# WEEK 5 : CLOUD AND API DEPLOYMENT

**Name :** Abhimanyu Gangani

**Batch Code :** LISUM15

**Submission Date : 1 December 2022**

**Submission To : Data Glacier**

# 1. Creating a model using sklearn library:

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle

dataset = pd.read_csv('diabetes.csv')

df[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age']] = df[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age']].replace(0,np.nan)

df.fillna(df.mean(),inplace=True)

X = df.iloc[:, :5]

y = df.iloc[:,-3]

#Splitting Training and Test Set
#Since we have a very small dataset, we will train our model with all available data.

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()

#Fitting model with trainig data
regressor.fit(X, y)

# Saving model to disk
pickle.dump(regressor, open('model.pkl','wb'))

# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[0,145,70,30,50]]))
```

# 2. Creating a flask application using python library

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))

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

@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    float_features = [float(x) for x in request.form.values()]
    final_features = [np.array(float_features)]
    prediction = model.predict(final_features)

    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text='DiabetesPedigreeFunction should be  {}'.format(output))

@app.route('/predict_api',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = model.predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)

if __name__ == "__main__":
    app.run(debug=True)
```

## 3. Creating the request file:

```
import requests

url = 'http://localhost:5000/predict_api'
r = requests.post(url,json={'Pregnancies':1, 'Glucose':143, 'BloodPressure':75, 'SkinThickness':35, 'Insulin':87})

print(r.json())
```

## 4. Create a HTML file and save it to Templates folder in same folder where python files are located:

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

</head>

<body>
 <div class="login">
        <h1>Predict Salary Analysis</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
        <input type="text" name="Pregnancies" placeholder="Pregnancies" required="required" />
        <input type="text" name="Glucose" placeholder="Glucose" required="required" />
        <input type="text" name="BloodPressure" placeholder="BloodPressure" required="required" />
        <input type="text" name="SkinThickness" placeholder="SkinThickness" required="required" />
        <input type="text" name="Insulin" placeholder="Insulin" required="required" />


        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
    </form>

   <br>
   <br>
   {{ prediction_text }}

 </div>


</body>
</html>
```

# 5. Create css file for styling in static/css folder as style.css :

```
@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; font-size: 13px; line-height: 18px; color: #333333; text-align: center;text-shadow: 0 1px 1px rgba(255, 255, 255, 0.75); vertical-align: middle; background-color: #f5f5f5; background-image: -moz-linear-gradient(top, #ffffff, #e6e6e6); background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6)); background-image: -webkit-linear-gradient(top, #ffffff, #e6e6e6); background-image: -o-linear-gradient(top, #ffffff, #e6e6e6); background-image: linear-gradient(top, #ffffff, #e6e6e6); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff, endColorstr=#e6e6e6, GradientType=0); border-color: #e6e6e6 #e6e6e6 #e6e6e6; border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25); border: 1px solid #e6e6e6; -webkit-border-radius: 4px; -moz-border-radius: 4px; border-radius: 4px; -webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); -moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); cursor: pointer; *margin-left: .3em; }
.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-border-radius: 5px; border-radius: 5px; }
.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position 0.1s linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s linear; transition: background-position 0.1s linear; }
.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }
.btn-primary.active { color: rgba(255, 255, 255, 0.75); }
.btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4); background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4)); background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4); background-image: -o-linear-gradient(top, #6eb6de, #4a77d4); background-image: linear-gradient(top, #6eb6de, #4a77d4); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de, endColorstr=#4a77d4, GradientType=0);  border: 1px solid #3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5); }
.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }
```

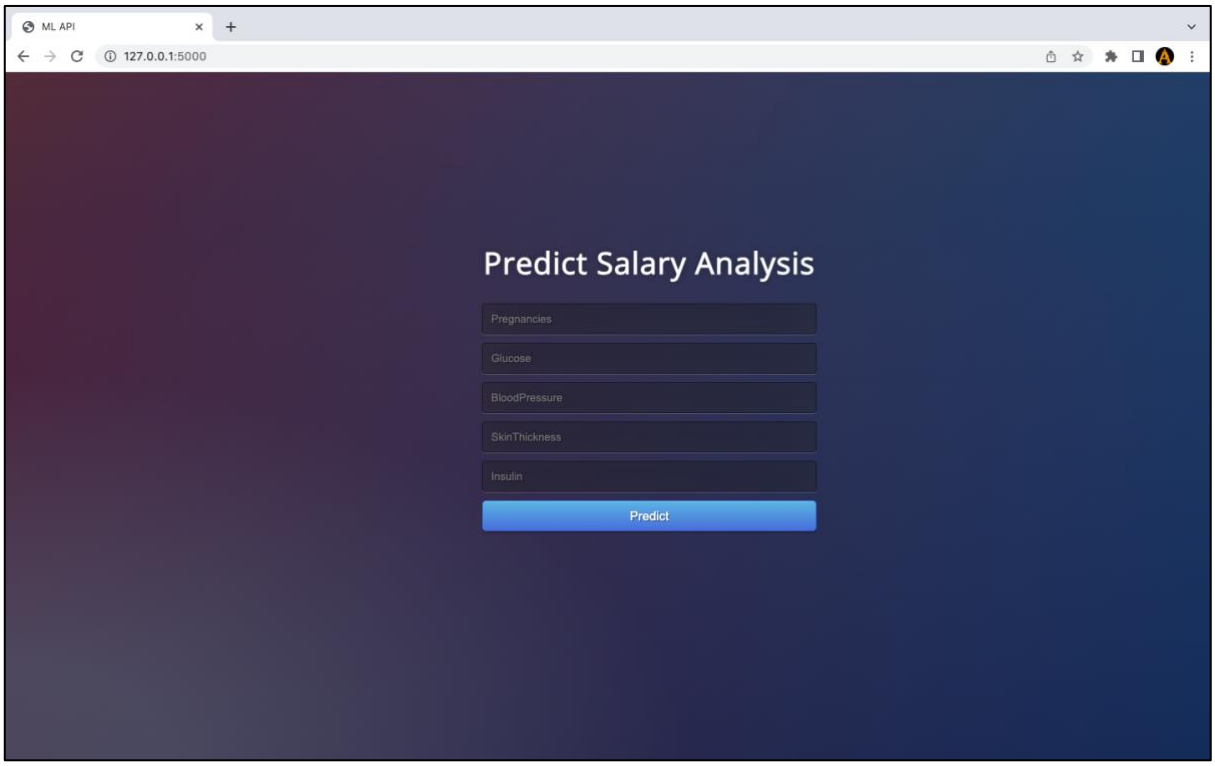
```
.btn-block { width: 100%; display:block; }

* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-sizing:border-box; -o-box-sizing:border-box; box-sizing:border-box; }

html { width: 100%; height:100%; overflow:hidden; }

body {
    width: 100%;
    height:100%;
    font-family: 'Open Sans', sans-serif;
    background: #092756;
    color: #fff;
    font-size: 18px;
    text-align:center;
    letter-spacing:1.2px;
    background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linear-gradient(top,  rgba(57,173,219,.25) 0%, rgba(42,60,87,.4) 100%), -moz-linear-gradient(-45deg,  #670d10 0%, #092756 100%);
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-gradient(top,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -webkit-linear-gradient(-45deg,  #670d10 0%,#092756 100%);
    background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -o-linear-gradient(-45deg,  #670d10 0%,#092756 100%);
    background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -ms-linear-gradient(-45deg,  #670d10 0%,#092756 100%);
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear-gradient(to bottom,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), linear-gradient(135deg,  #670d10 0%,#092756 100%);
    filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );

}
.login {
    position: absolute;
    top: 40%;
    left: 50%;
    margin: -150px 0 0 -150px;
    width:400px;
    height:400px;
}

.login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }
```

```
input {
    width: 100%;
    margin-bottom: 10px;
    background: rgba(0,0,0,0.3);
    border: none;
    outline: none;
    padding: 10px;
    font-size: 13px;
    color: #fff;
    text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
    border: 1px solid rgba(0,0,0,0.3);
    border-radius: 4px;
    box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px
rgba(255,255,255,0.2);
    -webkit-transition: box-shadow .5s ease;
    -moz-transition: box-shadow .5s ease;
    -o-transition: box-shadow .5s ease;
    -ms-transition: box-shadow .5s ease;
    transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px
rgba(255,255,255,0.2); }
```

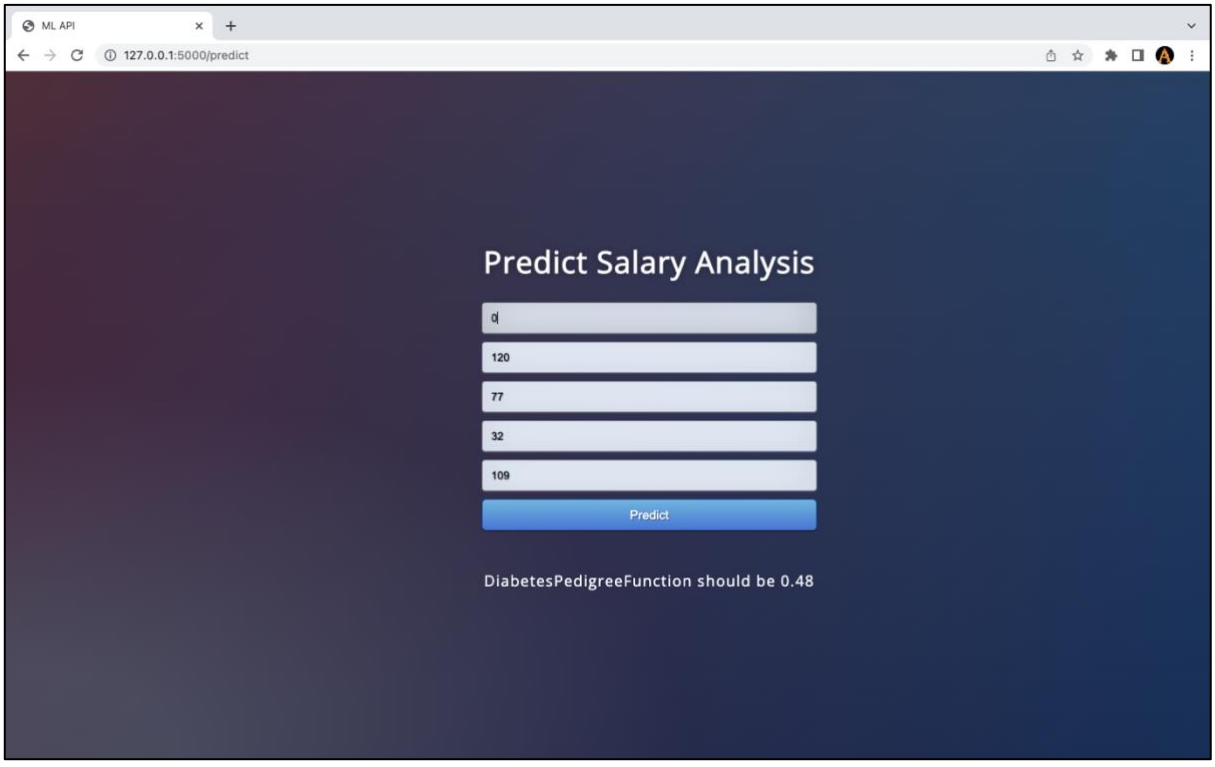
# 6. Open terminal and run the flask application file :

```
(base) abhimanyus-MacBook-Air:final_app abhimanyu$ ls
Untitled.ipynb  WEEK 4.docx     app.py          diabetes.csv    model.pkl       model.py        request.py      static          templates       ~$WEEK 4.docx
(base) abhimanyus-MacBook-Air:final_app abhimanyu$ python app.py
 * Serving Flask app "app" (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: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with watchdog (fsevents)
 * Debugger is active!
 * Debugger PIN: 594-515-009
```

## 7. Use the URL to access the application :




## 8. Enter any experience value to predict the salary :



Our application is running fine on the local machine lets try to deploy it on the Heroku cloud.

# Steps to deploy on Heroku:

1. **Create account on Heroku cloud.**
2. **Open terminal on your machine and install pipenv**

```
(base) abhimanyu@abhimanyus-MacBook-Air ~ % pip install pipenv
Requirement already satisfied: pipenv in ./opt/anaconda3/lib/python3.9/site-packages (2022.11.25)
Requirement already satisfied: virtualenv in ./opt/anaconda3/lib/python3.9/site-packages (from pipenv) (20.16.7)
Requirement already satisfied: certifi in ./opt/anaconda3/lib/python3.9/site-packages (from pipenv) (2021.10.8)
Requirement already satisfied: setuptools>=36.2.1 in ./opt/anaconda3/lib/python3.9/site-packages (from pipenv) (58.0.4)
Requirement already satisfied: virtualenv-clone>=0.2.5 in ./opt/anaconda3/lib/python3.9/site-packages (from pipenv) (0.5.7)
Requirement already satisfied: platformdirs<3,>=2.4 in ./opt/anaconda3/lib/python3.9/site-packages (from virtualenv->pipenv) (2.5.4)
Requirement already satisfied: distlib<1,>=0.3.6 in ./opt/anaconda3/lib/python3.9/site-packages (from virtualenv->pipenv) (0.3.6)
Requirement already satisfied: filelock<4,>=3.4.1 in ./opt/anaconda3/lib/python3.9/site-packages (from virtualenv->pipenv) (3.8.0)
(base) abhimanyu@abhimanyus-MacBook-Air ~ %
```

3. **Create a virtual environment with pipenv and install Flask and Gunicorn . (in directory where your files are present).**

```
(base) abhimanyu@abhimanyus-MacBook-Air final % pipenv install flask gunicron
Warning: the environment variable LANG is not set!
We recommend setting this in ~/.profile (or equivalent) for proper expected behavior.
Creating a virtualenv for this project...
Pipfile: /Users/abhimanyu/Desktop/Glacier/final/Pipfile
Using /Users/abhimanyu/opt/anaconda3/bin/python3 (3.9.7) to create virtualenv...
⠋ Creating virtual environment...created virtual environment CPython3.9.7.final.0-64 in 1445ms
  creator Venv(dest=/Users/abhimanyu/.local/share/virtualenvs/final-BCmYNCXP, clear=False, no_vcs_ignore=False, global=False, describe=CPython3Posix)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/Users/abhimanyu/Library/Application Support/virtualenv)
    added seed packages: pip==22.3.1, setuptools==65.5.1, wheel==0.38.4
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator

✔ Successfully created virtual environment!
Virtualenv location: /Users/abhimanyu/.local/share/virtualenvs/final-BCmYNCXP
Creating a Pipfile for this project...
Installing flask...
Installing gunicron...
⠙ Installing gunicron...[31m[1mError: [0m An error occurred while installing [32mgunicron[0m!
Error text:
[36mERROR: Could not find a version that satisfies the requirement gunicron (from versions: none)
ERROR: No matching distribution found for gunicron
[0m
✘ Installation Failed
(base) abhimanyu@abhimanyus-MacBook-Air final %
```

4. Create a ProcFile and write following code

```
(base) abhimanyu@abhimanyus-MacBook-Air final % cat Procfile
web: gunicorn app:app
(base) abhimanyu@abhimanyus-MacBook-Air final %
```

5. Create a requirement.txt file and mention all the versions required for application in that.

```
(base) abhimanyu@abhimanyus-MacBook-Air final % cat requirements.txt
Flask==1.1.1
gunicorn==19.9.0
itsdangerous==1.1.0
Jinja2==2.10.1
MarkupSafe==1.1.1
Werkzeug==0.15.5
numpy>=1.9.2
scipy>=0.15.1
scikit-learn>=0.18
matplotlib>=1.4.3
pandas>=0.19
(base) abhimanyu@abhimanyus-MacBook-Air final %
```

6. Run command pipenv shell

```
(base) abhimanyu@abhimanyus-MacBook-Air final % pipenv shell
Launching subshell in virtual environment...
 . /Users/abhimanyu/.local/share/virtualenvs/final-BCmYNCXP/bin/activate
(base) abhimanyu@abhimanyus-MacBook-Air final %  . /Users/abhimanyu/.local/share/virtualenvs/final-BCmYNCXP/bin/activate
(final) (base) abhimanyu@abhimanyus-MacBook-Air final %
```

## 7. Initialize an empty repo, add the files in the repo and commit all the changes.

```
(final) (base) abhimanyu@abhimanyus-MacBook-Air final % git init
Reinitialized existing Git repository in /Users/abhimanyu/Desktop/Glacier/final/.git/
(final) (base) abhimanyu@abhimanyus-MacBook-Air final % git add .
(final) (base) abhimanyu@abhimanyus-MacBook-Air final % git commit -m 'initial commit'
[master f6afa4c] initial commit
 1 file changed, 12 insertions(+)
 create mode 100644 Pipfile
(final) (base) abhimanyu@abhimanyus-MacBook-Air final %
```

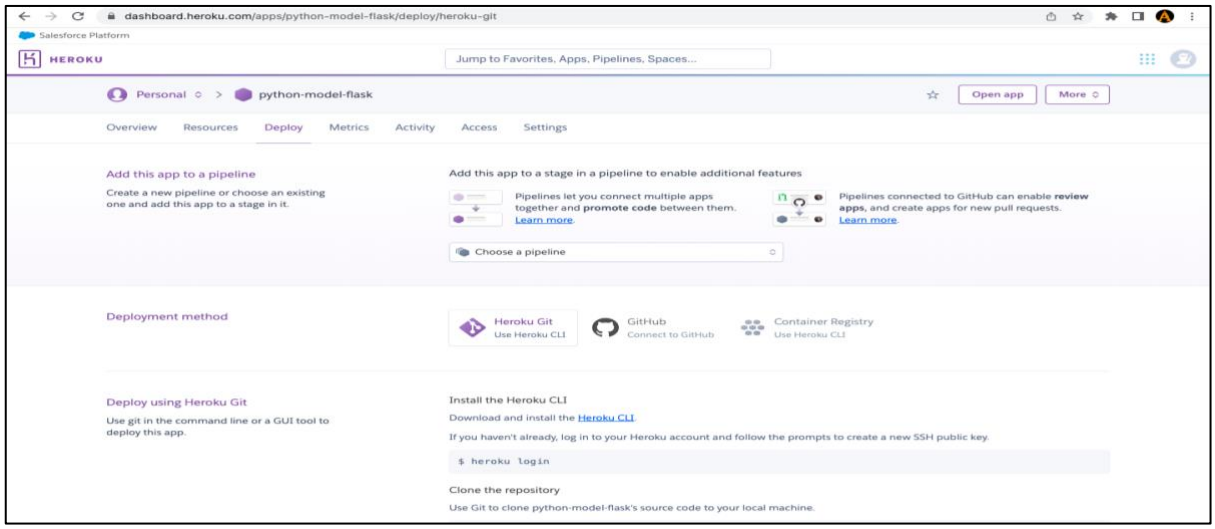
## 8. Install Heroku CLI and run Heroku login

```
(final) (base) abhimanyu@abhimanyus-MacBook-Air final % heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/fa69b898-614f-47d7-aa8d-ee2be1f7fc38?requestor=SFMyNTY.g2gDbQAAAAw4Mi4xLjIzMC4yMTluBgCw8t_OhAFiAAFRgA.6fz0tqUQ1
658t8
Logging in... done
Logged in as agangani97@gmail.com
(final) (base) abhimanyu@abhimanyus-MacBook-Air final %
```

## 9. Run command git push Heroku master.

```
(final) (base) abhimanyu@abhimanyus-MacBook-Air python-model-flask % git push heroku master
Everything up-to-date
(final) (base) abhimanyu@abhimanyus-MacBook-Air python-model-flask %
```

# 10. Run the app using Heroku




# 11. Check the application and try running it (https://python-model-flask.herokuapp.com/predict)