Bob’s Burgers - A Python API
By David Dell - dpd003

I like burgers and I love the tv show Bob's Burgers and I like programing so I combined the three for a proof of concept! For my final project I decided to build a full stack web application utilizing python as the backend using the micro framework flask and VueJs as the frontend To demonstrate the utilizations of these frameworks I have made a web application that allows a user to create additional users, edit a menu and place orders. This app is far from complete but it demonstrates how to utilize python and flask to make a restful API and VueJS as a client.

The crux of this project is the Python RESTFUL API. For my career I develop APIs in PHP, Javascript and JAVA, I was interested in learning web development from the Python perspective. I decided to use the Python micro framework called FLASK. I titled my project “Bob’s Burgers” (after the TV show), as I wanted my API to be used in the restaurant field. The current state of the API is functional but still incomplete as every project is never fully complete in the eyes of the clients. For the project I built out Create, Read, Update Delete (CRUD) endpoints for users, menu sections and menu items. I designed this application following Restful conventions. A consumer of the API can get a list of users through making a GET {domain}/api/users call. I provided a postman collection which can be imported using postman to simulate hitting the endpoint and to provide a list of available endpoints. A user can also make a curl request to consume the API. A consumer can also get a specific user by also making a GET {domain}/api/users/{id} where id represents the id of a user. Using the API, a consumer can also create and update users as well. To demonstrate how one would consume an API I built a VueJS web application to consume my API. The code for this can be viewed in the client directory of my project. This is useful as it provided a visual of the API being consumed. If you open up the console and view the network tab you can actively see the endpoints be consumed for various actions over the interface. My project is useful because I have laid the foundations for building and actual website for a restaurant to use. Given time I could build out this python API and client and make a useful website for a restaurant owner to use. I designed the API to scale well. I designed my API with Service Repository Pattern in mind. My app.py file acts as the intro point into the API and routes the various API calls to other services that handle the actual logic of the API. For example all user API calls get routed to my User service class that handles everything dealing with users. Meanwhile all menu and menu items/ sections API calls get mapped to my MenuService class. I also create a model per database table which extends my base model. My base model contains all the database logic such as querying the database for a result, deleting, creating and updating items as well. This allows me to easily create additional tables without having to worry about building in the CRUD operations. All I have to do is make a new model ex. Menu and extend the base model and I automatically gain access through inheritance. All the new model needs to know is the name of the database table it represents and the fields it contains. The instructions of how to set up the API are located in the readme of the project. The instructions to setup the API are found in the readme of the project. For testing I created unit testing and a test endpoint. Unit testing uses the unit test library. To run the test, once the environment is up one can just type python test.py and the test will run. This was pretty cool. I am familiar with unit testing in other languages so it was cool seeing it done in python. Overall I enjoyed creating this Python APP and applying what I have learned in class with Python and what I have learned in my career with API design.