Order Taking Application API Documentation

Welcome to the OTA API documentation! This API provides endpoints for managing user accounts, orders, and restaurant operations.

Base URL

The base URL for all API endpoints is:

```
https://5rg19e6tog.execute-api.ap-south-1.amazonaws.com/dev
```

Authentication

Authentication is required for certain endpoints. You need to obtain an JWT access token by signing in.

Endpoints

1.User Registration

Endpoint:

```
PUT /user/register
```

Request:

```
Content-Type: application/json
{
    "name": "Peter Pan",
    "password": "pass12345",
    "mobileNo": "9123456877",
    "emailId": "peterpan2145@gmail.com"
}
```

Parameter	Description
`name`	User name. Minimum accepted character 3, Maximum accepted character 100.
`password`	Provide user password. Minimum accepted character 8, Maximum accepted character 50.
`mobileNo`	User mobile number
`emailId`	User email id

Response:

```
{
    "message": "User account is created.",
    "data": {
        "userId": 83,
        "emailId": "peterpan2145@gmail.com",
        "name": "Peter Pan",
        "mobileNo": "9123456877"
    }
}
```

Parameter	Description
`message`	User accoount created message.
`data`	JSON object containing registered user data with the following parameters: • `userId` : User's unique identifer. • `emailId` : Registered user's email id. • `name` : Name of the user. • `emailId` : Mobile number of the user.

2.User Login

Endpoint:

```
POST /user/login
```

Request:

```
Content-Type: application/json
{
    "password": "pass12345",
    "emailId": "peterpan2145@gmail.com"
}
```

Parameter	Description
`password`	User password.
`emailId`	User email id

Response:

```
{
        "message": "Successfully logged in.",
        "token": "<USERJWTTOKEN>",
        "data": {
            "userId": 83,
            "emailId": "peterpan2145@gmail.com",
            "name": "Peter Pan",
            "mobileNo": "9123456877"
```

```
        }
    }
```

Parameter	Description
`message`	Login success message.
`token`	You'll receive a JWT token which should be included in the Authorization header for authentication. `Authorization:JWT <USERJWTTOKEN>`
`data`	JSON object containing registered user data with the following parameters: • `userId` : User's unique identifer. • `emailId` : Registered user's email id. • `name` : Name of the user. • `emailId` : Mobile number of the user.

3.User Logout

Endpoint:

```
GET /user/logout
```

Request:

```
Authorization: JWT <USERJWTTOKEN>
```

Response:

```
{
    "message": "User logged out."
}
```

Parameter	Description
`message`	User logout message.

4.User Cart - Add Item

Endpoint:

```
PUT /user/:id/item/:item_id
```

Request:

```
PUT /user/83/item/1
Authorization: JWT <USERJWTTOKEN>
Content-Type: application/json
{
    "qty":1,
```

```
    "restaurantId":1
}
```

Parameter	Description
`qty`	No.of quantity has to be ordered
`restaurantId`	Restaurant Id.

Response:

```
{
    "message": "Item added to cart."
}
```

Parameter	Description
`message`	Item added message.

5.User Cart - Remove Item

Endpoint:

```
DELETE /user/:id/item/:item_id
```

Request:

```
DELETE /user/83/item/3
Authorization: JWT <USERJWTTOKEN>
Content-Type: application/json
```

Response:

```
{
    "message": "Item removed from cart."
}
```

Parameter	Description
`message`	Item removed from user cart message.

6.User Cart Items

Endpoint:

```
GET /user/:id/cart
```

Request:

```
GET /user/83/cart
Authorization: JWT <USERJWTTOKEN>
```

Response:

```
{
"data": [
    {
        "id": 1,
        "user_id": 83,
        "res_id": 1,
        "item_id": 1,
        "qty": 1,
        "created_date": 1713898284,
        "name": "5 Leg Pc & 2 Dips Bucket",
        "category": 1,
        "price": "519.00",
    }
],
"subTotal": 519
}
```

Parameter	Description
`data`	JSON object containing cart information with the following parameters: • `id` : Cart Id. • `user_id` : User Id. • `res_id` : Restaurant id. • `item_id` : Item id. • `qty` : Item quantity. • `name` : Item name. • `category` : Category id. • `price` : Item price.
`subTotal`	Subtotal of the cart.

7.User Cart - Place Order

Endpoint:

```
PUT /user/:id/cart/place_order
```

Request:

```
PUT /user/83/cart/place order
Authorization: JWT <USERJWTTOKEN>
Content-Type: application/json
{}
```

Response:

```
{
    "message": "Order is placed.",
```

```
    "orderId": 70
}
```

Parameter	Description
`message`	Order created message.
`orderId`	Order id.

8.User - Get All Order

Endpoint:

```
GET /user/:id/orders?page=1
```

Request:

```
GET /user/83/orders?page=1
Authorization: JWT <USERJWTTOKEN>
```

Parameter	Description
`page`	Pagination number.

Response:

```
{
    "data": [
        {
            "id": 70,
            "user id": 83,
            "res id": 1,
            "subtotal": "519.00",
            "tax amount": "0.00",
            "total": "519.00",
            "status": "ACCEPTED",
            "created_date": 1714070965
        }
    ]
}
```

Parameter	Description
`data`	JSON object containing order data with the following parameters: • `id` : Order Id. • `user_id` : User Id. • `res_id` : Restaurant Id. • `subtotal` : Subtotal amount. • `total` : Total amount. • `status` : Order status. • `created_date` : Order created date in epoch seconds.

9.User - Get Order

Endpoint:

```
GET /user/:id/order/:order_id
```

Request:

```
GET /user/83/order/70
```

Response:


```
{
    "data": [
        {
            "id": 70,
            "user id": 83,
            "res id": 1,
            "subtotal": "519.00",
            "tax amount": "0.00",
            "total": "519.00",
            "status": "ACCEPTED",
            "created date": 1714070965,
            "items": [
                {
                    "id": 70,
                    "order id": 70,
                    "item id": 1,
                    "qty": 1,
                    "price": "519.00",
                    "name": "5 Leg Pc & 2 Dips Bucket"
                }
            ]
        }
    ]
}
```


Parameter	Description

`data`	JSON object containing order data with the following parameters:
	• `id` : Order Id.
	• `user_id` : User Id.
	• `res_id` : Restaurant Id.
	• `subtotal` : Subtotal amount.
	• `total` : Total amount.
	• `status` : Order status.
	• `created_date` : Order created date in epoch seconds.
	• `items` : items orders.
	  ◦ `order_id` : Order Id.
	  ◦ `item_id` : Item Id.
	  ◦ `qty` : Item quantity.
	  ◦ `price` : Item price.
	  ◦ `name` : Item name.

10.User - Get Active Order

Endpoint:

```
GET /user/:id/activeOrder
```

Request:

```
GET /user/83/activeOrder
Authorization: JWT <USERJWTTOKEN>
```

Response:

```
{
    "data": [
        {
            "id": 70,
            "user id": 83,
            "res id": 1,
            "subtotal": "519.00",
            "tax amount": "0.00",
            "total": "519.00",
            "status": "ACCEPTED",
            "created_date": 1714070965
        }
    ]
}
```

Parameter	Description
`data`	JSON object containing order data with the following parameters: • `id` : Order Id. • `user_id` : User Id. • `res_id` : Restaurant Id. • `subtotal` : Subtotal amount. • `total` : Total amount. • `status` : Order status. • `created_date` : Order created date in epoch seconds.

11.Restaurant - Get Token

Restaurant login is not available, call this api to get restaurant jwt token. Which used as Authorization token.

Endpoint:

```
GET /restaurant/token
```

Request:

```
GET /restaurant/token
```

Response:

```
{
    "token": <RESTAURANTJWTTOKEN>
}
```

Parameter	Description
`token`	You'll receive a JWT token which should be included in the Authorization header for authentication. `Authorization:JWT <RESTAURANTJWTTOKEN>`

12.Restaurant - Add Menu Item

Endpoint:

```
PUT /restaurant/:id/item
```

Request:

```
PUT /restaurant/1/item
Authorization: JWT <RESTAURANTJWTTOKEN>
Content-Type: application/json
{
    "name": "Classic Chicken Burger",
    "category": 6,
    "price": 119
}
```

Parameter	Description
`name`	Item name.
`category`	Item category.
`price`	Item price

Response:

```
{
        "message": "Item Is added.",
    "data": {
        "id": 41,
        "res id": 1,
        "name": "Classic Chicken Burger",
        "category": 6,
        "status": "ACTIVE",
        "price": "119.00",
        "is available": 1,
        "created_date": 1714072727
    }
}
```

Parameter	Description
`message`	Item created message.
`data`	JSON object containing item data with the following parameters:

	• `id` : Item Id. • `qty` : Item quantity. • `name` : Item name. • `category` : Category id. • `price` : Item price.

13.Restaurant - Get All Menu Items

Endpoint:

```
GET /restaurant/:id/items
```

Request:

```
GET /restaurant/1/items?page=1
Authorization: JWT <RESTAURANTJWTTOKEN>
```

Response:

```
{
        "data": [
            {
                "id": 1,
                "res id": 1,
                "name": "5 Leg Pc & 2 Dips Bucket",
                "category": 1,
                "status": "ACTIVE",
                "price": "519.00",
                "is available": 1,
                "created_date": 1713898284
            }
}
```

Parameter	Description
`data`	JSON object containing item data with the following parameters: • `id` : Item Id. • `qty` : Item quantity. • `name` : Item name. • `category` : Category id. • `price` : Item price.

14.Restaurant - Update Item Detail

Endpoint:

```
PUT /user/register
```

Request:

```
 PATCH /restaurant/1/item/1
Authorization: JWT <RESTAURANTJWTTOKEN>
{"is_available":true}
```

Parameter	Description
`name`	Item name.
`price`	Item Price
`is_available`	Is item available to order

Response:

```
{
        "message": "Item Is updated.",
        "data": {
            "id": 1,
            "res_id": 1,
            "name": "5 Leg Pc & 2 Dips Bucket",
            "category": 1,
            "status": "ACTIVE",
            "price": "519.00",
            "is_available": 1,
            "created_date": 1713898284
        }
    }
```

Parameter	Description
`message`	Item updated message.
`data`	JSON object containing updated item data.

15.Restaurant - Delete Item

Endpoint:

```
 DELETE /restaurant/:id/item/:item_id
```

Request:

```
 DELETE /restaurant/1/item/13
Authorization: JWT <RESTAURANTJWTTOKEN>
Content-Type: application/json
```

Response:

```
 {
    "message": "Item is removed."
}
```

Parameter	Description
`message`	Item removed message.

16.Restaurant - Update Order Status

Endpoint:

```
PATCH /restaurant/:id/order/:order_id
```

Request:

```
pATCH /restaurant/1/order/70
Authorization: JWT <RESTAURANTJWTTOKEN>
Content-Type: application/json
{"status":"PREPARING"}
```

Parameter	Description
`status`	Update Order status with one of `[COMPLETED, PREPARING]`.

Response:

```
{
        "message": "Order status updated."
    }
```

Parameter	Description
`message`	Order status updated message.

Status Codes

Status Code	Description
200	Successful request
201	Resource created (e.g., added new user)
400	Bad request (e.g., missing required parameters)
401	Unauthorized (invalid API key)
404	Not found (requested resource not found)
500	Internal server error