Restaurant API User Manual

Group 10

August 8, 2025

Table of Contents

1. What is this?

This is a restaurant management system that helps you run your restaurant online. Customers can order food online, staff can manage orders and inventory, and managers can see how the business is doing.

2. Main Features

- **Customer Ordering**: People can browse your menu and place orders online
- **Staff Management**: Your staff can update orders, manage inventory, and add new menu items
- **Payment Processing**: Handles both cash and card payments
- **Order Tracking**: Customers get tracking numbers for takeout and delivery
- **Analytics**: See which dishes are popular and revenue

---

3. Setup

3.1 First Time Setup

Before customers can start ordering, you need to set up your restaurant:

3.1.1 Configure Restaurant Information

1. **Edit the config file**
2. **Update restaurant details**:

- Change to your restaurant's name
- Update the address, phone, and email
- Set your tax rate
- Adjust timezone if needed

```
// api/config/restaurant_config.json
{
    "restaurant": {
        "name": "Your Restaurant Name"
    },
    "contact": {
        "address": {
            "street": "123 Main Street",
            "city": "Your City",
            "state": "Your State",
            "zip_code": "12345"
        },
        "phone": "555-123-4567",
        "email": "info@yourrestaurant.com"
    },
    "business_settings": {
        "tax_rate": 0.08
    }
}
```

3.1.1 Setup the Databse

1. In `` `api/dependencies/config.py` `` add your database settings.

3.1.2 Access the API Interface

1. **Start the server**: Run `` `uvicorn api.main:app --reload` `` 2 **Verify the database has been built**: ???
2. **Open your browser**: Go to `` `http://localhost:8000/docs` ``
3. **See the interface**: You'll see all the different things you can do organized by user type

3.1.3 Add Your Menu

1. **Add menu items**: Use the staff interface or run `` `python scripts/add_test_food.py` `` for sample items
2. **Set up inventory**: Add resources and stock levels
3. **Test the system**: Make sure everything works before going live

3.2 Understanding the Layout

The API is organized into three main sections:

- **Customer Actions**: Things customers do (order food, pay, track orders)
- **Staff Actions**: Things your employees do (manage orders, update inventory)
- **Analytics**: Reports and data for managers

4. Customer Actions

4.1 How Customers Order Food

Customers follow this flow when ordering:

4.1.1 Browse the Menu

- **What it does**: Shows all available food items
- **How to use**: Use `` `GET /customer/menu` ``
- **What to enter**: Nothing required
- **What happens**: System retrieves all menu items that have sufficient ingredients
- **What is returned**: A list of all menu items with names, prices, descriptions, and calories

4.1.2 Add Items to Cart

- **What it does**: Adds food items to their order
- **How to use**: Use `` `POST /customer/add-to-cart` ``
- **What to enter**:
  - Menu item name from the dropdown
  - Quantity (how many they want)
- **What happens**: The system checks if there's enough ingredients and adds it to their cart, the order id is stored in the browser
- **What is returned**: The system confirms the item was added and returns the order ID

4.1.3 Choose Order Type

- **What it does**: Sets whether they're eating in, taking out, or getting delivery
- **How to use**: Use `` `POST /customer/choose-order-type` ``
- **What to enter**: Order type - choose from:
  - `` `dine_in` `` - eating at the restaurant
  - `` `takeout` `` - picking up their order
  - `` `delivery` `` - having it delivered
- **What happens**: Order type gets saved to their current order
- **What is returned**: Confirmation with order ID and selected order type

4.1.4 Add Customer Information

- **What it does**: Collects their contact information
- **How to use**: Use `` `POST /customer/add-customer-information` ``
- **What to enter**:
  - Name
  - Phone number
  - Email address
  - Address (required for delivery orders)
- **What happens**: Customer information gets saved and linked to the order

- **What is returned**: Customer details with assigned customer ID

4.1.5 Add Payment Method

- **How to use**: Use `` `POST /customer/add-payment` ``
- **What to enter**:
    - Payment type: Choose `` `cash` ``, `` `credit_card` ``, or `` `debit_card` ``
    - Card number: Only needed for card payments (leave blank for cash)
- **What happens**: Payment method gets attached to their order

4.1.6 Add Promo Code (Optional)

- **How to use**: Use `` `POST /customer/add-promo-code-to-order` ``
- **What to enter**: The promo code (like "SAVE10" or "STUDENT15")
- **What happens**: Discount gets applied to their order total
- **Note**: This step can be skipped if no promo code is available

4.1.7 Complete the Order

- **How to use**: Use `` `POST /customer/checkout` ``
- **What happens**:
    - Order goes to the kitchen
    - Receipt gets printed (for dine-in/takeout) or emailed (for delivery)
    - Tracking number is created (for takeout/delivery)

4.2 Other Customer Features

4.2.1 Remove Item from Cart

- **How to use**: Use `` `DELETE /customer/remove-from-cart` ``
- **What to enter**: The menu item ID they want to remove
- **What happens**: Item gets removed from their cart and inventory is updated
- **Note**: This only works before checkout - once an order is completed, it can't be changed

4.2.2 Track an Order

- **How to use**: Use `` `GET /customer/track_order/{tracking_number}` ``
- **What to enter**: The 8-digit tracking number they received
- **What they'll see**: Current status like "in progress" or "ready for pickup"

4.2.3 Leave a Review

- **How to use**: Use `` `POST /customer/review-dish` ``
- **What to enter**:
    - Dish name
    - Rating (1-5 stars)
    - Written review

5. Staff Actions

5.1 Managing Orders

Staff members handle orders through these actions:

5.1.1 View Orders by Status

- **What it does**: Shows all orders with a specific status
- **How to use**: Use `` `GET /staff_actions/orders/status/{status}` ``
- **Status options**:
    - `` `pending` `` - just placed, waiting to start
    - `` `confirmed` `` - accepted and being prepared
    - `` `in_progress` `` - currently being made
    - `` `awaiting_pickup` `` - ready for customer pickup
    - `` `out_for_delivery` `` - on the way to customer
    - `` `completed` `` - finished and/or delivered

5.1.2 View Orders by Date

- **What it does**: Shows orders from a specific time period
- **How to use**: Use `` `GET /staff_actions/orders/date-range` ``
- **What to enter**: Start date and end date

5.1.3 Update Order Status

- **What it does**: Changes an order's status as it progresses
- **How to use**: Use `` `PUT /staff_actions/orders/{order_id}/status` ``
- **What to enter**:
    - Order ID number
    - New status
- **Example flow**: pending → confirmed → in_progress → awaiting_pickup → completed

5.2 Managing the Menu

5.2.1 Add New Menu Items

- **What it does**: Creates a new dish that customers can order
- **How to use**: Use `` `POST /staff_actions/add-menu-item` ``
- **What to enter**:
    - Dish name
    - Description
    - Price
    - Calories
    - Food category (regular, vegetarian, vegan)
    - Ingredients needed (as a list)

5.3 Managing Inventory

5.3.1 Update Stock Levels

- **What it does**: Adds or removes ingredients from inventory
- **How to use**: Use **`` `PUT /staff_actions/update-stock` ``**
- **What to enter**:
  - Ingredient name
  - Amount to add (positive number) or remove (negative number)
- **Example**: Add 50 tomatoes or remove 10 cheese portions

5.4 Viewing Reports

5.4.1 Daily Revenue

- **What it does**: Shows how much money was made on a specific day
- **How to use**: Use **`` `GET /staff_actions/revenue/{target_date}` ``**
- **What to enter**: Date in YYYY-MM-DD format (like 2024-03-15)

5.4.2 Dish Performance

- **What it does**: Shows which dishes are selling well
- **How to use**: Use **`` `GET /staff_actions/dish-analytics` ``**
- **What you'll see**: List of dishes with sales numbers and ratings

5.4.3 Customer Reviews

- **What it does**: Shows all customer reviews and ratings
- **How to use**: Use **`` `GET /staff_actions/view-reviews` ``**
- **What you'll see**: All reviews with ratings and comments

---

6. Analytics

6.1 Understanding Your Business

Managers can use these tools to make better decisions:

6.1.1 Dish Rating Analysis

- **What it does**: Shows average ratings for each dish over time
- **How to use**: Use **`` `GET /analytics/dish-analytics-average-rating` ``**
- **Options**: Filter by week, month, year, or all time
- **What you'll see**: Which dishes customers love and which need improvement

6.1.2 Dish Popularity Analysis

- **What it does**: Shows how often each dish is ordered

- **How to use**: Use **`GET /analytics/dish-analytics-popularity`**
- **Options**: Filter by time period and sort by different criteria
- **What you'll see**: Your best-selling and least popular items

7. Administrator Actions

7.1 Database Utilities

Administrators have access to special endpoints for managing test data and the database:

7.1.1 Add Test Data

- **How to use**: Use **`POST /admin/add-test-data/{data_type}`**
- **What to enter**: **`data_type`** can be **`customers`** or **`food`**
- **What happens**:
    - If **`customers`**, adds a set of test customers to the database.
    - If **`food`**, adds a set of test food items to the database.
- **What is returned**: A message indicating success or failure.

7.1.2 Purge Database

- **How to use**: Use **`DELETE /admin/purge-db`**
- **What happens**: Deletes all data from all tables in the database. **This action cannot be undone!**
- **What is returned**: A message indicating whether the database was purged successfully or if there was an error.

8. Basic CRUD Operations

Most resources in the system (orders, order details, resources, menu items, ingredients, payment methods, customers, reviews, promotions, etc.) support the following standard actions:

Method	Endpoint Example	Description
GET	**`/resource/`**	Read all
POST	**`/resource/`**	Create new
GET	**`/resource/{item_id}`**	Read one by ID
PUT	**`/resource/{item_id}`**	Update by ID
DELETE	**`/resource/{item_id}`**	Delete by ID

Available Endpoints:

- **`` `/orders/` ``**
- **`` `/orderdetails/` ``**
- **`` `/resources/` ``**
- **`` `/menu_item_ingredients/` ``**
- **`` `/menu_items/` ``**
- **`` `/payment_method/` ``**
- **`` `/customers/` ``**
- **`` `/reviews/` ``**
- **`` `/promotions/` ``**

8.1 Create (POST)

- **What it does**: Makes something new
- **How it works**: Fill out the form and click "Execute"
- **Example**: Adding a new customer or menu item

8.2 Read (GET)

- **What it does**: Shows existing information
- **How it works**: Just click "Execute" (sometimes you need to enter an ID)
- **Example**: Viewing all orders or a specific customer

8.3 Update (PUT)

- **What it does**: Changes existing information
- **How it works**: Enter the ID of what you want to change, then the new information
- **Example**: Updating a customer's phone number

8.4 Delete (DELETE)

- **What it does**: Removes something permanently
- **How it works**: Enter the ID of what you want to delete
- **Warning**: Be careful - this can't be undone!