UberCat - *CS2021 Final Project*

Link | **GitHub** | **Documentation**

Author - Rishabh Ravindra

Introduction

Overview

UberCat is an online Uber fare estimator for students at UC. The idea was inspired by Uber's fare estimator available online. My project is a UC specific version to help students estimate fares before they go out for the night. It is a Python/Flask application powered by UBER's and Google Maps' Python API. The project is deployed on Heroku.

- Enter your starting address
- Select your dorm
- Get price estimates for all types of Ubers

API/Framework Selection Process

The first choice I made was to use Uber's Python API to get price estimates. This was a no brainer as Uber officially supported the Python API.

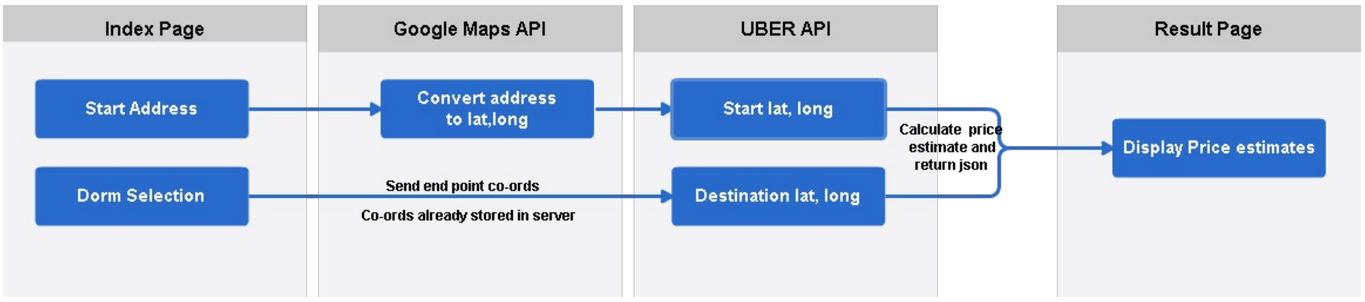
The next part was to decide where to get the co-ordinates from since the Uber API uses co-ordinates to define its starting and ending point. I used the Google Maps API python wrapper as I have used it before.

The last major decision was to choose what web framework to use to get information: Django or Flask? I settled with Flask because of its simplicity and unopinionated nature.

Process breakdown

The user is welcomed with a form that takes the starting address, and dorm name as input. A *POST* request is sent to the server when the user clicks on the submit button. The Maps API geocodes the address given to co-ordinates. There is a dictionary defined in the app.py which has co-ordinates of all dormitories. The server matches the user input with the dictionaries. The co-ordinates of the user starting point and dorm are then sent to the Uber API

The API calculates price estimates of all types of Uber's available and returns a JSON file with the details. The json is stored in a variable and is sent to render the results page. In the render HTML page, there is a loop set up to render an info card with trip details about each Uber.



Sample flowchart created during ideation

Project Results

Thoughts and challenges

The program consisted of three main parts: requesting data from Uber and Google Maps, getting user input and rendering output, and deploying on a cloud based server. The first part was handled by using the respective APIs and getting back data in JSON format. The second part was handled by Flask which sent back and forth data to the server and renderred the form and output pages. The last part was handled by Heroku, a cloud hosting platform. I had to install Gunicorn, a Python HTTP server that ran the app on Heroku.

During the whole program design, I had to make sure that my program code was good at handling different APIs and avoid any conflicts. As can be seen from the flowchart, a lot of processing goes between the two html pages rendered. I had to use a synchronous platform to accomplish the middle tasks.

For example, I had to make sure the staring address string was first converted to co-ordinates by the Maps API before I sent the co-ordinates to the Uber API. This all had to be done in a sequential pattern, else the UBER API would not be able to calculate the price estimate - a feature integral to the program.

Also, I improvised on the front-end. I had a basic html form input and a basic output render page. However, I did not like how the app looked. So I used UIKit, a HTML and CSS library with built-in web components that I could use.

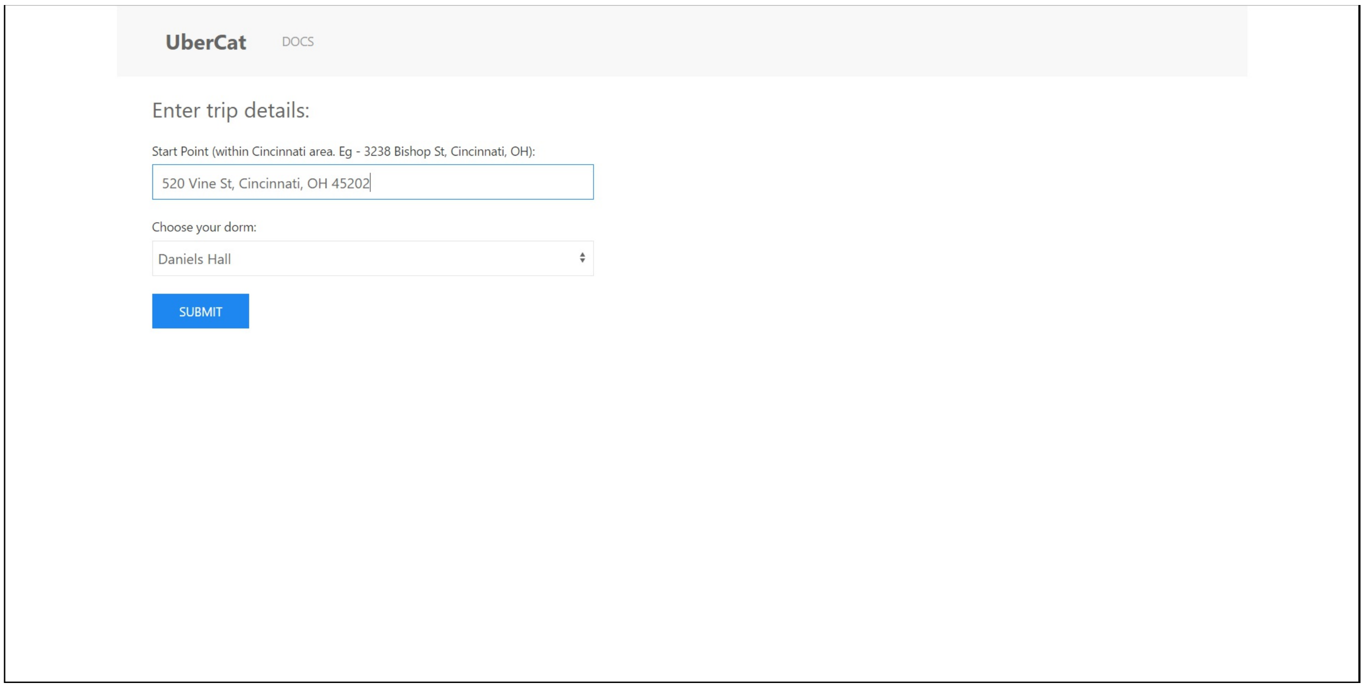
Future improvements

One improvement that immediately comes into my mind is integrating Lyft’s API to compare the pricess. This feature would actually yield more value to the application as users have to always switch between the two apps to get the price estimates. I did not implement this feature in the current project because Lyft does not have an official Python client currently and does not provide ride estimates through its API.

Another improvement is to update the file structure. Currently, the files are all in one folder. Since the app involves a Model-View-Controller structure, the folders should reflect it. This would help any future developers working on it understand the code better.

Screenshots

Get input from user


UberCat DOCS

Enter trip details:

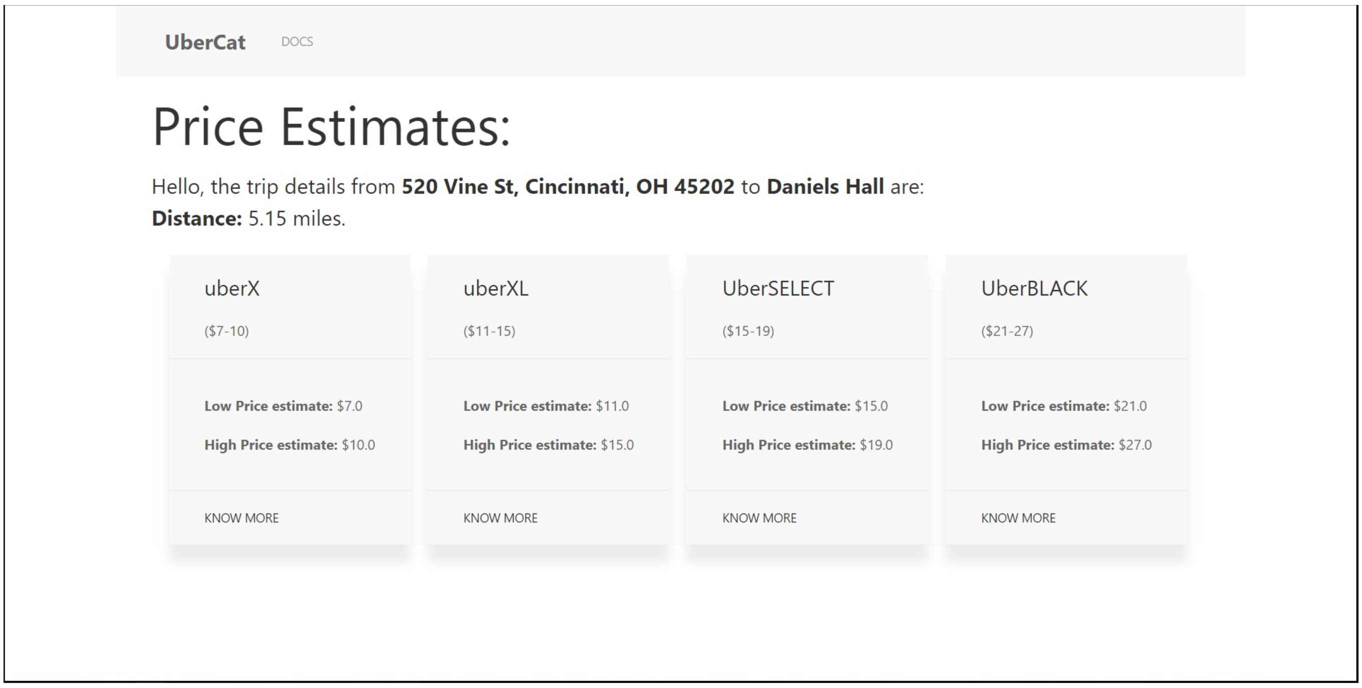
Start Point (within Cincinnati area. Eg - 3238 Bishop St, Cincinnati, OH):

520 Vine St, Cincinnati, OH 45202

Choose your dorm:

Daniels Hall

Results page with availbale fare rates for all types of UBERs running in Cincinnati



Price Estimates:

Hello, the trip details from **520 Vine St, Cincinnati, OH 45202** to **Daniels Hall** are:
Distance: 5.15 miles.

uberX	uberXL	UberSELECT	UberBLACK
($7-10)	($11-15)	($15-19)	($21-27)
Low Price estimate: $7.0	**Low Price estimate:** $11.0	**Low Price estimate:** $15.0	**Low Price estimate:** $21.0
High Price estimate: $10.0	**High Price estimate:** $15.0	**High Price estimate:** $19.0	**High Price estimate:** $27.0
KNOW MORE	KNOW MORE	KNOW MORE	KNOW MORE

Bibliography

A number of online resources and articles were instrumental in converting the program idea into fruition:

- Handle a POST Request In Flask
- Uber Rides Python API
- Python Client for Google Maps Services
- Deploying a simple Flask app to the cloud via Heroku

Links to API/Frameworks/Libraries used

UberCat was only possible because of the generous open-source/public libraries on the web.

API/Framework	Description	README
Flask	Web	flask.pocoo.org/

	framework for Python	
Maps API python	Python client for Google Maps API	pypi.python.org/pypi/googlemaps/
Gunicorn	HTTP server for Python	gunicorn.org/
Uber Rides API	Uber rides SDK for Python	github.com/uber/rides-python-sdk
UIkit	Front-end framework to quickly create web components	getuikit.com

Code Appendix

GitHub

Website