**API Documentation**

1. Installation

* Clone this project.
* Create new user mysql name “apijwt”, following this command:

CREATE USER 'apijwt'@'localhost' IDENTIFIED BY 'apijwt';

GRANT ALL PRIVILEGES ON \* . \* TO 'apijwt'@'localhost';

* Build and running docker with following command:

docker-compose build && docker-compose up -d

* Check docker container is running or not:

docker container ls

* Create .env file with copy .env.example in folder “src” and change the connection:

DB\_CONNECTION=mysql

DB\_HOST=mysql

DB\_PORT=3306

DB\_DATABASE=db\_apijwt

DB\_USERNAME=apijwt

DB\_PASSWORD=apijwt

* Do composer install, generated key, migrate and seeder with this command:

docker-compose exec php composer install

docker-compose exec php php artisan key:generate

docker-compose exec php php artisan migrate:fresh –seed

* Access laravel project in http://localhost:8088
* If want connect to mysql container, use this setting:

Host: localhost

Port: 4306

Username: apijwt

Password: apijwt

1. API Features

Base URL: **127.0.0.1:8000/api/auth/**

This API using JWT authentication and already test using postman.

1. **Register [ /register ]**

Before start to testing another API. First, create a user with role **“admin”** with role\_id **“1”**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| name | Required | Name of user |
| email | Required | Email user |
| password | Required | Password login user |
| password\_confirmation | Required | Password confirmation |
| role\_id | Required | Role ID:  1 🡺 admin  2 🡺 company  3 🡺 customer |

Example: **Figure 1. Register**

1. **User Login [ /login ]**

After that, do login to get **access\_token** for other authentication.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| email | Required | Email user |
| password | Required | Password login user |

Example: **Figure 2. User Login**

1. **User Profile [ /user-profile ]**

To see the details of the logged in user, you can use **access\_token** in User Login API and put that into bearer token in User Profile API. No required key. Example: **Figure 3. User Profile.**

1. **Refresh [ /refresh ]**

To get new token, it can be done by doing a refresh token with this API. You can use **access\_token** in User Login. No required key. Example: **Figure 4. Refresh**.

1. **Logout [ /logout ]**

To logout from the system use this API. You can use **access\_token** in User Login. No required key. Example: **Figure 5. Logout**.

1. **Import Restaurant JSON Data [ / import-restaurant ]**

To import data, you must login with the **admin role** and use **bearer token**, then select restaurant JSON file. Required key [file]. After import success, the result will look like below. This process may take some time.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| file | Required | Restaurant JSON file |

Example: **Figure 6. Import Restaurant JSON Data**

1. **Import Users JSON Data [ / import-user ]**

To import data, you must login with the **admin role** and use bearer token, then select restaurant JSON file. Required key [file]. After import success, the result will look like below. This process may take some time.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| file | Required | User JSON file |

Example: **Figure 7. Import User JSON Data**

1. **List all restaurants that are open at a certain datetime [ /list-restaurant-by-datetime ]**

This API can be used to see if a restaurant is open or not at a certain time. This API can be accessed on the **customer role**. Required key [datetime, offset, limit].

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| datetime | Required | Date time format:  **Y-m-d H:i:s**  Example:  **2021-01-01 12:12:12** |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 8. List all restaurants that are open at a certain datetime**

1. **List all restaurants within the vicinity of the user’s location or (any location), ranked by distance [ /list-restaurant-by-distance ]**

This API can be used to search for restaurants based on latitude and longitude. This API can be accessed on the **customer role**. If latitude or longtitude is null, then the location used to find the nearest restaurant is the location of the **user who logs in**. Required key [latitude, longitude, offset, limit].

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| latitude | Optional | Point format between:  [-90, 90] |
| logtitude | Optional | Point format between:  [-180, 180] |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 9. List all restaurants within the vicinity of the user’s location or (any location), ranked by distance**

1. **List all restaurants that are open for x-z hours per day or week [ /list-restaurant-by-open-hours ]**

This API can be used to search for restaurants based on range open hours. This API can be accessed on the **customer role**. Required key [start\_range\_time, end\_range\_time, offset, limit].

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| start\_range\_time | Required | Start range hours. Numeric format hours |
| end\_range\_time | Required | End range hours. Numeric format hours |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 10. List all restaurants that are open for x-z hours per day or week**

1. **List all restaurants that have x-z number of dishes within a price range [ /list-restaurant-by-price ]**

This API can be used to search for restaurants based on a valid price range from the restaurant's menu. This API can be accessed on the **customer role**. Required key [lowest\_price, highest\_price, offset, limit].

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| lowest\_price | Required | Lowest range price. Double format price |
| highest\_price | Required | Highest range hours. Double format price |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 11. List all restaurants that have x-z number of dishes within a price range**

1. **Search for restaurants or dishes by name, ranked by relevance to search term [ /list-restaurant-dish ]**

This API can be used to search for restaurants or dish that matching search term. This API can be accessed on the **customer role**. Required key [search, offset, limit].

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| search | Required | Name of dish or restaurant. String format. |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 12. Search for restaurants or dishes by name, ranked by relevance to search term**

1. **Search for restaurants that has a dish matching search term [ /list-restaurant-by-dish ]**

This API can be used to search for restaurants that has a dish matching search term. This API can be accessed on the **customer role**. Required key [dish, offset, limit].

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| dish | Required | Name of dish. String format. |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 13. Search for restaurants that has a dish matching search term**

1. **The top x users by total transaction amount within a date range [ /** **list-user-by-transaction ]**

This API can be used to search top x for user based on total transaction amount within a date range. This API can be accessed on the **admin and** **company role**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| start\_date | Required | Start date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| end\_date | Required | End date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 14. The top x users by total transaction amount within a date range**

1. **The most popular restaurants by transaction volume, either by number of transactions or transaction amount [ /** **list-restaurant-by-transaction ]**

This API can be used to search most popular restaurant based on amount or number of transaction within date range. This API can be accessed on the **admin and** **customer role**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| start\_date | Required | Start date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| end\_date | Required | End date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 15. The most popular restaurants by transaction volume, either by number of transactions or transaction amount**

1. **Total number of users who made transactions above or below $v within a date range [ /total-user-by-transaction-amount ]**

This API can be used to show total user based on below or above amount of transaction within date range. This API can be accessed on the **admin and** **company role**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| start\_date | Required | Start date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| end\_date | Required | End date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| sign | Required | Sign below or above. Enum format [<, <= , =, >=, >] |
| amount | Required | Amount in dollar. Numeric data format |

Example: **Figure 16. Total number of users who made transactions above or below $v within a date range**

1. **List all transactions belonging to a restaurant [ /list-transaction ]**

This API can be used to show list all transaction from the company that logged in. This API can be accessed on the **company role**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| start\_date | Required | Start date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| end\_date | Required | End date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 17. List all transactions belonging to a restaurant**

1. **List all transactions belonging to a user [ /list-transaction ]**

This API can be used to show list all transaction from the user that logged in. This API can be accessed on the **customer role**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| start\_date | Required | Start date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| end\_date | Required | End date transaction. Date format: Y-m-d. Example: 2021-12-12 |
| offset | Required |  |
| limit | Required |  |

Example: **Figure 18. List all transactions belonging to a user**

1. **Check balances customer or company [ /check-balances ]**

This API can be used to show last balance from the user that logged in. This API can be accessed on the **customer or company role**. No required key.

Example: **Figure 19. Check balances customer or company**

1. **Create purchase order [ /** **purchase-order ]**

This API can be used to create purchase order by ID from the user that logged in. This API can be accessed on the **customer role**. For example data, can use JSON data below:

|  |
| --- |
| [  {  "dish\_id": 1,  "restaurant\_id": 1,  "qty": 2  },  {  "dish\_id": 2,  "restaurant\_id": 1,  "qty": 1  }  ] |

In this process, purchase order process already paid. Data purchase will insert in **purchases** table **Figure 20. New data purchase order insert in purchases table** and create purchase detail in **purchase\_detail** table as in **Figure 21.** **New data purchase order insert in purchase\_detail table**. For customer balances will insert as credit and company balances will insert as debit as in **Figure 22. Insert data to customer\_balances as credit** and **Figure 23. Insert data to company\_balances as debit**.

* Parameter

|  |  |  |
| --- | --- | --- |
| Key | Required / Optional | Description |
| data | Required | Purchase order JSON data format. |

Example: **Figure 24. Create purchase order**

1. Figure

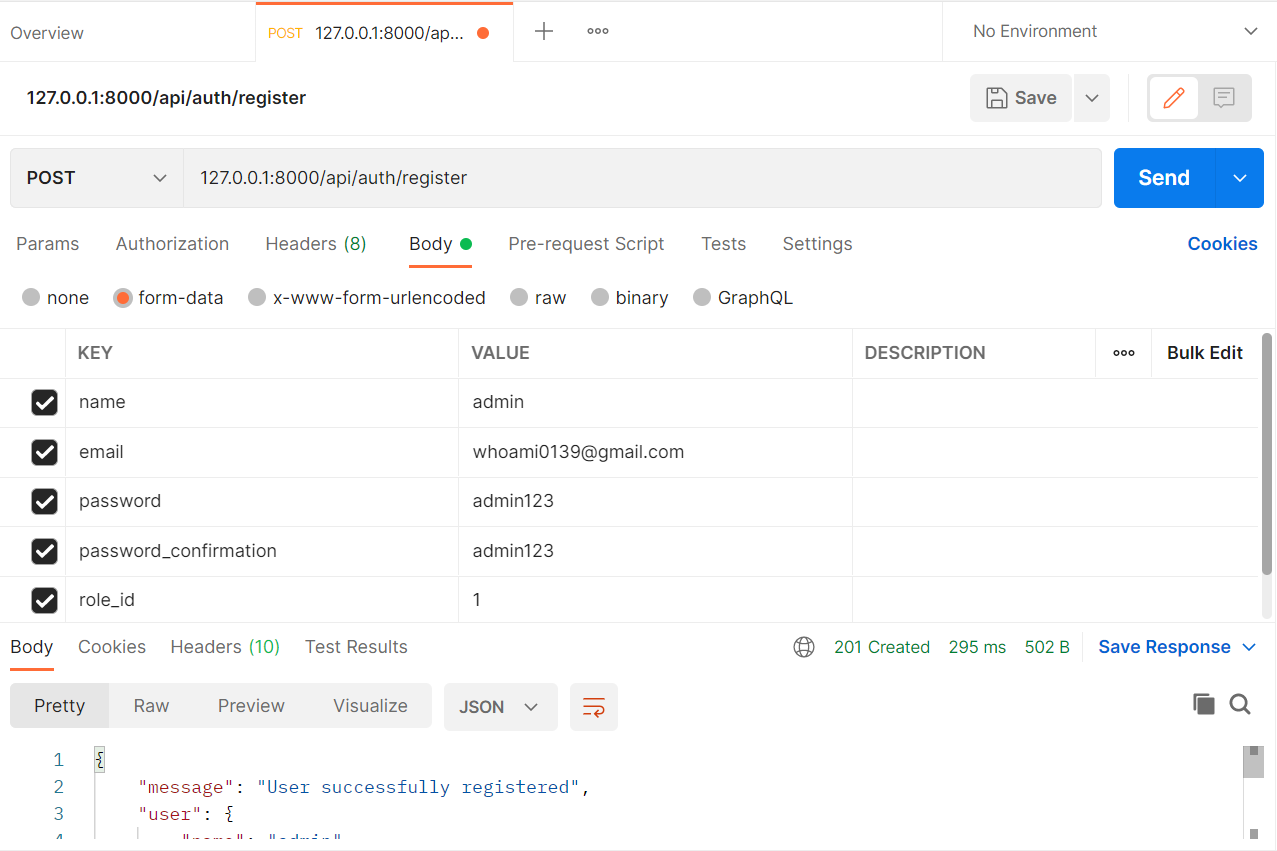


Figure 1. Register

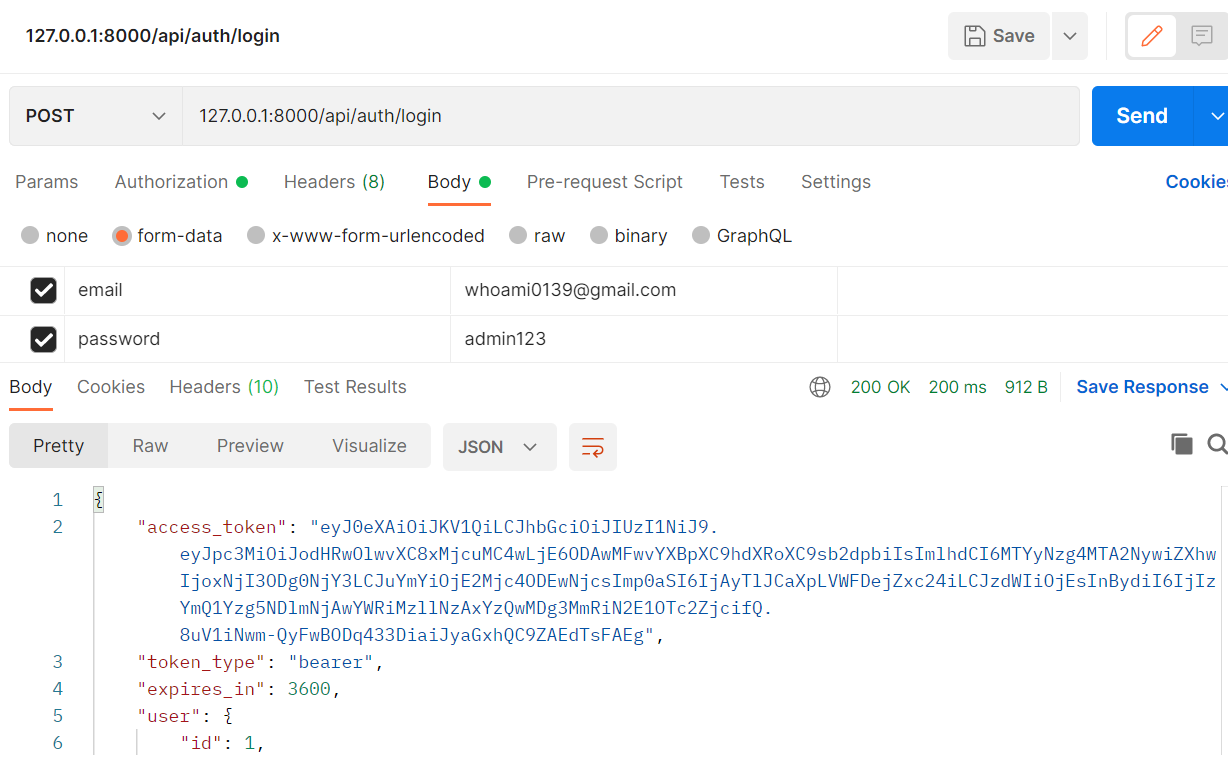


Figure 2. User Login

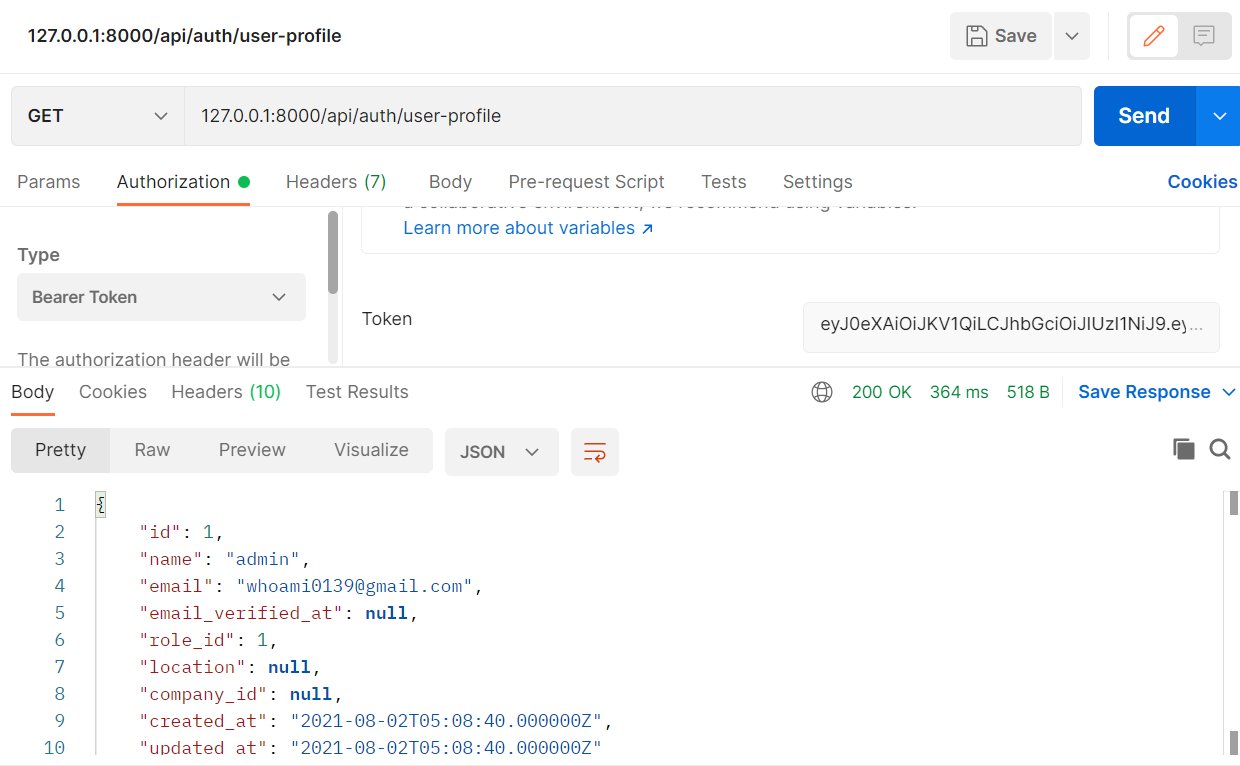


Figure 3. User Profile

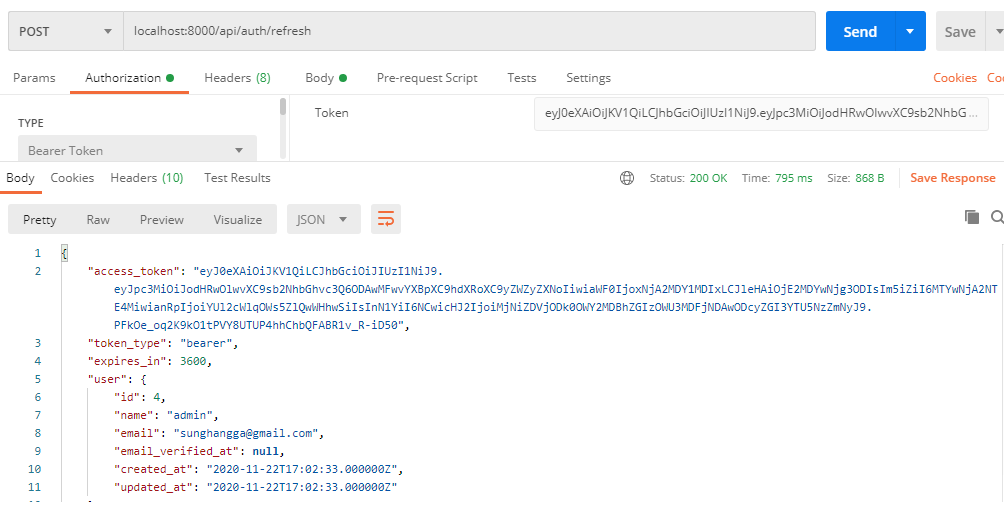


Figure 4. Refresh

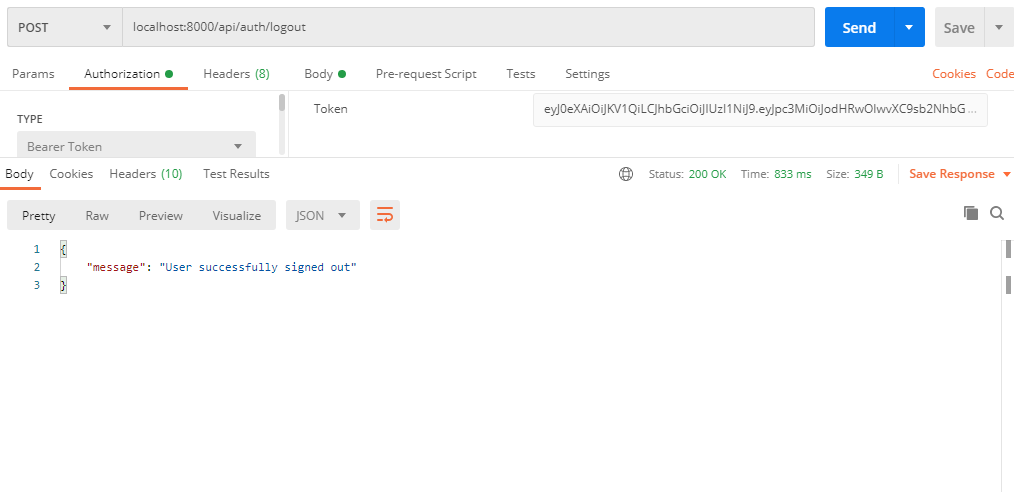


Figure 5. Logout

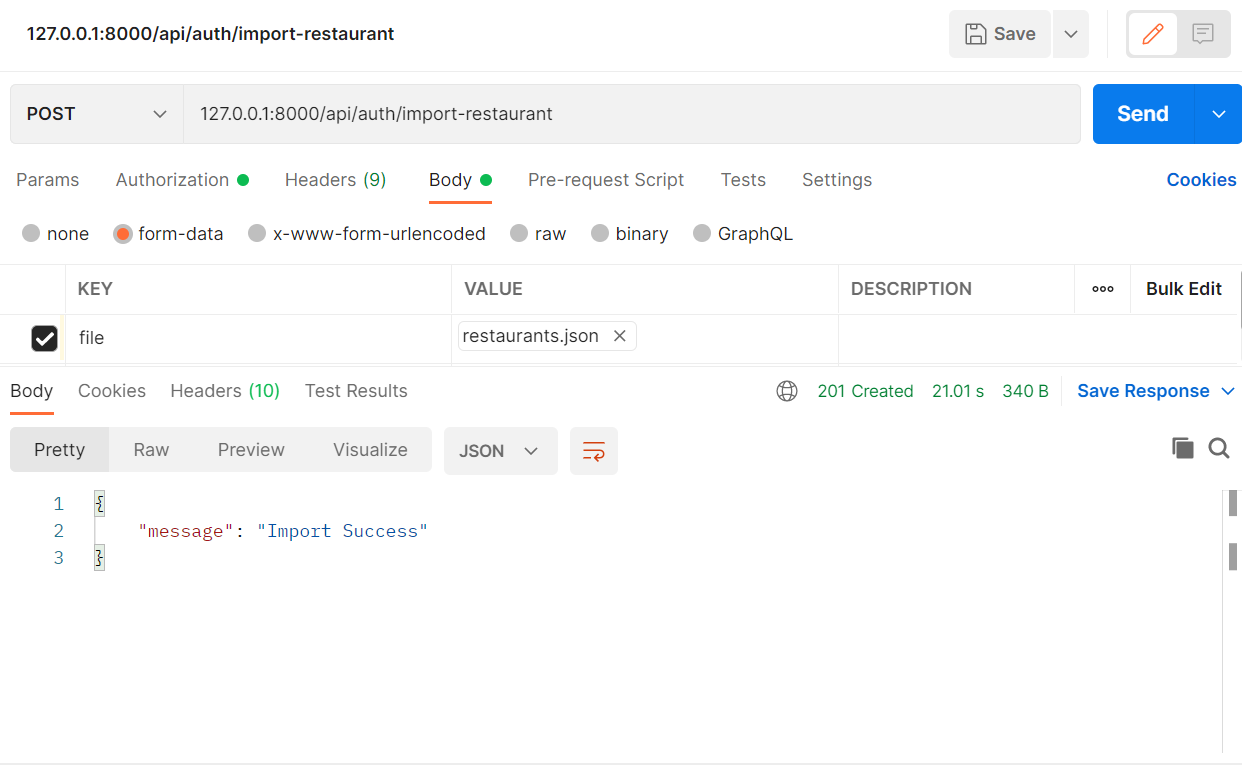


Figure 6. Import Restaurant JSON Data

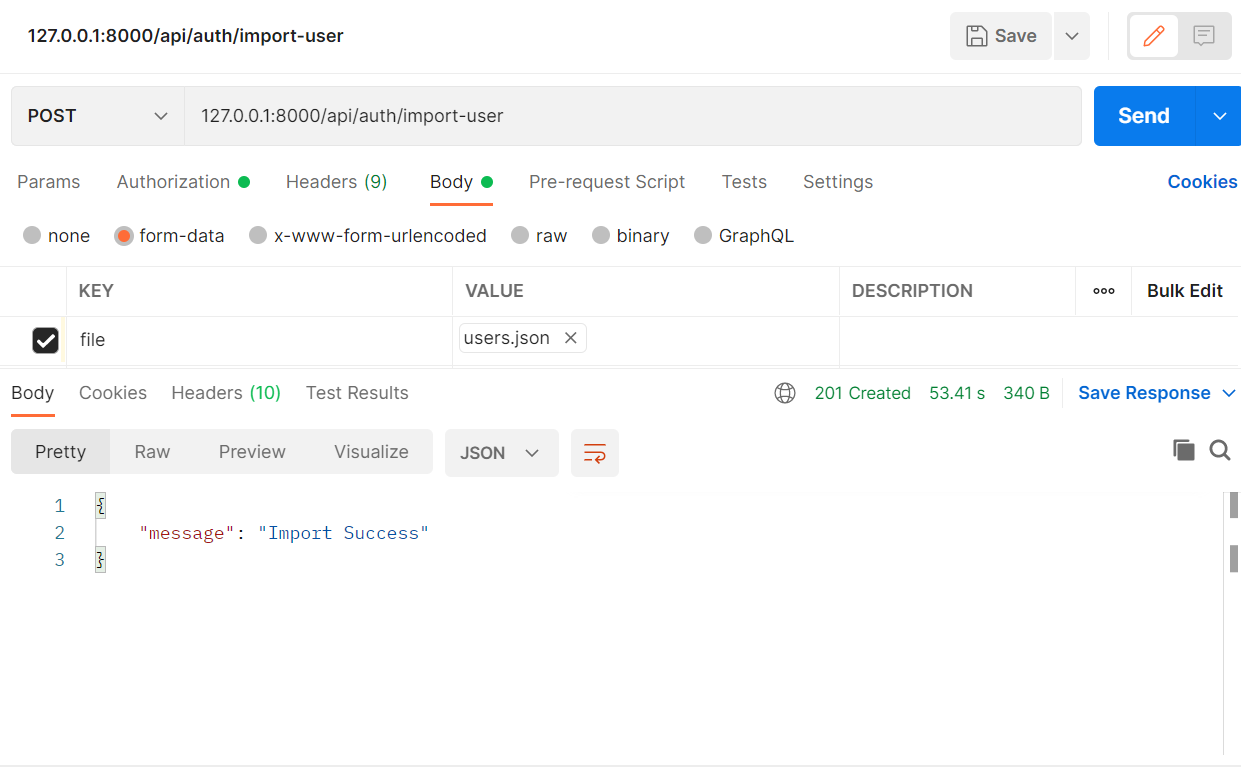


Figure 7. Import User JSON Data

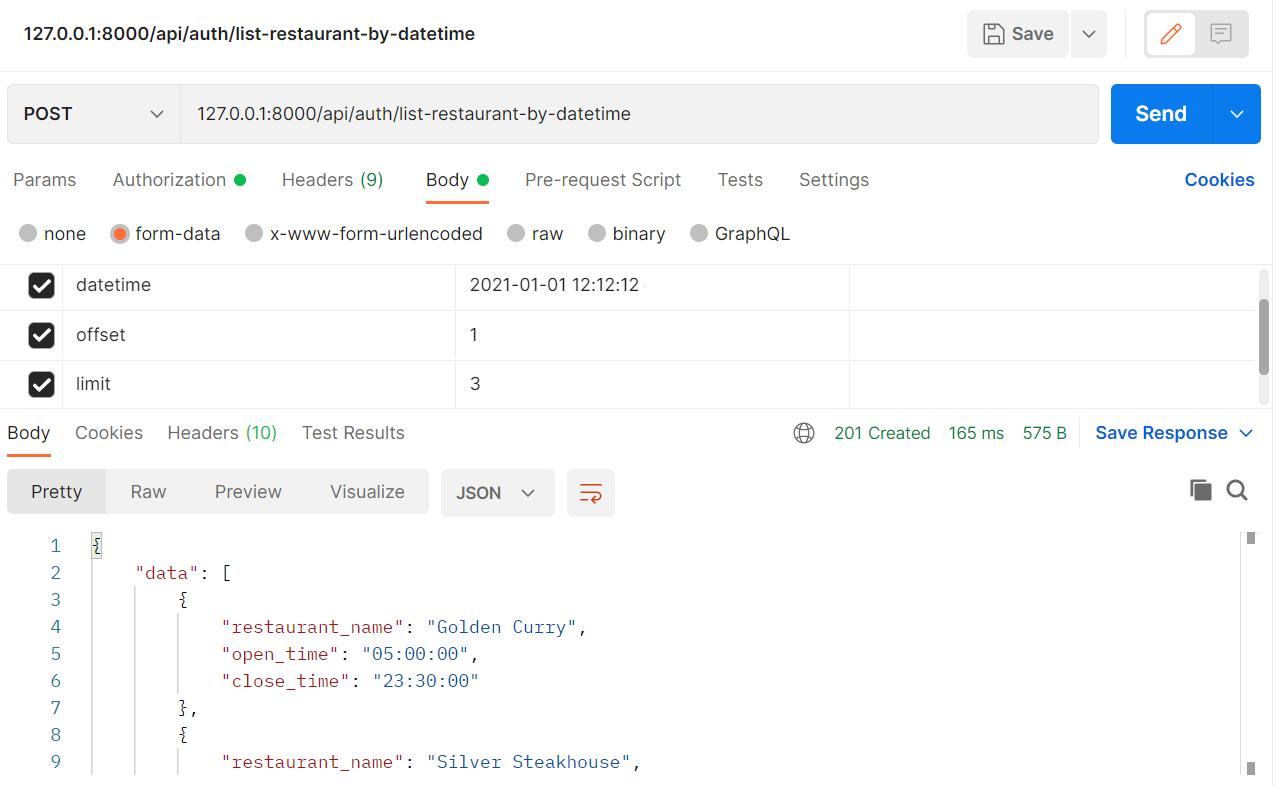


Figure 8. List all restaurants that are open at a certain datetime

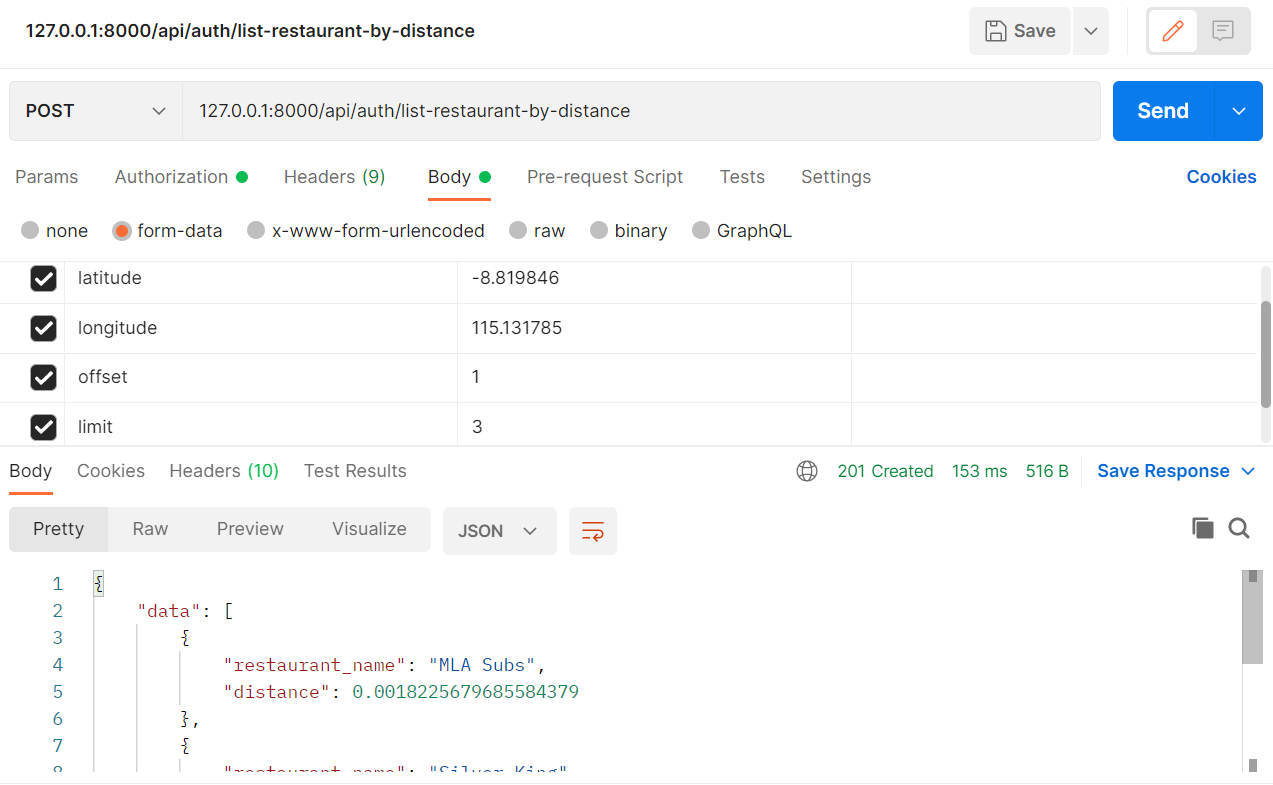


Figure 9. List all restaurants within the vicinity of the user’s location or (any location), ranked by distance

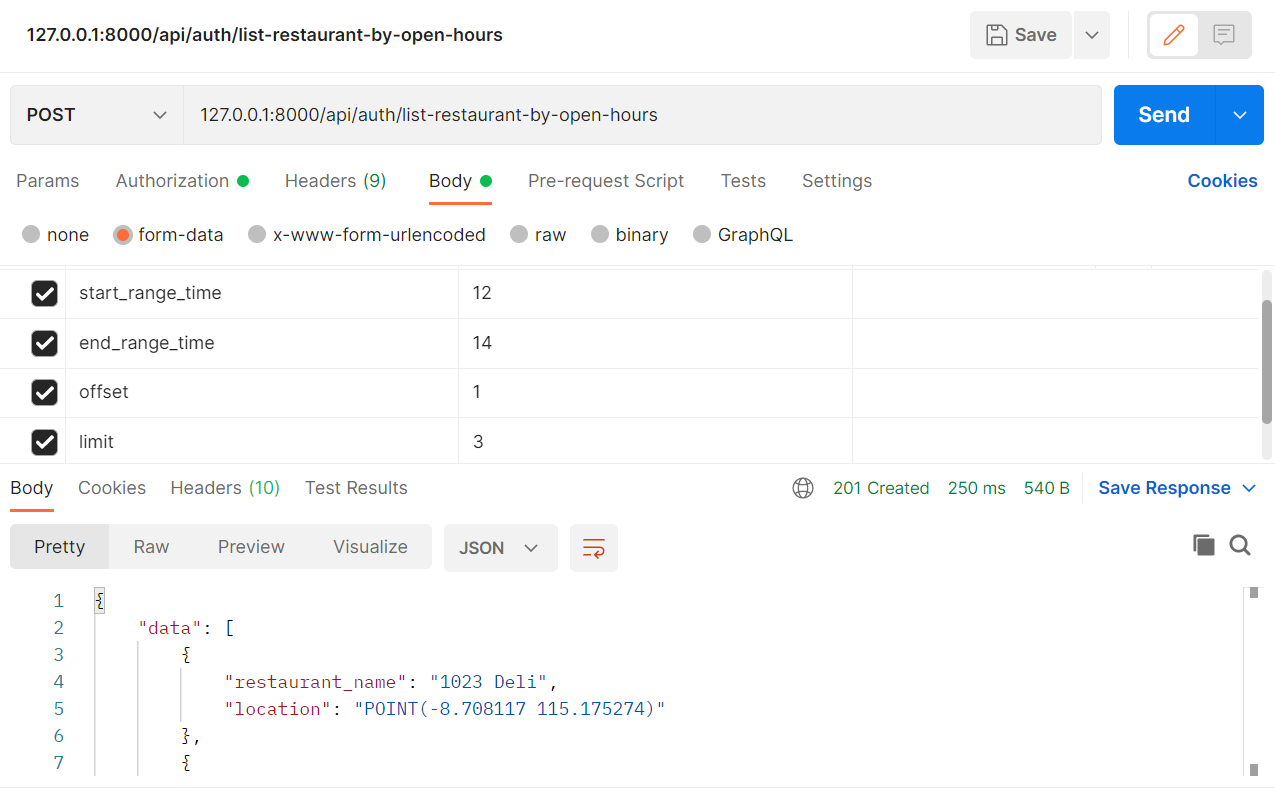


Figure 10. List all restaurants that are open for x-z hours per day or week

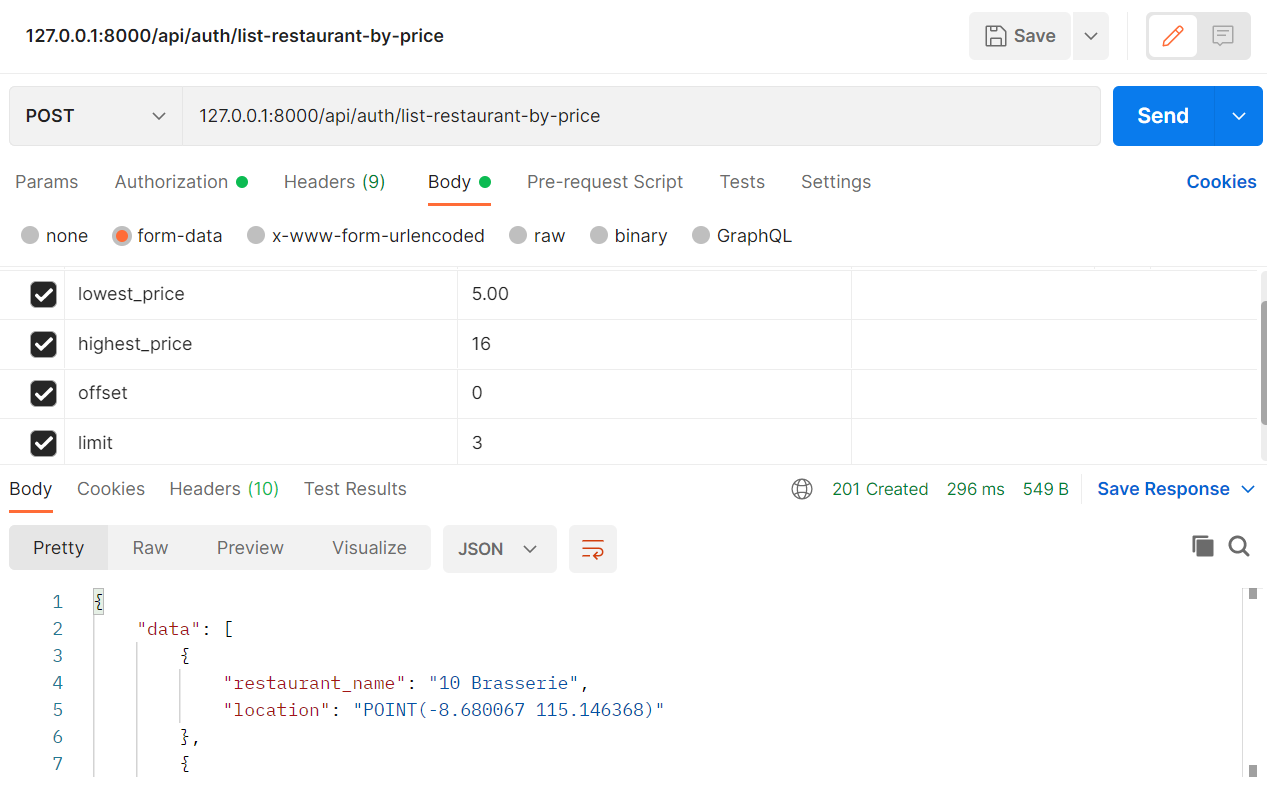


Figure 11. List all restaurants that have x-z number of dishes within a price range

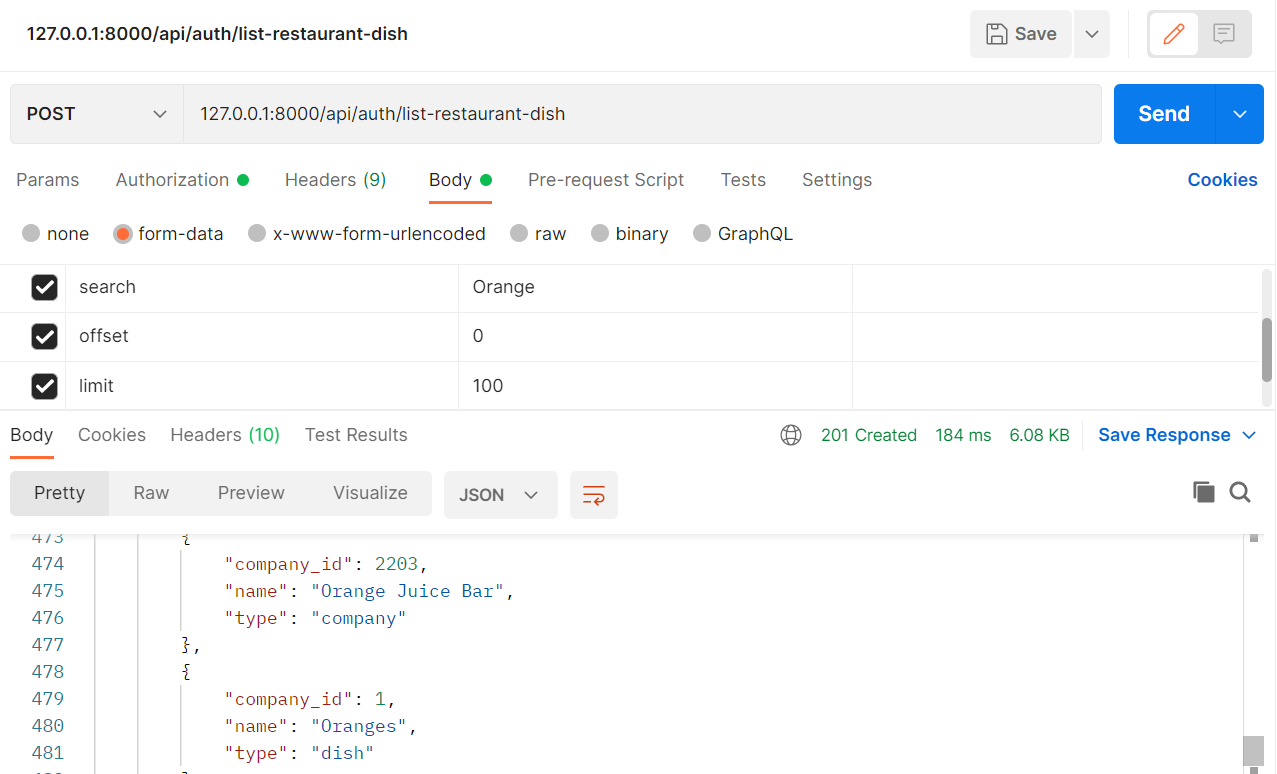


Figure 12. Search for restaurants or dishes by name, ranked by relevance to search term

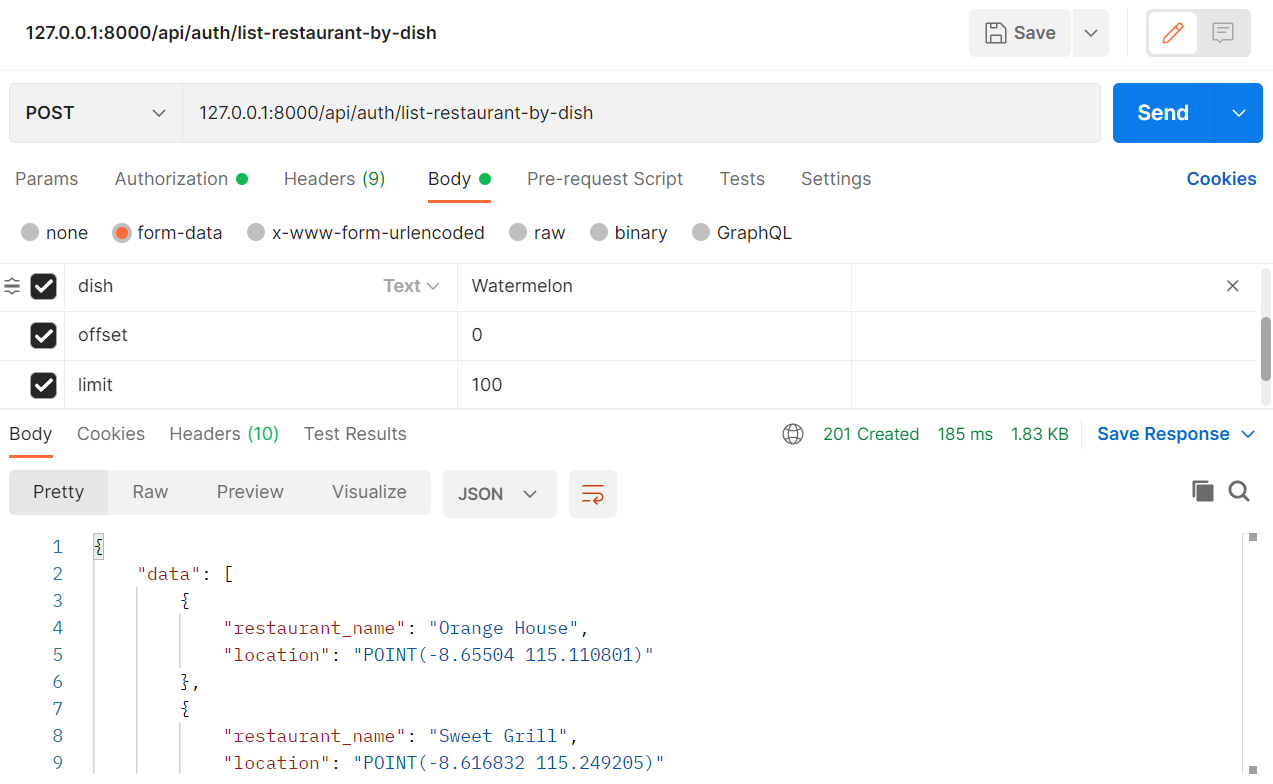


Figure 13. Search for restaurants that has a dish matching search term

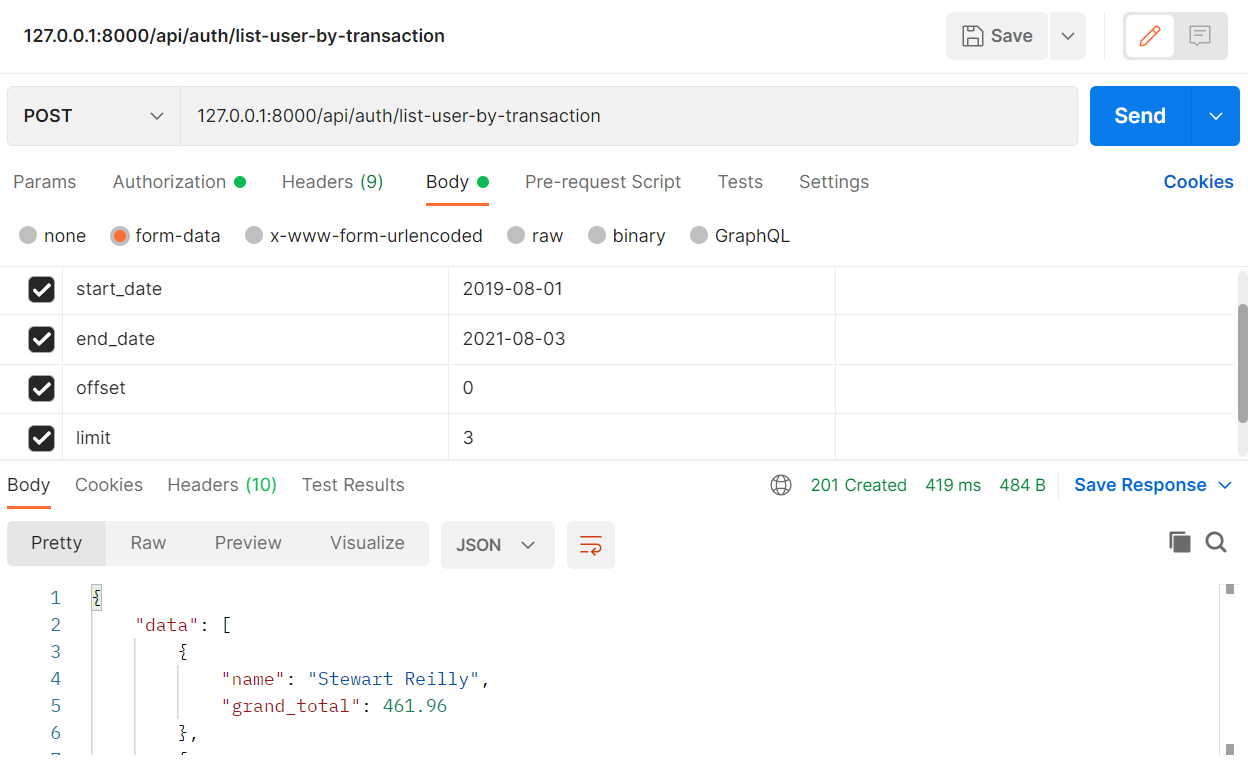


Figure 14. The top x users by total transaction amount within a date range

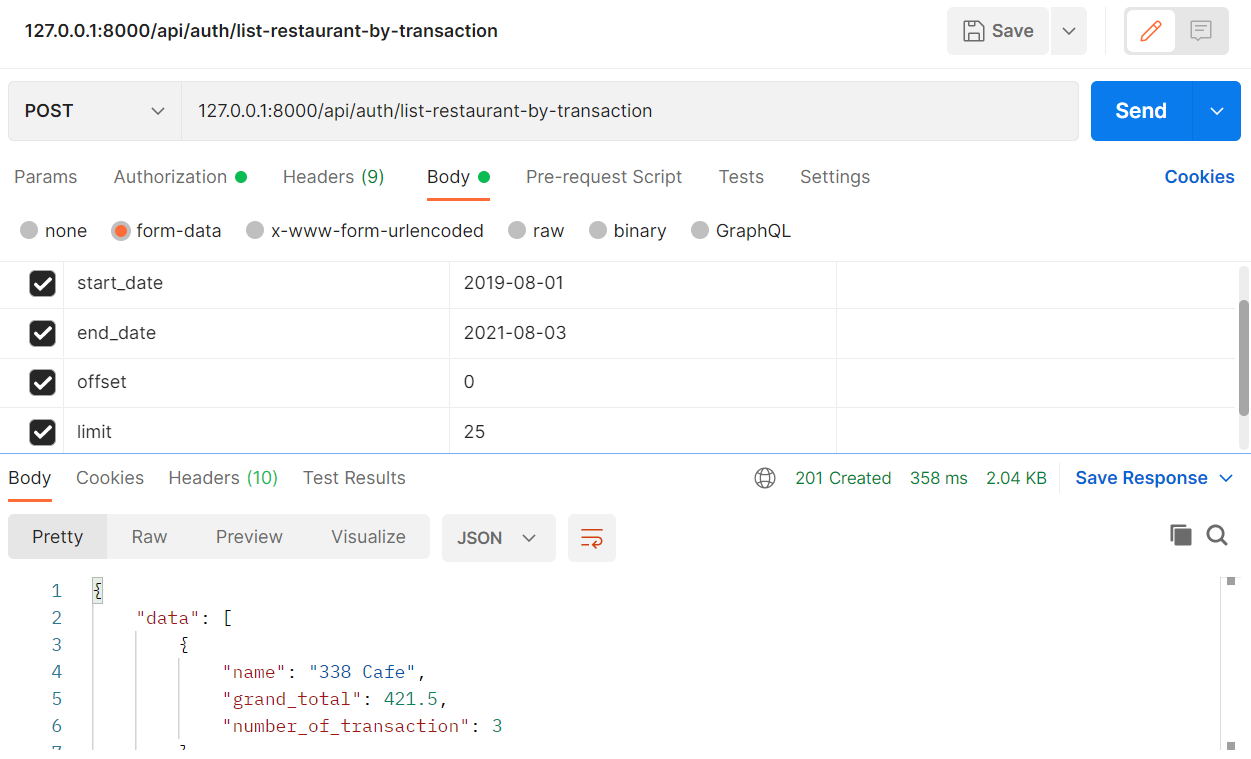


Figure 15. The most popular restaurants by transaction volume, either by number of transactions or transaction amount

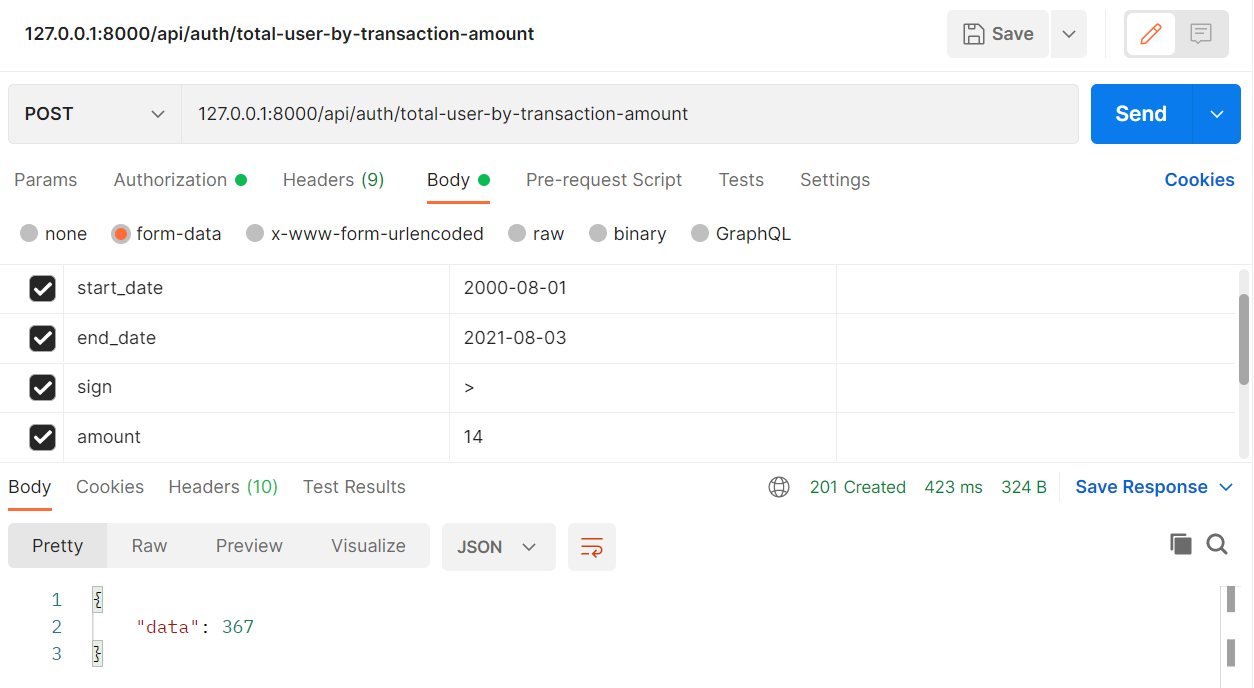


Figure 16. Total number of users who made transactions above or below $v within a date range

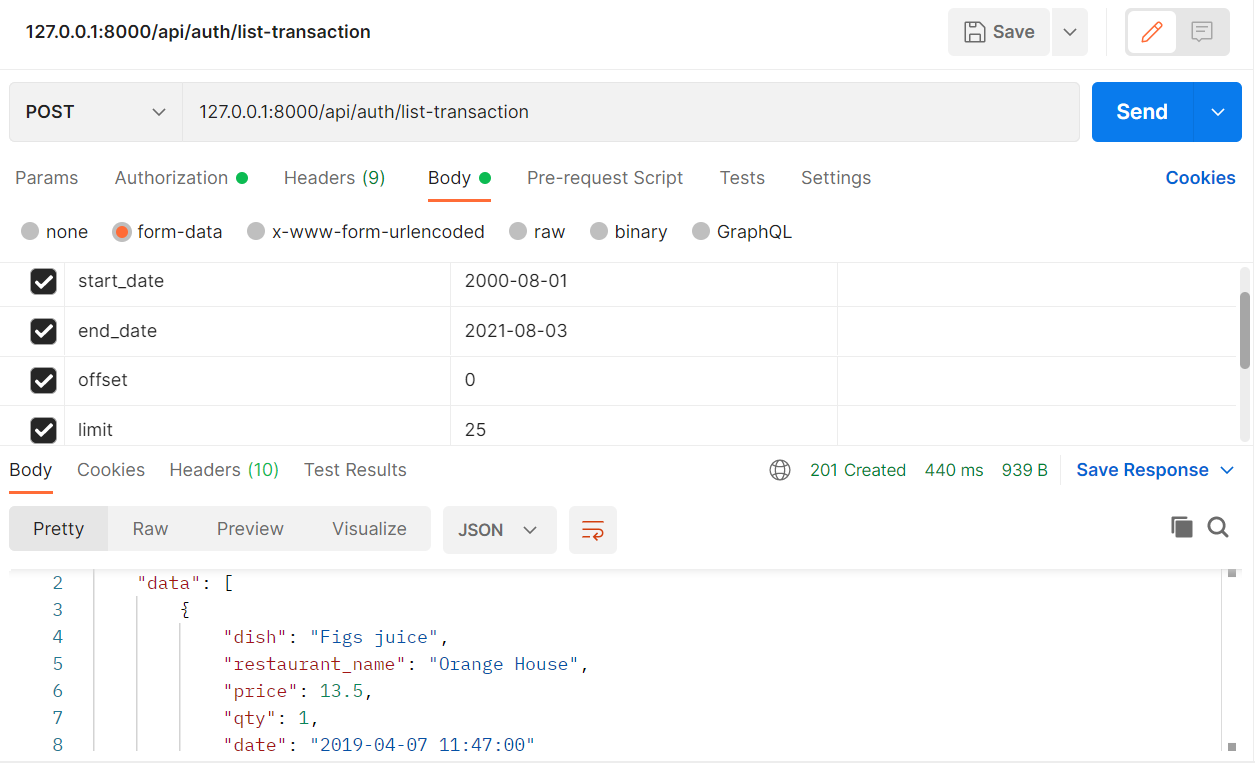


Figure 17. List all transactions belonging to a restaurant

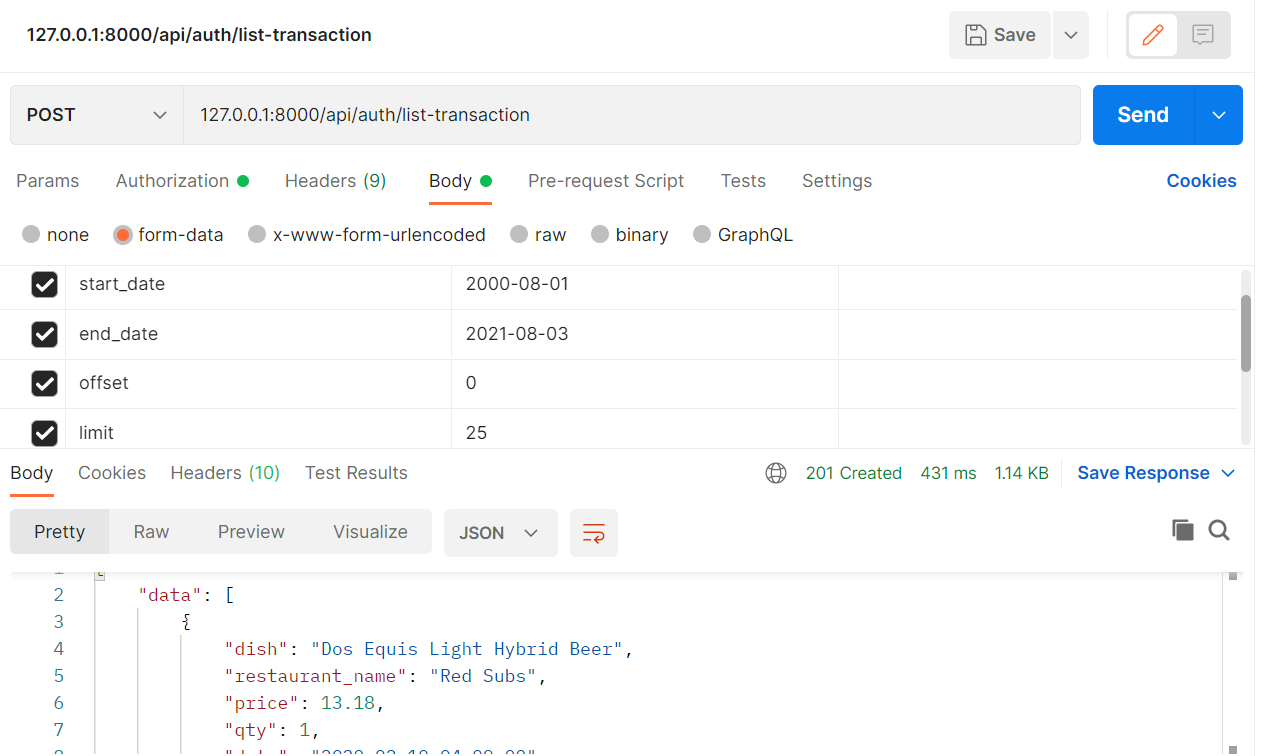


Figure 18. List all transactions belonging to a restaurant

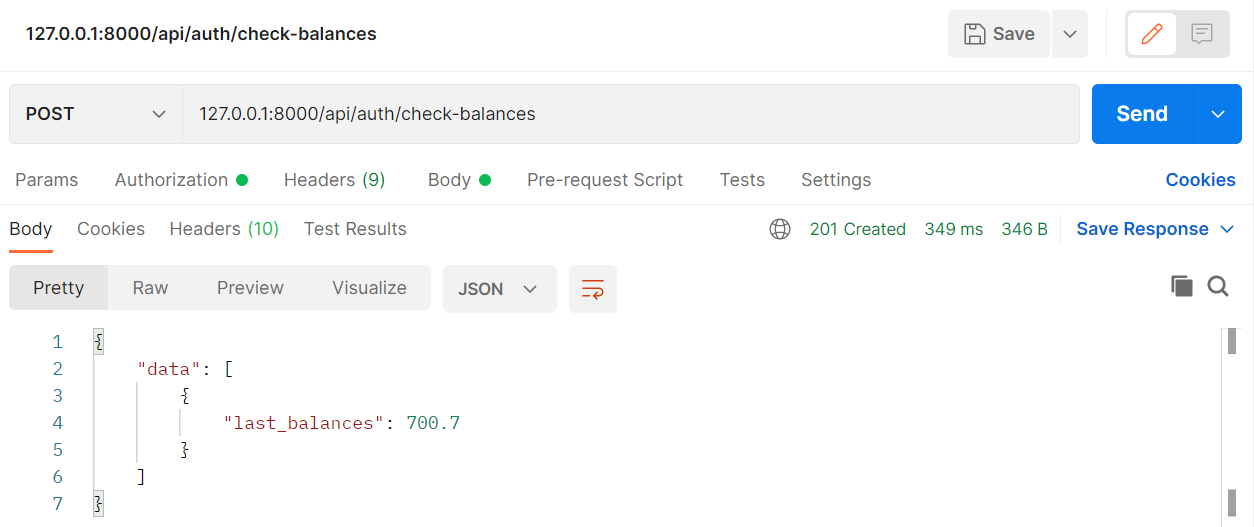


Figure 19. Check balances customer or company

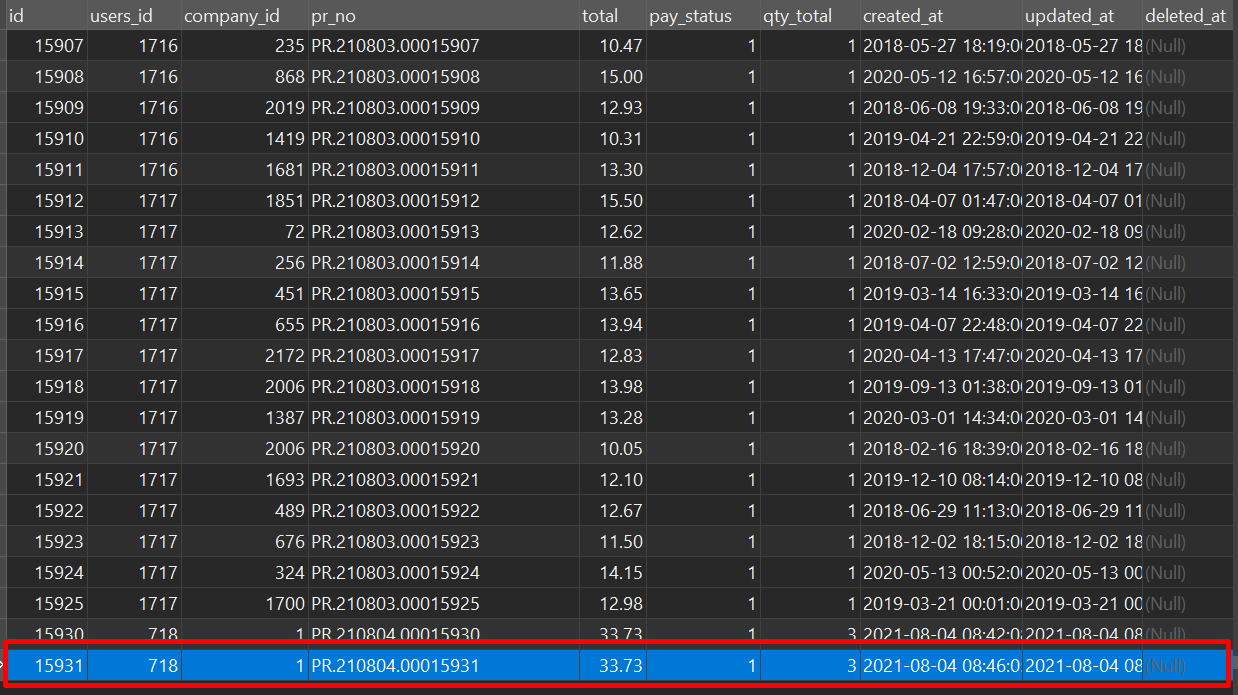


Figure 20. New data purchase order insert in purchases table

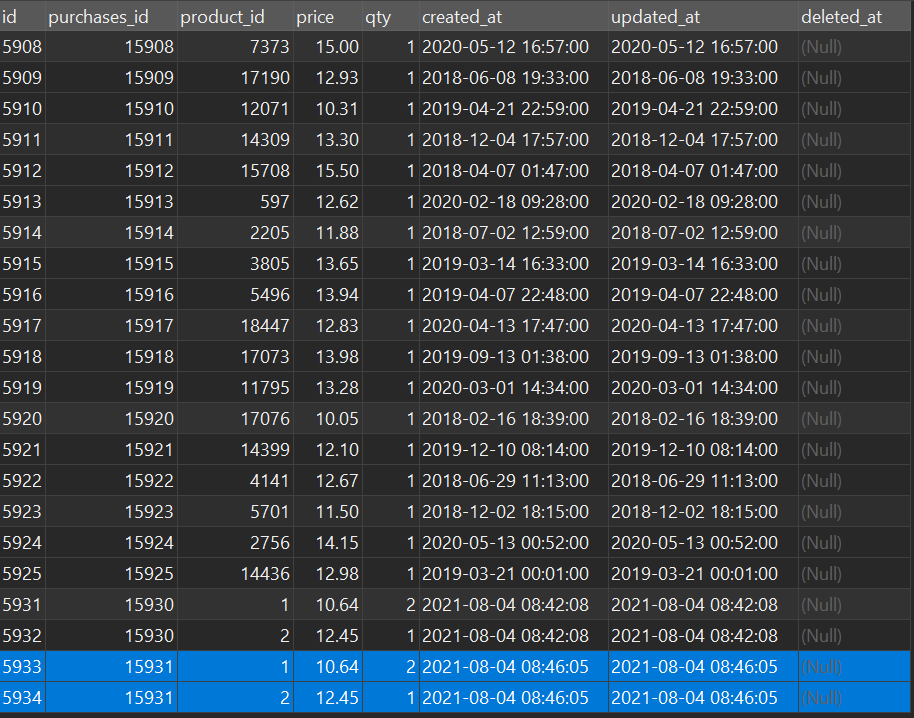


Figure 21. New data purchase order insert in purchase\_detail table

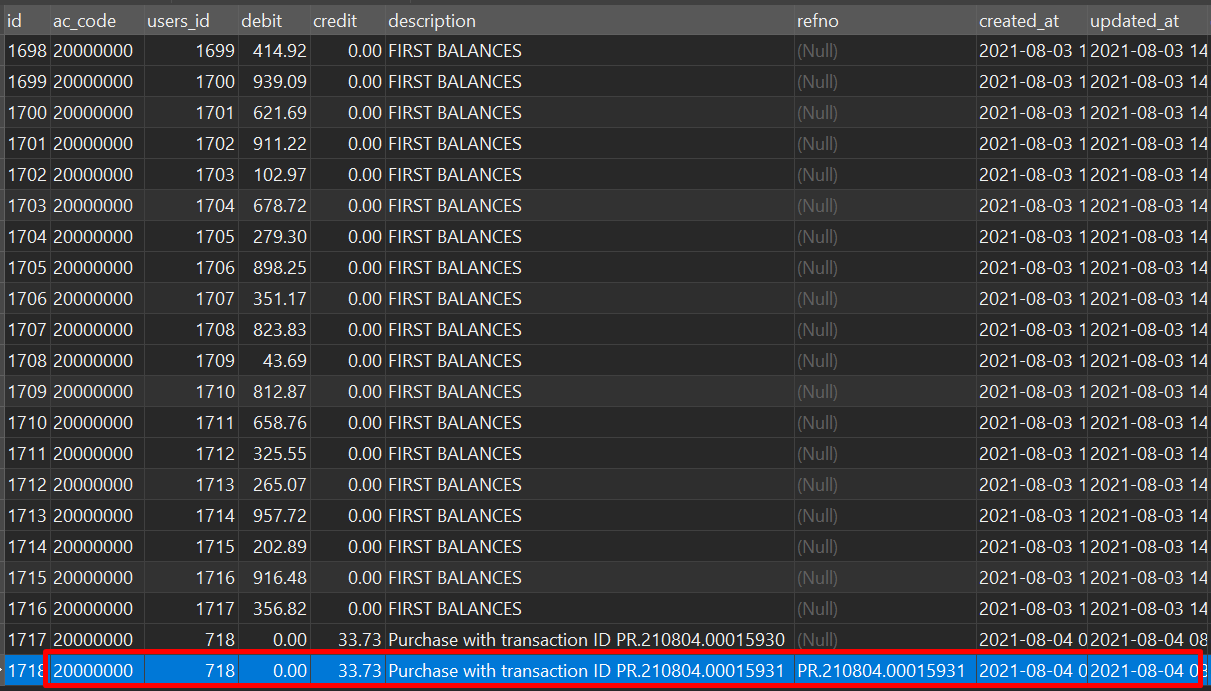


Figure 22. Insert data to customer\_balances as credit

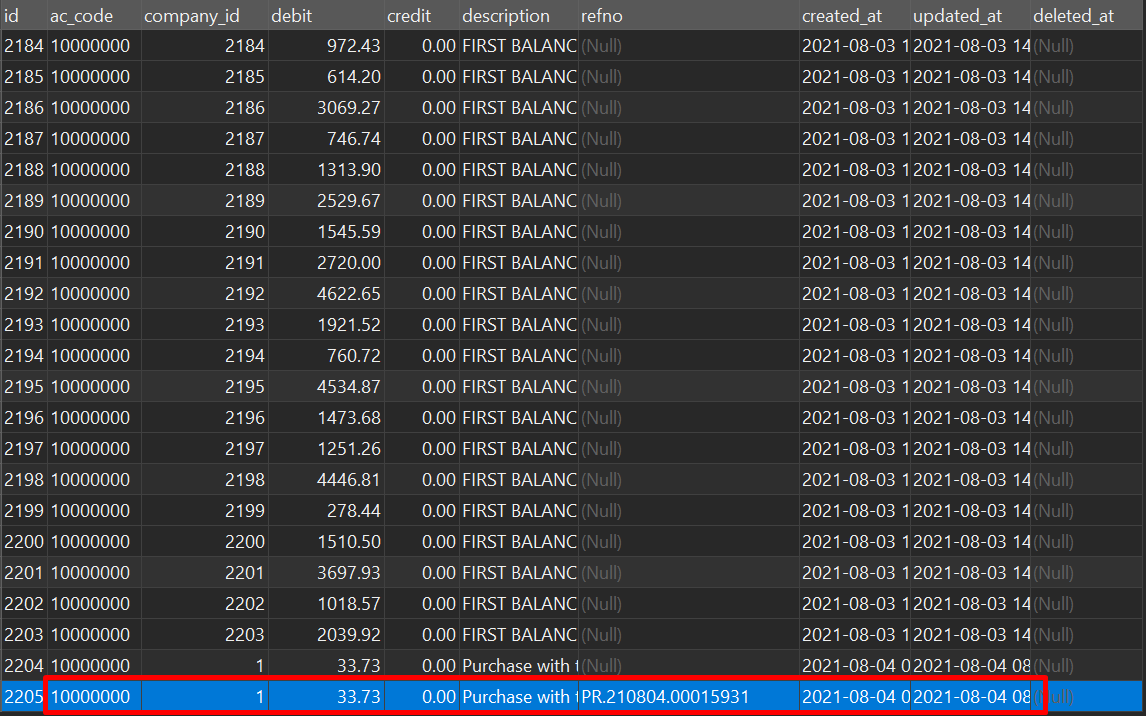


Figure 23. Insert data to company\_balances as debit

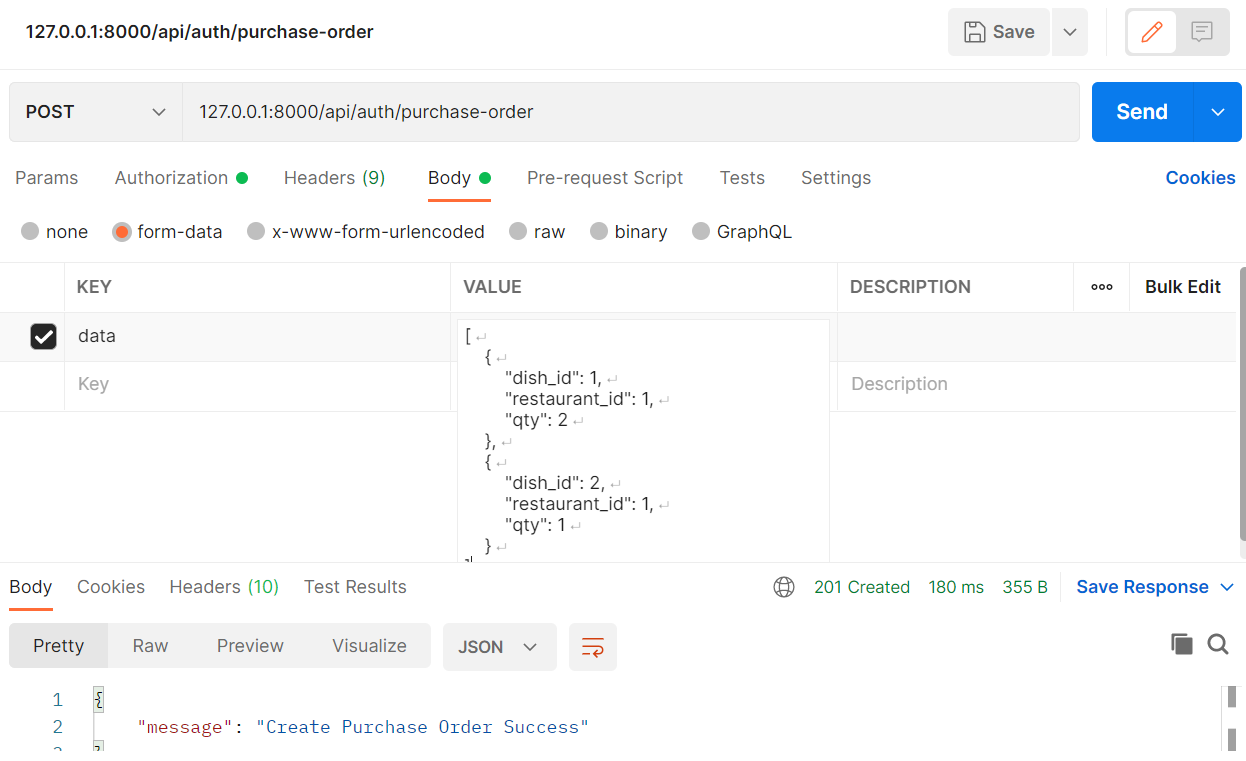


Figure 24. Create purchase order