Mock Payment Processing API

# Mock Payment Processing

## Introduction

The Mock Payment Processing API simulates payment processes within an e-commerce system. It enables the creation, management and updating of payment statuses for orders. The API is implemented as a RESTful service and supports various payment providers.

## Features

* **Create Payment:** Create new payment requests for orders.
* **Get Payment Details:** View information about a specific payment.
* **Update Payment Status:** Set the status of a payment to "PENDING", "COMPLETED" or "FAILED".
* **Webhook Support:** Enables external systems to submit payment status updates.

## API Endpoints

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

|  |  |  |
| --- | --- | --- |
| Method | Endpoint | Description |
| GET | /payments/ | |  | | --- | | Retrieve all payments. |  |  | | --- | |  | |
| POST | / payments/ | |  | | --- | | Create a new payment. | |
| GET | /payments/{id}/ | |  | | --- | | Retrieve details of a specific payment. |  |  | | --- | |  | |
| PUT | /payments/{id}/ | |  | | --- | | Update an existing payment. |  |  | | --- | |  | |
| DELETE | /payments/{id}/ | Delete a payment. |
| POST | /payments/webhook/ | Webhook to update payment status. |

## Moch Payment Processing Webhook

The webhook system allows real-time updates of payment statuses from external payment systems.

### Webhook Endpoint

Endpoint: **/payments/webhook**

### Example for expected Request Format (POST)

{

"payment\_id": "550e8400-e29b-41d4-a716-446655440000",

"status": "COMPLETED"

}

### Possible Responses

#### Successful Update

{

"message": "Payment status updated: COMPLETED"

}

#### Missing or Invalid Payment

{

"error": "Payment not found"

}

Or

{

"error": "Order not associated with this payment"

}

#### Invalid Status

{

"error": "Invalid status"

}

## Example for Payment Processing

**Create Payment (POST)**

* **Endpoint:** /payments/
* **Request Type:**
* **Request Body (JSON):**

{

"order": 1,

"amount": 99.99,

"currency": "USD",

"provider": "stripe",

"status": "PENDING"

}

* **Response:**

{

"payment\_id": "550e8400-e29b-41d4-a716-446655440000",

"order": 1,

"amount": 99.99,

"currency": "USD",

"provider": "stripe",

"status": "PENDING",

"created\_at": "2025-02-20T14:35:56.459584Z"

}

## Testing the API

* **Manual Testing**

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

* **Automated Testing**

Test case class *EmailNotificationTestCase (TestCase)* in *tests.py* to ensure API functionality:

Run the tests using:

**python manage.py test**

## Additional Notes

* Database Integration: The API is designed to integrate seamlessly with a relational database.
* Error Handling: Non-existent or invalid payment IDs trigger structured error messages.
* Security: Endpoints are secured with transaction locks ***(transaction.atomic())*** to prevent data inconsistencies.