**API Documentation: Inventory**

**Management System**

**Project Overview**  
This project is designed to develop a robust **Garments Inventory Management System** using **Spring Boot** to efficiently manage all aspects of garment inventory, order processing, customer relations, and supplier management. The system aims to streamline operational processes, improve accuracy, and enhance overall business performance in the retail and garment industry.

**Key Figures**

* **Total Controllers**: 18
* **Total APIs**: 60

**Technical Stack**

* **Backend**: Spring Boot
* **Database**: MySQL
* **Security**: Spring Security
* **API Architecture**: RESTful APIs

**Core Features**:

**Payment Controller**: Handles all operations related to processing payments for customer orders, including payment validation and transaction updates.

**Order Controller**: Manages order creation, updating, and retrieval. It tracks customer orders, their status, and associated data like order date and total value.

**Warehouse Controller**: Oversees inventory storage locations, managing the organization of stock within the warehouse and tracking product movements.

**User Controller**: Handles user authentication and authorization, including registration, login, profile management, and role-based access.

**Supplier Controller**: Manages supplier information, including the creation, updating, and retrieval of supplier records for procurement purposes.

**Shipment Controller**: Manages the shipping process, tracking shipment status, and updating the system with shipment details for each order.

**Return Controller**: Handles return requests, managing the return process for products, including return approval and updating inventory accordingly.

**Report Controller**: Provides various business reports like sales, inventory status, and financial summaries to aid decision-making and performance analysis.

**Purchase Order Controller**: Manages purchase orders to suppliers, handling order creation, updates, and monitoring the status of incoming stock.

**Product Controller**: Manages product details, including creation, updating, and deletion of product information. It supports product catalog management.

**Order Item Controller**: Handles the details of individual items in a customer’s order, including quantity, price, and product specifications.

**Notification Controller**: Manages notifications to customers and employees, such as order updates, shipment status, or system alerts.

**Inventory Controller**: Manages stock levels for products, including inventory adjustments, stock updates, and ensuring accurate stock tracking.

**Employee Controller**: Handles employee-related operations, such as managing employee records, roles, and permissions within the system.

**Discount Controller**: Manages discount offers, creating, updating, and applying discounts to orders or specific products for customer promotions.

**Customer Controller**: Handles customer profiles, including creation, updating, and managing customer contact and billing information.

**Category Controller**: Manages product categories, allowing for classification and organization of products for easier navigation and search.

**Audit Log Controller**: Tracks and stores system activities for auditing purposes, ensuring transparency and accountability in user actions and system changes.

The system is built to be highly modular, scalable, and secure, providing a comprehensive solution that supports both operational efficiency and customer satisfaction. The 18 controllers and 85 APIs facilitate smooth integration across all business functions, enabling streamlined management of inventory, orders, and business reports.

**API Version:** v1.0.0  
**Base URL:** <http://localhost:8090>

1. **Payment API:** Handles all operations related to processing payments for customer orders, including payment validation and transaction updates.

**1.1 Get All Payments**

* **URL:** /payments
* **Method:** GET
* **Description:** Retrieve all payments.
* **Response (200):**

json

[

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"

},

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"

}]

**1.2 Get Payment by ID**

* **URL:** /payments/{id}
* **Method:** GET
* **Description:** Retrieve payment details by ID.
* **Parameters:**
  + id (path) - Payment ID (integer)
* **Response (200):**

json

{

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"

}}

**1.3 Create Payment**

* **URL:** /payments
* **Method:** POST
* **Description:** Create a new payment.
* **Request Body:**

json

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"

}

* **Response (201):**

json

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"

}

**1.4 Update Payment**

* **URL:** /payments/{id}
* **Method:** PUT
* **Description:** Update an existing payment by ID.
* **Parameters:**
  + id (path) - Payment ID (integer)
* **Request Body:**

json

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"}

* **Response (200):**

json

{

"id": 0,

"orderId": 0,

"amount": 0,

"paymentMethod": "string",

"paymentStatus": "string"

}

**1.5 Delete Payment**

* **URL:** /payments/{id}
* **Method:** DELETE
* **Description:** Delete a payment by ID.
* **Parameters:**
  + id (path) - Payment ID (integer)
* **Response (200):**

json

{

"message": "Payment deleted successfully"

}

**2. Order API**

**2.1 Get All Orders**

* **URL:** /orders
* **Method:** GET
* **Description:** Retrieve all orders.
* **Response (200):**

json

[

{

"id": 0,

"customerName": "string",

"customerContact": "string",

"orderItemIds": [

0

],

"totalAmount": 0,

"status": "string"

}

]

**2.2 Get Order by ID**

* **URL:** /orders/{id}
* **Method:** GET
* **Description:** Retrieve order details by ID.
* **Parameters:**
  + id (path) - Order ID (integer)
* **Response (200):**

json

{

"id": 1,

"customerId": 1,

"orderDate": "2025-01-01",

"totalAmount": 500.00

}

**2.3 Create Order**

* **URL:** /orders
* **Method:** POST
* **Description:** Create a new order.
* **Request Body:**

json

{

"id": 0,

"customerName": "string",

"customerContact": "string",

"orderItemIds": [

0

],

"totalAmount": 0,

"status": "string"

}

* **Response (200):**

json

{

"id": 3,

"customerId": 1,

"orderDate": "2025-01-01",

"totalAmount": 500.00

}

**2.4 Update Order**

* **URL:** /orders/{id}
* **Method:** PUT
* **Description:** Update an existing order by ID.
* **Parameters:**
  + id (path) - Order ID (integer)
* **Request Body:**

json

{

"customerId": 1,

"orderDate": "2025-01-01",

"totalAmount": 600.00

}

* **Response (200):**

json

{

"id": 1,

"customerId": 1,

"orderDate": "2025-01-01",

"totalAmount": 600.00

}

**2.5 Delete Order**

* **URL:** /orders/{id}
* **Method:** DELETE
* **Description:** Delete an order by ID.
* **Parameters:**
  + id (path) - Order ID (integer)
* **Response (200):**

json

{

"message": "Order deleted successfully"

}

**3. Warehouse API**

**3.1 Get All Warehouses**

* **URL:** /api/warehouses
* **Method:** GET
* **Description:** Retrieve all warehouses.
* **Response (200):**

json

[

{

"id": 1,

"name": "Main Warehouse",

"location": "New York"

},

{

"id": 2,

"name": "Secondary Warehouse",

"location": "Los Angeles"

}

]

**3.2 Get Warehouse by ID**

* **URL:** /api/warehouses/{id}
* **Method:** GET
* **Description:** Retrieve warehouse details by ID.
* **Parameters:**
  + id (path) - Warehouse ID (integer)
* **Response (200):**

json

{

"id": 1,

"name": "Main Warehouse",

"location": "New York"

}

**3.3 Create Warehouse**

* **URL:** /api/warehouses
* **Method:** POST
* **Description:** Create a new warehouse.
* **Request Body:**

json

{

"name": "New Warehouse",

"location": "Chicago"

}

* **Response (201):**

json

{

"id": 3,

"name": "New Warehouse",

"location": "Chicago"

}

**3.4 Update Warehouse**

* **URL:** /api/warehouses/{id}
* **Method:** PUT
* **Description:** Update an existing warehouse by ID.
* **Parameters:**
  + id (path) - Warehouse ID (integer)
* **Request Body:**

json

{

"name": "Updated Warehouse",

"location": "San Francisco"

}

* **Response (200):**

json

{

"id": 1,

"name": "Updated Warehouse",

"location": "San Francisco"

}

**3.5 Delete Warehouse**

* **URL:** /api/warehouses/{id}
* **Method:** DELETE
* **Description:** Delete a warehouse by ID.
* **Parameters:**
  + id (path) - Warehouse ID (integer)
* **Response (200):**

json

{

"message": "Warehouse deleted successfully"

}

**4. User API**

**4.1 Get All Users**

* **URL:** /api/users
* **Method:** GET
* **Description:** Retrieve all users.
* **Response (200):**

json

[

{

"id": 1,

"name": "Alice Brown",

"email": "aliceb@example.com",

"role": "Admin"

},

{

"id": 2,

"name": "Bob White",

"email": "bobw@example.com",

"role": "User"

}

]

**4.2 Get User by ID**

* **URL:** /api/users/{id}
* **Method:** GET
* **Description:** Retrieve user details by ID.
* **Parameters:**
  + id (path) - User ID (integer)
* **Response (200):**

json

{

"id": 1,

"name": "Alice Brown",

"email": "aliceb@example.com",

"role": "Admin"

}

**4.3 Create User**

* **URL:** /api/users
* **Method:** POST
* **Description:** Create a new user.
* **Request Body:**

json

{

"name": "Charlie Green",

"email": "charlieg@example.com",

"role": "Manager"

}

* **Response (201):**

json

{

"id": 3,

"name": "Charlie Green",

"email": "charlieg@example.com",

"role": "Manager"

}

**4.4 Update User**

* **URL:** /api/users/{id}
* **Method:** PUT
* **Description:** Update an existing user by ID.
* **Parameters:**
  + id (path) - User ID (integer)
* **Request Body:**

json

{

"name": "David Blue",

"email": "davidb@example.com",

"role": "Editor"

}

* **Response (200):**

json

{

"id": 1,

"name": "David Blue",

"email": "davidb@example.com",

"role": "Editor"

}

**4.5 Delete User**

* **URL:** /api/users/{id}
* **Method:** DELETE
* **Description:** Delete a user by ID.
* **Parameters:**
  + id (path) - User ID (integer)
* **Response (200):**

json

{

"message": "User deleted successfully"

}

**5. Supplier API**

**5.1 Get All Suppliers**

* **URL:** /api/suppliers
* **Method:** GET
* **Description:** Retrieve all suppliers.
* **Response (200):**

json

[

{

"id": 1,

"name": "Supplier X",

"contact": "supplierx@example.com"

},

{

"id": 2,

"name": "Supplier Y",

"contact": "suppliery@example.com"

}

]

**5.2 Get Supplier by ID**

* **URL:** /api/suppliers/{id}
* **Method:** GET
* **Description:** Retrieve supplier details by ID.
* **Parameters:**
  + id (path) - Supplier ID (integer)
* **Response (200):**

json

{

"id": 1,

"name": "Supplier X",

"contact": "supplierx@example.com"

}

**5.3 Create Supplier**

* **URL:** /api/suppliers
* **Method:** POST
* **Description:** Create a new supplier.
* **Request Body:**

json

{

"name": "Supplier Z",

"contact": "supplierz@example.com"

}

* **Response (201):**

json

{

"id": 3,

"name": "Supplier Z",

"contact": "supplierz@example.com"

}

**5.4 Update Supplier**

* **URL:** /api/suppliers/{id}
* **Method:** PUT
* **Description:** Update an existing supplier by ID.
* **Parameters:**
  + id (path) - Supplier ID (integer)
* **Request Body:**

json

{

"name": "Supplier W",

"contact": "supplierw@example.com"

}

* **Response (200):**

json

{

"id": 1,

"name": "Supplier W",

"contact": "supplierw@example.com"

}

**5.5 Delete Supplier**

* **URL:** /api/suppliers/{id}
* **Method:** DELETE
* **Description:** Delete a supplier by ID.
* **Parameters:**
  + id (path) - Supplier ID (integer)
* **Response (200):**

json

{

"message": "Supplier deleted successfully"

}

**6. Shipment API**

**6.1 Get All Shipments**

* **URL:** /api/shipments
* **Method:** GET
* **Description:** Retrieve all shipments.
* **Response (200):**

json

[

{

"id": 1,

"trackingNumber": "TRK123",

"status": "Shipped",

"destination": "New York"

},

{

"id": 2,

"trackingNumber": "TRK124",

"status": "Pending",

"destination": "Los Angeles"

}

]

**6.2 Get Shipment by ID**

* **URL:** /api/shipments/{id}
* **Method:** GET
* **Description:** Retrieve shipment details by ID.
* **Parameters:**
  + id (path) - Shipment ID (integer)
* **Response (200):**

json

{

"id": 1,

"trackingNumber": "TRK123",

"status": "Shipped",

"destination": "New York"

}

**6.3 Create Shipment**

* **URL:** /api/shipments
* **Method:** POST
* **Description:** Create a new shipment.
* **Request Body:**

json

{

"trackingNumber": "TRK125",

"status": "Shipped",

"destination": "Chicago"

}

* **Response (201):**

json

{

"id": 3,

"trackingNumber": "TRK125",

"status": "Shipped",

"destination": "Chicago"

}

**6.4 Update Shipment**

* **URL:** /api/shipments/{id}
* **Method:** PUT
* **Description:** Update an existing shipment by ID.
* **Parameters:**
  + id (path) - Shipment ID (integer)
* **Request Body:**

json

{

"trackingNumber": "TRK126",

"status": "Delivered",

"destination": "Chicago"

}

* **Response (200):**

json

{

"id": 1,

"trackingNumber": "TRK126",

"status": "Delivered",

"destination": "Chicago"

}

**6.5 Delete Shipment**

* **URL:** /api/shipments/{id}
* **Method:** DELETE
* **Description:** Delete a shipment by ID.
* **Parameters:**
  + id (path) - Shipment ID (integer)
* **Response (200):**

json

{

"message": "Shipment deleted successfully"

}

**7. Return API**

**7.1 Get All Returns**

* **URL:** /api/returns
* **Method:** GET
* **Description:** Retrieve all returns.
* **Response (200):**

json

[

{

"id": 1,

"orderId": 123,

"status": "Pending",

"reason": "Wrong item"

},

{

"id": 2,

"orderId": 124,

"status": "Completed",

"reason": "Defective product"

}

]

**7.2 Get Return by ID**

* **URL:** /api/returns/{id}
* **Method:** GET
* **Description:** Retrieve return details by ID.
* **Parameters:**
  + id (path) - Return ID (integer)
* **Response (200):**

json

{

"id": 1,

"orderId": 123,

"status": "Pending",

"reason": "Wrong item"

}

**7.3 Create Return**

* **URL:** /api/returns
* **Method:** POST
* **Description:** Create a new return.
* **Request Body:**

json

{

"orderId": 125,

"status": "Pending",

"reason": "Item damaged"

}

* **Response (201):**

json

{

"id": 3,

"orderId": 125,

"status": "Pending",

"reason": "Item damaged"

}

**7.4 Update Return**

* **URL:** /api/returns/{id}
* **Method:** PUT
* **Description:** Update an existing return by ID.
* **Parameters:**
  + id (path) - Return ID (integer)
* **Request Body:**

json

{

"orderId": 123,

"status": "Completed",

"reason": "Wrong item"

}

* **Response (200):**

json

{

"id": 1,

"orderId": 123,

"status": "Completed",

"reason": "Wrong item"

}

**7.5 Delete Return**

* **URL:** /api/returns/{id}
* **Method:** DELETE
* **Description:** Delete a return by ID.
* **Parameters:**
  + id (path) - Return ID (integer)
* **Response (200):**

json

{

"message": "Return deleted successfully"

}

**8. Report API**

**8.1 Get All Reports**

* **URL:** /api/reports
* **Method:** GET
* **Description:** Retrieve all reports.
* **Response (200):**

json

[

{

"id": 1,

"type": "Sales Report",

"date": "2025-01-01",

"details": "Monthly sales data"

},

{

"id": 2,

"type": "Inventory Report",

"date": "2025-01-02",

"details": "Inventory status"

}

]

**8.2 Get Report by ID**

* **URL:** /api/reports/{id}
* **Method:** GET
* **Description:** Retrieve report details by ID.
* **Parameters:**
  + id (path) - Report ID (integer)
* **Response (200):**

json

{

"id": 1,

"type": "Sales Report",

"date": "2025-01-01",

"details": "Monthly sales data"

}

**8.3 Create Report**

* **URL:** /api/reports
* **Method:** POST
* **Description:** Create a new report.
* **Request Body:**

json

{

"type": "Performance Report",

"date": "2025-01-05",

"details": "Quarterly performance review"

}

* **Response (201):**

json

{

"id": 3,

"type": "Performance Report",

"date": "2025-01-05",

"details": "Quarterly performance review"

}

**8.4 Update Report**

* **URL:** /api/reports/{id}
* **Method:** PUT
* **Description:** Update an existing report by ID.
* **Parameters:**
  + id (path) - Report ID (integer)
* **Request Body:**

json

{

"type": "Updated Sales Report",

"date": "2025-01-05",

"details": "Updated sales data"

}

* **Response (200):**

json

{

"id": 1,

"type": "Updated Sales Report",

"date": "2025-01-05",

"details": "Updated sales data"

}

**8.5 Delete Report**

* **URL:** /api/reports/{id}
* **Method:** DELETE
* **Description:** Delete a report by ID.
* **Parameters:**
  + id (path) - Report ID (integer)
* **Response (200):**

json

{

"message": "Report deleted successfully"

}

**9. Purchase Order API**

**9.1 Get All Purchase Orders**

* **URL:** /api/purchase-orders
* **Method:** GET
* **Description:** Retrieve all purchase orders.
* **Response (200):**

json

[

{

"id": 1,

"supplierId": 10,

"status": "Ordered",

"orderDate": "2025-01-01"

},

{

"id": 2,

"supplierId": 12,

"status": "Delivered",

"orderDate": "2025-01-02"

}

]

**9.2 Get Purchase Order by ID**

* **URL:** /api/purchase-orders/{id}
* **Method:** GET
* **Description:** Retrieve purchase order details by ID.
* **Parameters:**
  + id (path) - Purchase Order ID (integer)
* **Response (200):**

json

{

"id": 1,

"supplierId": 10,

"status": "Ordered",

"orderDate": "2025-01-01"

}

**9.3 Create Purchase Order**

* **URL:** /api/purchase-orders
* **Method:** POST
* **Description:** Create a new purchase order.
* **Request Body:**

json

{

"supplierId": 14,

"status": "Ordered",

"orderDate": "2025-01-05"

}

* **Response (201):**

json

{

"id": 3,

"supplierId": 14,

"status": "Ordered",

"orderDate": "2025-01-05"

}

**9.4 Update Purchase Order**

* **URL:** /api/purchase-orders/{id}
* **Method:** PUT
* **Description:** Update an existing purchase order by ID.
* **Parameters:**
  + id (path) - Purchase Order ID (integer)
* **Request Body:**

json

{

"supplierId": 12,

"status": "Shipped",

"orderDate": "2025-01-02"

}

* **Response (200):**

json

{

"id": 1,

"supplierId": 12,

"status": "Shipped",

"orderDate": "2025-01-02"

}

**9.5 Delete Purchase Order**

* **URL:** /api/purchase-orders/{id}
* **Method:** DELETE
* **Description:** Delete a purchase order by ID.
* **Parameters:**
  + id (path) - Purchase Order ID (integer)
* **Response (200):**

json

{

"message": "Purchase order deleted successfully"

}

**10. Product API**

**10.1 Get All Products**

* **URL:** /api/products
* **Method:** GET
* **Description:** Retrieve all products.
* **Response (200):**

json

[

{

"id": 1,

"name": "Product A",

"price": 100,

"quantity": 50

},

{

"id": 2,

"name": "Product B",

"price": 150,

"quantity": 30

}

]

**10.2 Get Product by ID**

* **URL:** /api/products/{id}
* **Method:** GET
* **Description:** Retrieve product details by ID.
* **Parameters:**
  + id (path) - Product ID (integer)
* **Response (200):**

json

{

"id": 1,

"name": "Product A",

"price": 100,

"quantity": 50

}

**10.3 Create Product**

* **URL:** /api/products
* **Method:** POST
* **Description:** Create a new product.
* **Request Body:**

json

{

"id": 0,

"name": "string",

"description": "string",

"size": "string",

"color": "string",

"brand": "string",

"price": 0,

"categoryId": 0

}

* **Response (201):**

json

{

"id": 3,

"name": "Product C",

"price": 200,

"quantity": 40

}

**10.4 Update Product**

* **URL:** /api/products/{id}
* **Method:** PUT
* **Description:** Update an existing product by ID.
* **Parameters:**
  + id (path) - Product ID (integer)
* **Request Body:**

json

{

"name": "Product A Updated",

"price": 110,

"quantity": 55

}

* **Response (200):**

json

{

"id": 1,

"name": "Product A Updated",

"price": 110,

"quantity": 55

}

**10.5 Delete Product**

* **URL:** /api/products/{id}
* **Method:** DELETE
* **Description:** Delete a product by ID.
* **Parameters:**
  + id (path) - Product ID (integer)
* **Response (200):**

json

{

"message": "Product deleted successfully"

}

**11. Order Items API**

**11.1 Get Order Item by ID**

* **URL**: /api/order-items/{id}
* **Method**: GET
* **Description**: Retrieve order item details by ID.
* **Parameters**:
  + id (path) - Order Item ID (integer)
* **Response (200)**:

json

{

"id": 1,

"orderId": 123,

"productId": 45,

"quantity": 2,

"price": 100

}

**11.2 Update Order Item**

* **URL**: /api/order-items/{id}
* **Method**: PUT
* **Description**: Update an existing order item by ID.
* **Parameters**:
  + id (path) - Order Item ID (integer)
* **Request Body**:

json

{

"orderId": 123,

"productId": 45,

"quantity": 3,

"price": 120

}

* **Response (200)**:

json

{

"id": 1,

"orderId": 123,

"productId": 45,

"quantity": 3,

"price": 120

}

**11.3 Delete Order Item**

* **URL**: /api/order-items/{id}
* **Method**: DELETE
* **Description**: Delete an order item by ID.
* **Parameters**:
  + id (path) - Order Item ID (integer)
* **Response (200)**:

json

{

"message": "Order item deleted successfully"

}

**12. Notifications API**

**12.1 Get Notification by ID**

* **URL**: /api/notifications/{id}
* **Method**: GET
* **Description**: Retrieve notification details by ID.
* **Parameters**:
  + id (path) - Notification ID (integer)
* **Response (200)**:

json

{

"id": 1,

"message": "New order placed",

"timestamp": "2025-01-29T12:00:00Z"

}

**12.2 Update Notification**

* **URL**: /api/notifications/{id}
* **Method**: PUT
* **Description**: Update an existing notification by ID.
* **Parameters**:
  + id (path) - Notification ID (integer)
* **Request Body**:

json

{

"message": "Order shipped",

"timestamp": "2025-01-30T09:00:00Z"

}

* **Response (200)**:

json

{

"id": 1,

"message": "Order shipped",

"timestamp": "2025-01-30T09:00:00Z"

}

**12.3 Delete Notification**

* **URL**: /api/notifications/{id}
* **Method**: DELETE
* **Description**: Delete a notification by ID.
* **Parameters**:
  + id (path) - Notification ID (integer)
* **Response (200)**:

json

{

"message": "Notification deleted successfully"

}

**13. Inventories API**

**13.1 Get Inventory by ID**

* **URL**: /api/inventories/{id}
* **Method**: GET
* **Description**: Retrieve inventory details by ID.
* **Parameters**:
  + id (path) - Inventory ID (integer)
* **Response (200)**:

json

{

"id": 1,

"productId": 45,

"stock": 100,

"location": "Warehouse A"

}

**13.2 Update Inventory**

* **URL**: /api/inventories/{id}
* **Method**: PUT
* **Description**: Update an existing inventory by ID.
* **Parameters**:
  + id (path) - Inventory ID (integer)
* **Request Body**:

json

{

"productId": 45,

"stock": 120,

"location": "Warehouse B"

}

* **Response (200)**:

json

{

"id": 1,

"productId": 45,

"stock": 120,

"location": "Warehouse B"

}

**13.3 Delete Inventory**

* **URL**: /api/inventories/{id}
* **Method**: DELETE
* **Description**: Delete an inventory by ID.
* **Parameters**:
  + id (path) - Inventory ID (integer)
* **Response (200)**:

json

{

"message": "Inventory deleted successfully"

}

**14. Employees API**

**14.1 Get Employee by ID**

* **URL**: /api/employees/{id}
* **Method**: GET
* **Description**: Retrieve employee details by ID.
* **Parameters**:
  + id (path) - Employee ID (integer)
* **Response (200)**:

json

{

"id": 1,

"firstName": "John",

"lastName": "Doe",

"role": "Manager",

"department": "Sales"

}

**14.2 Update Employee**

* **URL**: /api/employees/{id}
* **Method**: PUT
* **Description**: Update an existing employee by ID.
* **Parameters**:
  + id (path) - Employee ID (integer)
* **Request Body**:

json

{

"firstName": "John",

"lastName": "Doe",

"role": "Senior Manager",

"department": "Sales"

}

* **Response (200)**:

json

{

"id": 1,

"firstName": "John",

"lastName": "Doe",

"role": "Senior Manager",

"department": "Sales"

}

**14.3 Delete Employee**

* **URL**: /api/employees/{id}
* **Method**: DELETE
* **Description**: Delete an employee by ID.
* **Parameters**:
  + id (path) - Employee ID (integer)
* **Response (200)**:

json

{

"message": "Employee deleted successfully"

}

**15. Discounts API**

**15.1 Get Discount by ID**

* **URL**: /api/discounts/{id}
* **Method**: GET
* **Description**: Retrieve discount details by ID.
* **Parameters**:
  + id (path) - Discount ID (integer)
* **Response (200)**:

json

{

"id": 1,

"discountCode": "SUMMER2025",

"percentage": 20,

"validUntil": "2025-08-31"

}

**15.2 Update Discount**

* **URL**: /api/discounts/{id}
* **Method**: PUT
* **Description**: Update an existing discount by ID.
* **Parameters**:
  + id (path) - Discount ID (integer)
* **Request Body**:

json

Copy

{

"discountCode": "SUMMER2025",

"percentage": 25,

"validUntil": "2025-09-30"

}

* **Response (200)**:

json

{

"id": 1,

"discountCode": "SUMMER2025",

"percentage": 25,

"validUntil": "2025-09-30"

}

**15.3 Delete Discount**

* **URL**: /api/discounts/{id}
* **Method**: DELETE
* **Description**: Delete a discount by ID.
* **Parameters**:
  + id (path) - Discount ID (integer)
* **Response (200)**:

json

{

"message": "Discount deleted successfully"

}

**16. Customers API**

**16.1 Get Customer by ID**

* **URL**: /api/customers/{id}
* **Method**: GET
* **Description**: Retrieve customer details by ID.
* **Parameters**:
  + id (path) - Customer ID (integer)
* **Response (200)**:

json

{

"id": 1,

"firstName": "Alice",

"lastName": "Smith",

"email": "alice.smith@example.com",

"phone": "123-456-7890"

}

**16.2 Update Customer**

* **URL**: /api/customers/{id}
* **Method**: PUT
* **Description**: Update an existing customer by ID.
* **Parameters**:
  + id (path) - Customer ID (integer)
* **Request Body**:

json

{

"firstName": "Alice",

"lastName": "Smith",

"email": "alice.smith@example.com",

"phone": "987-654-3210"

}

* **Response (200)**:

json

{

"id": 1,

"firstName": "Alice",

"lastName": "Smith",

"email": "alice.smith@example.com",

"phone": "987-654-3210"

}

**16.3 Delete Customer**

* **URL**: /api/customers/{id}
* **Method**: DELETE
* **Description**: Delete a customer by ID.
* **Parameters**:
  + id (path) - Customer ID (integer)
* **Response (200)**:

json

{

"message": "Customer deleted successfully"

}

**17. Categories API**

**17.1 Get All Category**

* **URL**: /api/categories
* **Method**: GET
* **Description**: Retrieve a list of all Category.
* **Response (200)**:

json

[

{

"id": 1,

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

},

{

"id": 2,

"action": "Order Shipped",

"timestamp": "2025-01-29T15:00:00Z",

"userId": 124,

"details": "Order ID 102 shipped"

}

]

**17.2 Get Category by ID**

* **URL**: /api/audit-logs/{id}
* **Method**: GET
* **Description**: Retrieve a specific audit log by ID.
* **Response (200)**:

json

{

"id": 1,

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

}

**17.3 Create Category**

* **URL**: /api/audit-logs
* **Method**: POST
* **Description**: Create a new audit log.
* **Request Body**:

json

{

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

}

* **Response (201)**:

json

{

"id": 1,

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

}

**17.4 Update Category**

**URL**: /api/audit-logs/update/{id}

* **Method**: PUT
* **Description**: Update an existing audit log by ID.
* **Request Body**:

json

{

"action": "Order Shipped",

"timestamp": "2025-01-29T15:00:00Z",

"userId": 124,

"details": "Order ID 102 shipped"

}

* **Response (200)**:

json

{

"id": 1,

"action": "Order Shipped",

"timestamp": "2025-01-29T15:00:00Z",

"userId": 124,

"details": "Order ID 102 shipped"

}

**17.5 Delete Category**

* **URL**: /api/audit-logs/{id}
* **Method**: DELETE
* **Description**: Delete an audit log by ID.
* **Response (204)**:

json

{

"message": "Categorydeleted successfully"

}

**18. Audit Logs API**

**18.1 Get All Audit Logs**

* **URL**: /api/audit-logs
* **Method**: GET
* **Description**: Retrieve a list of all audit logs.
* **Response (200)**:

json

[

{

"id": 1,

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

},

{

"id": 2,

"action": "Order Shipped",

"timestamp": "2025-01-29T15:00:00Z",

"userId": 124,

"details": "Order ID 102 shipped"

}

]

**18.2 Get Audit Log by ID**

* **URL**: /api/audit-logs/{id}
* **Method**: GET
* **Description**: Retrieve a specific audit log by ID.
* **Response (200)**:

json

{

"id": 1,

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

}

**18.3 Create Audit Log**

* **URL**: /api/audit-logs
* **Method**: POST
* **Description**: Create a new audit log.
* **Request Body**:

json

{

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

}

* **Response (201)**:

json

{

"id": 1,

"action": "Order Created",

"timestamp": "2025-01-29T14:00:00Z",

"userId": 123,

"details": "Order created with ID 101"

}

**18.4 Update Audit Log**

* **URL**: /api/audit-logs/update/{id}
* **Method**: PUT
* **Description**: Update an existing audit log by ID.
* **Request Body**:

json

{

"action": "Order Shipped",

"timestamp": "2025-01-29T15:00:00Z",

"userId": 124,

"details": "Order ID 102 shipped"

}

* **Response (200)**:

json

{

"id": 1,

"action": "Order Shipped",

"timestamp": "2025-01-29T15:00:00Z",

"userId": 124,

"details": "Order ID 102 shipped"

}

**18.5 Delete Audit Log**

* **URL**: /api/audit-logs/{id}
* **Method**: DELETE
* **Description**: Delete an audit log by ID.
* **Response (204)**:

json

{

"message": "Audit Logdeleted successfully"

}

**That’s All For Today. Thank You.**