Week5: UI with Docker and CI/CD

Name: Madoka Fujii

Batch Code: LISUM39

Submission Date: 12/04/2024

Submission to: Data Glacier

Step1: Select any toy data

I found Loan Prediction dataset

```
df_train.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	U
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	U
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	U
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	U

There are originally 13 variables.
Then, I have adjusted the dataset for building a logistic regression model.
-Omitted Loan_Status variable
-Created One-Hot encoding
-Mapped boolean answer to 1/0.
-Imputed missing value with mean
-Adjusted some row (eg. converted from 3+ to 3 in Dependents variable)

After the adjustments, the variable became 13. (it was the same as before but the contents were changed.)

```
29]: train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Married                 614 non-null    int64
 1   Dependents              614 non-null    int32
 2   Education               614 non-null    int64
 3   Self_Employed           614 non-null    int64
 4   ApplicantIncome         614 non-null    int64
 5   CoapplicantIncome       614 non-null    float64
 6   LoanAmount              614 non-null    float64
 7   Loan_Amount_Term        614 non-null    float64
 8   Credit_History          614 non-null    float64
 9   Loan_Status             614 non-null    int64
 10  Gender_Female           614 non-null    bool
 11  Gender_Male             614 non-null    bool
 12  Property_Area_Rural     614 non-null    bool
 13  Property_Area_Semiurban 614 non-null    bool
 14  Property_Area_Urban     614 non-null    bool
dtypes: bool(5), float64(4), int32(1), int64(5)
memory usage: 48.7 KB
```

Model Building

```
31]: #Spliting the dataset into features and target
x=train.drop('Loan_Status',axis=1)
y=train['Loan_Status']
```

Step2: Save the model

```
import pickle

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

# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[1, 2, 1, 0, 6000, 500, 100.01, 360, 1.0, 0, 1, 0, 0, 1]]))
```

```
[1]
```

Using pickle, saved the model to deploy on Flask.

Step3: Deploy the model on Flask

In PyCharm, I created app.py file.

Data Mapping between appy.py and index.html.

app.py 2 | index.html

C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week4 > app.py > ...

```python
import numpy as np
from flask import Flask, request, 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
    '''
    form_data = request.form

    # conver strings to int or float as need
    applicant_income = float(form_data['applicantIncome'])
    coapplicant_income = float(form_data['coapplicantIncome'])
    loan_amount = float(form_data['loanamount'])
    loan_amount_term = float(form_data['loanamountterm'])
    credit_history = 1 if form_data['credit_history'] == 'Yes' else 0


    #One-Hot encoding
    married = 1 if form_data['married'] == 'Yes' else 0  # married_yes -> 1
    dependents = form_data['dependents']
    dependents_value = 3 if dependents == '3+' else int(dependents)
    education = 1 if form_data['education'] == 'Yes' else 0  # education_yes -> 1
    self_employed = 1 if form_data['self_employed'] == 'Yes' else 0  # self_employed_yes -> 1

    gender_male = 1 if form_data['gender_male'] == 'Yes' else 0  # Gender Male -> 1
    gender_female = 1 if form_data['gender_male'] == 'No' else 0  # Gender Female -> 1

    # encoding for Property Area
    property_area = form_data['property_area']
```

app.py 2 | index.html

C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week4 > app.py > predict

```python
def predict():
    # encoding for Property Area
    property_area = form_data['property_area']
    rural = 1 if property_area == 'Rural' else 0
    semiurban = 1 if property_area == 'Semiurban' else 0
    urban = 1 if property_area == 'Urban' else 0

    # put array the future characteristics
    final_features = np.array([
        applicant_income,
        coapplicant_income,
        loan_amount,
        loan_amount_term,
        credit_history,
        married,
        dependents_value,
        education,
        self_employed,
        gender_male,
        gender_female,
        rural,
        semiurban,
        urban
    ]).reshape(1, -1)


    #predict implimentaion
    prediction = model.predict(final_features)

    output = "Yes" if prediction[0] == 1 else "No"

    return render_template('index.html', prediction_text='Loan Acceptance {}'.format(output))

if __name__ == "__main__":
    app.run(port=5000, debug=True)
```

Created index.html file.
Data Mapping between appy.py and index.html.

app.py 2 index.html

C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week4 > templates > index.html > ...

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>ML Logistics Reg 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>Loan Prediction</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">


    <label for="gender_male">Are you Male?:</label>
        <select name="gender_male" id="gender_male">
            <option value="Yes">Yes</option>
            <option value="No">No</option>
        </select>
    <br>
    <br>
    <label for="married">Married?:</label>
        <select name="married" id="married">
            <option value="Yes">Yes</option>
            <option value="No">No</option>
        </select>
    <br>

    <label for="dependents">How many Dependents?:</label>
        <select name="dependents" id="dependents">
            <option value="0">0</option>
```

app.py 2 index.html

C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week4 > templates > index.html > ...

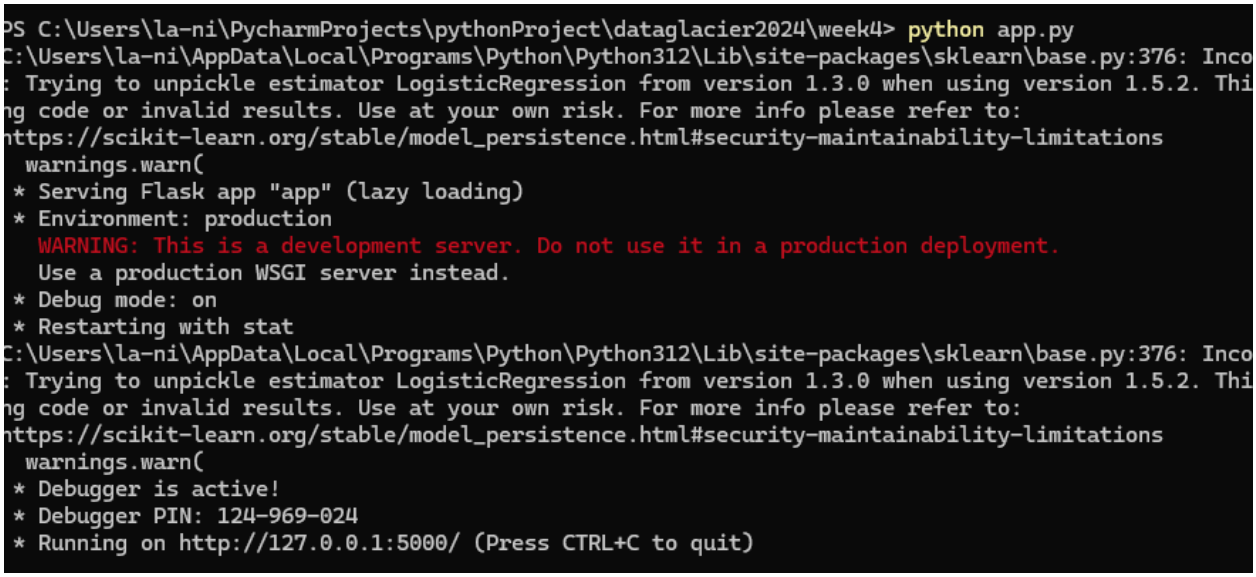
```
 2  <html >
14  <body>
15   <div class="login">
19      <form action="{{ url_for('predict')}}"method="post">
36      <label for="dependents">How many Dependents?:</label>
37          <select name="dependents" id="dependents">
38              <option value="0">0</option>
39              <option value="1">1</option>
40              <option value="2">2</option>
41              <option value="3">3+</option>
42          </select>
43      <br>
44
45      <label for="education">Education is graduate?:</label>
46          <select name="education" id="education">
47              <option value="Yes">Yes</option>
48              <option value="No">No</option>
49          </select>
50      <br>
51
52      <label for="self_employed">Self Employed?:</label>
53          <select name="self_employed" id="self_employed">
54              <option value="Yes">Yes</option>
55              <option value="No">No</option>
56          </select>
57      <br>
58
59
60      <label for="applicantIncome">Applicant Income:</label>
61      <input type="text" name="applicantIncome" placeholder="eg: 2500" required="required" />
62
63      <label for="coapplicantIncome">Coapplicant Income:</label>
64      <input type="text" name="coapplicantIncome" placeholder="eg: 1500.54" required="required" />
65
66      <label for="bloanamount">Loan Amount:</label>
67      <input type="text" name="loanamount" placeholder="eg: 120.06" required="required" />
68
69      <label for="loanamountterm">Loan Amount Term:</label>
```

app.py 2 | index.html

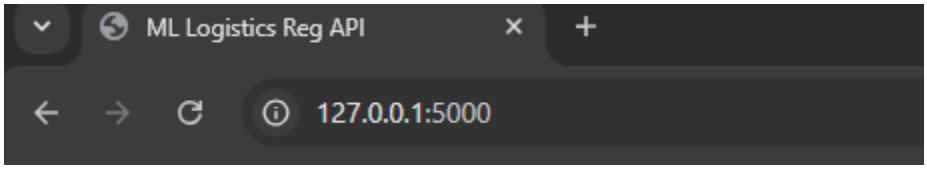
C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week4 > templates > index.html > html > body > div.login > form > label

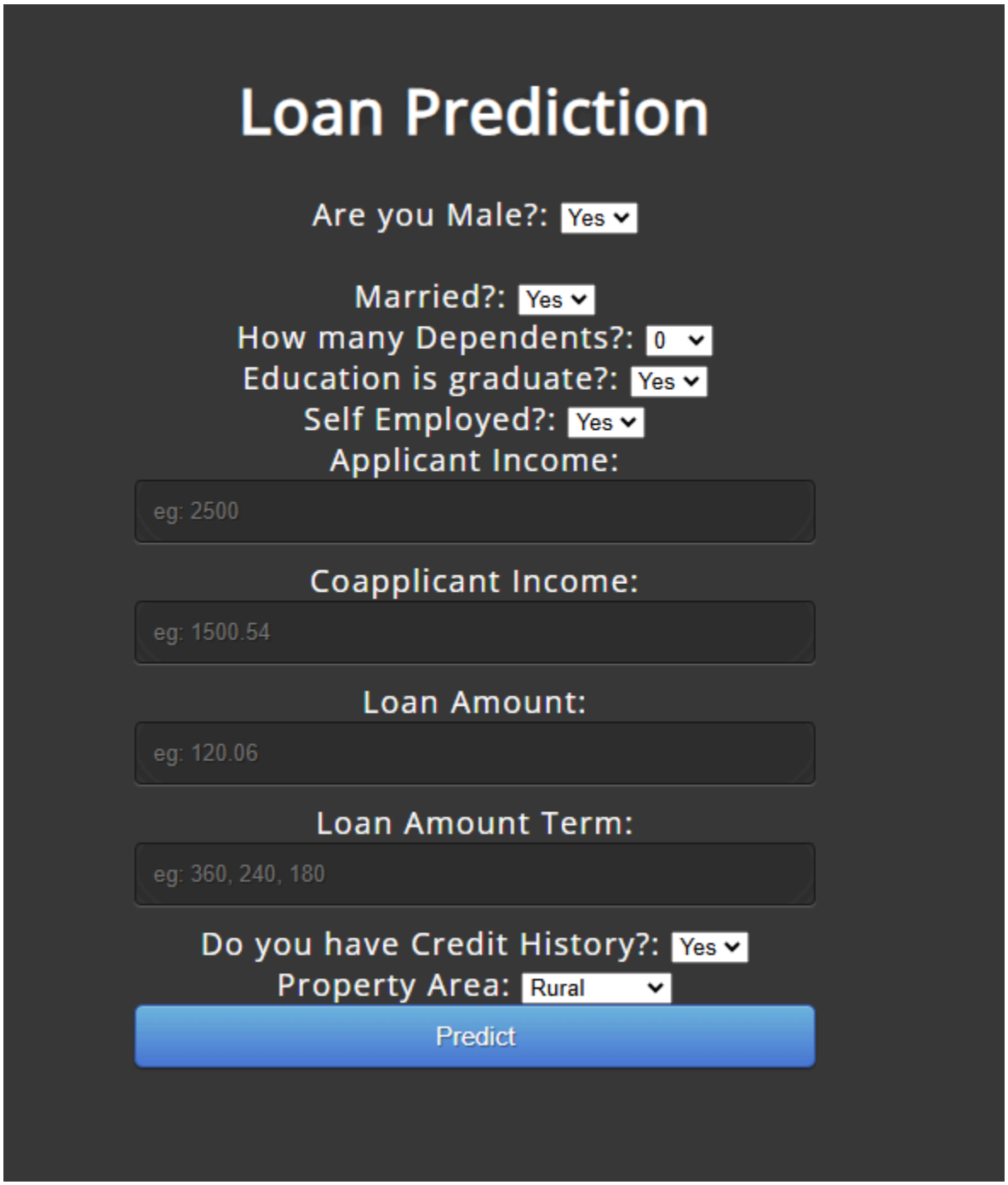
```
2   <html >
14  <body>
15   <div class="login">
19      <form action="{{ url_for('predict')}}"method="post">
65
66      <label for="bloanamount">Loan Amount:</label>
67      <input type="text" name="loanamount" placeholder="eg: 120.06" required="required" />
68
69      <label for="loanamountterm">Loan Amount Term:</label>
70      <input type="text" name="loanamountterm" placeholder="eg: 360, 240, 180" required="required" />
71
72      <label for="credit_history">Do you have Credit History?:</label>
73          <select name="credit_history" id="credit_history">
74              <option value="Yes">Yes</option>
75              <option value="No">No</option>
76          </select>
77      <br>
78
79      <label for="property_area">Property Area:</label>
80          <select name="property_area" id="property_area">
81            <option value="Rural">Rural</option>
82            <option value="Semiurban">Semiurban</option>
83            <option value="Urban">Urban</option>
84          </select>
85      <br>
86
87          <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
88      </form>
89
90     <br>
91
92     {{ prediction_text }}
93
94   </div>
95   <!--img src="/static/images/Original.svg" style="width: 400px;position: absolute;bottom: 10px;left: 10px;" alt="Company Logo"/-->
96
97  </body>
```

Run app.py in Command prompt.

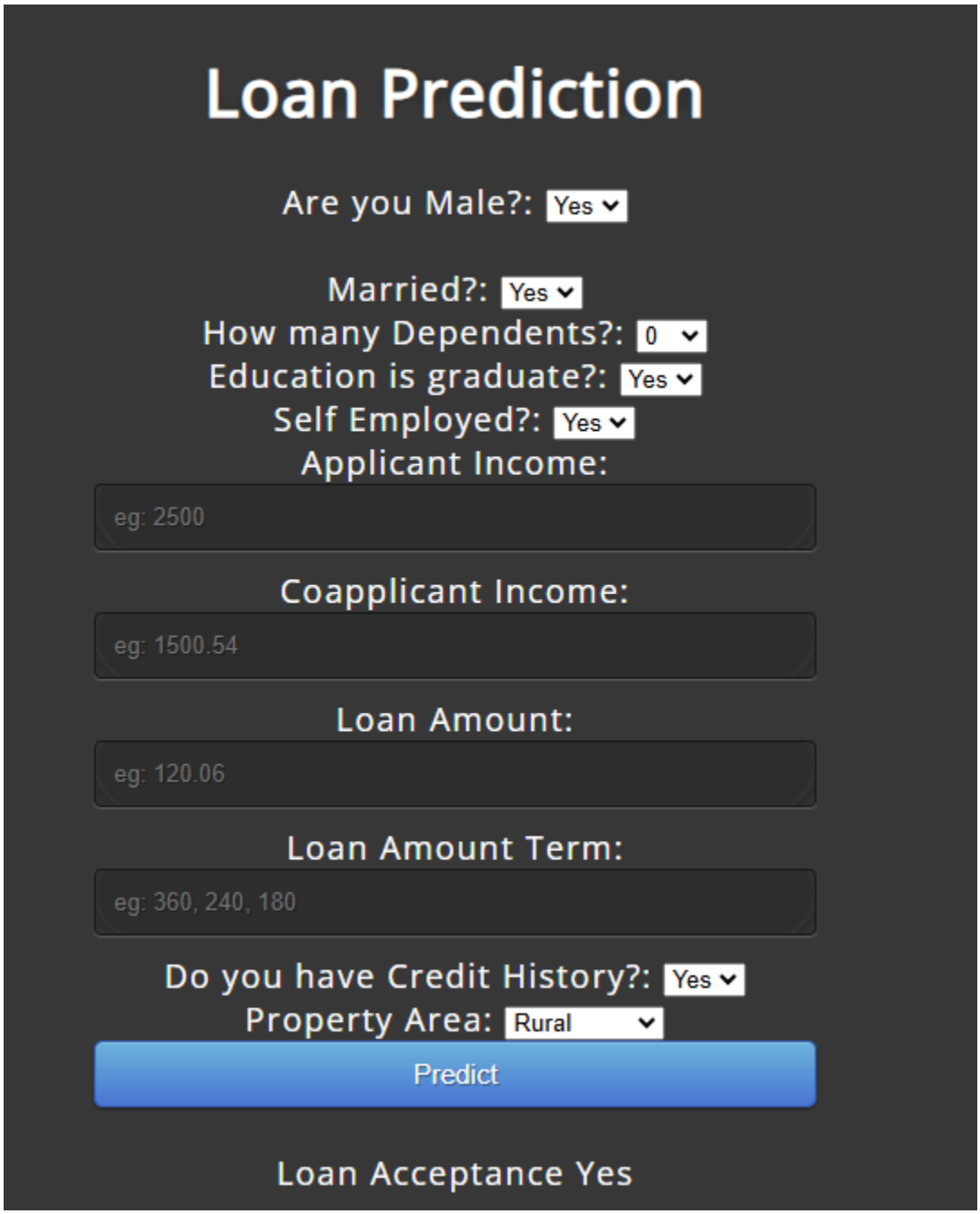


Open the link in the new browser.





Fill in the format and then press the button.
The result appears.



Step4: Add requirements.txt and Procfile

requirements.txt

```
altair==4.2.2
attrs==22.2.0
beautifulsoup4==4.12.2
bleach==6.0.0
blinker==1.6.2
build==0.9.0
cachetools==5.3.0
certifi==2022.12.7
charset-normalizer==3.1.0
click==8.1.3
colorama==0.4.6
decorator==5.1.1
docutils==0.20.1
entrypoints==0.4
Flask==2.3.2
gitdb==4.0.10
GitPython==3.1.31
idna==3.4
importlib-metadata==6.2.0
itsdangerous==2.1.2
jaraco.classes==3.2.3
Jinja2==3.1.2
joblib==1.3.1
jsonschema==4.17.3
keyring==24.2.0
lxml==4.9.2
markdown-it-py==2.2.0
MarkupSafe==2.1.2
mdurl==0.1.2
more-itertools==9.1.0
numpy==1.24.2
```

Procfile

```
1   web: gunicorn app:app
```

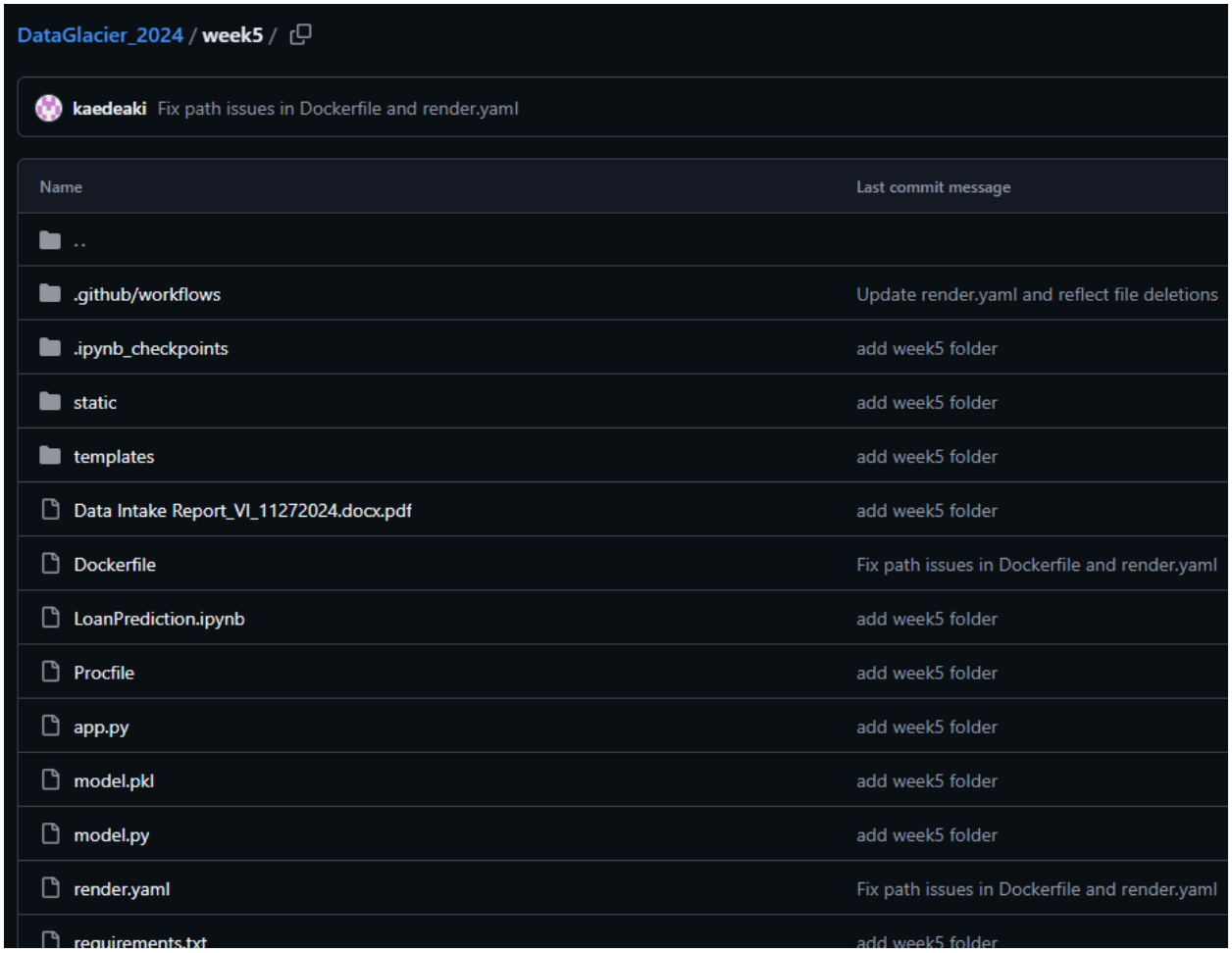
Step5: Create Dockerfile and build image

Dockerfile

C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier202

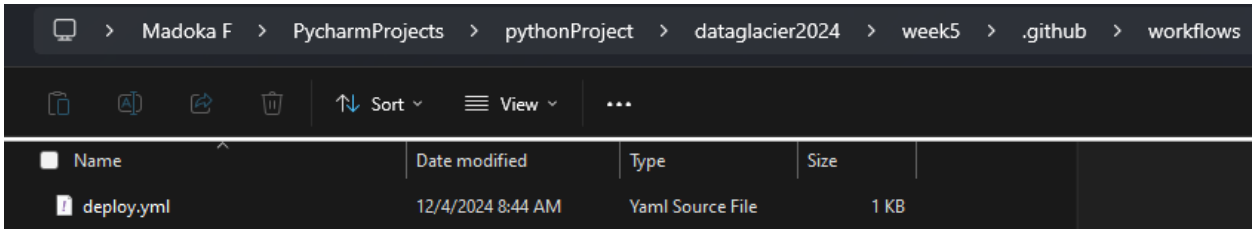
```
# assign base image
FROM python:3.9-slim

# setting the directory
WORKDIR /app

# install and copy
COPY week5/requirements.txt requirements.txt
RUN pip install -r requirements.txt

# copy application file
COPY week5/ .

# open port
EXPOSE 5000

# run application
CMD ["python", "app.py"]

```

Step6: Push to GitHub

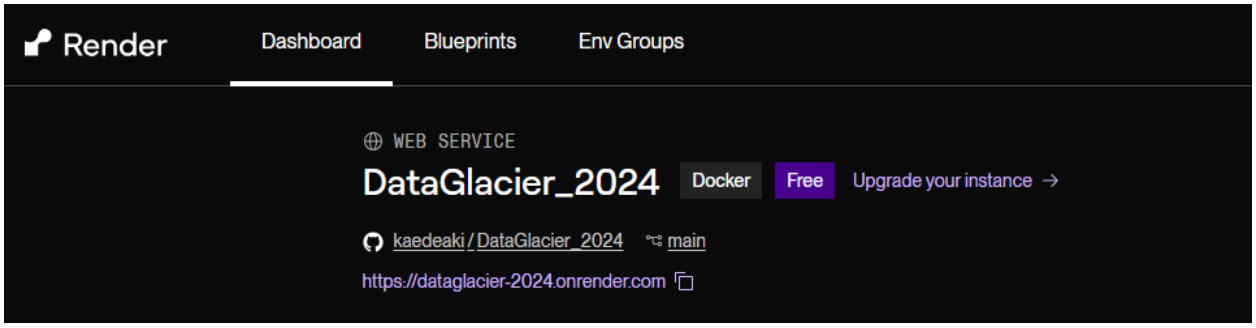


Step7: Create CI/CD

In local repository, create the folder and yml file for CI/CD



Step8 : Create a new project in render.com



Step9 : Create render.yaml in local directory and adjust Dockerfile to render to reach the subfile as week5 in GItHub

render.yaml

C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week5 > render.yaml

```yaml
services:
  - type: web
    name: DataGlacier_UI
    env: docker
    branch: main
    buildCommand: docker build -t your-app -f week5/Dockerfile .
    startCommand: python app.py
```

Dockerfile

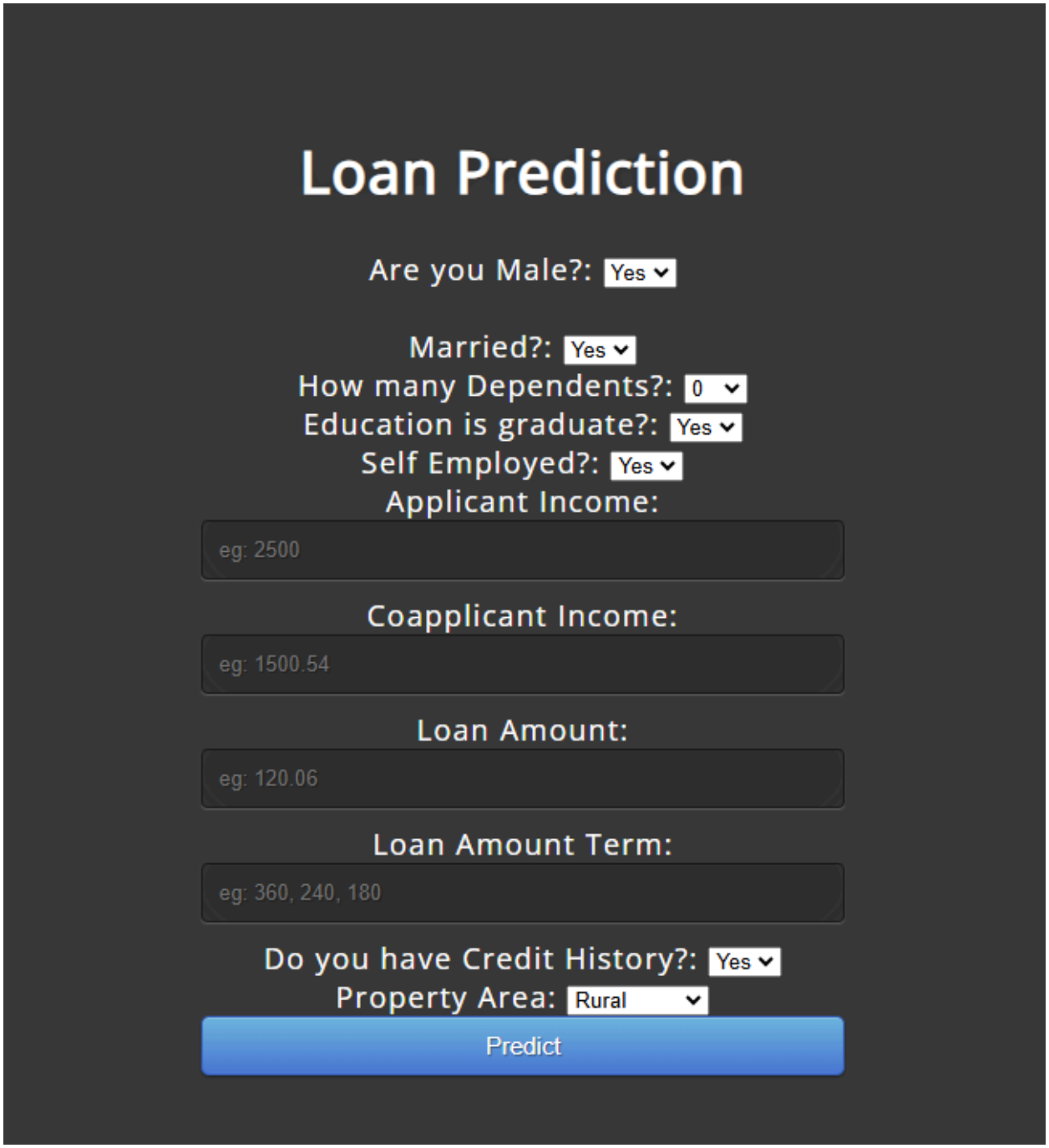
C: > Users > la-ni > PycharmProjects > pythonProject > dataglacier2024 > week5 > Dockerfile >

```dockerfile
# assign base image
FROM python:3.9-slim

# setting the directory
WORKDIR /app

# install and copy
COPY week5/requirements.txt requirements.txt
RUN pip install -r requirements.txt

# copy application file
COPY week5/ .

# open port
EXPOSE 5000

# run application
CMD ["python", "app.py"]
```

Then, push to GitHub

Step10 : Make sure the deployment is LIVE and test if the web works

https://dataglacier-2024.onrender.com/



Input the variables then press predict and check if works.