\_sepconv2 (SeparableConv (None, 74, 74, 256) 67840 ['block3\_sepconv2\_act[0][0]']

2D)

block3\_sepconv2\_bn (BatchNorma (None, 74, 74, 256) 1024 ['block3\_sepconv2[0][0]']

lization)

conv2d\_1 (Conv2D) (None, 37, 37, 256) 32768 ['add[0][0]']

block3\_pool (MaxPooling2D) (None, 37, 37, 256) 0 ['block3\_sepconv2\_bn[0][0]']

batch\_normalization\_1 (BatchNo (None, 37, 37, 256) 1024 ['conv2d\_1[0][0]']

rmalization)

add\_1 (Add) (None, 37, 37, 256) 0 ['block3\_pool[0][0]',

'batch\_normalization\_1[0][0]']

block4\_sepconv1\_act (Activatio (None, 37, 37, 256) 0 ['add\_1[0][0]']

n)

block4\_sepconv1 (SeparableConv (None, 37, 37, 728) 188672 ['block4\_sepconv1\_act[0][0]']

2D)

block4\_sepconv1\_bn (BatchNorma (None, 37, 37, 728) 2912 ['block4\_sepconv1[0][0]']

lization)

block4\_sepconv2\_act (Activatio (None, 37, 37, 728) 0 ['block4\_sepconv1\_bn[0][0]']

n)

block4\_sepconv2 (SeparableConv (None, 37, 37, 728) 536536 ['block4\_sepconv2\_act[0][0]']

2D)

block4\_sepconv2\_bn (BatchNorma (None, 37, 37, 728) 2912 ['block4\_sepconv2[0][0]']

lization)

conv2d\_2 (Conv2D) (None, 19, 19, 728) 186368 ['add\_1[0][0]']

block4\_pool (MaxPooling2D) (None, 19, 19, 728) 0

['block4\_sepconv2\_bn[0][0]']

batch\_normalization\_2 (BatchNo (None, 19, 19, 728) 2912 ['conv2d\_2[0][0]']

rmalization)

add\_2 (Add) (None, 19, 19, 728) 0 ['block4\_pool[0][0]',

'batch\_normalization\_2[0][0]']

block5\_sepconv1\_act (Activatio (None, 19, 19, 728) 0 ['add\_2[0][0]']

n)

block5\_sepconv1 (SeparableConv (None, 19, 19, 728) 536536 ['block5\_sepconv1\_act[0][0]']

2D)

block5\_sepconv1\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block5\_sepconv1[0][0]']

lization)

block5\_sepconv2\_act (Activatio (None, 19, 19, 728) 0 ['block5\_sepconv1\_bn[0][0]']

n)

block5\_sepconv2 (SeparableConv (None, 19, 19, 728) 536536 ['block5\_sepconv2\_act[0][0]']

2D)

block5\_sepconv2\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block5\_sepconv2[0][0]']

lization)

block5\_sepconv3\_act (Activatio (None, 19, 19, 728) 0 ['block5\_sepconv2\_bn[0][0]']

n)

block5\_sepconv3 (SeparableConv (None, 19, 19, 728) 536536 ['block5\_sepconv3\_act[0][0]']

2D)

block5\_sepconv3\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block5\_sepconv3[0][0]']

lization)

add\_3 (Add) (None, 19, 19, 728) 0

['block5\_sepconv3\_bn[0][0]', 'add\_2[0][0]']

block6\_sepconv1\_act (Activatio (None, 19, 19, 728) 0 ['add\_3[0][0]']

n)

block6\_sepconv1 (SeparableConv (None, 19, 19, 728) 536536 ['block6\_sepconv1\_act[0][0]']

2D)

block6\_sepconv1\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block6\_sepconv1[0][0]']

lization)

block6\_sepconv2\_act (Activatio (None, 19, 19, 728) 0 ['block6\_sepconv1\_bn[0][0]']

n)

block6\_sepconv2 (SeparableConv (None, 19, 19, 728) 536536 ['block6\_sepconv2\_act[0][0]']

2D)

block6\_sepconv2\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block6\_sepconv2[0][0]']

lization)

block6\_sepconv3\_act (Activatio (None, 19, 19, 728) 0 ['block6\_sepconv2\_bn[0][0]']

n)

block6\_sepconv3 (SeparableConv (None, 19, 19, 728) 536536 ['block6\_sepconv3\_act[0][0]']

2D)

block6\_sepconv3\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block6\_sepconv3[0][0]']

lization)

add\_4 (Add) (None, 19, 19, 728) 0 ['block6\_sepconv3\_bn[0][0]',

'add\_3[0][0]']

block7\_sepconv1\_act (Activatio (None, 19, 19, 728) 0 ['add\_4[0][0]']

n)

block7\_sepconv1 (SeparableConv (None, 19, 19, 728) 536536 ['block7\_sepconv1\_act[0][0]']

2D)

block7\_sepconv1\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block7\_sepconv1[0][0]']

lization)

block7\_sepconv2\_act (Activatio (None, 19, 19, 728) 0 ['block7\_sepconv1\_bn[0][0]']

n)

block7\_sepconv2 (SeparableConv (None, 19, 19, 728) 536536 ['block7\_sepconv2\_act[0][0]']

2D)

block7\_sepconv2\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block7\_sepconv2[0][0]']

lization)

block7\_sepconv3\_act (Activatio (None, 19, 19, 728) 0 ['block7\_sepconv2\_bn[0][0]']

n)

block7\_sepconv3 (SeparableConv (None, 19, 19, 728) 536536 ['block7\_sepconv3\_act[0][0]']

2D)

block7\_sepconv3\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block7\_sepconv3[0][0]']

lization)

add\_5 (Add) (None, 19, 19, 728) 0 ['block7\_sepconv3\_bn[0][0]',

'add\_4[0][0]']

block8\_sepconv1\_act (Activatio (None, 19, 19, 728) 0 ['add\_5[0][0]']

n)

block8\_sepconv1 (SeparableConv (None, 19, 19, 728) 536536 ['block8\_sepconv1\_act[0][0]']

2D)

block8\_sepconv1\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block8\_sepconv1[0][0]']

lization)

block8\_sepconv2\_act (Activatio (None, 19, 19, 728) 0 ['block8\_sepconv1\_bn[0][0]']

n)

block8\_sepconv2 (SeparableConv (None, 19, 19, 728) 536536 ['block8\_sepconv2\_act[0][0]']

2D)

block8\_sepconv2\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block8\_sepconv2[0][0]']

lization)

block8\_sepconv3\_act (Activatio (None, 19, 19, 728) 0 ['block8\_sepconv2\_bn[0][0]']

n)

block8\_sepconv3 (SeparableConv (None, 19, 19, 728) 536536 ['block8\_sepconv3\_act[0][0]']

2D)

block8\_sepconv3\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block8\_sepconv3[0][0]']

lization)

add\_6 (Add) (None, 19, 19, 728) 0 ['block8\_sepconv3\_bn[0][0]',

'add\_5[0][0]']

block9\_sepconv1\_act (Activatio (None, 19, 19, 728) 0 ['add\_6[0][0]']

n)

block9\_sepconv1 (SeparableConv (None, 19, 19, 728) 536536 ['block9\_sepconv1\_act[0][0]']

2D)

block9\_sepconv1\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block9\_sepconv1[0][0]']

lization)

block9\_sepconv2\_act (Activatio (None, 19, 19, 728) 0 ['block9\_sepconv1\_bn[0][0]']

n)

block9\_sepconv2 (SeparableConv (None, 19, 19, 728) 536536

['block9\_sepconv2\_act[0][0]'] 2D)

block9\_sepconv2\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block9\_sepconv2[0][0]']

lization)

block9\_sepconv3\_act (Activatio (None, 19, 19, 728) 0 ['block9\_sepconv2\_bn[0][0]']

n)

block9\_sepconv3 (SeparableConv (None, 19, 19, 728) 536536 ['block9\_sepconv3\_act[0][0]']

2D)

block9\_sepconv3\_bn (BatchNorma (None, 19, 19, 728) 2912 ['block9\_sepconv3[0][0]']

lization)

add\_7 (Add) (None, 19, 19, 728) 0 ['block9\_sepconv3\_bn[0][0]',

'add\_6[0][0]']

block10\_sepconv1\_act (Activati (None, 19, 19, 728) 0 ['add\_7[0][0]']

on)

block10\_sepconv1 (SeparableCon (None, 19, 19, 728) 536536 ['block10\_sepconv1\_act[0][0]']

v2D)

block10\_sepconv1\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block10\_sepconv1[0][0]']

alization)

block10\_sepconv2\_act (Activati (None, 19, 19, 728) 0 ['block10\_sepconv1\_bn[0][0]']

on)

block10\_sepconv2 (SeparableCon (None, 19, 19, 728) 536536 ['block10\_sepconv2\_act[0][0]']

v2D)

block10\_sepconv2\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block10\_sepconv2[0][0]']

alization)

block10\_sepconv3\_act (Activati (None, 19, 19, 728) 0 ['block10\_sepconv2\_bn[0][0]']

on)

block10\_sepconv3 (SeparableCon (None, 19, 19, 728) 536536 ['block10\_sepconv3\_act[0][0]']

v2D)

block10\_sepconv3\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block10\_sepconv3[0][0]']

alization)

add\_8 (Add) (None, 19, 19, 728) 0 ['block10\_sepconv3\_bn[0][0]',

'add\_7[0][0]']

block11\_sepconv1\_act (Activati (None, 19, 19, 728) 0 ['add\_8[0][0]']

on)

block11\_sepconv1 (SeparableCon (None, 19, 19, 728) 536536 ['block11\_sepconv1\_act[0][0]']

v2D)

block11\_sepconv1\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block11\_sepconv1[0][0]']

alization)

block11\_sepconv2\_act (Activati (None, 19, 19, 728) 0 ['block11\_sepconv1\_bn[0][0]']

on)

block11\_sepconv2 (SeparableCon (None, 19, 19, 728) 536536 ['block11\_sepconv2\_act[0][0]']

v2D)

block11\_sepconv2\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block11\_sepconv2[0][0]']

alization)

block11\_sepconv3\_act (Activati (None, 19, 19, 728) 0 ['block11\_sepconv2\_bn[0][0]']

on)

block11\_sepconv3 (SeparableCon (None, 19, 19, 728) 536536 ['block11\_sepconv3\_act[0][0]']

v2D)

block11\_sepconv3\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block11\_sepconv3[0][0]']

alization)

add\_9 (Add) (None, 19, 19, 728) 0 ['block11\_sepconv3\_bn[0][0]',

'add\_8[0][0]']

block12\_sepconv1\_act (Activati (None, 19, 19, 728) 0 ['add\_9[0][0]']

on)

block12\_sepconv1 (SeparableCon (None, 19, 19, 728) 536536 ['block12\_sepconv1\_act[0][0]']

v2D)

block12\_sepconv1\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block12\_sepconv1[0][0]']

alization)

block12\_sepconv2\_act (Activati (None, 19, 19, 728) 0 ['block12\_sepconv1\_bn[0][0]']

on)

block12\_sepconv2 (SeparableCon (None, 19, 19, 728) 536536 ['block12\_sepconv2\_act[0][0]']

v2D)

block12\_sepconv2\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block12\_sepconv2[0][0]']

alization)

block12\_sepconv3\_act (Activati (None, 19, 19, 728) 0 ['block12\_sepconv2\_bn[0][0]']

on)

block12\_sepconv3 (SeparableCon (None, 19, 19, 728) 536536 ['block12\_sepconv3\_act[0][0]']

v2D)

block12\_sepconv3\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block12\_sepconv3[0][0]']

alization)

add\_10 (Add) (None, 19, 19, 728) 0 ['block12\_sepconv3\_bn[0][0]',

'add\_9[0][0]']

block13\_sepconv1\_act (Activati (None, 19, 19, 728) 0 ['add\_10[0][0]']

on)

block13\_sepconv1 (SeparableCon (None, 19, 19, 728) 536536 ['block13\_sepconv1\_act[0][0]']

v2D)

block13\_sepconv1\_bn (BatchNorm (None, 19, 19, 728) 2912 ['block13\_sepconv1[0][0]']

alization)

block13\_sepconv2\_act (Activati (None, 19, 19, 728) 0 ['block13\_sepconv1\_bn[0][0]']

on)

block13\_sepconv2 (SeparableCon (None, 19, 19, 1024 752024 ['block13\_sepconv2\_act[0][0]']

v2D) )

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| block13\_sepconv2\_bn (BatchNorm  ['block13\_sepconv2[0][0]'] | (None, | 19, | 19, | 1024 | 4096 |
| alization) | ) | | | |  |
| conv2d\_3 (Conv2D) ['add\_10[0][0]'] | (None, 10, 10, 1024  ) | | | | 745472 |
| block13\_pool (MaxPooling2D) ['block13\_sepconv2\_bn[0][0]'] | (None, 10, 10, 1024  ) | | | | 0 |
| batch\_normalization\_3 (BatchNo | (None, 10, 10, 1024 | | | | 4096 |
| ['conv2d\_3[0][0]'] rmalization) | ) | | | |  |
| add\_11 (Add) | (None, 10, 10, 1024 | | | | 0 |

['block13\_pool[0][0]', ['block14\_sepconv1[0][0]'] alization) )

block14\_sepconv1\_act (Activati (None, 10, 10, 1536 0 ['block14\_sepconv1\_bn[0][0]']

on) )

block14\_sepconv2 (SeparableCon (None, 10, 10, 2048 3159552

['block14\_sepconv1\_act[0][0]'] v2D) )

block14\_sepconv2\_bn (BatchNorm (None, 10, 10, 2048 8192 ['block14\_sepconv2[0][0]']

alization) )

block14\_sepconv2\_act (Activati (None, 10, 10, 2048 0 ['block14\_sepconv2\_bn[0][0]']

on) )

ﬂatten (Flatten) (None, 204800) 0

['block14\_sepconv2\_act[0][0]']

dense (Dense) (None, 5) 1024005

['ﬂatten[0][0]']

==============================================================================

====================

Total params: 21,885,485

Trainable params: 1,024,005

Non-trainable params: 20,861,480

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Epoch 1/30  3/3 [==============================] | - | 55s | 17s/step | - | loss: | 11.8808 - | |
| accuracy: 0.3854 |  |  |  |  |  |  | |
| Epoch 2/30 |  |  |  |  |  |  | |
| 3/3 [==============================] | - | 43s | 12s/step | - | loss: | 12.2067 - | |
| accuracy: 0.5625 |  |  |  |  |  |  | |
| Epoch 3/30 |  |  |  |  |  |  | |
| 3/3 [==============================] | - 47s | | 13s/step | - loss: | | 9.0027 | - accuracy: |
| 0.5833 |  | |  |  | |  |  |
| Epoch 4/30 |  | |  |  | |  |  |
| 3/3 [==============================] | - 46s | | 13s/step | - loss: | | 6.3194 | - accuracy: |
| 0.5729 |  | |  |  | |  |  |
| Epoch 5/30 |  | |  |  | |  |  |
| 3/3 [==============================] | - 45s | | 13s/step | - loss: | | 5.9865 | - accuracy: |
| 0.5625 |  | |  |  | |  |  |
| Epoch 6/30 |  | |  |  | |  |  |
| 3/3 [==============================] | - 44s | | 13s/step | - loss: | | 4.8598 | - accuracy: |
| 0.7083 |  | |  |  | |  |  |
| Epoch 7/30 |  | |  |  | |  |  |
| 3/3 [==============================] | - 44s | | 13s/step | - loss: | | 4.9243 | - accuracy: |
| 0.5938 |  | |  |  | |  |  |
| Epoch 8/30 |  | |  |  | |  |  |
| 3/3 [==============================] | - 43s | | 13s/step | - loss: | | 5.4845 | - accuracy: |
| 0.6250 |  | |  |  | |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch 9/30 |  | | | | |
| 3/3 [==============================] | - 43s | 13s/step | - loss: | 4.2899 | - accuracy: |
| 0.6146 |  |  |  |  |  |
| Epoch 10/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 3.8048 | - accuracy: |
| 0.6458 |  |  |  |  |  |
| Epoch 11/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 3.5794 | - accuracy: |
| 0.6250 |  |  |  |  |  |
| Epoch 12/30 |  |  |  |  |  |
| 3/3 [==============================] | - 45s | 13s/step | - loss: | 3.4683 | - accuracy: |
| 0.6458 |  |  |  |  |  |
| Epoch 13/30 |  |  |  |  |  |
| 3/3 [==============================] | - 48s | 15s/step | - loss: | 2.8153 | - accuracy: |
| 0.7917 |  |  |  |  |  |
| Epoch 14/30 |  |  |  |  |  |
| 3/3 [==============================] | - 45s | 13s/step | - loss: | 3.0900 | - accuracy: |
| 0.6458 |  |  |  |  |  |
| Epoch 15/30 |  |  |  |  |  |
| 3/3 [==============================] | - 47s | 13s/step | - loss: | 2.9761 | - accuracy: |
| 0.6875 |  |  |  |  |  |
| Epoch 16/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 2.6205 | - accuracy: |
| 0.7292 |  |  |  |  |  |
| Epoch 17/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 2.8528 | - accuracy: |
| 0.6458 |  |  |  |  |  |
| Epoch 18/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 2.7091 | - accuracy: |
| 0.6458 |  |  |  |  |  |
| Epoch 19/30 |  |  |  |  |  |
| 3/3 [==============================] | - 45s | 13s/step | - loss: | 3.1372 | - accuracy: |
| 0.6667 |  |  |  |  |  |
| Epoch 20/30 |  |  |  |  |  |
| 3/3 [==============================] | - 45s | 13s/step | - loss: | 2.6082 | - accuracy: |
| 0.7500 |  |  |  |  |  |
| Epoch 21/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 3.9316 | - accuracy: |
| 0.7188 |  |  |  |  |  |
| Epoch 22/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 3.0207 | - accuracy: |
| 0.7292 |  |  |  |  |  |
| Epoch 23/30 |  |  |  |  |  |
| 3/3 [==============================] | - 48s | 14s/step | - loss: | 2.7646 | - accuracy: |
| 0.6250 |  |  |  |  |  |
| Epoch 24/30 |  |  |  |  |  |
| 3/3 [==============================] | - 36s | 13s/step | - loss: | 3.3575 | - accuracy: |
| 0.7308 |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch 25/30 |  |  |  |  |  |
| 3/3 [==============================] | - 53s | 14s/step | - loss: | 3.2141 | - accuracy: |
| 0.7396 |  |  |  |  |  |
| Epoch 26/30 |  |  |  |  |  |
| 3/3 [==============================] | - 48s | 15s/step | - loss: | 3.9002 | - accuracy: |
| 0.6979 |  |  |  |  |  |
| Epoch 27/30 |  |  |  |  |  |
| 3/3 [==============================] | - 45s | 13s/step | - loss: | 2.6503 | - accuracy: |
| 0.7083 |  |  |  |  |  |
| Epoch 28/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 2.6253 | - accuracy: |
| 0.7812 |  |  |  |  |  |
| Epoch 29/30 |  |  |  |  |  |
| 3/3 [==============================] | - 45s | 13s/step | - loss: | 3.3202 | - accuracy: |
| 0.6771 |  |  |  |  |  |
| Epoch 30/30 |  |  |  |  |  |
| 3/3 [==============================] | - 44s | 13s/step | - loss: | 4.4876 | - accuracy: |
| 0.6667  **7.2 Feature 2** |  |  |  |  |  |

## Application building (Using Python-Flask)

import numpy as np import os

from tensorﬂow import keras

from keras import models

from keras.models import load\_model from keras.preprocessing import image

from keras.applications.inception\_v3 import preprocess\_input import requests

from ﬂask import Flask, request, render\_template, redirect, url\_for from cloudant.client import Cloudant

model = load\_model(r"Updated-xception-diabetic-retinopathy.h5") app = Flask(\_name\_)

# Authenticate using an IAM API key

client = Cloudant.iam('367e91e7-6150-4f63-92f4-24625af53457-bluemix', 'EqKm5BOKxzGLIm9YsFnXKJ66ywOyL9tDVK9oN0\_FPD4G', connect=True)

# Create a database using an initialized client my\_database = client.create\_database('my\_db') if my\_database.exists():

print("Database '{0}' successfully created.".format('my\_db'))

# default home page or route @app.route('/')

def index():

return render\_template('index.html')

@app.route('/index') def home():

return render\_template("index.html")

'''@ app.route('/register') def register():

return render\_template("register.html")'''

# registration page

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

if request.method == "POST":

name = request.form.get("name") mail = request.form.get("emailid") mobile = request.form.get("num") pswd = request.form.get("pass") data = {

'name': name,

'mail': mail, 'mobile': mobile, 'psw': pswd

}

print(data)

query = {'mail': {'$eq': data['mail']}}

docs = my\_database.get\_query\_result(query) print(docs)

print(len(docs.all()))

if (len(docs.all()) == 0):

url = my\_database.create\_document(data)

return render\_template("register.html", pred=" Registration Successful , please login using your details ")

else:

return render\_template('register.html', pred=" You are already a member , please login using your details ")

else:

return render\_template('register.html')

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

if request.method == "POST":

user = request.form.get('name') passw = request.form.get('pass') print(user, passw)

query = {'\_id': {'$eq': user}}

docs = my\_database.get\_query\_result(query) print(docs)

print(len(docs.all()))

if (len(docs.all()) == 0):

return render\_template('login.html', pred="The username is not found.") else:

if ((user == docs[0][0]['\_id'] and passw == docs[0][0]['pswd'])): return redirect(url\_for('prediction'))

else:

print('Invalid User')

else:

return render\_template('login.html')

@app.route('/logout') def logout():

return render\_template('logout.html')

@app.route("/predict") def predict():

return render\_template("prediction.html")

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

if request.method == "POST":

f = request.ﬁles['image']

# getting the current path 1.e where app.py is present basepath = os.path.dirname(\_ﬁle\_)

# print ( " current path " , basepath )

# from anywhere in the system we can give image but we want that ﬁlepath = os.path.join(basepath, 'uploads', f.ﬁlename)

# print ( " upload folder is " , ﬁlepath ) f.save(ﬁlepath)

img = image.load\_img(ﬁlepath, target\_size=(299, 299)) x = image.img\_to\_array(img) # ing to array

x = np.expand\_dims(x, axis=0) # used for adding one more dimension # print ( x )

img\_data = preprocess\_input(x)

prediction = np.argmax(model.predict(img\_data), axis=1)

# prediction = model.predict ( x ) #instead of predict\_classes ( x ) we can use predict ( X ) >

predict\_classes ( x ) gave error

# print ( " prediction is prediction )

index = [' No Diabetic Retinopathy ', ' Mild DR ',

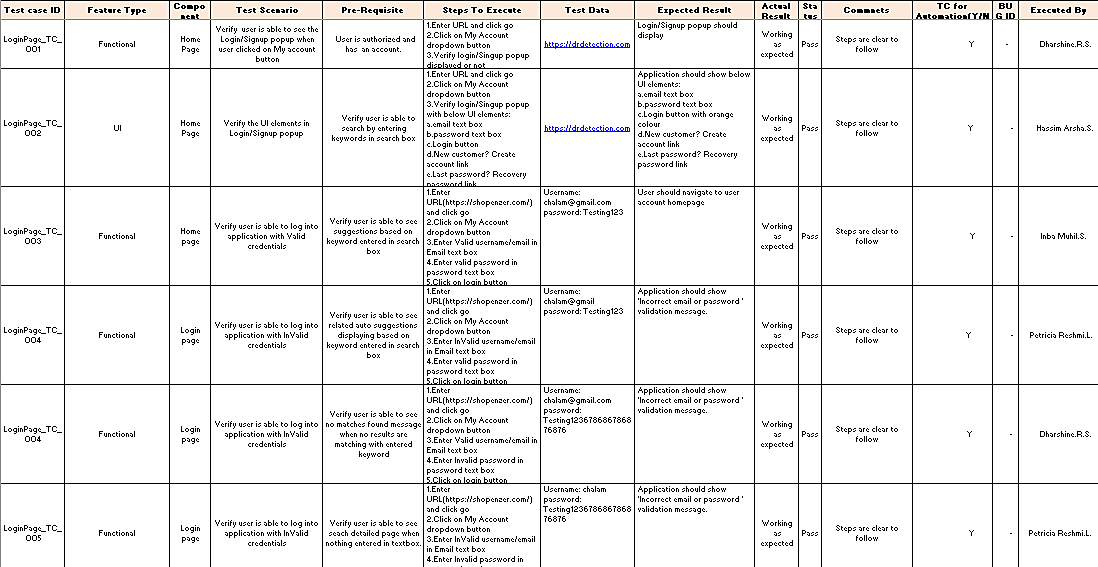
' Moderate DR ', ' Severe DR ', ' Proliferative DR '] # result = str ( index [ output [ 011 )

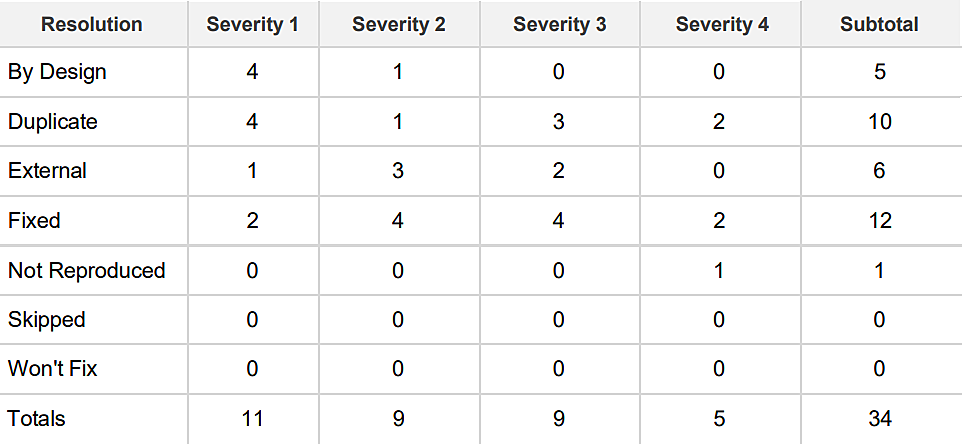
result = str(index[prediction[0]]) print(result)

return render\_template('prediction.html', prediction=result)

if \_name\_ == "\_main\_": app.run(debug=False)

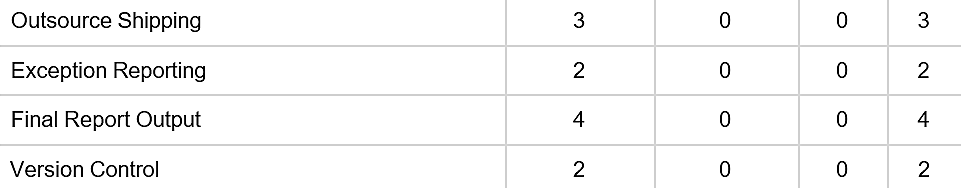
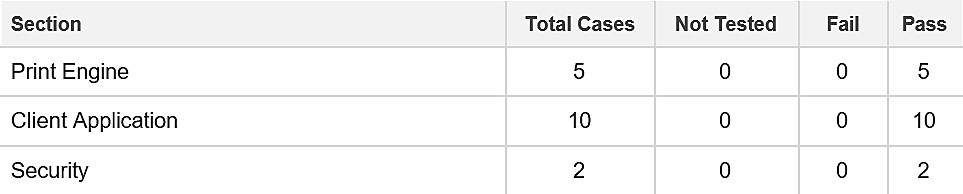
## TESTING

* 1. **Test Cases**
  2. **User Acceptance Testing Defect Analysis**

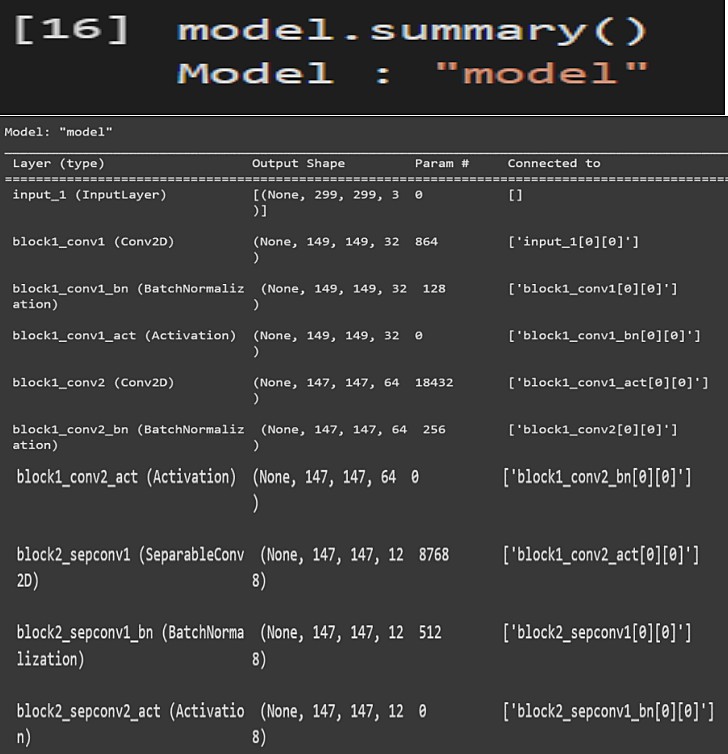
This report shows the number of resolved or closed bugs at each severity level, and how they were resolved.

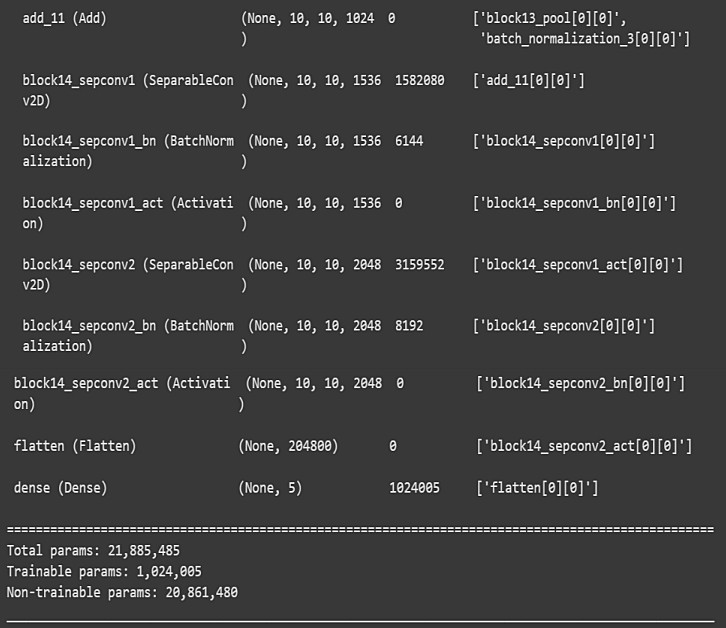
**Test Case Analysis**

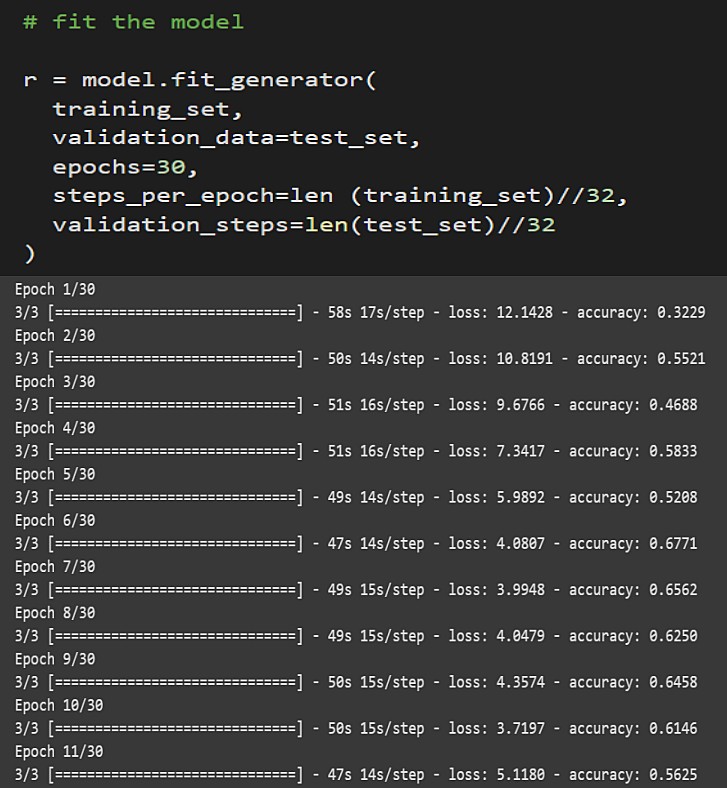
This report shows the number of test cases that have passed, failed,and untested.

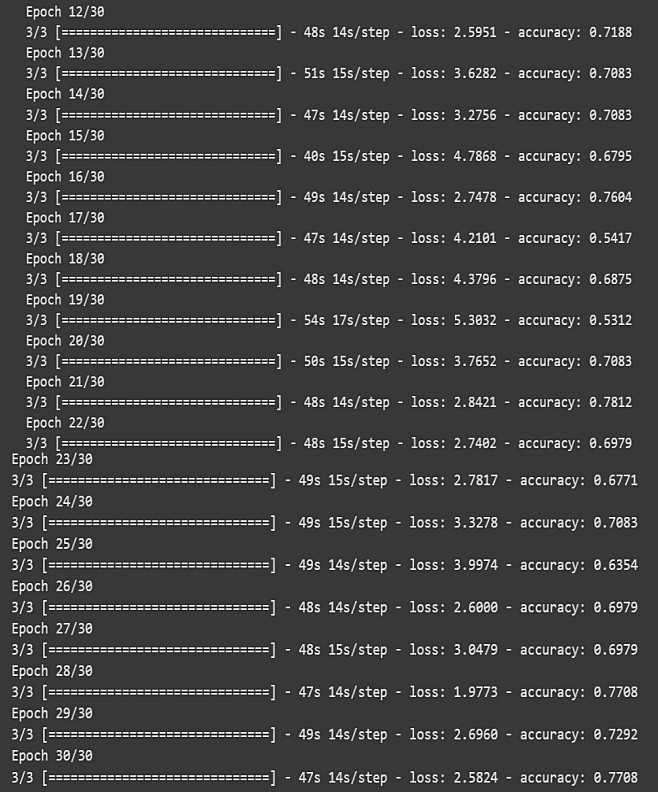


1. **RESULTS**
   1. **Performance Metrics Model Summary**





**Accuracy (Training and Validation Accuracy)**



1. **ADVANTAGES & DISADVANTAGES**
   1. **Advantages**

1) Early detection and treatment can reduce the risk of vision loss for diabetics by 25%. 2)The economic cost of early diagnosis and treatment is low.

1. Treatment for patients with recognizable disease is safe, effective and universally agreeable.
2. It is an appropriate screening procedure which is acceptable both to the public and health care professionals.
   1. **Disadvantages**

1) It requires very large amount of data in order to perform better than other techniques. 2)It is extremely expensive to train due to complex data models. Moreover deep learning requires expensive GPUs and hundreds of machines. This increases cost to the users.

3)There is no standard theory to guide you in selecting right deep learning tools as it requires knowledge of topology, training method and other parameters. As a result it is diﬃcult to be adopted by less skilled people.

1. **CONCLUSION**

Diabetic retinopathy is a serious [complication of diabetes mellitus](https://www.sciencedirect.com/topics/medicine-and-dentistry/complications-of-diabetes-mellitus), leading to progressive damage and even blindness of the retina. Its early detection and treatment is important in order to prevent its deterioration and the [retina's damage](https://www.sciencedirect.com/topics/medicine-and-dentistry/retina-injury). The interest in applying deep learning in detecting diabetic retinopathy has increased during the past years and as several DL systems evolve and become integrated into the clinical practice, they will enable the clinicians to treat the patients in need more effectively and eﬃciently. This article presents the current state of research regarding the application of deep learning in diagnosing diabetic retinopathy. Although deep learning has paved the way for more accurate diagnosis and treatment, further improvements are still necessary regarding performance, interpretability and trustworthiness from ophthalmologists.

1. **FUTURE SCOPE**

Use of AI in medical diagnostics, especially in ophthalmology heralds a new era. If proven to be sensitive and speciﬁc enough this technology can totally change the way we look at screening programs and community-based ophthalmology programs. Most of the present systems use conventional of 30–50° fundus images.A lot of work is also being done on identifying serum biomarkers for early detection and monitoring of diseases like diabetic retinopathy. Thus, a comprehensive analysis of fundus imaging, systemic parameter proﬁle and other serum biomarkers using AI might provide better insights, perhaps even better conclusions than what human intelligence is capable of deriving.

1. **APPENDIX**

Github Link: <https://github.com/IBM-EPBL/IBM-Project-13274-1659515636>

Team Id: PNT2022TMID12524