# Week 4 Report
# Deployment on Flask

Name: Deplotment on Flask
Report date: 10-28-2024
Internship Batch: LISUM38
Version:<1.0>
Data intake by: Ky Dang
Data intake reviewer:
Data storage location:

**Tabular data details:**

| | |
|---|---|
| **Total number of observations** | 158 |
| **Total number of files** | 1 |
| **Total number of features** | 10 |
| **Base format of the file** | .csv |
| **Size of the data** | 9Kb |

## 1. Building Model and save

### 1.1 Data Input

In this scenario, the data will be provided by the Hyndai Heavy Industries, which is one of the world's largest ship manufacturers and builds cruise liners

The task is to help them give accurate estimates of how many crew members a ship will require.

They are currently building new ships for some customers and want you to create a model and use it to predict how many crew members the ships will need.

Description: Measurements of ship size, capacity, crew, and age for 158 cruises ships.

**The data is composed of 10 fields and 158 records. And because of its simple and cleaning, the first beginning can start with going straight into building a model from a scratch.**

```
In [1]: import pandas as pd

In [2]: df = pd.read_csv('cruise_ship_info.csv')

In [3]: df.head()

Out[3]:
```

| | Ship_name | Cruise_line | Age | Tonnage | passengers | length | cabins | passenger_density | crew |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Journey | Azamara | 6 | 30.277 | 6.94 | 5.94 | 3.55 | 42.64 | 3.55 |
| 1 | Quest | Azamara | 6 | 30.277 | 6.94 | 5.94 | 3.55 | 42.64 | 3.55 |
| 2 | Celebration | Carnival | 26 | 47.262 | 14.86 | 7.22 | 7.43 | 31.80 | 6.70 |
| 3 | Conquest | Carnival | 11 | 110.000 | 29.74 | 9.53 | 14.88 | 36.99 | 19.10 |
| 4 | Destiny | Carnival | 17 | 101.353 | 26.42 | 8.92 | 13.21 | 38.36 | 10.00 |

```
In [16]: df.shape

Out[16]: (158, 10)
```

*Figure 1: Expore data*

```
In [4]: # import essential libraries
        from sklearn.preprocessing import LabelEncoder, StandardScaler
        from sklearn.compose import ColumnTransformer
        from sklearn.pipeline import Pipeline
        from sklearn.ensemble import RandomForestClassifier

In [8]: df.columns

Out[8]: Index(['Ship_name', 'Cruise_line', 'Age', 'Tonnage', 'passengers', 'length',
               'cabins', 'passenger_density', 'crew'],
              dtype='object')

In [9]: # encoding the category column
        label_encoder = LabelEncoder()
        df['Cruise_line_encoded']=label_encoder.fit_transform(df['Cruise_line'])

In [30]: df['Cruise_line_encoded']

Out[30]: 0       0
         1       0
         2       1
         3       1
         4       1
                ..
         153    18
```

*Figure 2:Import the essential labraries and prepare formodel*

One of the important input features is the category column “Cruise_line”, which represents different cruise lines and is essential for predicting crew requirements.

This step outlines the encoding process for the category column 'Cruise_line.

## 1.2 Build and save a Model

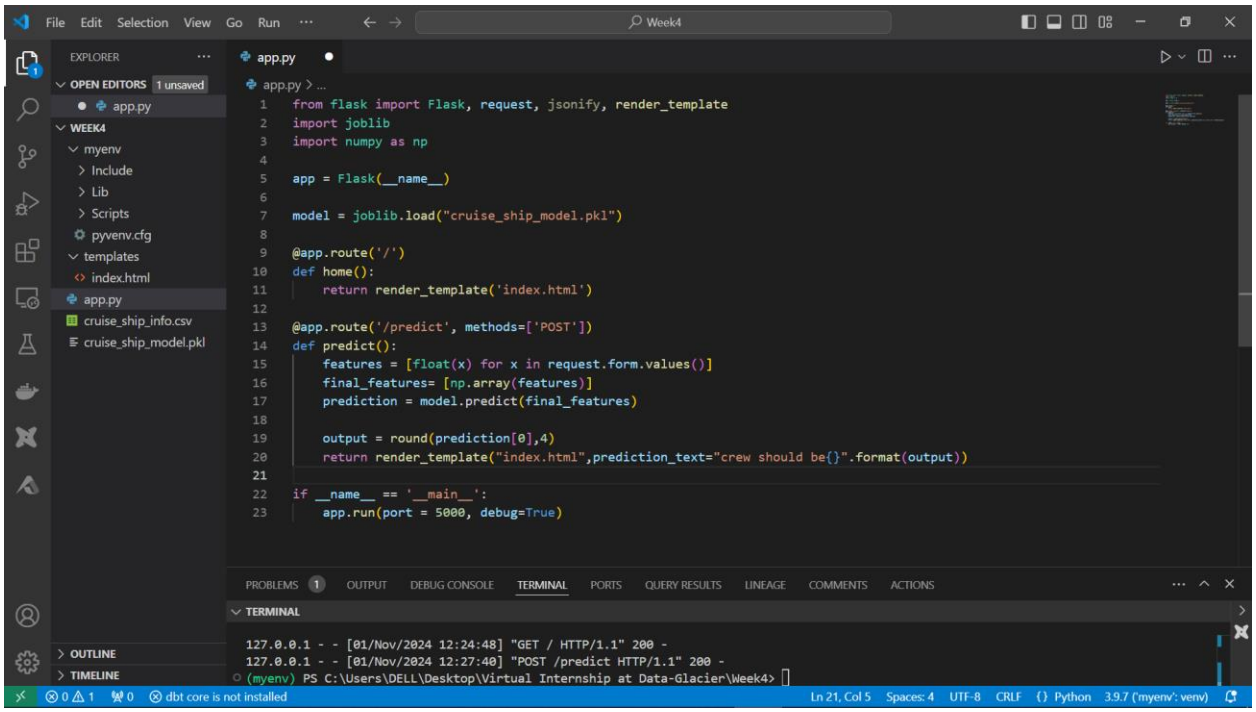
```
In [19]: X = df[['Age', 'Tonnage', 'passengers', 'length', 'cabins', 'passenger_density', 'Cruise_line_encoded']]
         y = df['crew']

In [20]: from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LinearRegression

In [22]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

In [23]: lr_model= LinearRegression()
         lr_model.fit(X_train, y_train)

Out[23]: LinearRegression()

In [28]: train = lr_model.score(X_train, y_train)
         test = lr_model.score(X_test, y_test)
         print(f"train R square:{train}")
         print(f"test R square:{test}")

         train R square:0.917357868638892
         test R square:0.9429444349278397
```

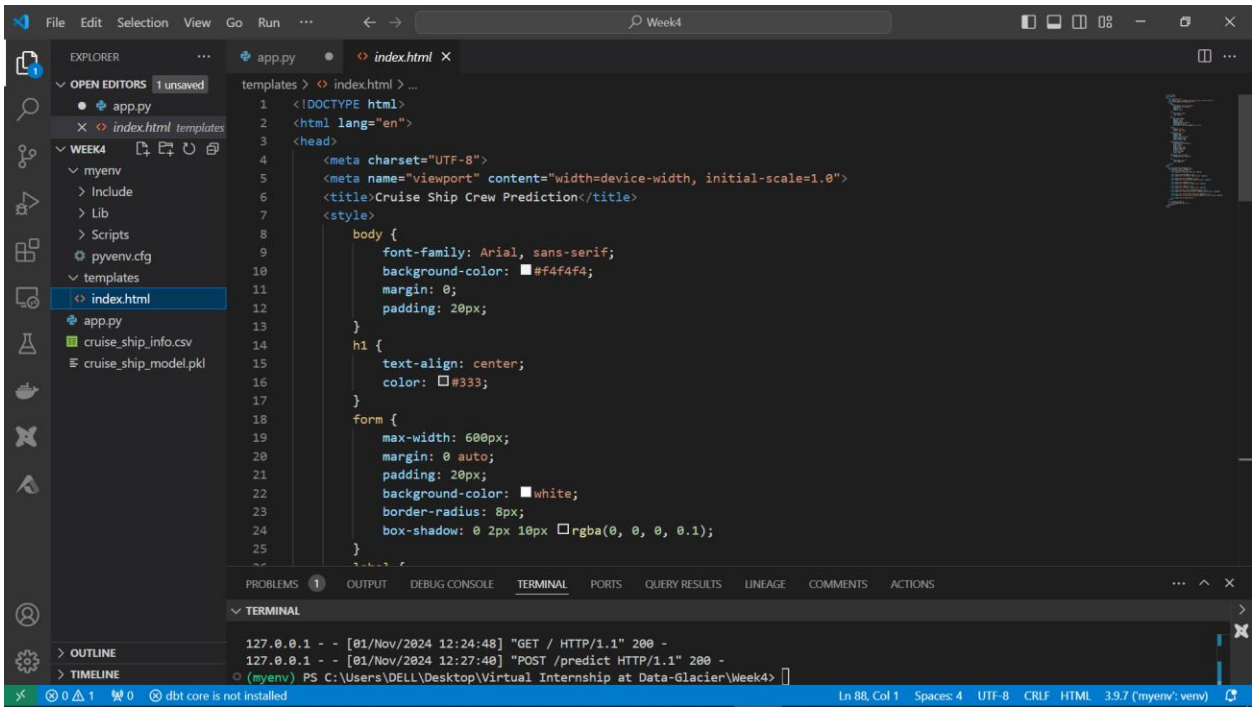
*Figure 3: Build a model*

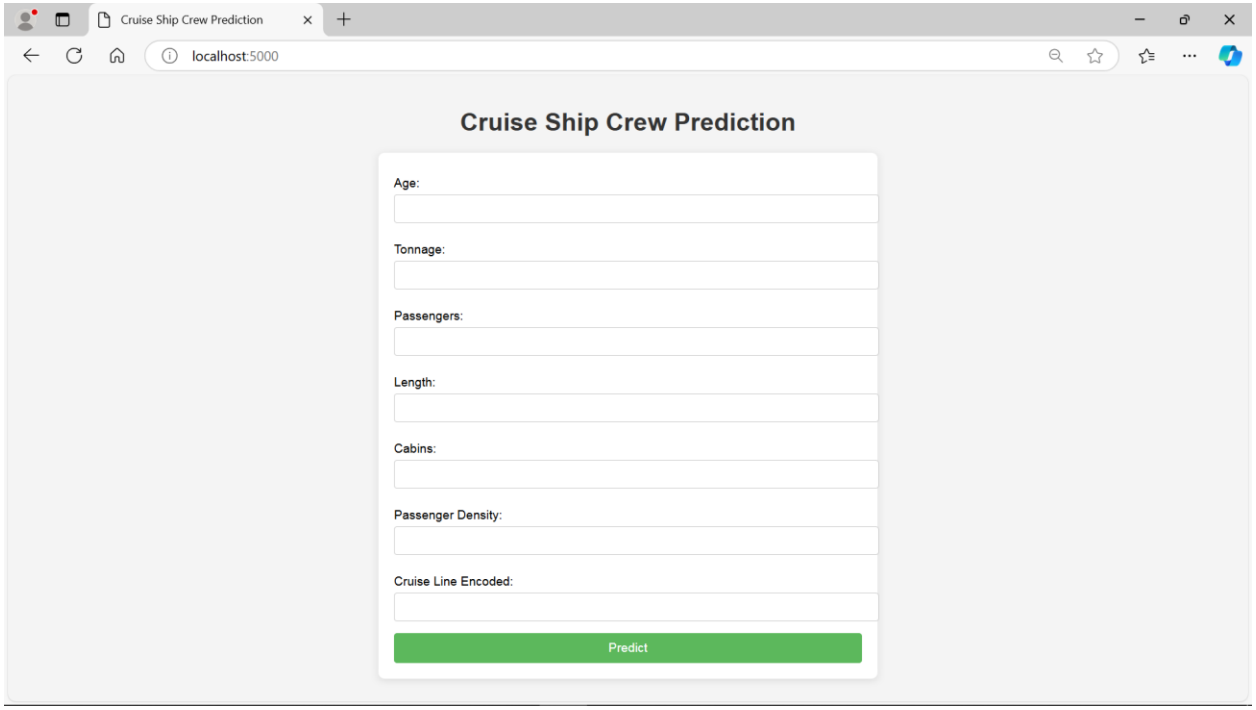
```
In [29]: import joblib
         joblib.dump(lr_model, 'cruise_ship_model.pkl')

Out[29]: ['cruise_ship_model.pkl']

In [ ]:
```

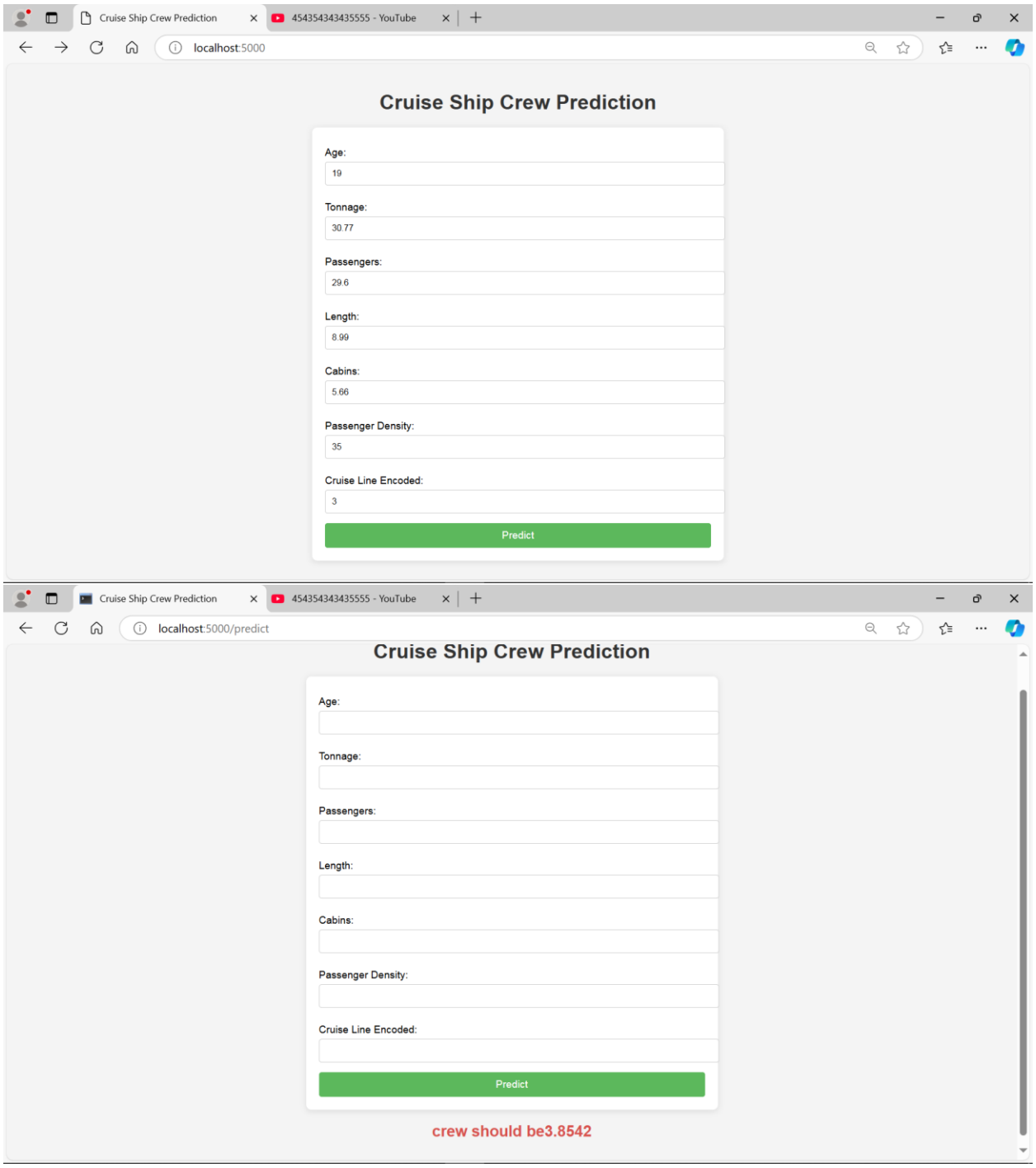
*Figure 4: Save a model*

## 2. Deploying the model on Flask




File Edit Selection View Go Run
Week4
EXPLORER
OPEN EDITORS 1 unsaved
app.py
index.html templates
WEEK4
myenv
Include
Lib
Scripts
pyvenv.cfg
templates
index.html
app.py
cruise_ship_info.csv
cruise_ship_model.pkl
OUTLINE
TIMELINE
app.py
index.html
templates > index.html > ...
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Cruise Ship Crew Prediction</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f4f4f4;
margin: 0;
padding: 20px;
}
h1 {
text-align: center;
color: #333;
}
form {
max-width: 600px;
margin: 0 auto;
padding: 20px;
background-color: white;
border-radius: 8px;
box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
}
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
QUERY RESULTS
LINEAGE
COMMENTS
ACTIONS
TERMINAL
127.0.0.1 - - [01/Nov/2024 12:24:48] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [01/Nov/2024 12:27:40] "POST /predict HTTP/1.1" 200 -
(myenv) PS C:\Users\DELL\Desktop\Virtual Internship at Data-Glacier\Week4>
dbt core is not installed
Ln 88, Col 1
Spaces: 4
UTF-8
CRLF
HTML
3.9.7 ('myenv': venv)


Cruise Ship Crew Prediction
localhost:5000
Cruise Ship Crew Prediction
Age:
Tonnage:
Passengers:
Length:
Cabins:
Passenger Density:
Cruise Line Encoded:
Predict




**Workflow:**

1. Create the virtual environment (venv) within the project folder.
2. Install Flask, Numpy, joblib and other vital libraries in the venv.
3. Write the app.py script.
4. Develop the index.html file and put it into templates folder.
5. Run the app.py file.
6. Get straight into the interface of web app (http://localhost:5000).