# Manilogix Order-Service Documentation

## 1. Module Overview

The **Order-Service** handles customer order creation, status updates, supplier assignment, and order tracking within the Manilogix platform. It integrates with **Inventory-Service** and **Supplier-Service** and uses **MySQL** for transactional data.

## 2. Project Structure

order-service/  
├── src/main/java/com/manilogix/order/  
│ ├── controller/  
│ │ └── OrderController.java  
│ ├── service/  
│ │ └── OrderService.java  
│ ├── dto/  
│ │ ├── OrderRequest.java  
│ │ └── OrderResponse.java  
│ ├── model/  
│ │ └── Order.java  
│ ├── repository/  
│ │ └── OrderRepository.java  
│ └── OrderServiceApplication.java  
└── resources/  
 └── application.yml

## 3. Controller and Responsibilities

### OrderController

**Path:** /api/orders

#### Responsibilities:

* Create new order (POST /create)
* Get all orders (GET /all)
* Get order by ID (GET /{id})
* Update order status (PUT /status/{id})
* Delete order (DELETE /delete/{id})

## 4. DTOs (Data Transfer Objects)

### OrderRequest.java

* String customerId
* List<String> itemIds
* String supplierId
* String deliveryAddress

### OrderResponse.java

* Long orderId
* String customerId
* List<String> itemIds
* String supplierId
* String deliveryAddress
* OrderStatus status
* LocalDateTime createdAt

## 5. Model

### Order.java

* @Id Long orderId
* String customerId
* List<String> itemIds
* String supplierId
* String deliveryAddress
* OrderStatus status (Enum: PENDING, CONFIRMED, DISPATCHED, DELIVERED, CANCELLED)
* LocalDateTime createdAt

## 6. Service Layer

### OrderService

* Place a new order
* Assign supplier and check inventory
* Update order status
* Cancel or delete order
* Retrieve order history or by ID

## 7. Repository

### OrderRepository

* Extends JpaRepository<Order, Long>
* Includes methods for filtering orders by customer, status, date

## 8. Application Properties (YAML)

spring:  
 datasource:  
 url: jdbc:mysql://localhost:3306/orderdb  
 username: root  
 password: yourpassword  
 jpa:  
 hibernate:  
 ddl-auto: update  
 show-sql: true  
 application:  
 name: order-service

## 9. Endpoints Summary

| Method | Endpoint | Description |
| --- | --- | --- |
| POST | /api/orders/create | Place new order |
| GET | /api/orders/all | List all orders |
| GET | /api/orders/{id} | Get order by ID |
| PUT | /api/orders/status/{id} | Update order status |
| DELETE | /api/orders/delete/{id} | Cancel/delete order |

## 10. Dependencies

* Spring Web
* Spring Data JPA
* MySQL Driver
* Lombok
* Validation API
* Spring Boot DevTools

## 11. Notes

* Order-service is central to business workflow
* Can publish events to Notification-Service via Kafka
* May integrate with external logistics APIs in future