Order Management API

# Introduction

The Order Management API facilitates the creation, management, and tracking of customer orders. This backend service is implemented with RESTful endpoints for CRUD operations (Create, Read, Update, Delete). It is designed to work independently of a frontend and can seamlessly integrate with a database in the future.

# Features

The Order Management API provides the following core features:

1. **Create Orders**: Add new customer orders with relevant details.
2. **Retrieve Orders**: Fetch all or specific orders.
3. **Update Orders**: Modify the details or status of an order.
4. **Delete Orders**: Remove orders when necessary.

# API Endpoints

* **Base URL:** The API is accessible at **/orders/**
* Endpoints and HTTP Methods:

|  |  |  |
| --- | --- | --- |
| Method | Endpoint | Description |
| GET | /orders/ | |  | | --- | | Retrieve all orders. |  |  | | --- | |  | |
| POST | /orders/ | |  | | --- | | Create a new order. |  |  | | --- | |  | |
| GET | /orders/{id}/ | |  | | --- | | Retrieve details of a specific order. |  |  | | --- | |  | |
| PUT | /orders/{id}/ | |  | | --- | | Update an existing order. |  |  | | --- | |  | |
| DELETE | /orders/{id}/ | Delete an order. |

## Examples

**Create an Order (POST)**

* **Endpoint:** /orders/
* **Request Type:** POST
* **Request Body (JSON):**

{

"customer\_name": "John Doe",

"customer\_email": "johndoe@example.com",

"products": [

{"name": "Laptop", "quantity": 1},

{"name": "Mouse", "quantity": 2}

],

"total\_price": 1200.50,

"status": "PENDING"

}

* **Response:**

{

"id": 1,

    "customer\_name": "Max Mustermann",

    "customer\_email": "max.mustermann@example.com",

    "products": {

        "products": [

            {

                "name": "Laptop",

                "quantity": 1

            },

            {

                "name": "Mouse",

                "quantity": 2

            }

        ]

    },

    "total\_price": "1200.50",

    "status": "PENDING",

    "created\_at": "2025-01-24T18:42:56.459584Z",

    "updated\_at": "2025-01-24T18:42:56.459609Z"

}

## Testing the API

* **Manual Testing**

Use tools like **Postman** or **cURL** to manually test the endpoints.

* **Automated Testing**

test cases class *OrderTestCase(TestCase)* in *tests.py* to ensure API functionality:

Run the tests using:

**python manage.py test**

## Additional Notes

1. Database Integration: The system is prepared to integrate with a database, though it is not currently required.
2. Frontend Integration: The API is independent of the frontend but provides all necessary endpoints for frontend interaction.
3. Error Handling: Validation errors for missing or incorrect data are handled by the serializer, and appropriate error messages are returned.