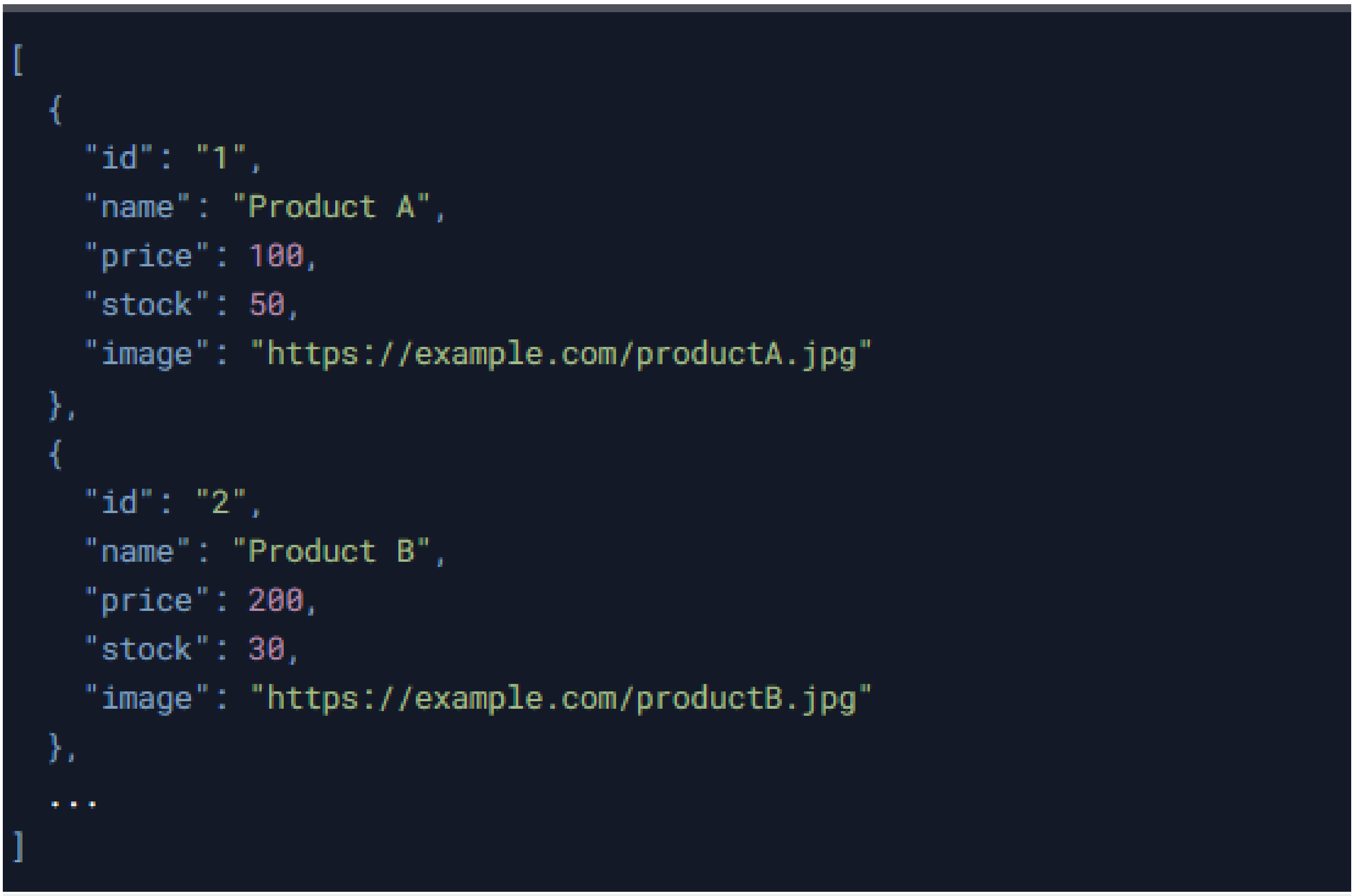
API Requirements

Endpoint: /products

Method: GET

Description:
Fetch all available products from Sanity.

Response:



```
[
  {
    "id": "1",
    "name": "Product A",
    "price": 100,
    "stock": 50,
    "image": "https://example.com/productA.jpg"
  },
  {
    "id": "2",
    "name": "Product B",
    "price": 200,
    "stock": 30,
    "image": "https://example.com/productB.jpg"
  },
  ...
]
```

Response Fields:

- id (string): Unique identifier for the product.
- name (string): Name of the product.
- price (number): Price of the product.
- stock (number): Available stock quantity.
- image (string): URL to the product image.

Status Codes:

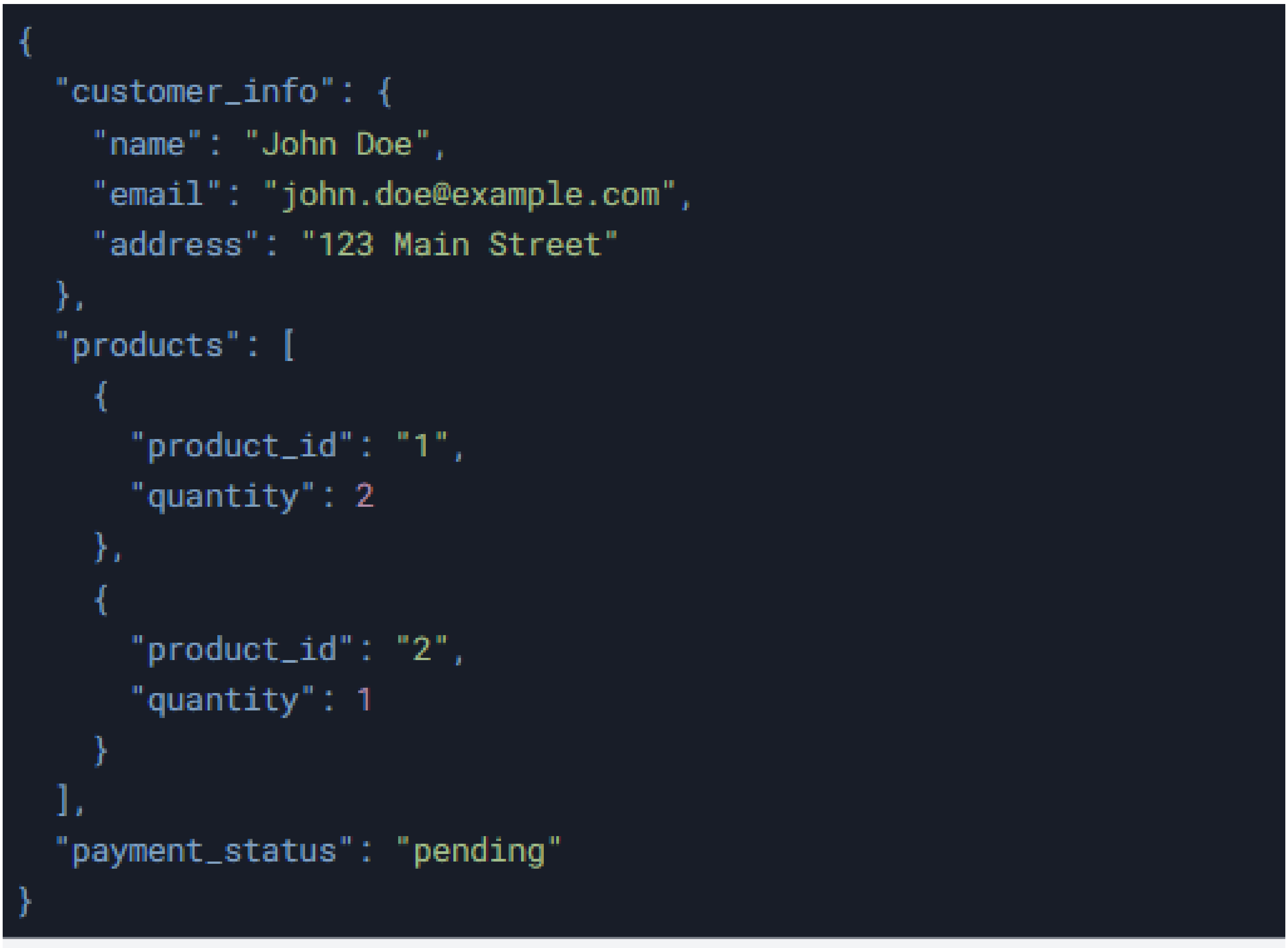
- 200 OK: Successful operation.

- 500 Internal Server Error: Server error.

Endpoint: /orders

Method: POST

Description:
Create a new order in Sanity.

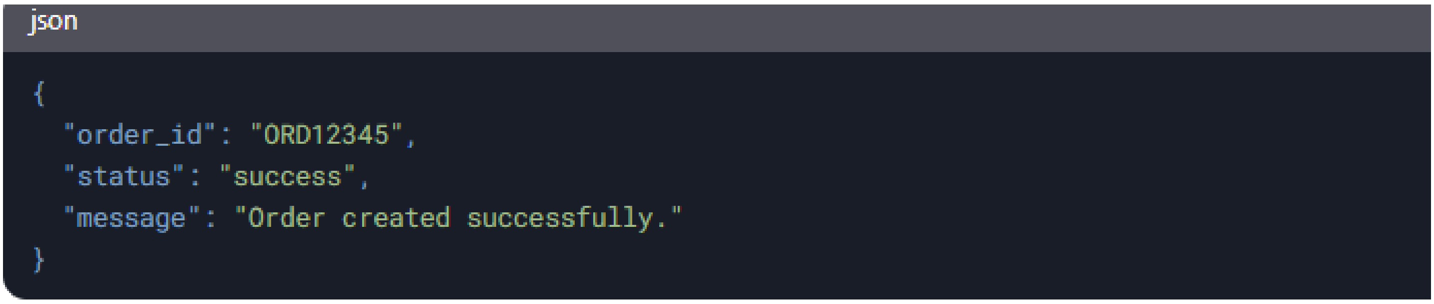


```
{
  "customer_info": {
    "name": "John Doe",
    "email": "john.doe@example.com",
    "address": "123 Main Street"
  },
  "products": [
    {
      "product_id": "1",
      "quantity": 2
    },
    {
      "product_id": "2",
      "quantity": 1
    }
  ],
  "payment_status": "pending"
}
```

Request Fields:

- customer_info (object):
  - name (string): Customer's name.
  - email (string): Customer's email.
  - address (string): Shipping address.
- products (array of objects):
  - product_id (string): ID of the product.
  - quantity (number): Quantity ordered.
- payment_status (string): Status of the payment (e.g., "pending", "paid").

Response:



```json
{
  "order_id": "ORD12345",
  "status": "success",
  "message": "Order created successfully."
}
```

1. **Response Fields:**

- **order_id (string):** Unique identifier for the order.
- **status (string):** Status of the operation.
- **message (string):** Success or error message.

Status Codes:

- **201 Created:** Order created successfully.
- **400 Bad Request:** Invalid request payload.
- **500 Internal Server Error:** Server error.

Endpoint: /shipment

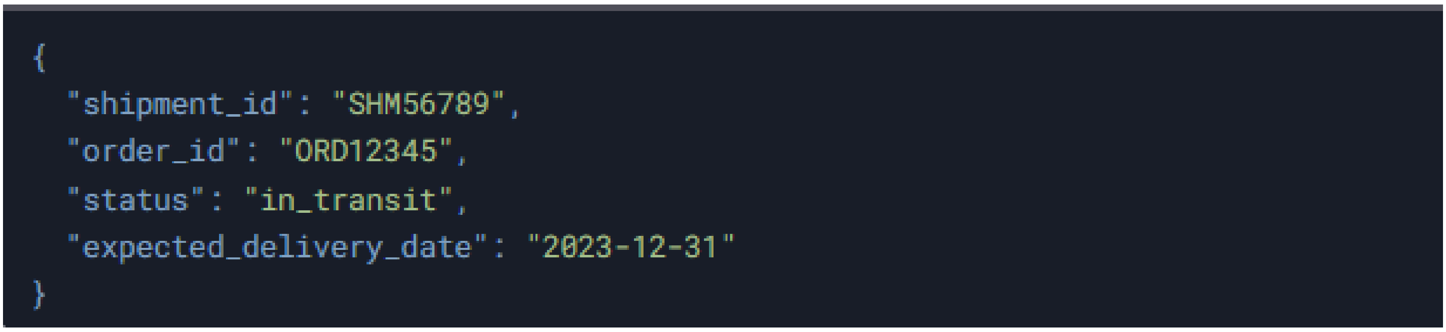
Method: GET

Description:
Track order status via third-party API.

Query Parameters:

- order_id (string, required): Unique identifier for the order.

Response:



```
{
  "shipment_id": "SHM56789",
  "order_id": "ORD12345",
  "status": "in_transit",
  "expected_delivery_date": "2023-12-31"
}
```

Response Fields:

- shipment_id (string): Unique identifier for the shipment.
- order_id (string): Unique identifier for the order.
- status (string): Current status of the shipment (e.g., "pending", "in_transit", "delivered").

- expected_delivery_date (string): Expected delivery date in ISO format.

Status Codes:

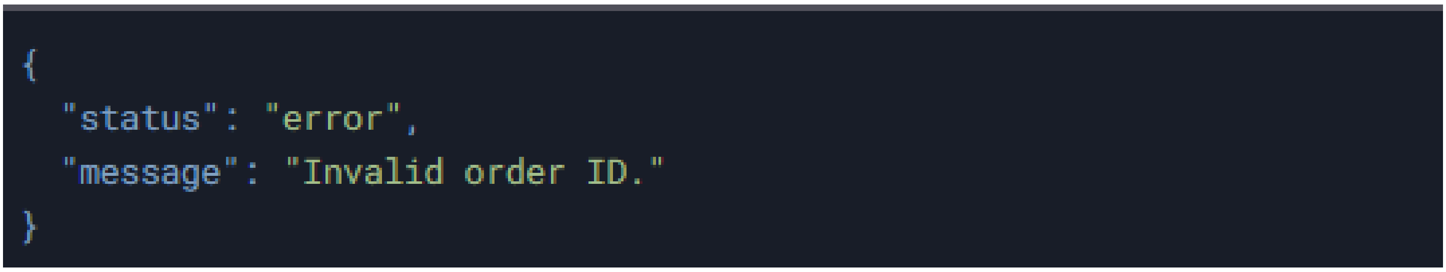
- 200 OK: Successful operation.
- 404 Not Found: Order not found.
- 500 Internal Server Error: Server error.

Authentication:

- **Authentication:** None for these endpoints.
- **Note:** In a production environment, authentication (e.g., JWT) should be implemented for security.

Error Handling:

- Error responses will include a status field and a message field.
- Example error response



```
{
  "status": "error",
  "message": "Invalid order ID."
}
```

- Ensure that the API documentation aligns with marketplace-specific workflows for clarity in implementation.
- Consider implementing versioning (e.g., /api/v1) to handle future changes.
- Include pagination for the /products endpoint if the number of products is large.
- Specify any required headers or additional parameters for each endpoint.