**Project Purpose:**

The purpose of this project is to understand the inner workings of the REST API. For this project, I have built a simple REST API that showcases food menus from different restaurants. The app contains different choices of Restaurants with their cuisines. Each item has a price and quantity to choose from. I tried to perform CRUD operations using Django REST API.

Platform: VSCode

API: Django

Framework: Django framework

API Testing Tool: Postman

**Development Instructions:**

1. Download and install latest 3.x version of Python.
2. Set up Django Project:
   1. Create Directory:

>mkdir <project\_name>

* 1. Navigate to the directory:
  2. > cd <project\_path>
  3. Create a virtual environment in Python that isolates the dependencies with multiple Python objects:

Python -m venv <project\_name\_env>

* 1. Activate the Virtual Environment from the terminal:

<project\_name\_venv>\Script\activate

* 1. In the project folder, install the following frameworks:

>pip install Django

>pip intsall Django\_rest\_framework

* 1. Create the Django project:

>Django-admin startproject <project\_name>

> cd <project\_name>

* 1. Create the new app for the API:

>Django-admin startapp <project\_name\_api>

* 1. Run the initial migrations:

>Python manage.py migrate

* 1. Open VScode:
  2. Edit the run configuration:

The run configuration for VScode has different settings that you can go through. Please refer to the following link:

<https://code.visualstudio.com/docs/python/tutorial-django>

* 1. Once the run configuration is created, I can run/debug: Django within VScode
  2. Create superuser:

>python manage.py createsuperuser.

For this project refer to the following username:

Username: psm

Password:fall2023

1. Postman Setup:
   1. Go to the VScode extensions.
   2. Install Postman extension for VSCode.
   3. Note: To use this extension, you must sign in with your Postman account.
2. Create Restaurant Menu:
   1. Created following files:
      1. Serializers.py
      2. Urls.py
      3. Models.py
      4. Views.py