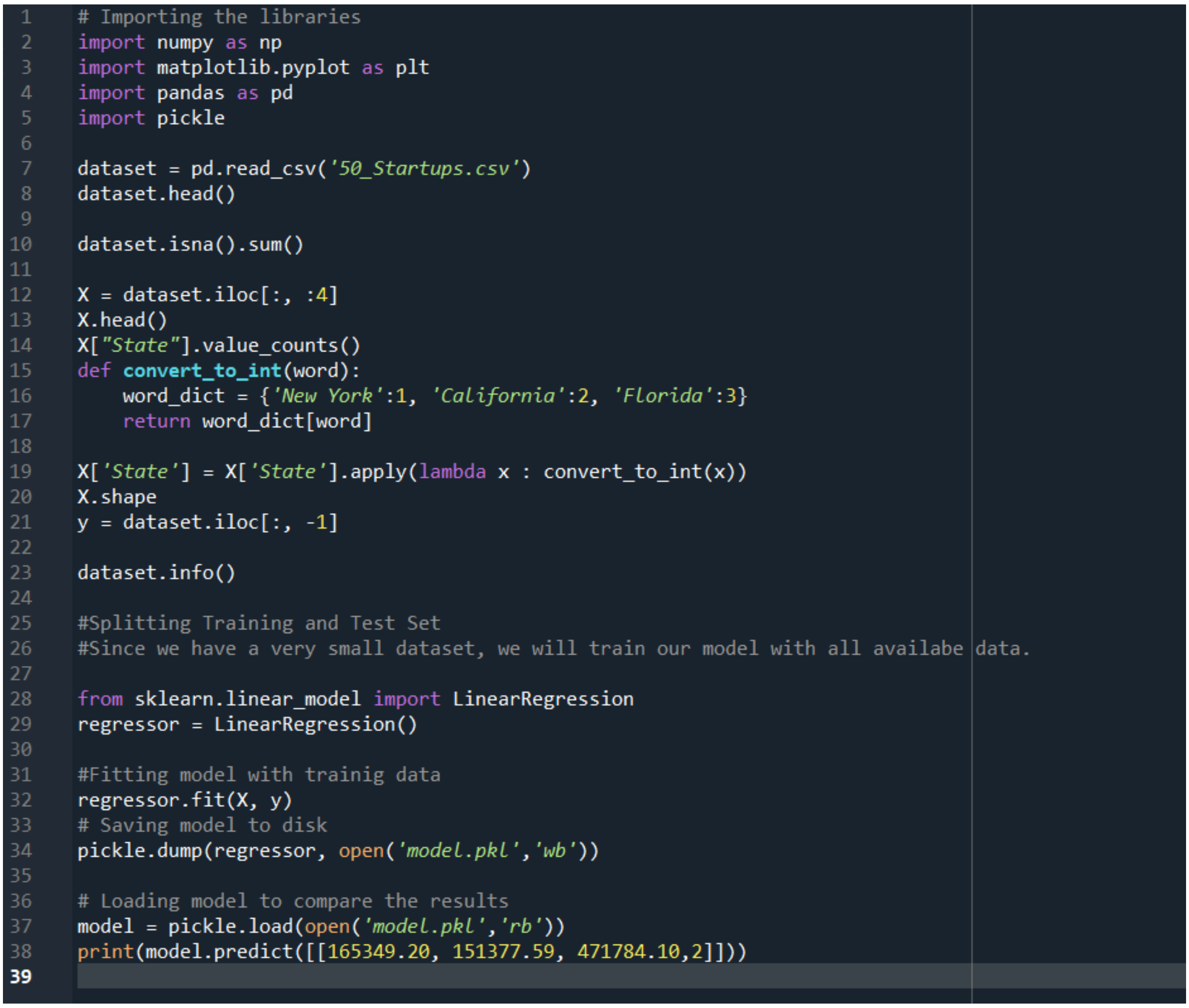
NAME: Jagriti Bablani

BATCH CODE: LISUM02

SUBMISSION DATE:22-Augest-2021

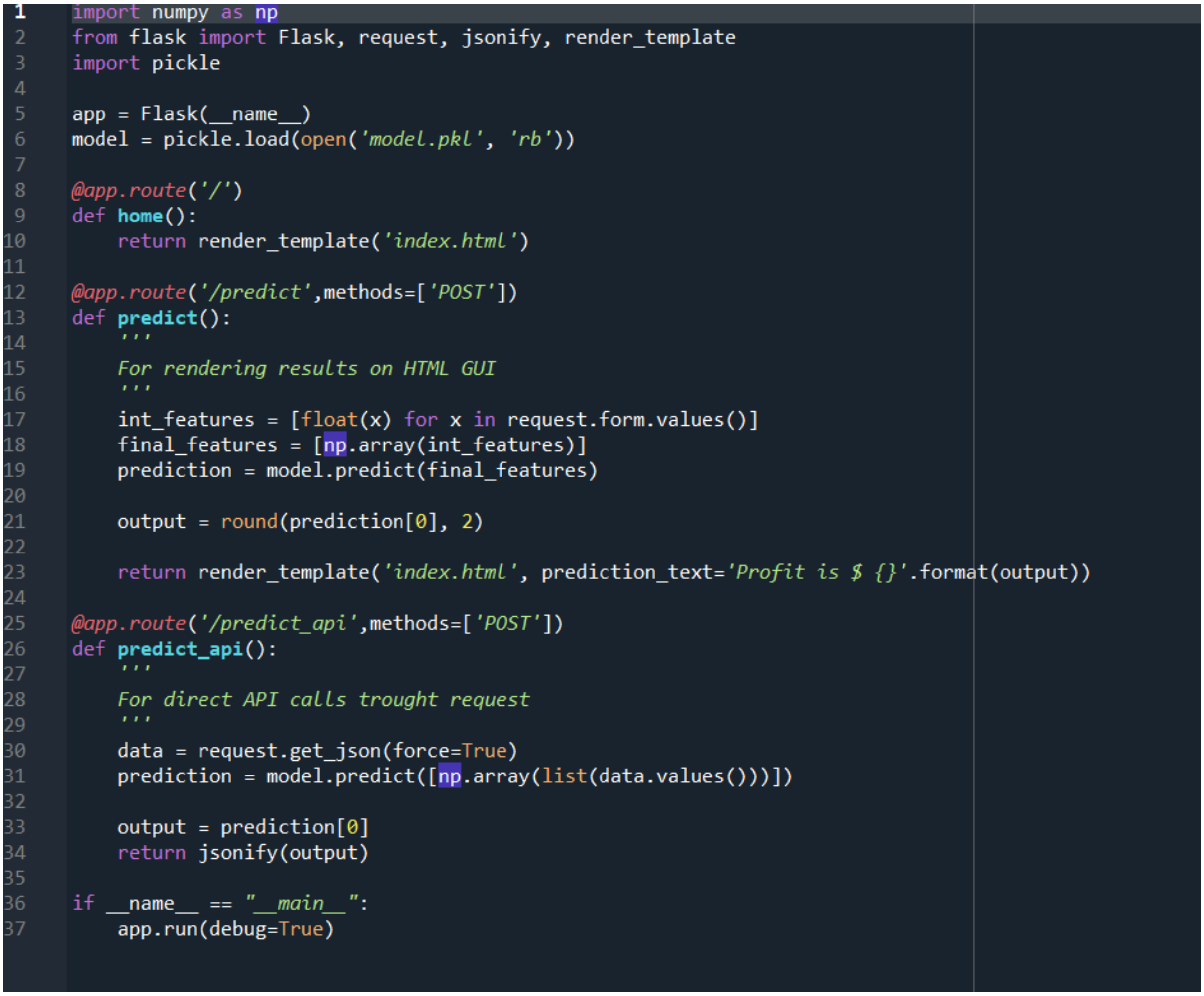
SUBMITTED TO: Data Glacier

- Code for model building:



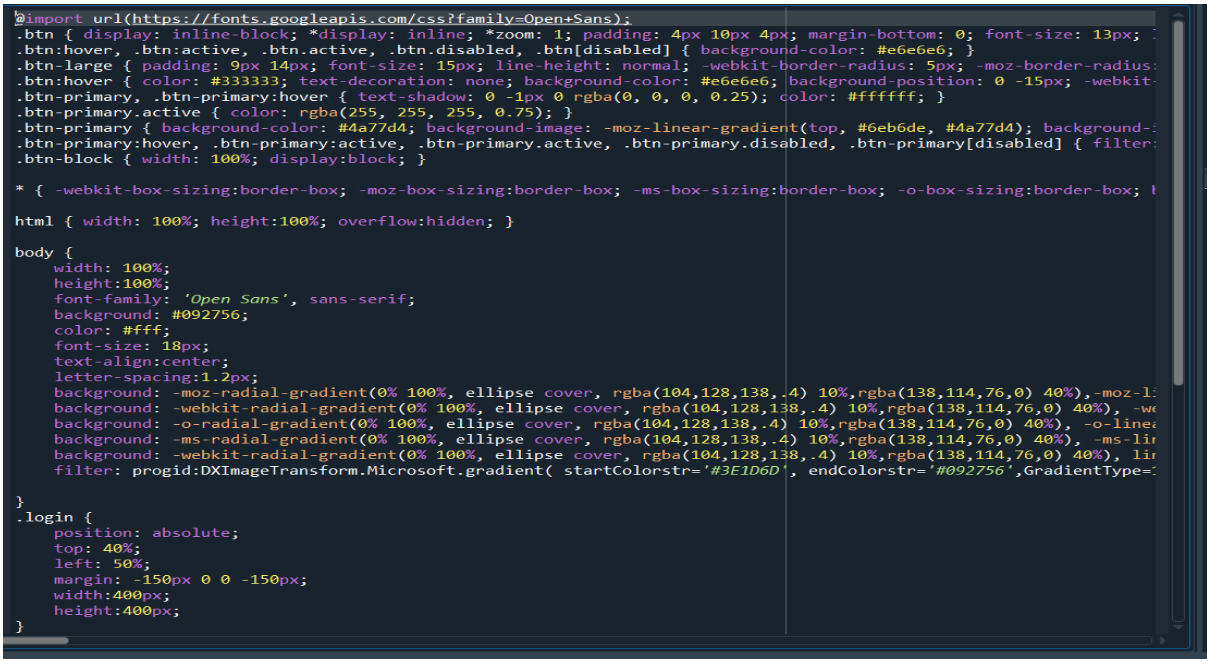
```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle

dataset = pd.read_csv('50_Startups.csv')
dataset.head()

dataset.isna().sum()

X = dataset.iloc[:, :4]
X.head()
X["State"].value_counts()
def convert_to_int(word):
    word_dict = {'New York':1, 'California':2, 'Florida':3}
    return word_dict[word]

X['State'] = X['State'].apply(lambda x : convert_to_int(x))
X.shape
y = dataset.iloc[:, -1]

dataset.info()

#Splitting Training and Test Set
#Since we have a very small dataset, we will train our model with all availabe 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([[165349.20, 151377.59, 471784.10,2]]))
```

- Code for deployment:

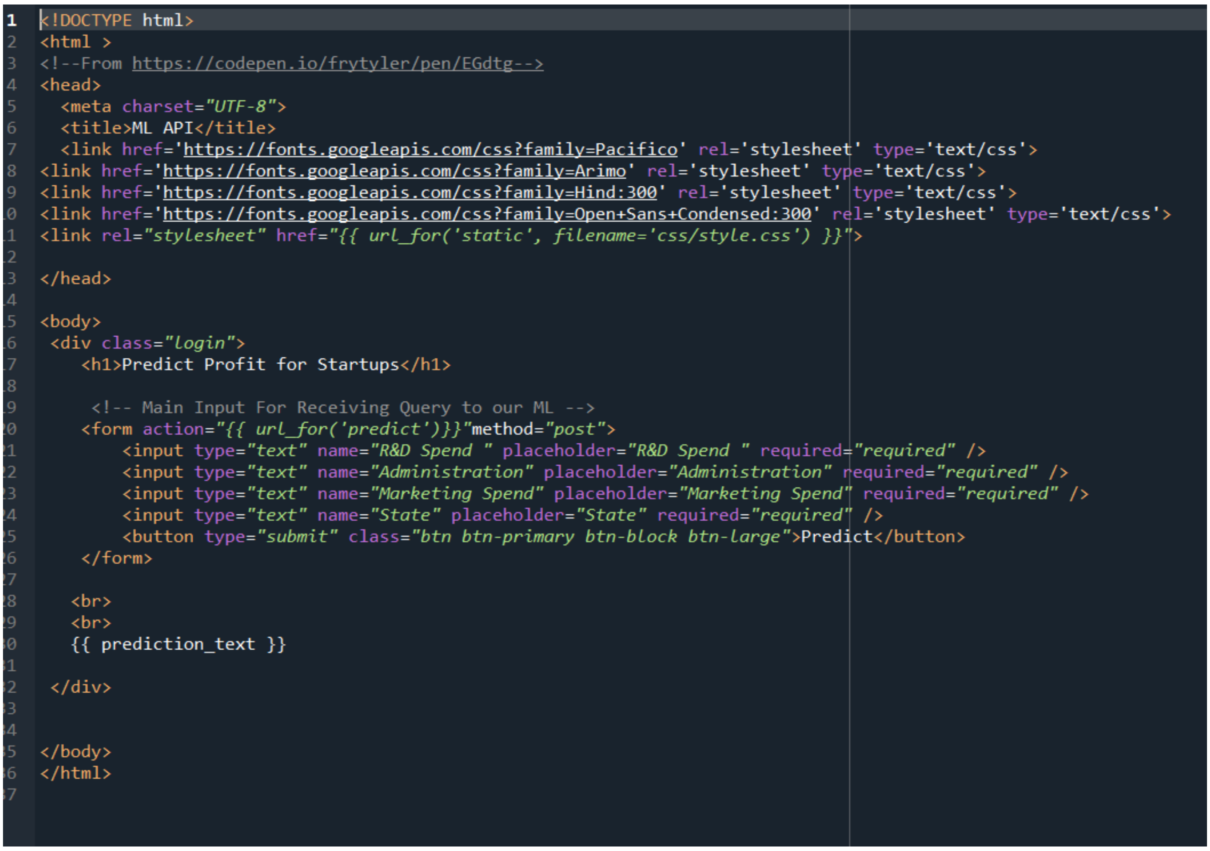


```python
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

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
    '''
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text='Profit is $ {}'.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)
```

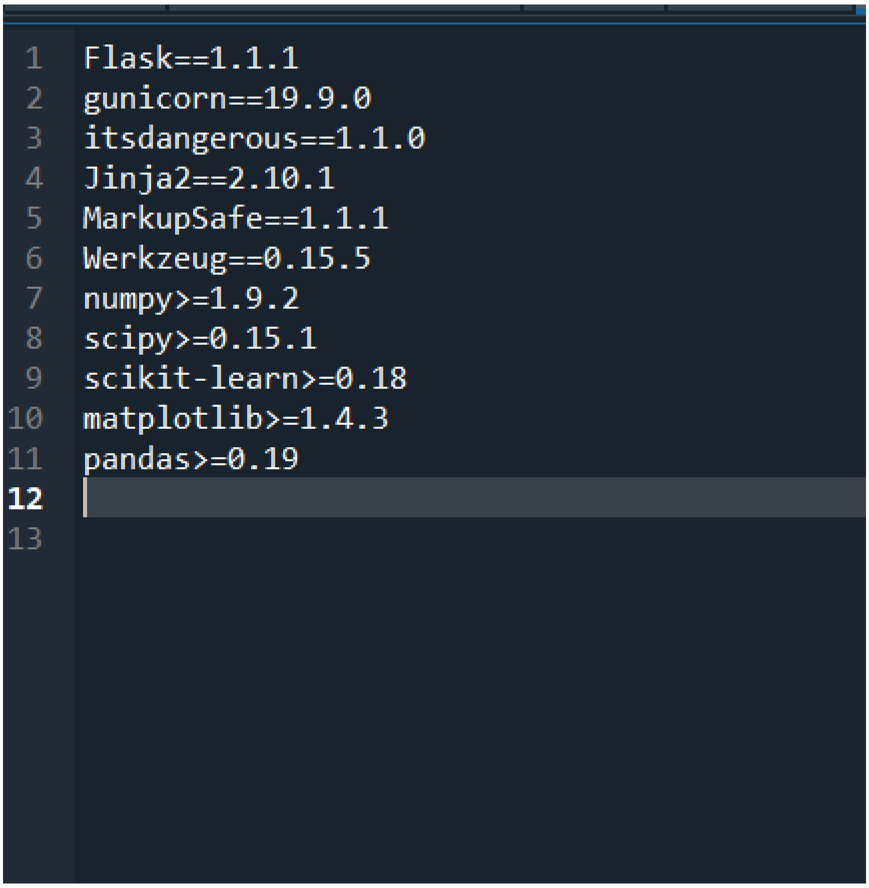
- Code for style, Index and requirement:



```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;
.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
.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-
.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-
.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter
.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;

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-l
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -we
    background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-line
    background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-lin
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), lin
    filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=

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

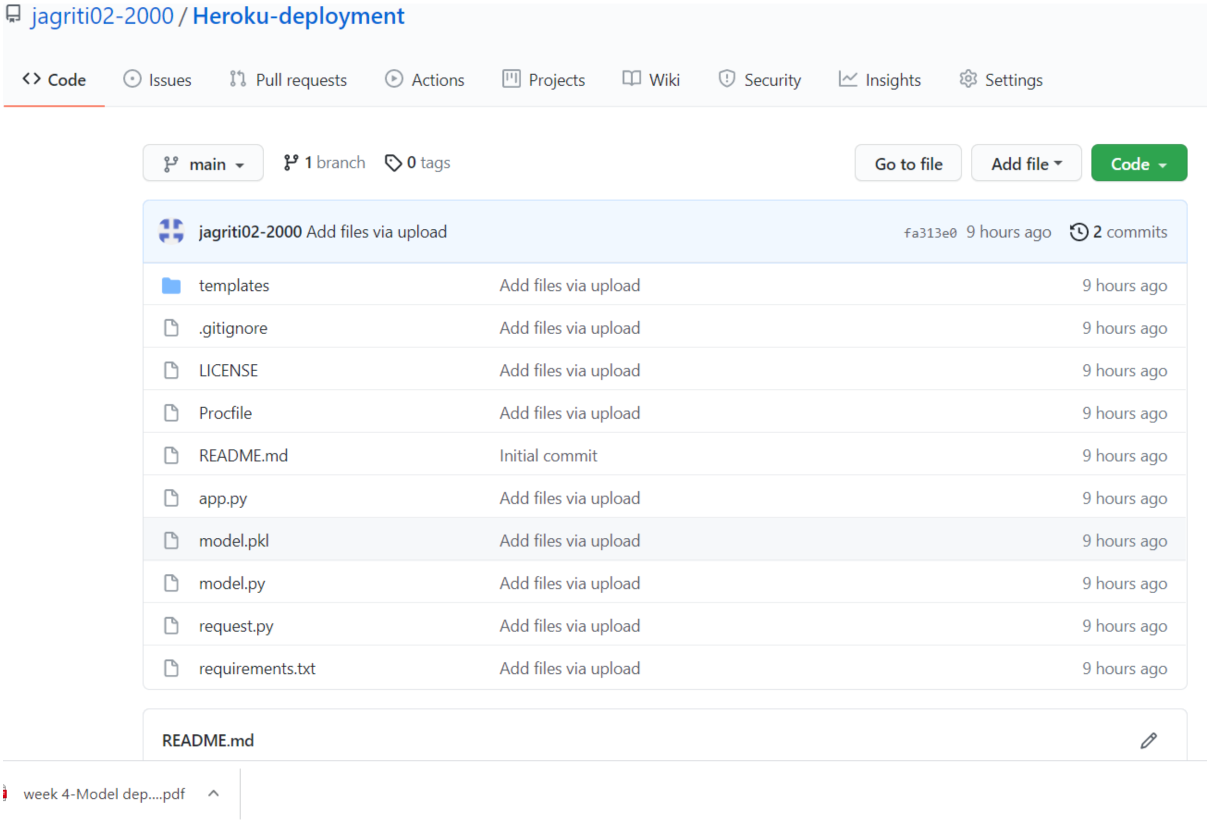


```html
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' 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 Profit for Startups</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
        <input type="text" name="R&D Spend " placeholder="R&D Spend " required="required" />
        <input type="text" name="Administration" placeholder="Administration" required="required" />
        <input type="text" name="Marketing Spend" placeholder="Marketing Spend" required="required" />
        <input type="text" name="State" placeholder="State" required="required" />
        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
    </form>

   <br>
   <br>
   {{ prediction_text }}

 </div>


</body>
</html>
```



```
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

```

- Adding files to GitHub

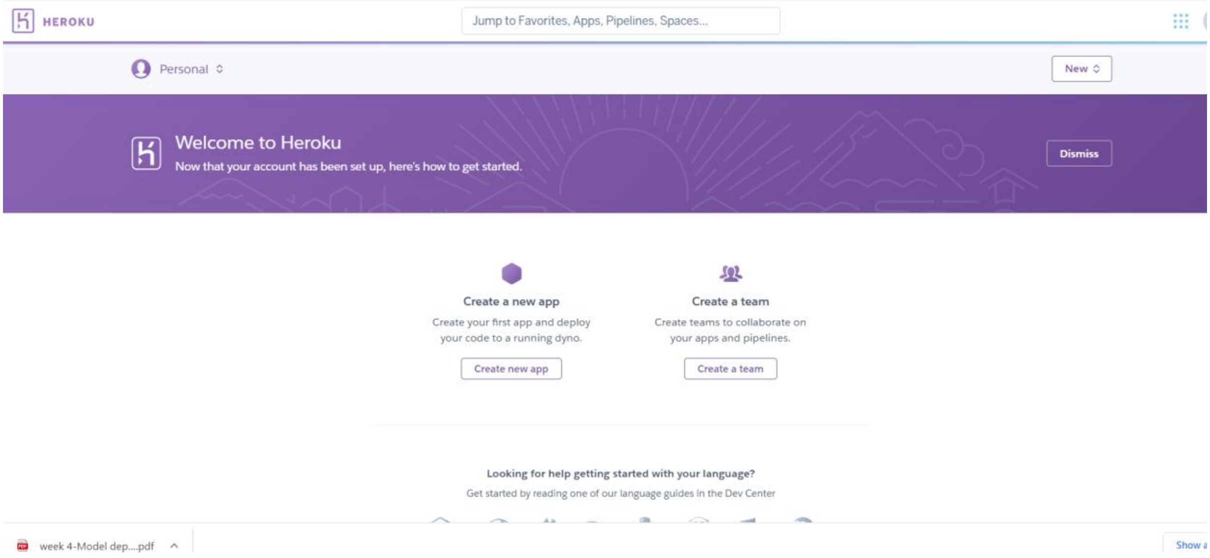


jagriti02-2000 Add files via upload fa313e0 9 hours ago 2 commits

templates	Add files via upload	9 hours ago
.gitignore	Add files via upload	9 hours ago
LICENSE	Add files via upload	9 hours ago
Procfile	Add files via upload	9 hours ago
README.md	Initial commit	9 hours ago
app.py	Add files via upload	9 hours ago
model.pkl	Add files via upload	9 hours ago
model.py	Add files via upload	9 hours ago
request.py	Add files via upload	9 hours ago
requirements.txt	Add files via upload	9 hours ago

README.md

- Login into Heroku



Welcome to Heroku

Now that your account has been set up, here's how to get started.

Create a new app

Create your first app and deploy your code to a running dyno.

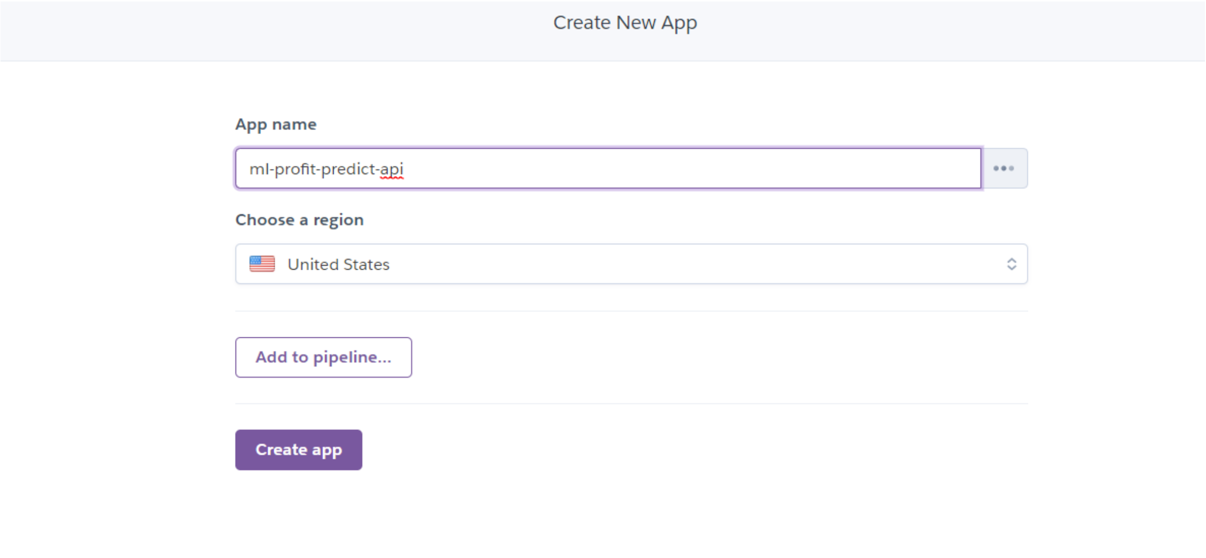
Create a team

Create teams to collaborate on your apps and pipelines.

Looking for help getting started with your language?

Get started by reading one of our language guides in the Dev Center

- Creating a new app



Create New App

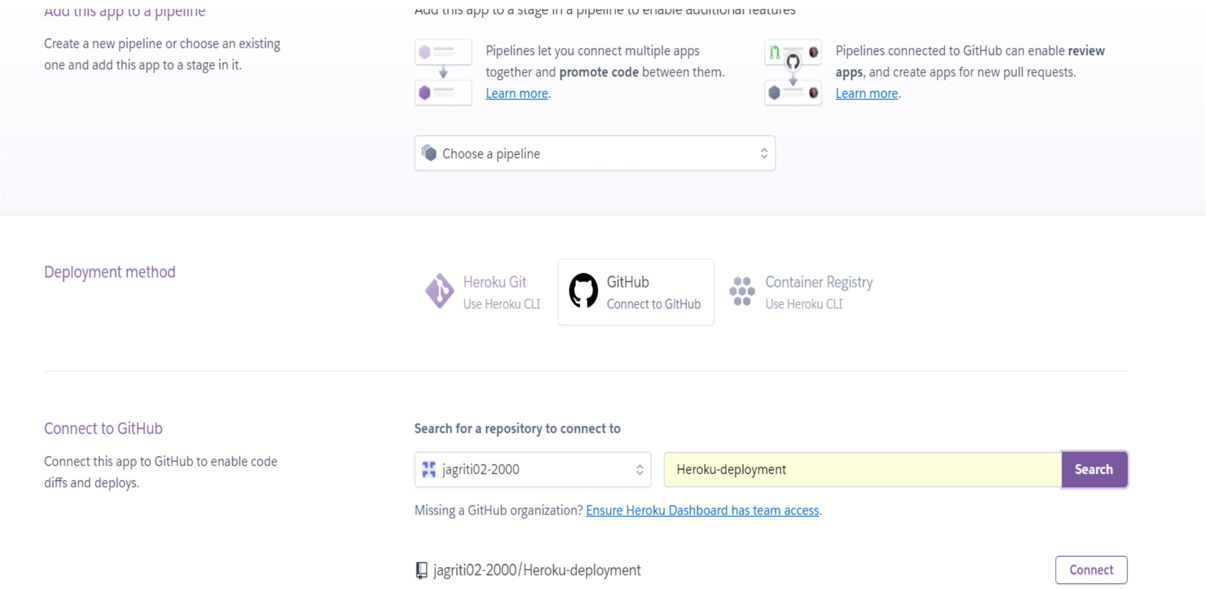
App name

ml-profit-predict-api

Choose a region

United States

- Connecting with GitHub



Add this app to a pipeline

Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and **promote code** between them. Learn more.

Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. Learn more.

Choose a pipeline

Deployment method

Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.

Search for a repository to connect to

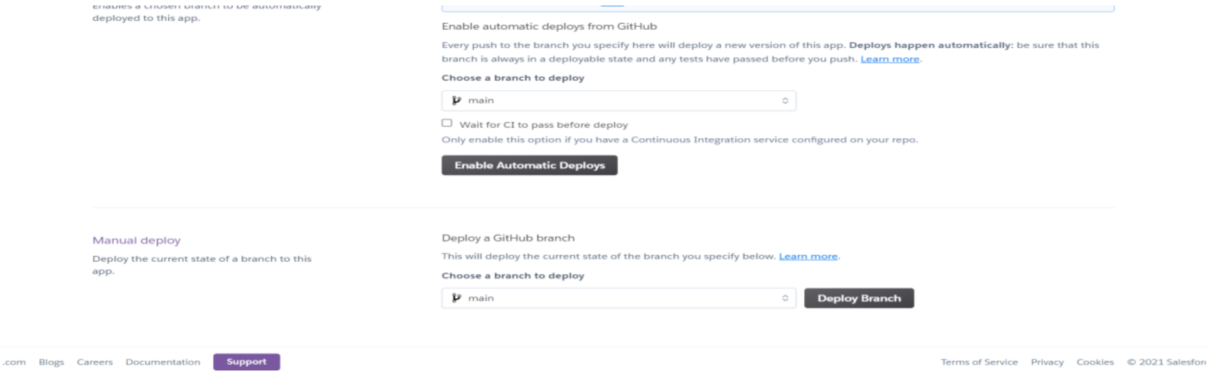
jagriti02-2000

Heroku-deployment

Missing a GitHub organization? Ensure Heroku Dashboard has team access.

jagriti02-2000/Heroku-deployment

- Manual Deployment



Enables a chosen branch to be automatically deployed to this app.

Enable automatic deploys from GitHub

Every push to the branch you specify here will deploy a new version of this app. **Deploys happen automatically:** be sure that this branch is always in a deployable state and any tests have passed before you push. Learn more.

Choose a branch to deploy

main

□ Wait for CI to pass before deploy

Only enable this option if you have a Continuous Integration service configured on your repo.

Manual deploy

Deploy the current state of a branch to this app.

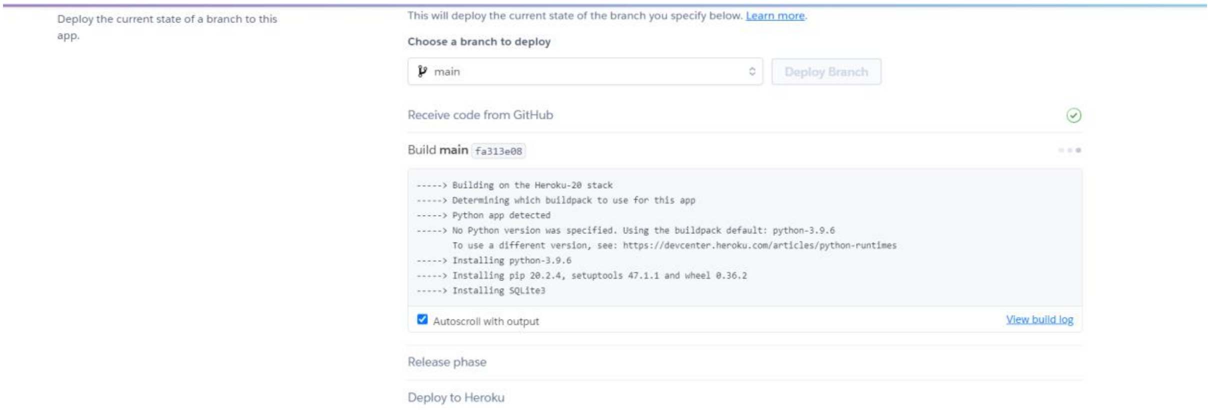
Deploy a GitHub branch

This will deploy the current state of the branch you specify below. Learn more.

Choose a branch to deploy

main

- Model Deployment:



Deploy the current state of a branch to this app.

This will deploy the current state of the branch you specify below. Learn more.

Choose a branch to deploy

main

Receive code from GitHub ✓

Build main fa313e08

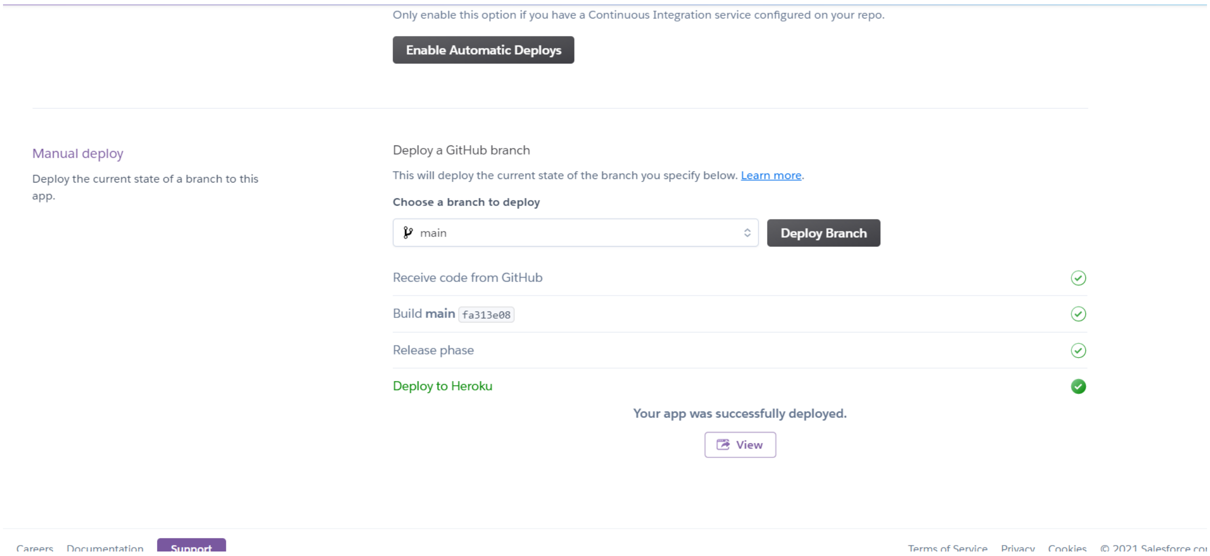
```
-----> Building on the Heroku-20 stack
-----> Determining which buildpack to use for this app
-----> Python app detected
-----> No Python version was specified. Using the buildpack default: python-3.9.6
       To use a different version, see: https://devcenter.heroku.com/articles/python-runtimes
-----> Installing python-3.9.6
-----> Installing pip 20.2.4, setuptools 47.1.1 and wheel 0.36.2
-----> Installing SQLite3
```

☑ Autoscroll with output

Release phase

Deploy to Heroku

- Model Deployed



Only enable this option if you have a Continuous Integration service configured on your repo.

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. Learn more.

Choose a branch to deploy

main

Receive code from GitHub ✓

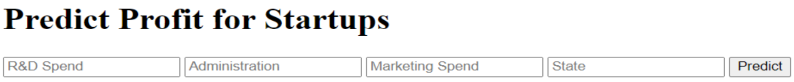
Build main fa313e08 ✓

Release phase ✓

Deploy to Heroku ✓

Your app was successfully deployed.

- Deployed Model



Predict Profit for Startups

- Link to the model: https://ml-profit-predict-api.herokuapp.com/