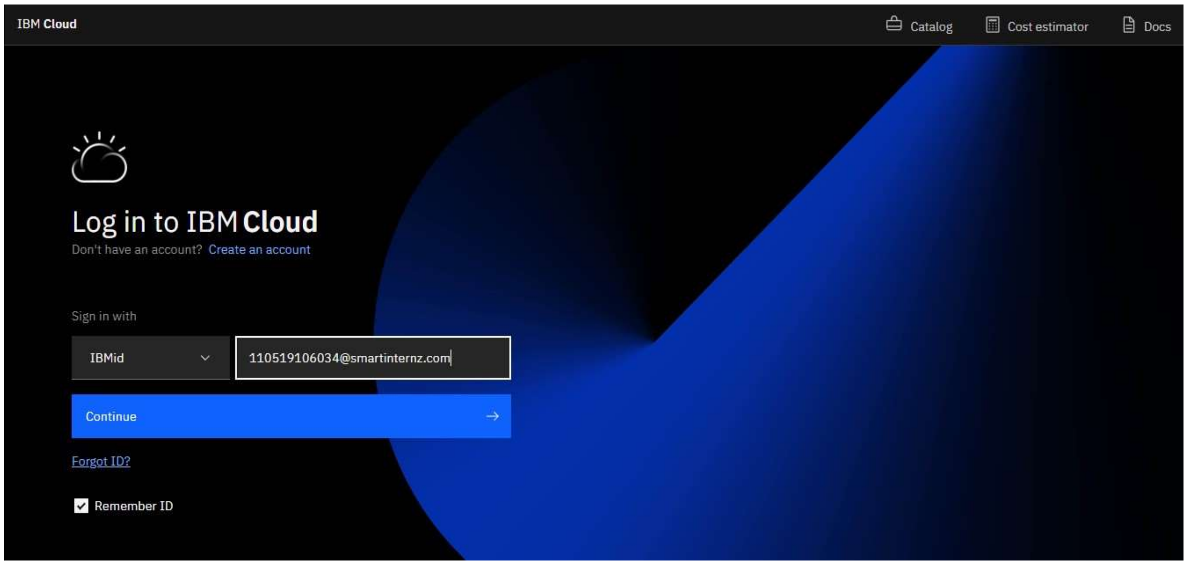
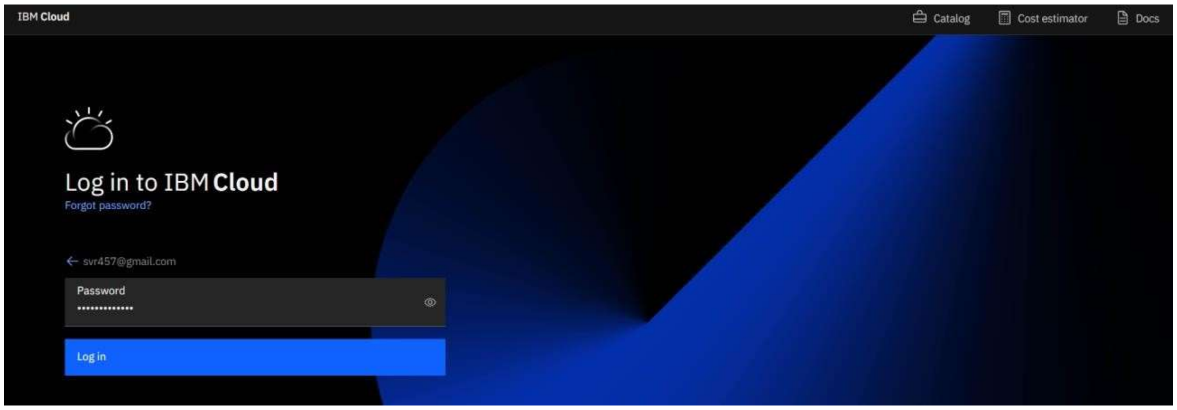
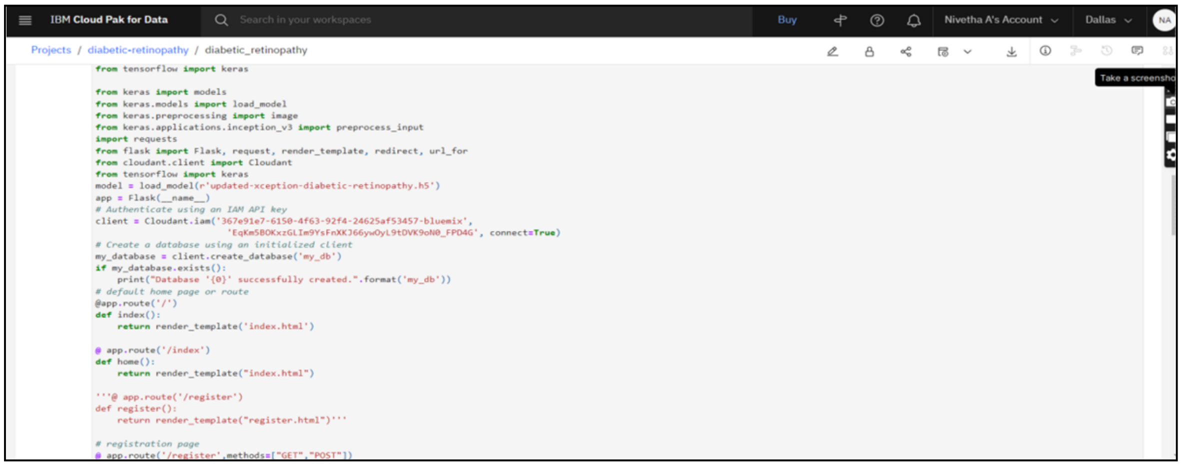
Team ID	PNT2022TMID36211
Project Name	Deep Learning Fundus Image Analysis for Detection of Diabetic Retinopathy

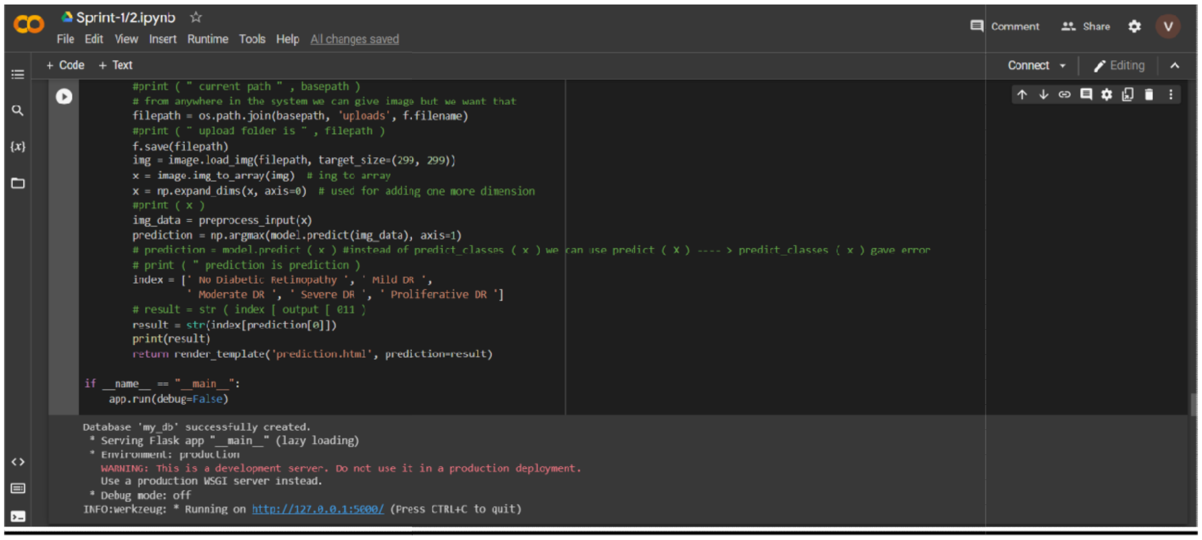





IBM Cloud Pak for Data
Search in your workspaces
Buy
Nivetha A's Account
Dallas
Projects / diabetic-retinopathy / diabetic_retinopathy
Take a screensho

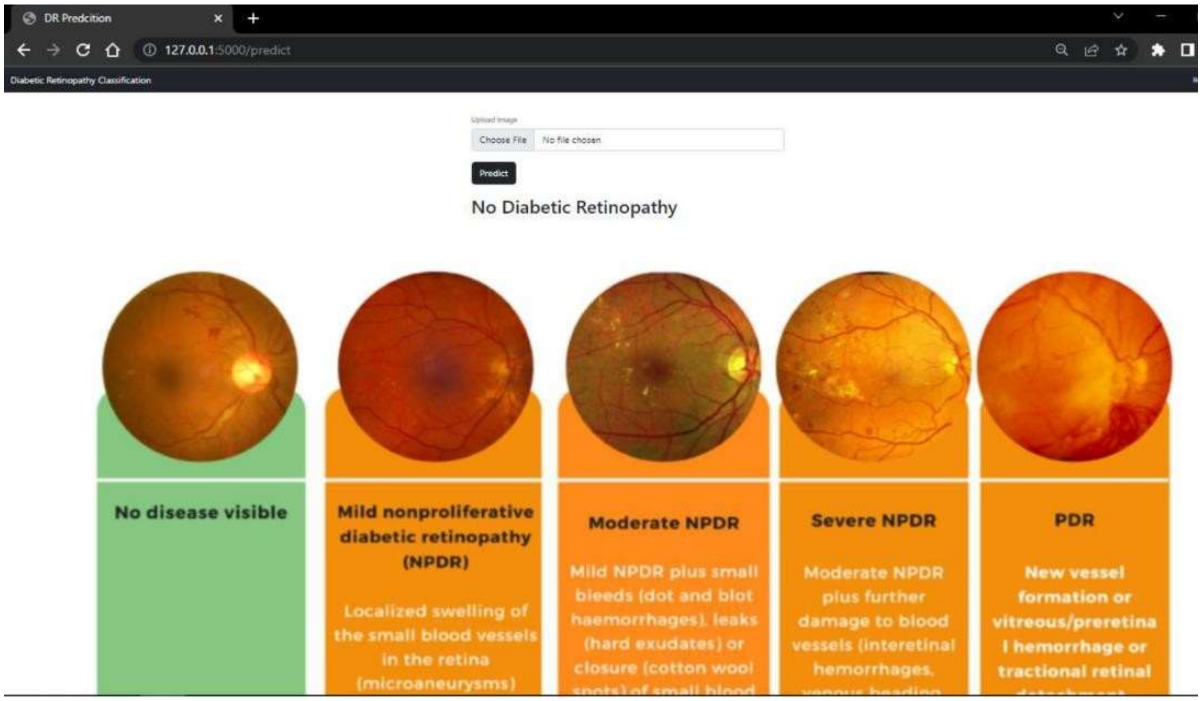
```python
from tensorflow import keras

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
from tensorflow import keras
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"])
```

Python Code



```
        #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)
```

```
Database 'my_db' successfully created.
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```


DR Predcition
127.0.0.1:5000/predict
Diabetic Retinopathy Classification
Upload image
Choose File
No file chosen
Predict
No disease visible
Mild nonproliferative diabetic retinopathy (NPDR)
Localized swelling of the small blood vessels in the retina (microaneurysms)
Moderate NPDR
Mild NPDR plus small bleeds (dot and blot haemorrhages), leaks (hard exudates) or closure (cotton wool spots) of small blood
Severe NPDR
Moderate NPDR plus further damage to blood vessels (interetinal hemorrhages,
PDR
New vessel formation or vitreous/preretinal hemorrhage or tractional retinal

No Diabetic Retinopathy