QuitQ – Backend Development Report

# 1. Introduction

QuitQ is a full-stack e-commerce application developed using Java Spring Boot for the backend. This document outlines the backend implementation details including architecture, technologies used, and security features.

# 2. Technology Stack

• Java 17+

• Spring Boot Framework

• Spring Security

• MySQL Database

• JWT (JSON Web Tokens)

• Swagger for API Documentation

• JUnit for Unit Testing

# 3. Architecture Overview

The backend follows a monolithic layered architecture:

• Controller Layer: Exposes REST APIs.

• Service Layer: Business logic.

• Repository Layer: Interacts with MySQL using JPA.

• Entity Layer: Maps to database tables.

# 4. Key Functional Modules

• User & Seller Registration and Login

• Product Catalog Management

• Shopping Cart Management

• Order Processing and Payment

• Admin Module (User/Product Control)

# 5. Authentication and Authorization

• JWT Authentication implemented for login and protected routes.

• Role-Based Access Control (User, Seller, Admin).

• Tokens contain userId, roles, and expiration time.

• Secure password handling using BCrypt hashing.

# 6. API Design

• RESTful API endpoints for all major functionalities.

• JSON used for request and response bodies.

• Example Endpoints:

- POST /api/auth/login

- GET /api/products

- POST /api/orders

- GET /api/users/profile

# 7. Error Handling and Validation

• Validation using annotations like @NotNull, @Email.

• Centralized Exception Handling using @ControllerAdvice.

• Returns user-friendly error messages in JSON format.

# 8. Logging and Testing

• Logging is implemented using SLF4J and Logback.

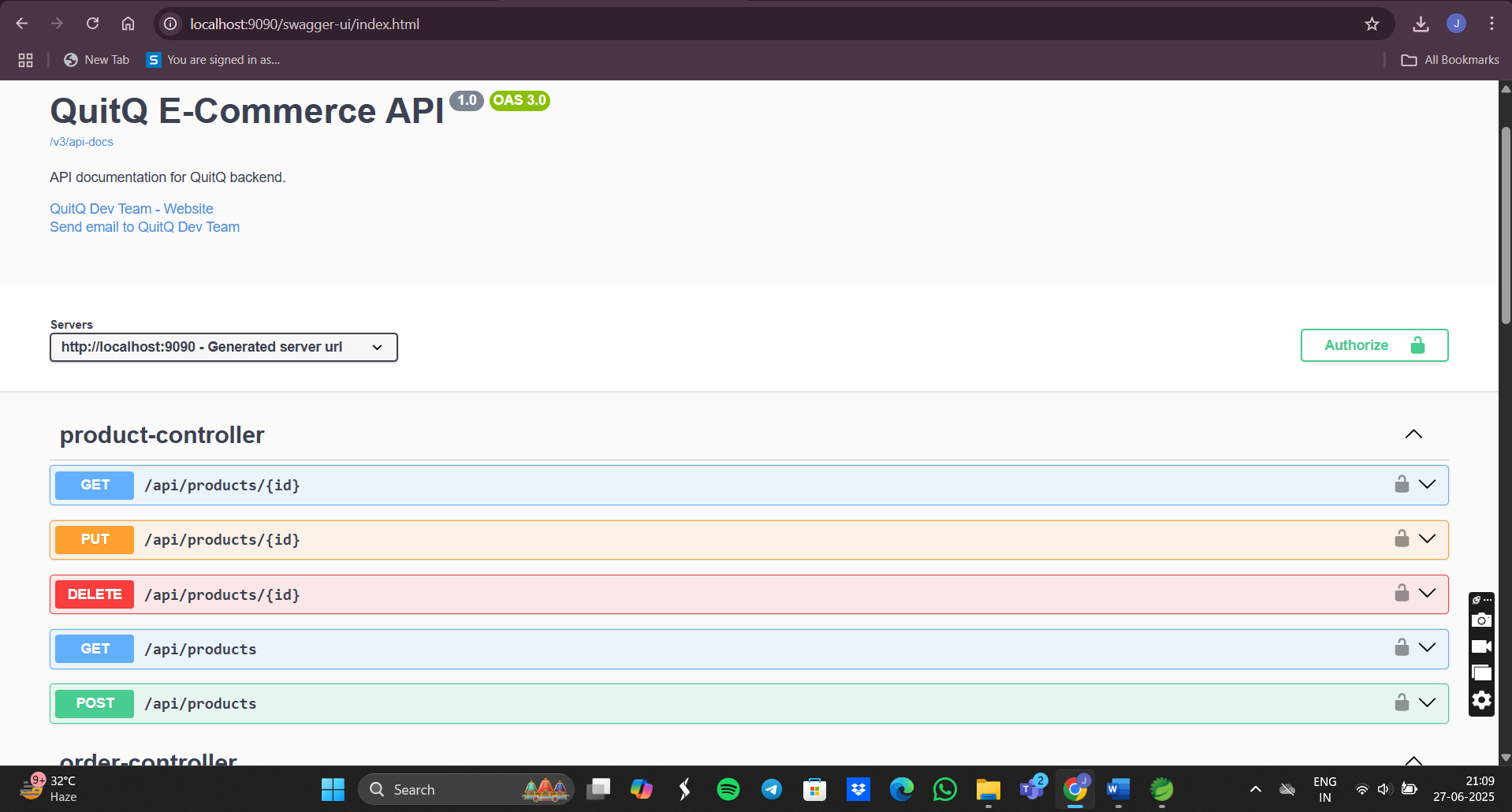
• Unit tests written using JUnit and Mockito for service layer.

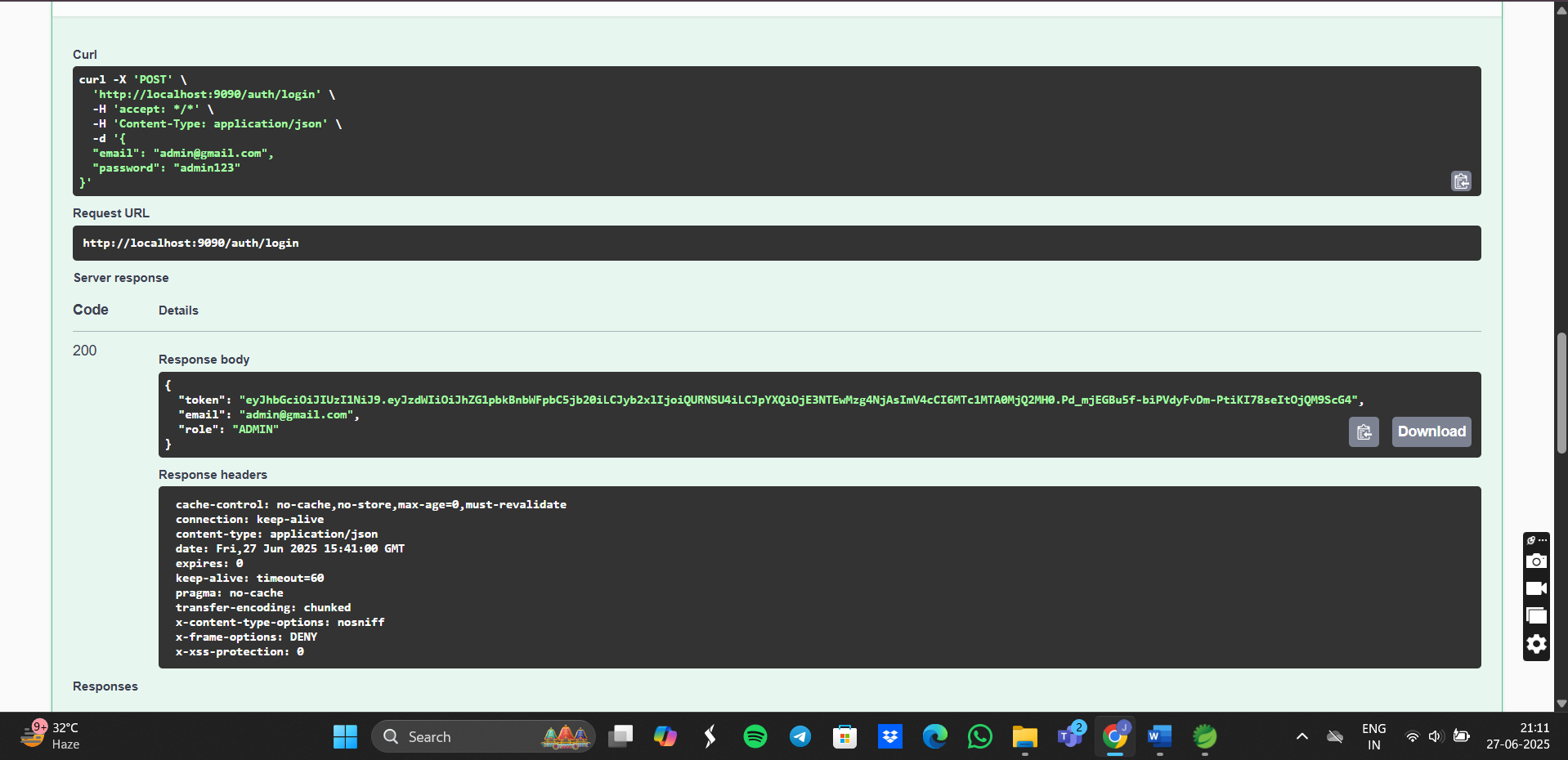
# 9. Swagger Integration

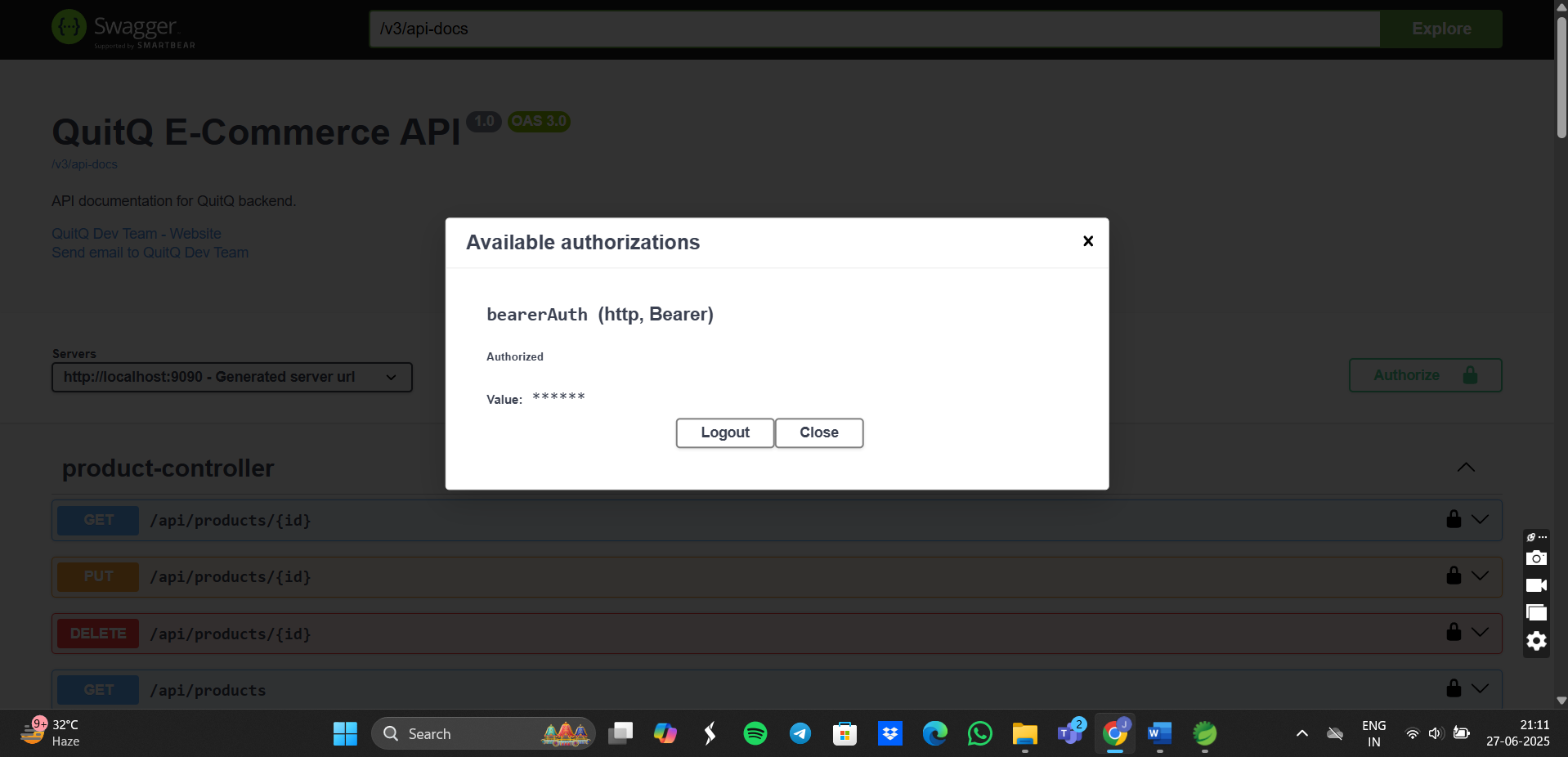
• Swagger UI provides interactive documentation and API testing.

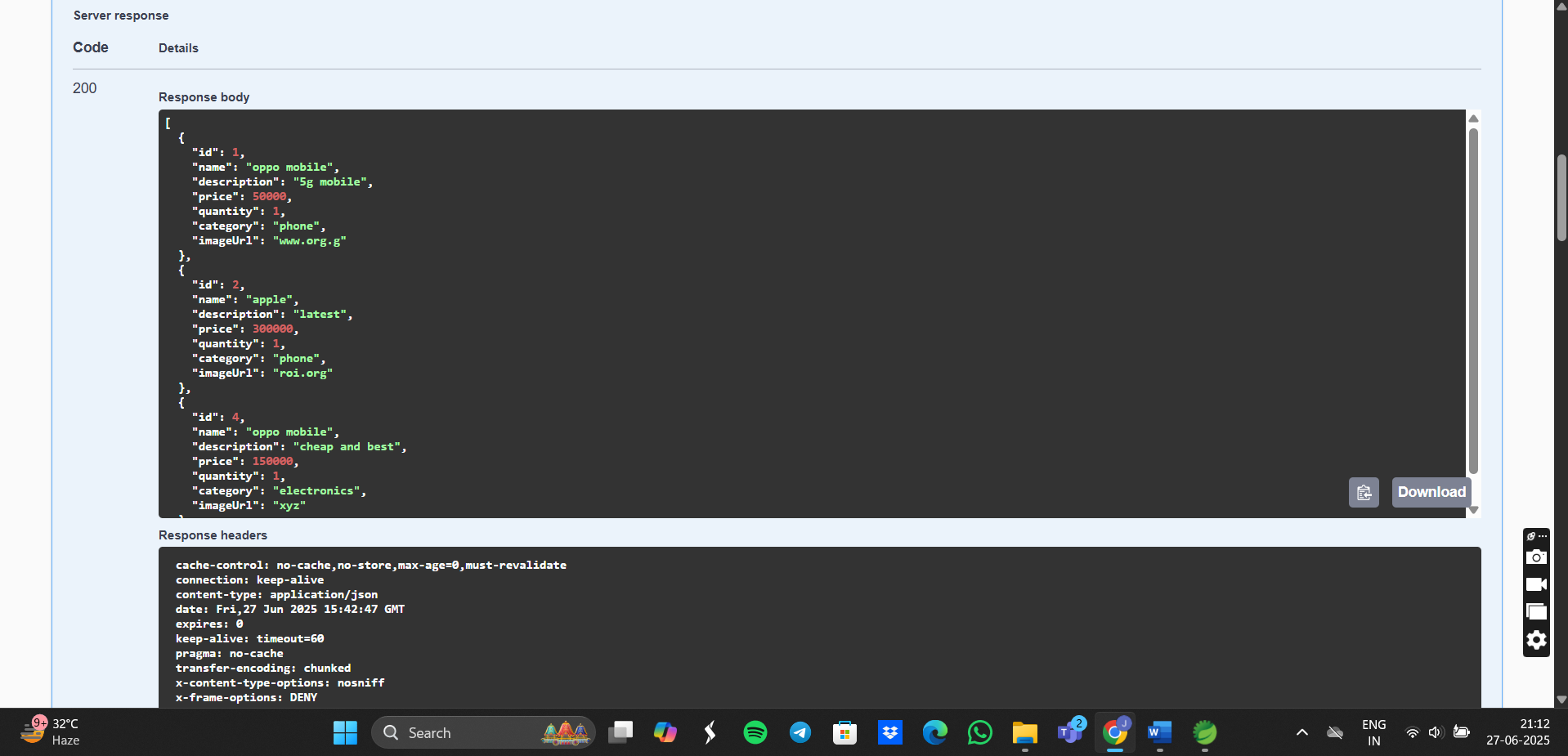
• Accessible at /swagger-ui.html

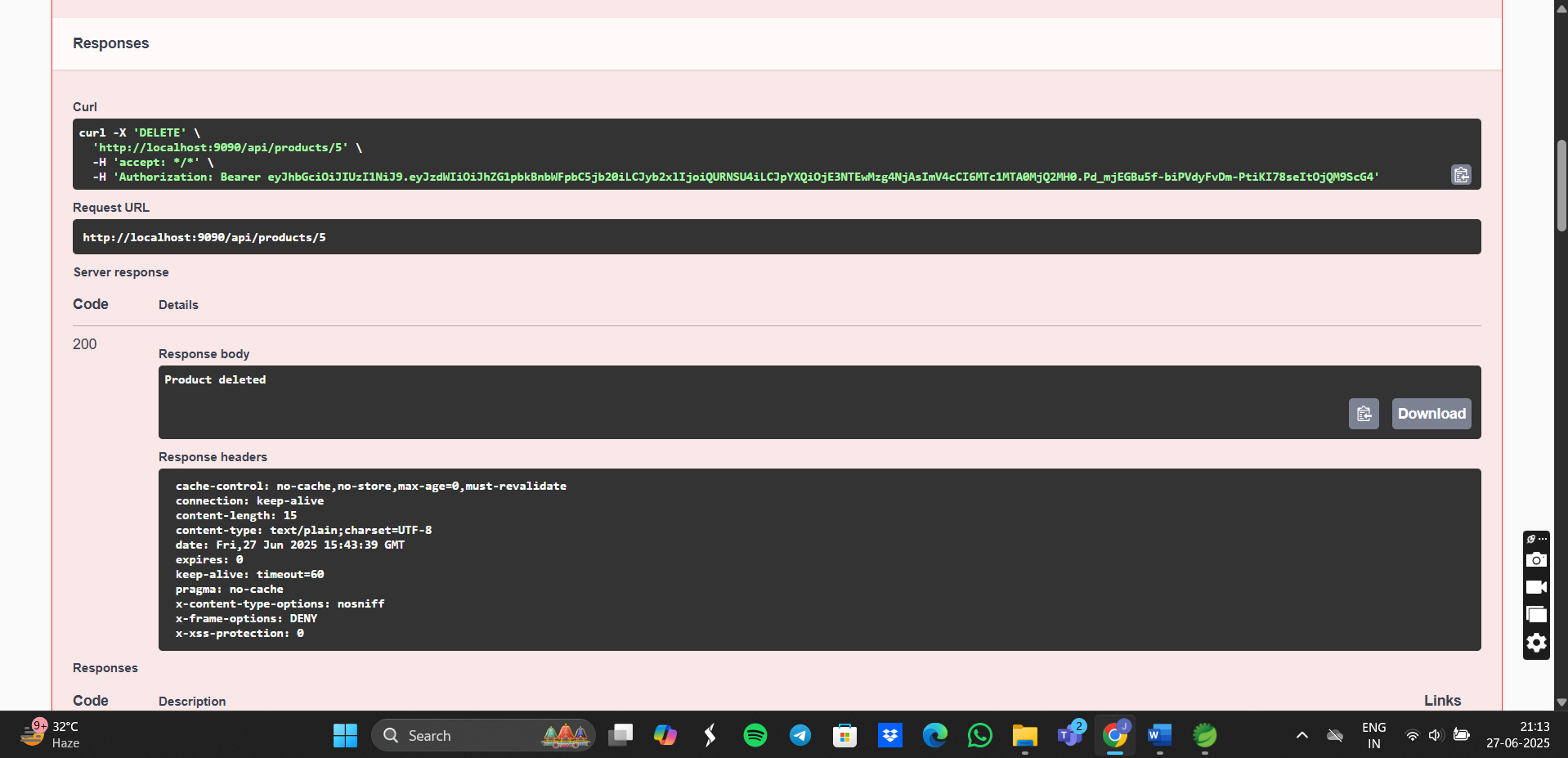
**Output Screenshots:-**

****

****

****

****

****