**INF 428/528 Final Project**

***Predictive Model for Service Calls Payment***

**Flask Application**

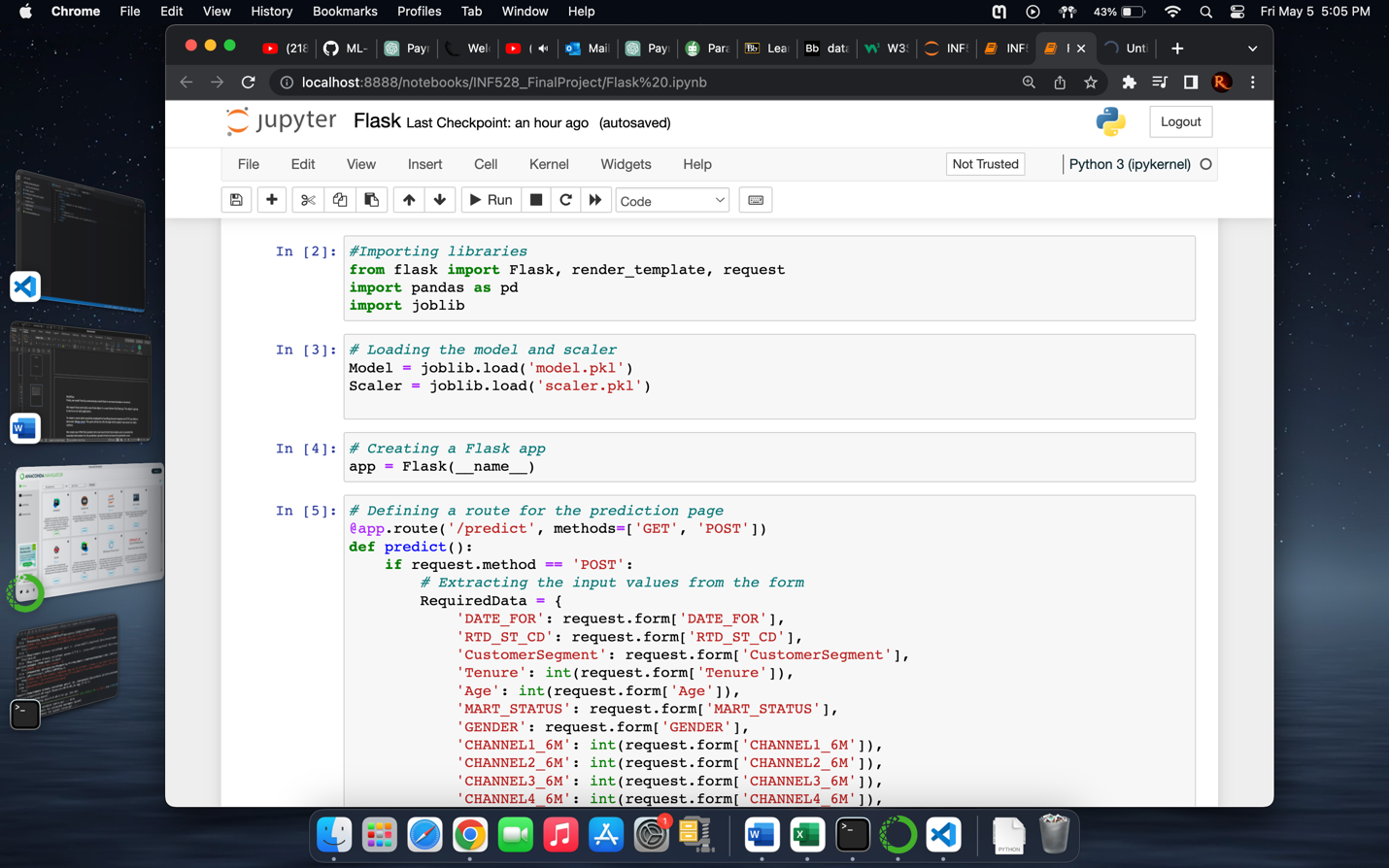
Yashoda Rishita Pothunuri

001570390

**Workflow**

Firstly, we install Flask by command pip install Flask in command window or terminal.

We import Flask and build a new Flask object in a new Python file (Flask.py). This object is going to serve as our web application. The first lines load the required libraries: Flask for building the web app, Pandas for data processing, and joblib for building the machine learning model and scaler.



A picture containing text, screenshot, computer, monitor

Description automatically generated

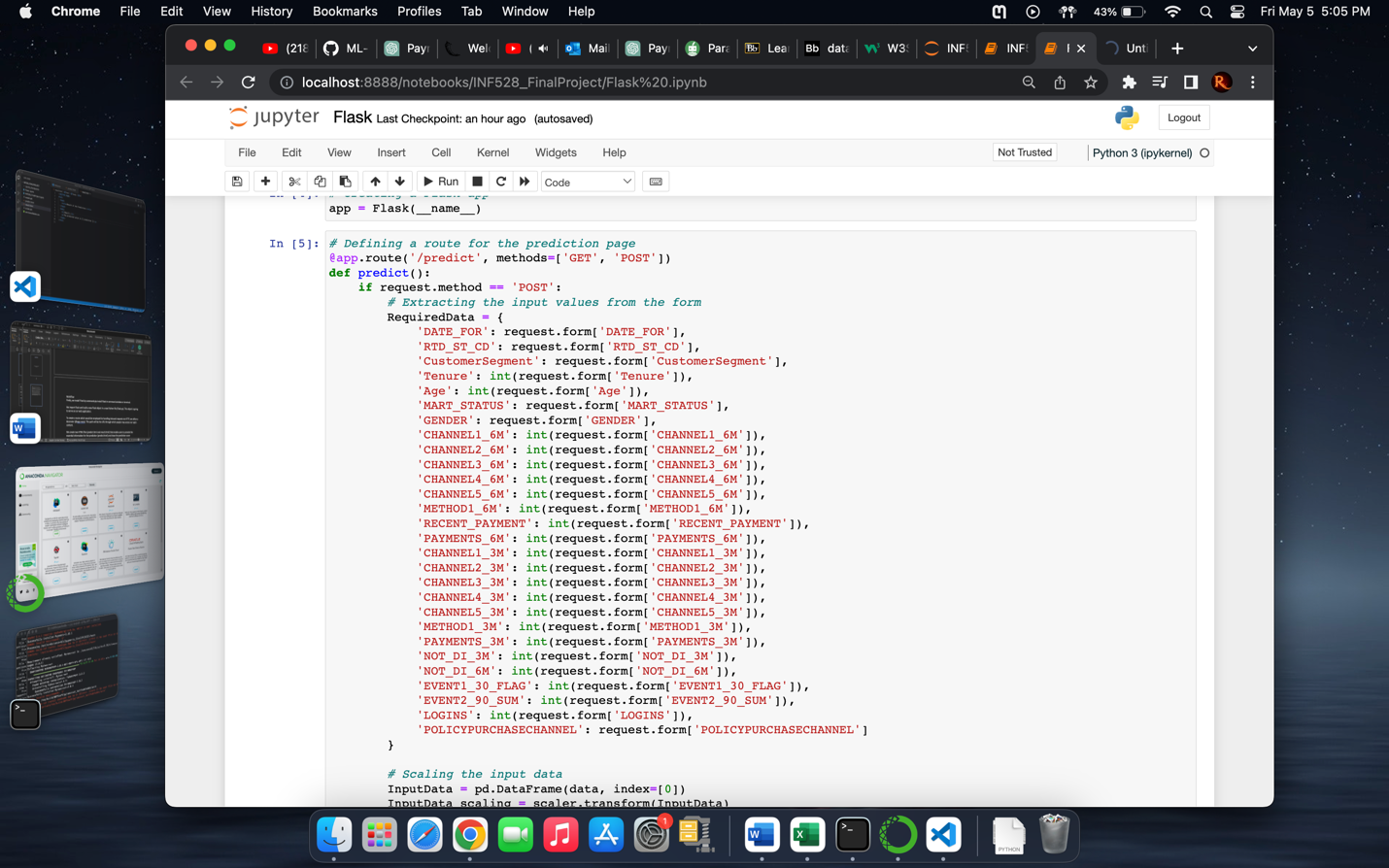
To create a route which would be employed for handling inbound requests via HTTP, we utilize a decorator (@app. route). This path will be the URL through which people may access our app's content.

A picture containing text, screenshot, computer, monitor

Description automatically generated

We create two HTML files (predict.html and result.html) that enable users to provide the essential information for the prediction (predict.html) and show the prediction score (result.html).

To handle form submissions, we construct a function (RequiredData). This function ought to retrieve the values from the form and then apply our model to anticipate the values.



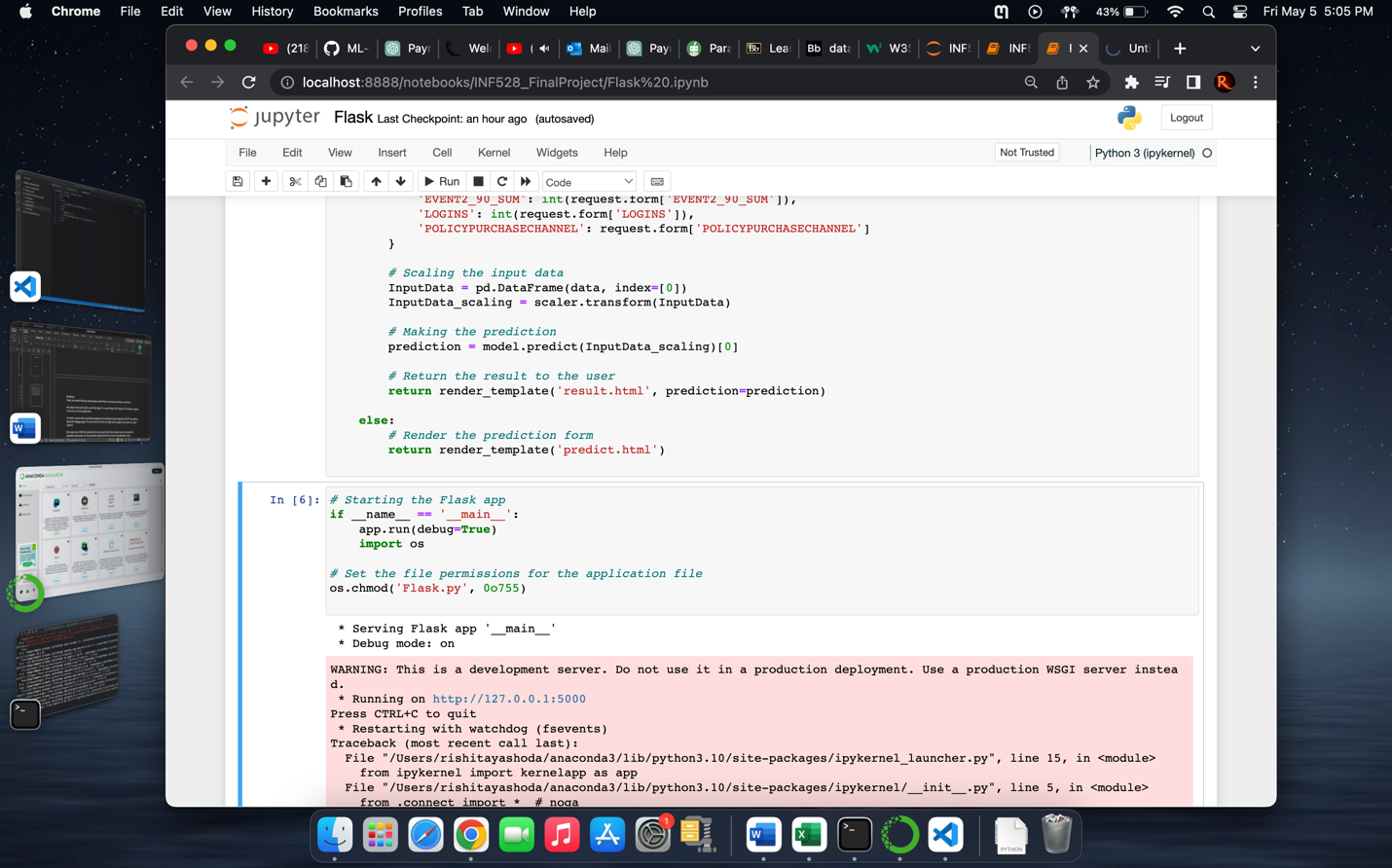
We return the outcome of the prediction to the user after it has been made. This may be accomplished by creating a new HTML template with the result of the prediction (result.html).

A picture containing text, screenshot, computer, indoor

Description automatically generated

**Result**

Finally, we use app.run() to start the Flask application. This will launch the development server and permit to use a web browser to connect to our application.



It appears to provide the URL where the application is running, but it still encounters an exception. It is unable to import the name 'Sequence' from 'collections', causing the site to load poorly.

**Code**

#Importing libraries

from flask import Flask, render\_template, request

import pandas as pd

import joblib

# Loading the model and scaler

Model = joblib.load('model.pkl')

Scaler = joblib.load('scaler.pkl')

# Creating a Flask app

app = Flask(\_\_name\_\_)

# Defining a route for the prediction page

@app.route('/predict', methods=['GET', 'POST'])

def predict():

if request.method == 'POST':

# Extracting the input values from the form

RequiredData = {

'DATE\_FOR': request.form['DATE\_FOR'],

'RTD\_ST\_CD': request.form['RTD\_ST\_CD'],

'CustomerSegment': request.form['CustomerSegment'],

'Tenure': int(request.form['Tenure']),

'Age': int(request.form['Age']),

'MART\_STATUS': request.form['MART\_STATUS'],

'GENDER': request.form['GENDER'],

'CHANNEL1\_6M': int(request.form['CHANNEL1\_6M']),

'CHANNEL2\_6M': int(request.form['CHANNEL2\_6M']),

'CHANNEL3\_6M': int(request.form['CHANNEL3\_6M']),

'CHANNEL4\_6M': int(request.form['CHANNEL4\_6M']),

'CHANNEL5\_6M': int(request.form['CHANNEL5\_6M']),

'METHOD1\_6M': int(request.form['METHOD1\_6M']),

'RECENT\_PAYMENT': int(request.form['RECENT\_PAYMENT']),

'PAYMENTS\_6M': int(request.form['PAYMENTS\_6M']),

'CHANNEL1\_3M': int(request.form['CHANNEL1\_3M']),

'CHANNEL2\_3M': int(request.form['CHANNEL2\_3M']),

'CHANNEL3\_3M': int(request.form['CHANNEL3\_3M']),

'CHANNEL4\_3M': int(request.form['CHANNEL4\_3M']),

'CHANNEL5\_3M': int(request.form['CHANNEL5\_3M']),

'METHOD1\_3M': int(request.form['METHOD1\_3M']),

'PAYMENTS\_3M': int(request.form['PAYMENTS\_3M']),

'NOT\_DI\_3M': int(request.form['NOT\_DI\_3M']),

'NOT\_DI\_6M': int(request.form['NOT\_DI\_6M']),

'EVENT1\_30\_FLAG': int(request.form['EVENT1\_30\_FLAG']),

'EVENT2\_90\_SUM': int(request.form['EVENT2\_90\_SUM']),

'LOGINS': int(request.form['LOGINS']),

'POLICYPURCHASECHANNEL': request.form['POLICYPURCHASECHANNEL']

}

# Scaling the input data

InputData = pd.DataFrame(data, index=[0])

InputData\_scaling = scaler.transform(InputData)

# Making the prediction

prediction = model.predict(InputData\_scaling)[0]

# Returning the result to the user

return render\_template('result.html', prediction=prediction)

else:

# Render the prediction form

return render\_template('predict.html')

# Starting the Flask app

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=5003)

import os

# Set the file permissions for the application file

os.chmod('Flask.py', 0o755)

from collections import Sequence

**References**

*Deploy machine learning models using Flask*. YouTube. (2021, March 5). Retrieved May 6, 2023, from https://youtu.be/0nr6TPKlrN0

Hastie, T., Tibshirani, R., & Friedman, J. H. (2017). *The elements of Statistical Learning: Data Mining, Inference, and prediction*. Springer.

*Learn about building, deploying, and managing your apps on Heroku.* Heroku Dev Center. (n.d.). Retrieved May 6, 2023, from https://devcenter.heroku.com/

*Learn: Machine learning in Python*. scikit. (n.d.). Retrieved May 6, 2023, from https://scikit-learn.org/stable/documentation.html

McKinney, W. (2018). *Python for Data Analysis: Data wrangling with pandas, NumPy, and ipython*. O'Reilly.

*Welcome to flask*. Welcome to Flask - Flask Documentation (2.1.x). (n.d.). Retrieved May 6, 2023, from https://flask.palletsprojects.com/en/2.1.x/

YouTube. (2021, April 10). *Deploy machine learning model flask*. YouTube. Retrieved May 6, 2023, from https://www.youtube.com/watch?v=MxJnR1DMmsY