NAME: RITA UZOKA

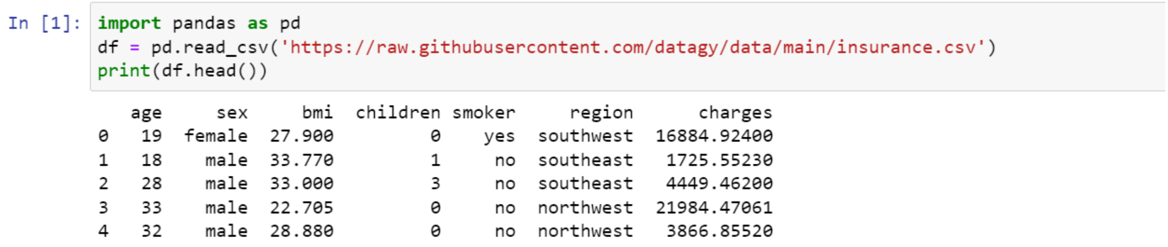
BATCH CODE: LISUM10

SUBMITION DATE: 28TH JUNE 2022

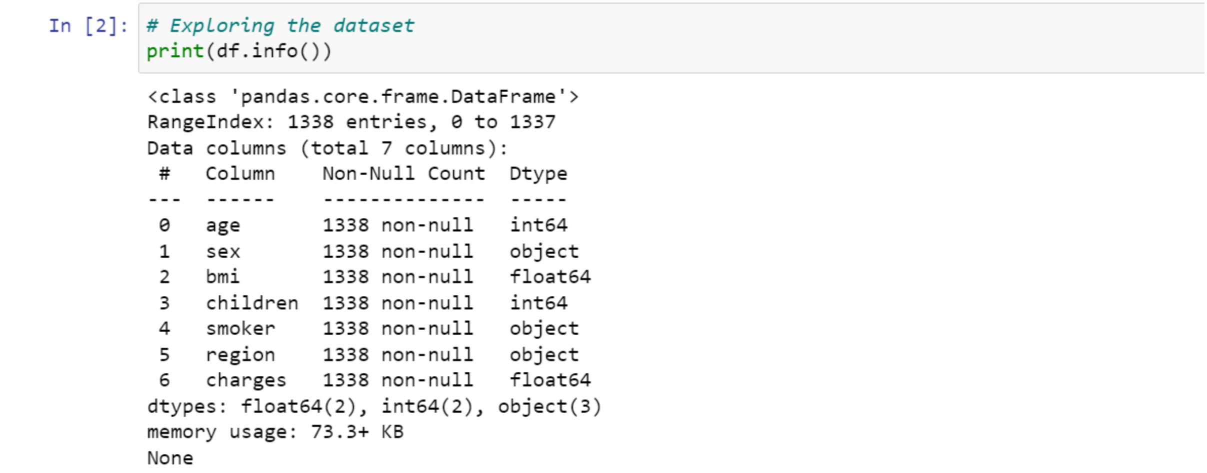
SUBMITTED TO: DATA GLACIER

WEEK 3 DATA GLACIER-DEPLOYMENT OF A MODEL ON FLASK

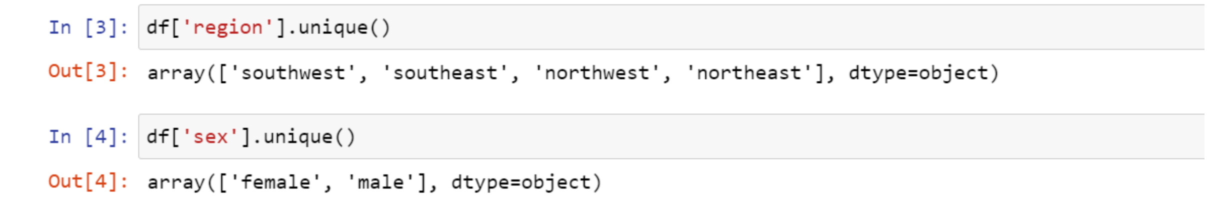
- **DUMMY DATASET(INSURANCE)**



```
In [1]: import pandas as pd
        df = pd.read_csv('https://raw.githubusercontent.com/datagy/data/main/insurance.csv')
        print(df.head())

           age     sex     bmi  children smoker     region      charges
        0   19  female  27.900         0    yes  southwest  16884.92400
        1   18    male  33.770         1     no  southeast   1725.55230
        2   28    male  33.000         3     no  southeast   4449.46200
        3   33    male  22.705         0     no  northwest  21984.47061
        4   32    male  28.880         0     no  northwest   3866.85520
```

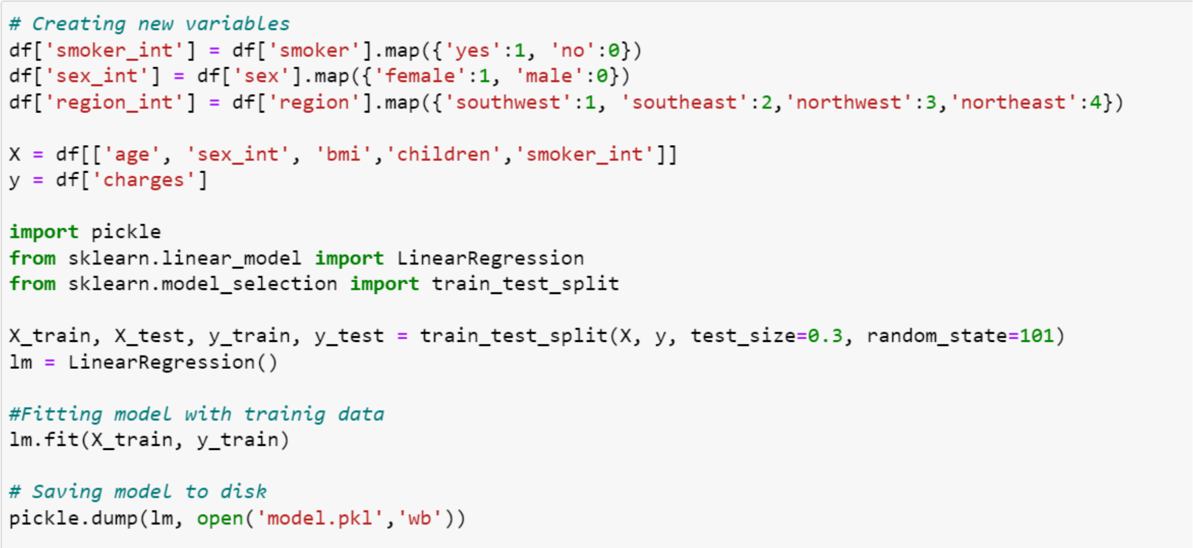


```
In [2]: # Exploring the dataset
        print(df.info())

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 1338 entries, 0 to 1337
        Data columns (total 7 columns):
         #   Column    Non-Null Count  Dtype  
        ---  ------    --------------  -----  
         0   age       1338 non-null   int64  
         1   sex       1338 non-null   object 
         2   bmi       1338 non-null   float64
         3   children  1338 non-null   int64  
         4   smoker    1338 non-null   object 
         5   region    1338 non-null   object 
         6   charges   1338 non-null   float64
        dtypes: float64(2), int64(2), object(3)
        memory usage: 73.3+ KB
        None
```

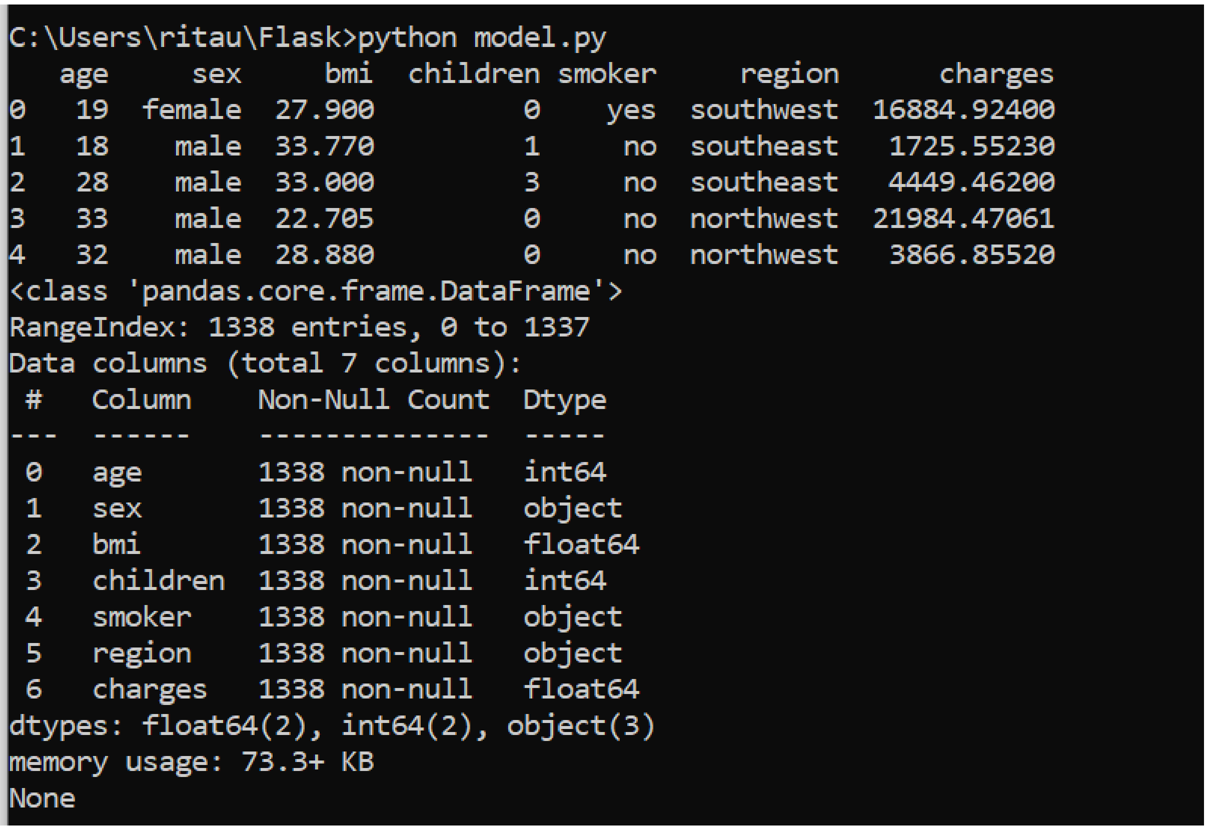


```
In [3]: df['region'].unique()

Out[3]: array(['southwest', 'southeast', 'northwest', 'northeast'], dtype=object)

In [4]: df['sex'].unique()

Out[4]: array(['female', 'male'], dtype=object)
```

- **MODEL TO PREDICT INSURANCE CHARGES**

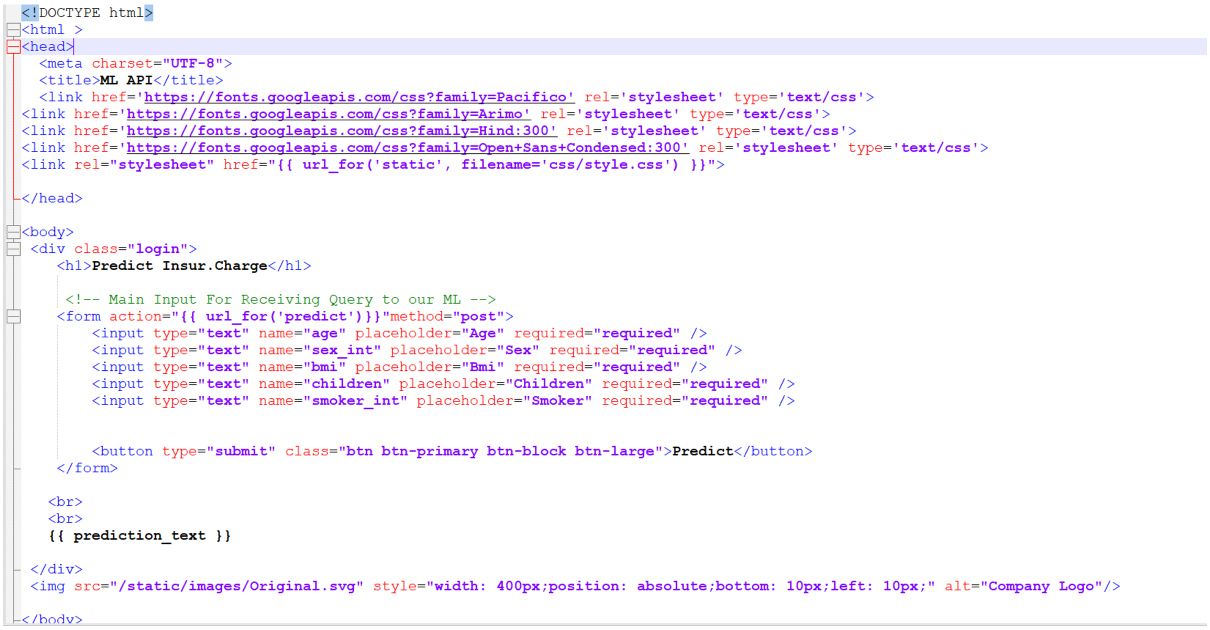


```python
# Creating new variables
df['smoker_int'] = df['smoker'].map({'yes':1, 'no':0})
df['sex_int'] = df['sex'].map({'female':1, 'male':0})
df['region_int'] = df['region'].map({'southwest':1, 'southeast':2,'northwest':3,'northeast':4})

X = df[['age', 'sex_int', 'bmi','children','smoker_int']]
y = df['charges']

import pickle
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
lm = LinearRegression()

#Fitting model with trainig data
lm.fit(X_train, y_train)

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

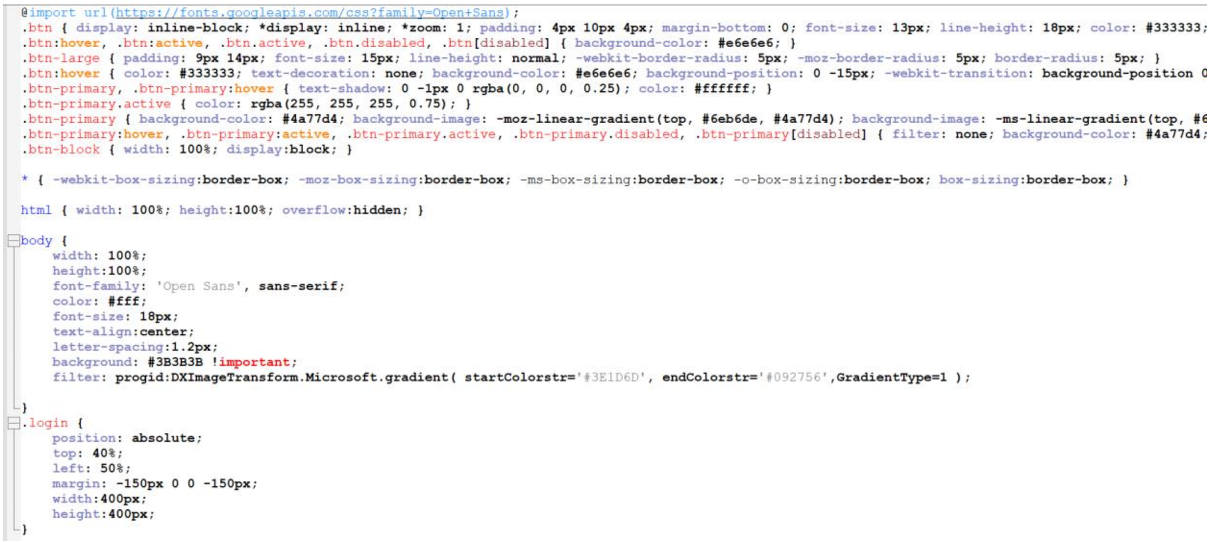


```
C:\Users\ritau\Flask>python model.py
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
None
```

- **DESIGN OF WEBAPP USING HTML AND CSS**

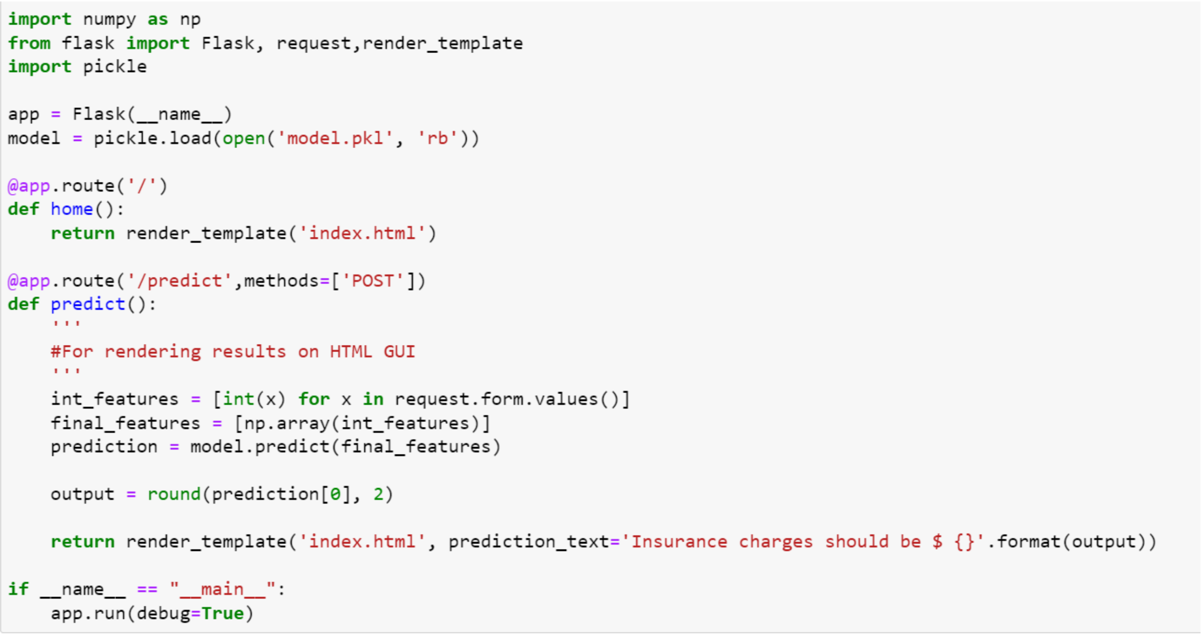


```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' 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 Insur.Charge</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
        <input type="text" name="age" placeholder="Age" required="required" />
        <input type="text" name="sex_int" placeholder="Sex" required="required" />
        <input type="text" name="bmi" placeholder="Bmi" required="required" />
        <input type="text" name="children" placeholder="Children" required="required" />
        <input type="text" name="smoker_int" placeholder="Smoker" required="required" />


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

   <br>
   <br>
   {{ prediction_text }}

 </div>
 <img src="/static/images/Original.svg" style="width: 400px;position: absolute;bottom: 10px;left: 10px;" alt="Company Logo"/>

</body>
```

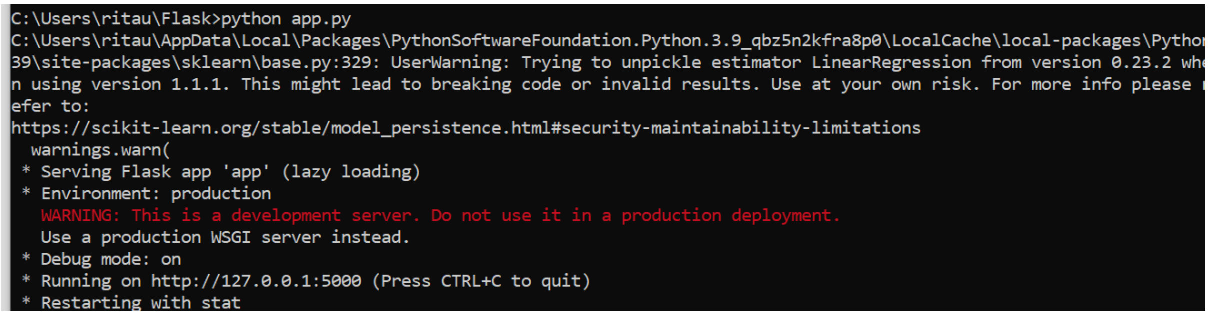


```
@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; font-size: 13px; line-height: 18px; color: #333333;
.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #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
.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, #6
.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;
    color: #fff;
    font-size: 18px;
    text-align:center;
    letter-spacing:1.2px;
    background: #3B3B3B !important;
    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;
}
```

- **DEPLOY THE MODEL ON FLASK**

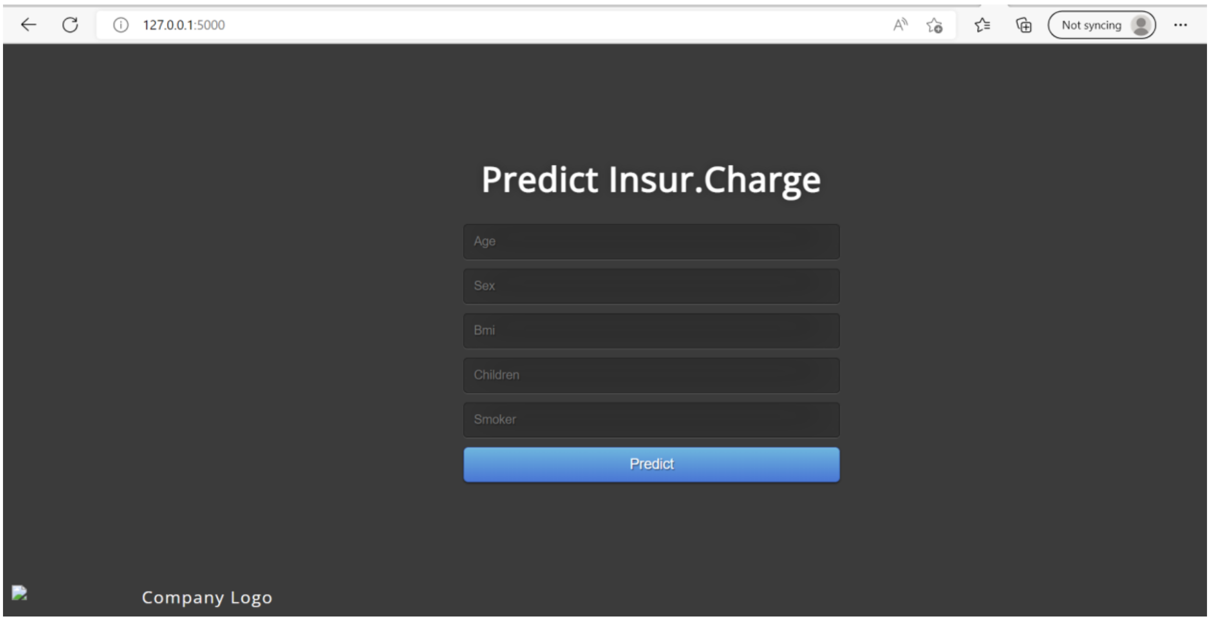


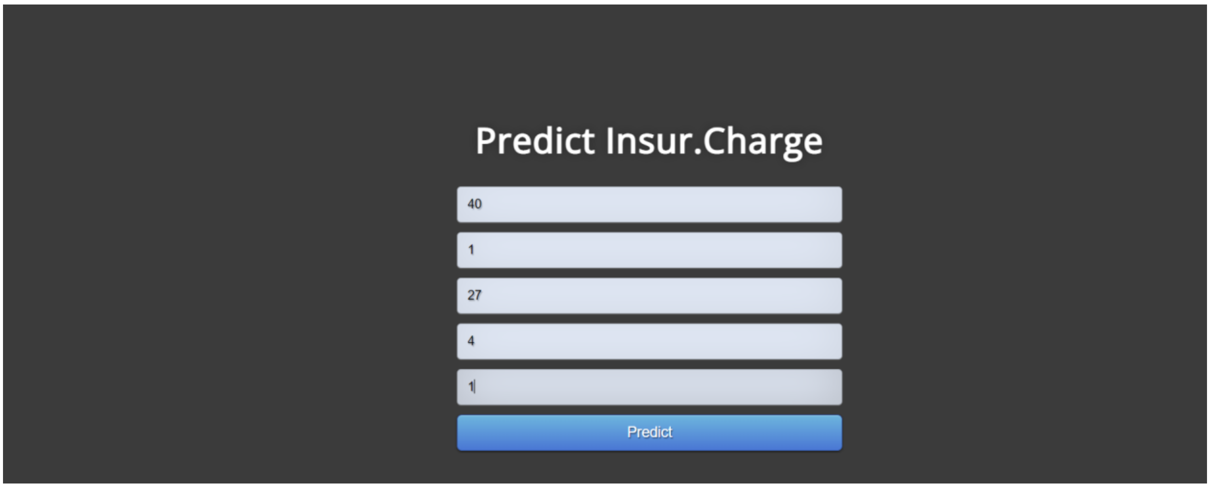
```
import numpy as np
from flask import Flask, request,render_template
import pickle

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
    '''
    int_features = [int(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='Insurance charges should be $ {}'.format(output))

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



```
C:\Users\ritau\Flask>python app.py
C:\Users\ritau\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2kfra8p0\LocalCache\local-packages\Pytho
39\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator LinearRegression from version 0.23.2 whe
n using version 1.1.1. This might lead to breaking code or invalid results. Use at your own risk. For more info please
efer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
 * 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 stat
```

- **TEST THE APPLICATION**





Predict Insur.Charge

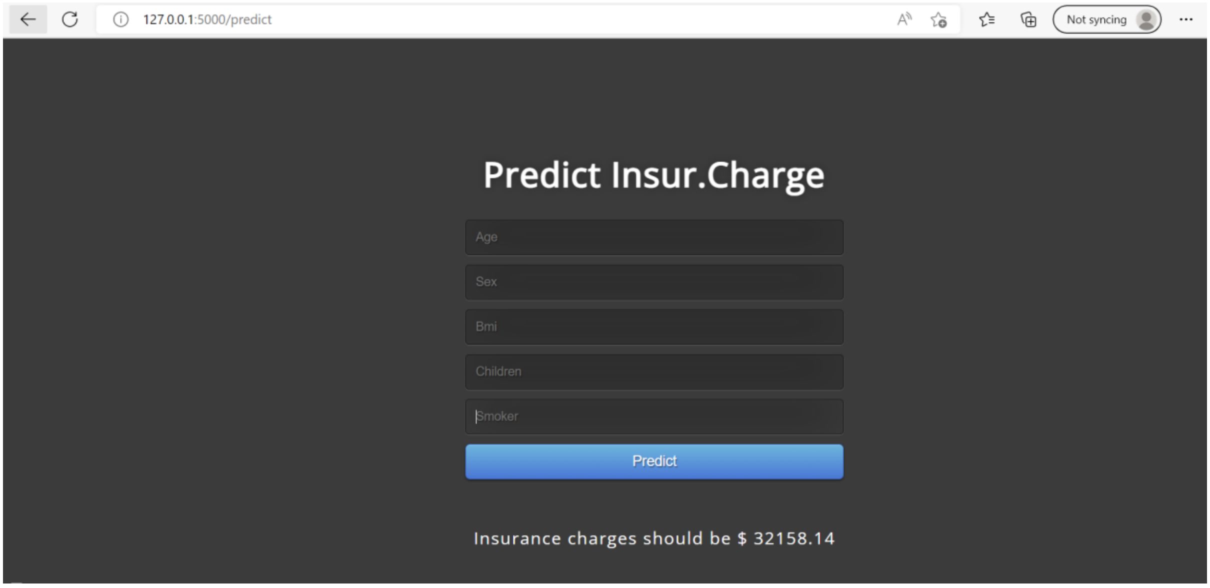
40

1

27

4

1



Predict Insur.Charge

Insurance charges should be $ 32158.14