**Online Restaurant Ordering System**

**Group 15**

**UNC Charlotte**

**Restaurant Ordering System User Guide - Group 15**

***Introduction:***

This API is designed to function as an online restaurant ordering system simulation using CRUD framework as a baseline. This is accomplished by using [FastAPI](https://fastapi.tiangolo.com/) docs to emulate the API calls in real time. Group 15’s [Python](https://www.python.org/downloads/) API setup is designed to connect with a [MySQL](https://www.mysql.com/) database via [SQLAlchemy](https://www.sqlalchemy.org/). It is set up to implement [PyTest](https://docs.pytest.org/en/stable/) as necessary to ensure proper execution of functions. Each object of the API is divided into four sections - model, schema, controller, and router. These four sections are described below:

* ***Model***: This is the way in which data is stored inside the database, ie. tables, rules, attributes, etc.
* ***Schema***: These pieces connect the database to the API by defining the correct structure for requests and responses.
* ***Controller***: Logical functions go here to manipulate data as needed by the object. This includes database interaction.
* ***Router***: This is the connection between the controller and the rest of the API. It defines endpoints from the CRUD operations.

Using this combination of tools, the backend of the restaurant ordering system accomplishes user needs and creates implementations to streamline the restaurant processes.

***Legend:***

***Code***

**File/Directory**

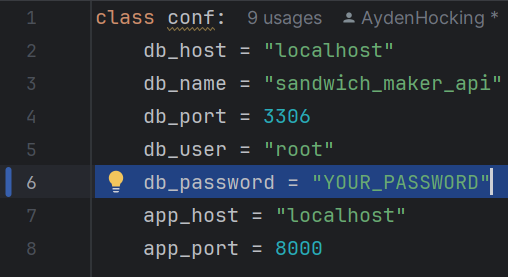
***Initial Setup:***

| **Requirements:** | **Use:** |
| --- | --- |
| ***MySQL 8.0*** | *Database Visualization and Setup* |
| ***Git v.2.47.0 or higher*** | *Cloning Repository to Local Machine* |
| ***Pycharm v.2024.3.1.1 or higher*** | *Code Editing and Compiling(Optional)* |

1. Open a new terminal, and navigate to the directory (use ***cd <directory>****)* in which you wish to add the repository.
2. Use the following command to clone the repository from GitHub:

* ***git clone*** [***https://github.com/AydenHocking/Python\_Online\_Restaurant\_Ordering\_System***](https://github.com/AydenHocking/Python_Online_Restaurant_Ordering_System)
  1. Open the code in an editor (such as Pycharm), and navigate to the requirements.txt file in the root directory. Install each of the following:
* ***fastapi***
* ***"uvicorn[standard]"***
* ***sqlalchemy***
* ***pymysql***
* ***pytest***
* ***pytest-mock***
* ***httpx***
* ***cryptography***
  1. NOTE: If you do not want to use a code editor, navigate to the root directory in a terminal and type “***pip install***” followed by the names of each python library listed.
* Example: ***pip install fastapi***

1. Open the file located at <root>/api/dependencies/config.py. Edit line 6 to include and set the MySQL password.



1. In File Explorer, navigate to <root>/api/database, then open the Populate\_Database.sql and Testing\_And\_Reseting.sql files in MySQL.
2. Once open in MySQL, run the following lines of the Testing\_And\_Reseting.sql by highlighting them and clicking .

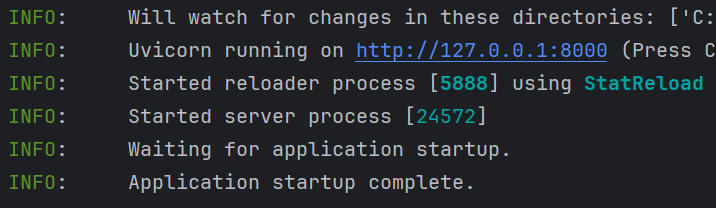
* **drop database sandwich\_maker\_api;**
* **create database sandwich\_maker\_api;**

This will create the new database to be used for the tables.

1. Navigate back to your terminal, and in the root directory run the following command to start the FastAPI:

* **uvicorn api.main:app --port 8001 --reload**

You should receive the following terminal message:



1. Back in MySQL, refresh schemas on the left side and double click on the sandwich\_maker\_api schema.
2. Highlight the following section of Testing\_And\_Reseting.sql and run by clicking .

* **SET FOREIGN\_KEY\_CHECKS = 0;**
* **TRUNCATE TABLE recipes;**
* **TRUNCATE TABLE ingredients;**
* **TRUNCATE TABLE menuitems;**
* **TRUNCATE TABLE customers;**
* **TRUNCATE TABLE payments;**
* **TRUNCATE TABLE promos;**
* **TRUNCATE TABLE orderitems;**
* **TRUNCATE TABLE orders;**
* **TRUNCATE TABLE reviews;**
* **SET FOREIGN\_KEY\_CHECKS = 1;**

This is to ensure that no data occurs in the tables prior to population.

1. Now run the entire Populate\_Database.sql. This will populate the database tables with values.
2. Finally, visit the URL below to access the FastAPI docs and generate API requests.

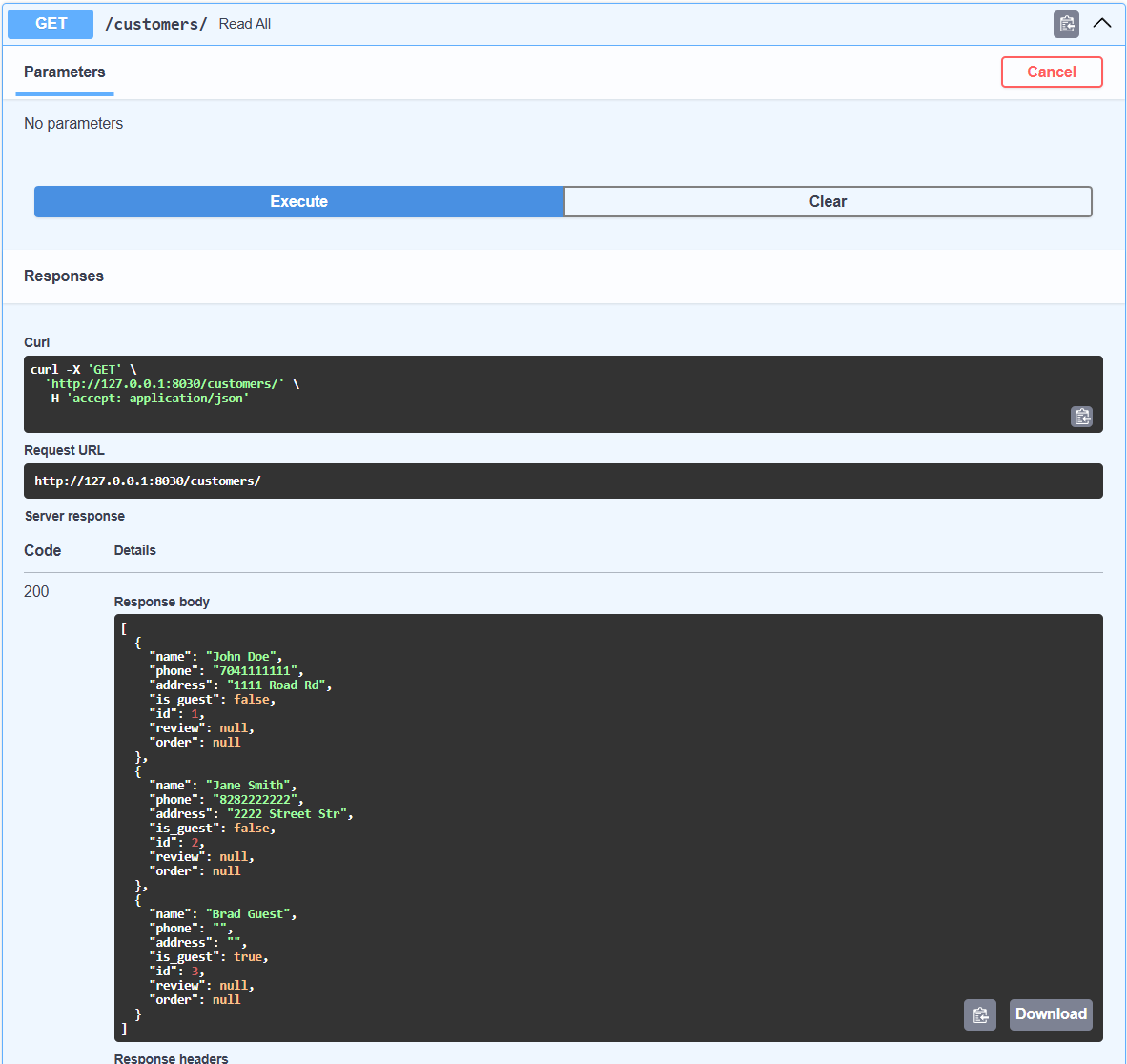
* [*http://127.0.0.1:8000/docs#*](http://127.0.0.1:8000/docs#)

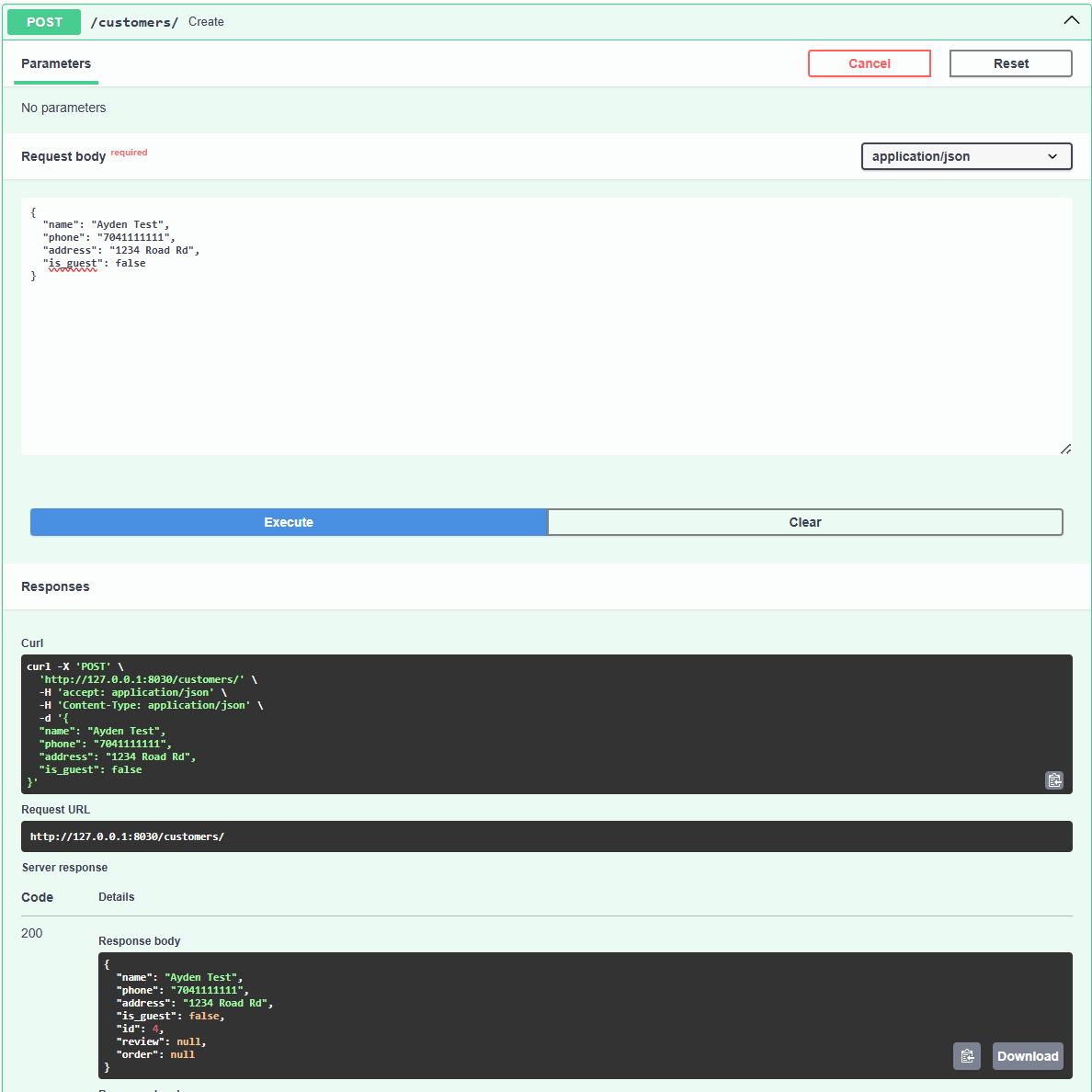
Requests can made by clicking the “Try it out” button and entering the values requested (See database for formatting)

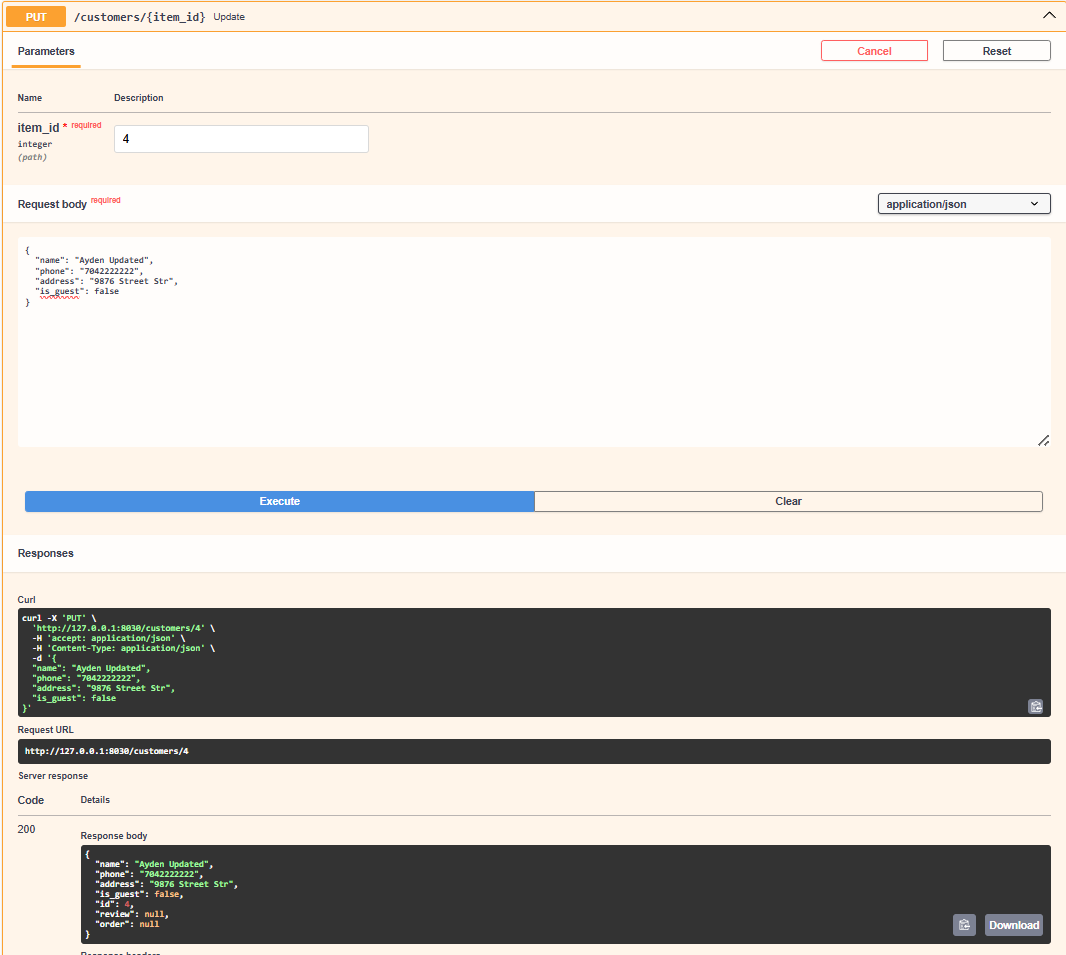
Access is now available to see the API calls in real time with FastAPI docs, and the database manipulation with MySQL. If any errors occur, press CTRL-C in the terminal, and repeat steps 6-11. This will reset the database and API server.

***API Usage Examples:***

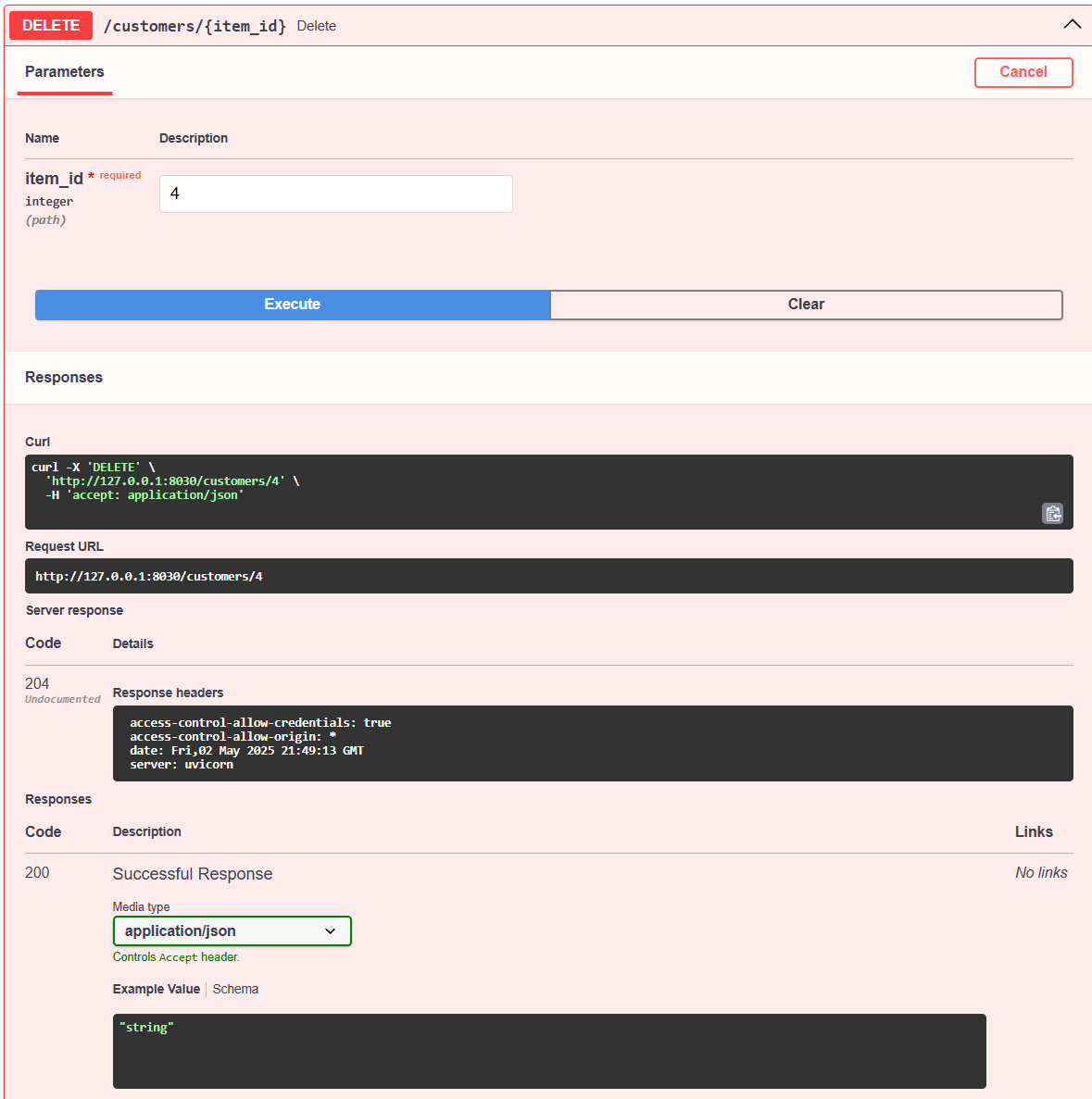
***GET request for Read All Customers:***

******

***POST request for Create Customer:***

***PUT request for Update Customer:  
***

***DELETE request for Deleting Customer:***

******