

# **ENTERPRISE SOFTWARE ENGINEERING**

**MSC COMPUTER SCIENCE(SOFTWARE ENGINEERING)**

# 

Asad Ullah (22040227)

Table of Contents

[Introduction: 2](#_Toc156066965)

[Why Use Spring Boot: 3](#_Toc156066966)

[Requirement Modeling: 4](#_Toc156066967)

[Use Case Diagrams 4](#_Toc156066968)

[Registration Activity Diagram: 8](#_Toc156066969)

[ERD Diagram 9](#_Toc156066970)

[API: 9](#_Toc156066971)

[Testing: 11](#_Toc156066972)

[Conclusion: 12](#_Toc156066973)

# Introduction:

The web application created is for an online clothing store that will allow users to buy and order clothing online. The application backend was built with the Spring Book framework and a SQL database to store and retrieve data. REST APIs are utilized to handle HTTP requests and responses with the use of @RestController, @GetMapping, @PostMapping, @PutMapping and @Deletemapping annotations.

The annotation provided by spring boot framework has been used as below.

* @RestController annotation has been used to make the class a controller.
* @Autowired annotations are used to resolve dependencies
* @Getter , @Setter @Entity annotation has been used to connect the bean class to particular tables in the database.
* @joincolumn annotation as a join column for an entity association or an element collection

The application consist of two type of roles admin and local user. Users must first register into the system in order to get started with the online clothing store. The functionalities and contents that users can have access to are limited based on their roles. When the user logs in to the system, it will redirect user to the homepage according to the user role.

Local User:

* Registration
* Login
* Get jwt token
* View products
* View order
* View profile

Admin:

* View Products
* Delete Products
* Add products

## Why Use Spring Boot:

Spring Boot provides numerous benefits, making it a popular alternative for developing Java-based apps. Here are some specific reasons why developers and businesses prefer Spring Boot:

Convention over Configuration: Simplified Configuration: Spring Boot adheres to the convention-over-configuration paradigm, requiring as few explicit setups as possible. Default settings and sensible defaults allow developers to get started quickly without having to write a lot of boilerplate code.

**Rapid Development:**

Standalone Applications: Spring Boot allows developers to create stand-alone applications with an embedded web server (like Tomcat, Jetty, or Undertow). This eliminates the need for external web server configuration, simplifying deployment and reducing development time.

**Microservices Architecture:**

Built for Microservices: Spring Boot is well-suited for building microservices-based architectures. It provides features like embedded web servers, easy externalized configuration, and a lightweight footprint, making it an excellent choice for microservices development.

Opinionated Defaults:

Smart Defaults: Spring Boot comes with a set of opinionated defaults for various configurations. This reduces the need for developers to make many decisions, especially in the early stages of development. It provides sensible defaults for databases, templating engines, and more.

Built-in Monitoring and Management:

**Spring Boot Actuator:** Actuator is a set of production-ready features that help monitor and manage applications. It includes health checks, metrics, info, and other endpoints that expose crucial information about the application. This facilitates better management and monitoring in production environments.

**Embedded Servers and Containers:**

Embedded Servers: Spring Boot applications can be packaged as executable JARs, including an embedded web server. This eliminates the need for external servlet containers, simplifying deployment and reducing operational complexity.

**Dependency Management:**

Starter Templates: Spring Boot provides starter templates that include a set of dependencies commonly used for specific tasks. Developers can include these starters in their projects, simplifying dependency management.

**Extensive Ecosystem:**

Integration with Spring Ecosystem: Spring Boot seamlessly integrates with the broader Spring ecosystem, including Spring Data, Spring Security, Spring Cloud, etc. This allows developers to leverage a wide range of libraries and tools for various functionalities.

Auto-Configuration:

**Automatic Configuration:** Spring Boot's auto-configuration feature automatically configures application components based on the project's dependencies. It intelligently sets up beans and configurations, reducing the need for manual setup.

**Testability:**

Testing Support: Spring Boot provides excellent support for testing, including unit testing and integration testing. It integrates well with testing frameworks like JUnit and provides tools for mocking and testing components.

**Community Support:**

Active Community: Spring Boot has a large and active community. This means ample resources, documentation, tutorials, and community support for developers facing challenges or seeking guidance.

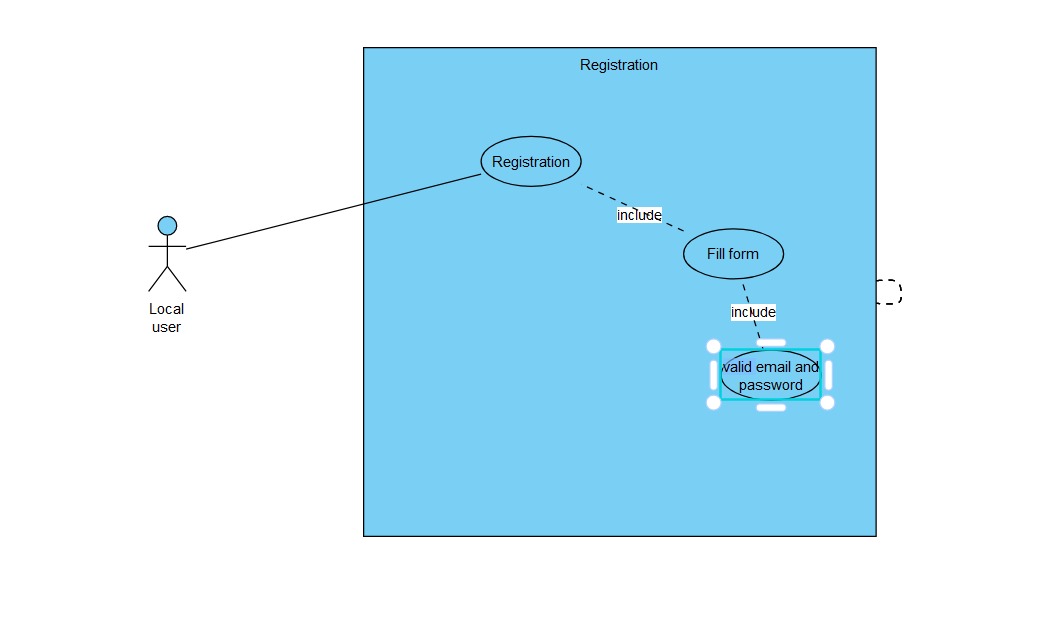
**Cloud-Native Development:**

Cloud Integration: Spring Boot supports cloud-native development and is well-integrated with cloud platforms. It makes it easier to develop applications that can be deployed and scaled in cloud environments.

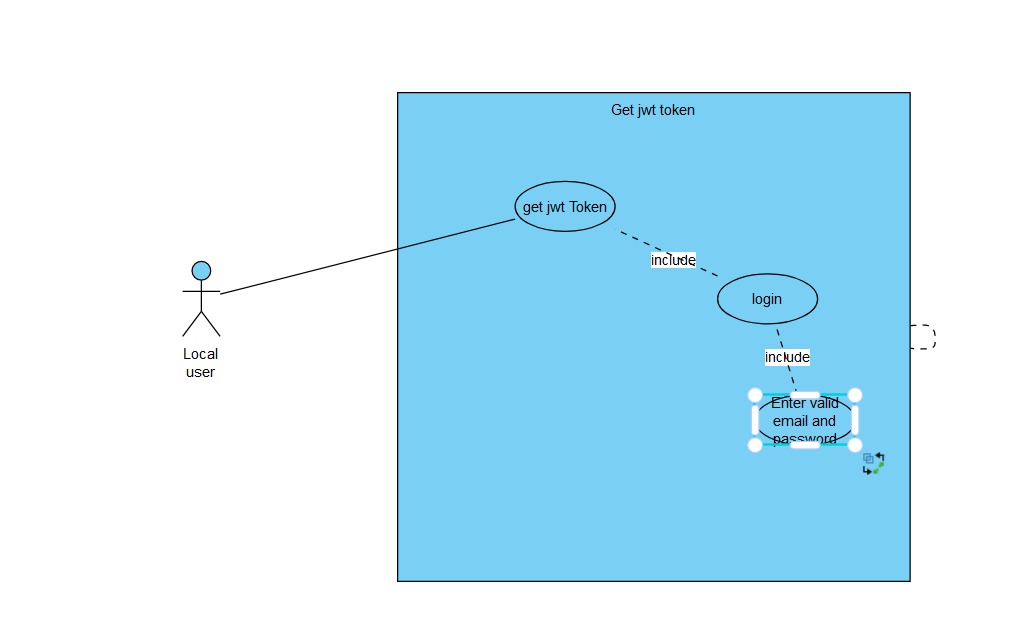
### Requirement Modeling:

#### Use Case Diagrams

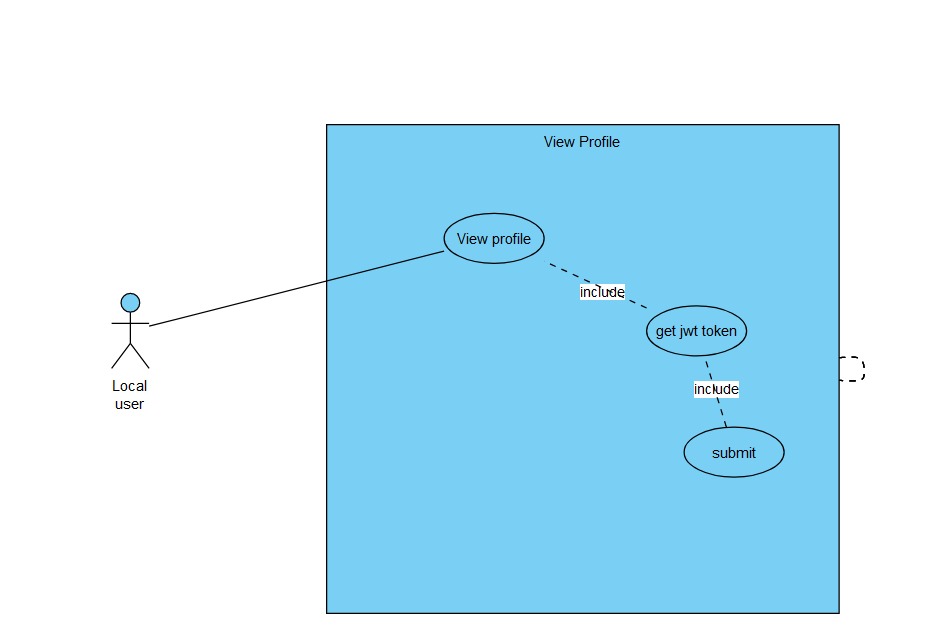
1. User Registration



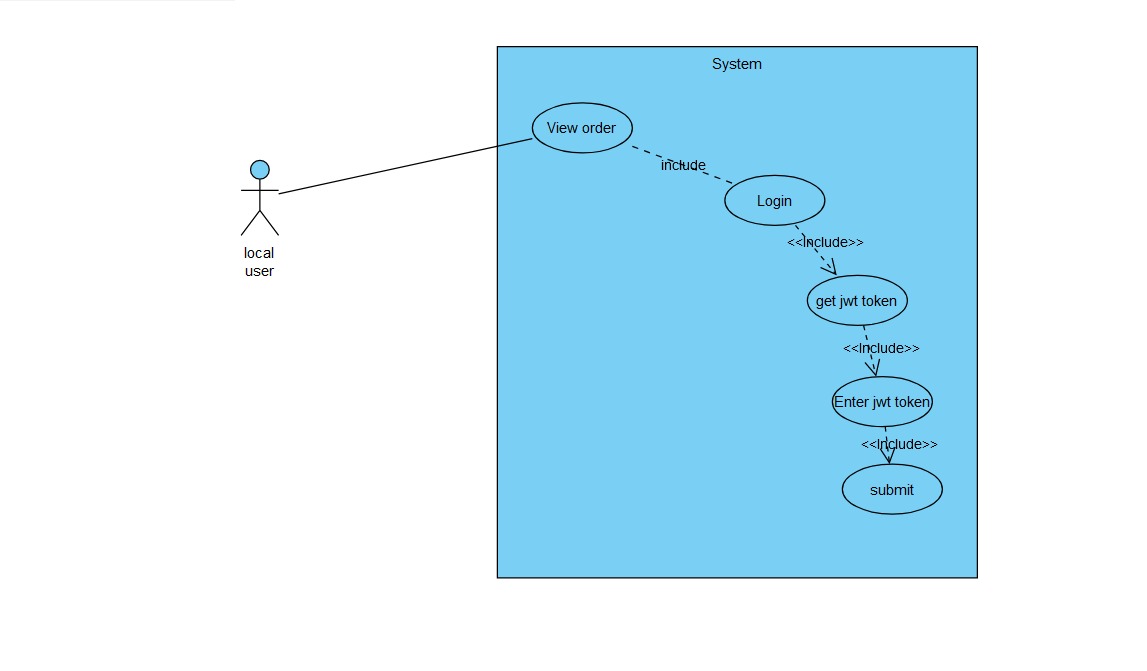
2.Get Jwt Token;



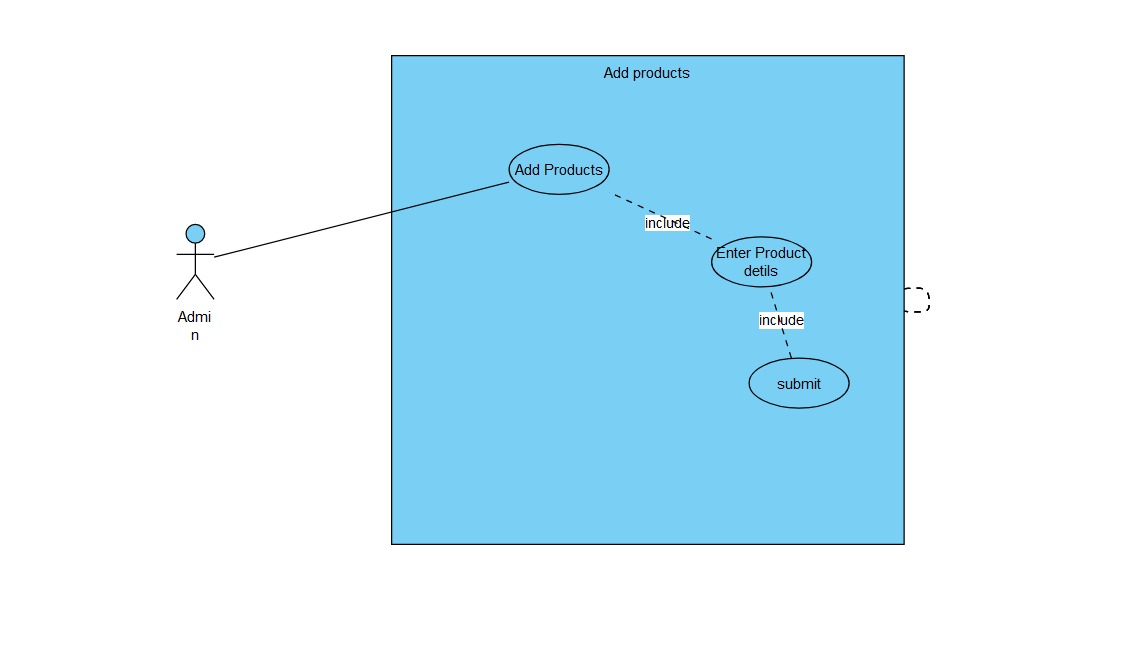
3.View profile:



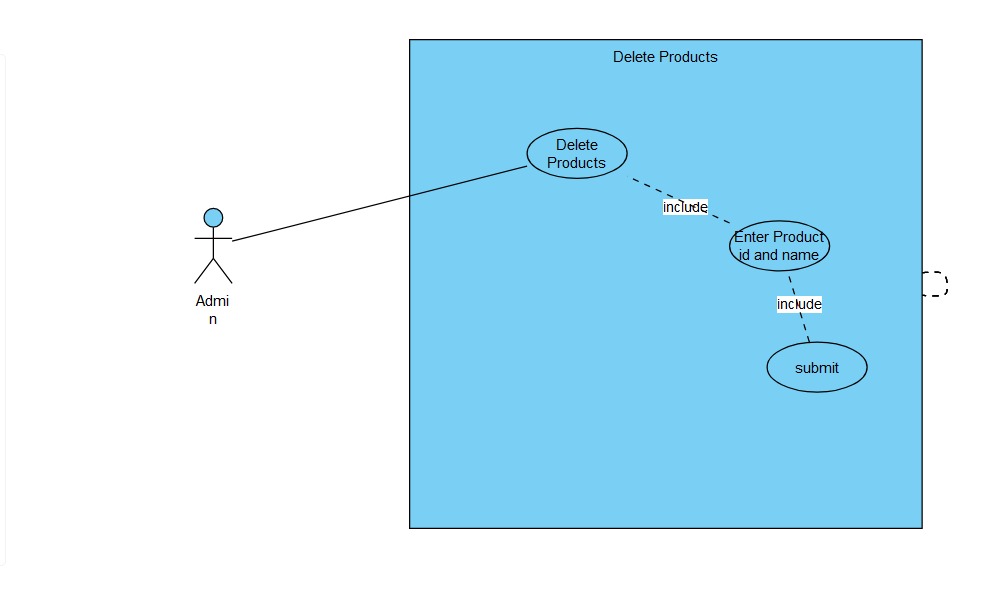
4.View orders:



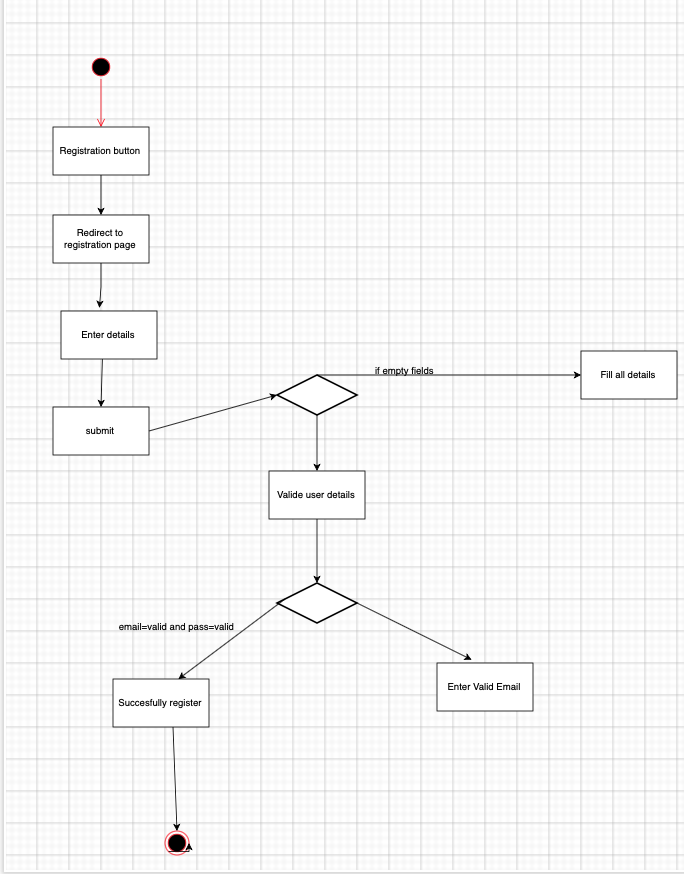
Add products:



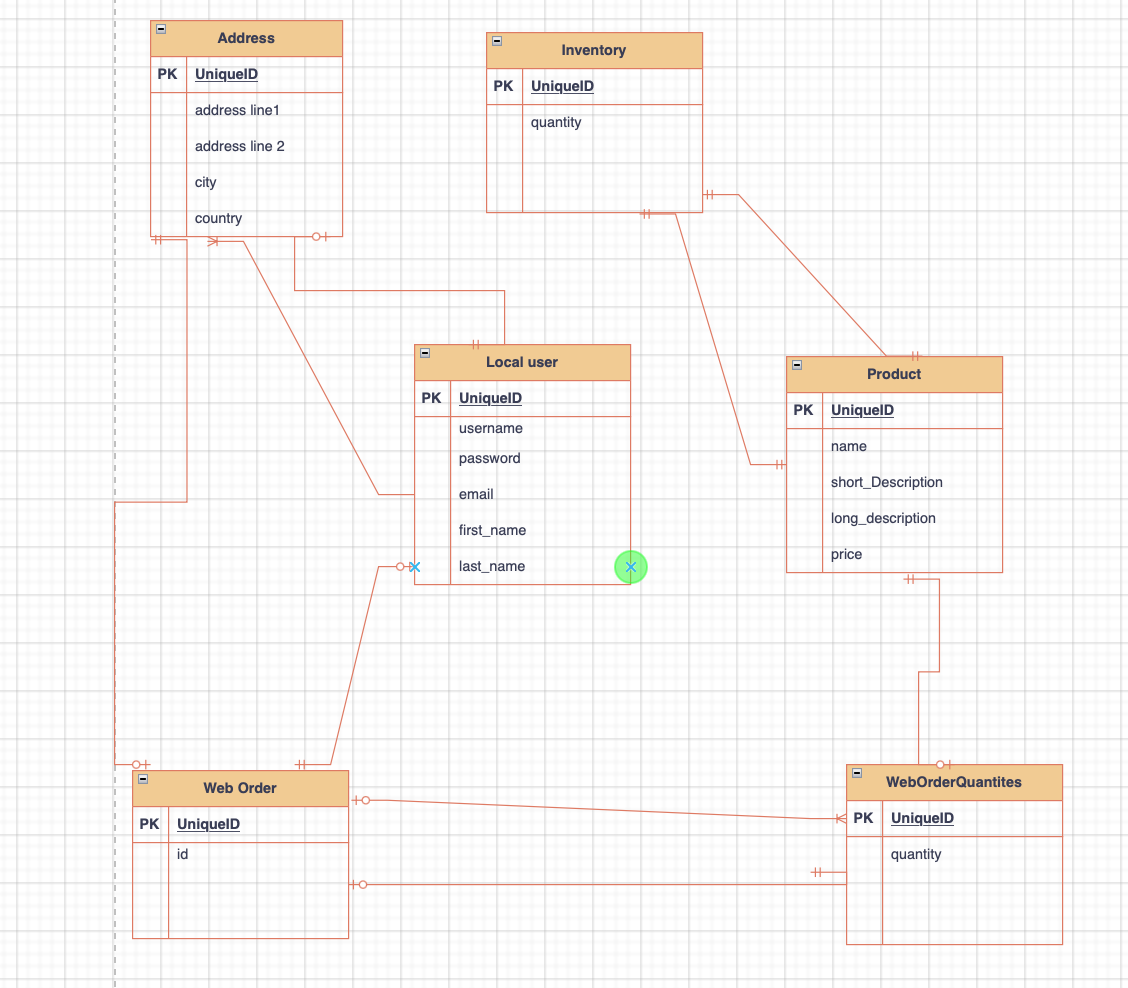
Delete Products:



##### Registration Activity Diagram:

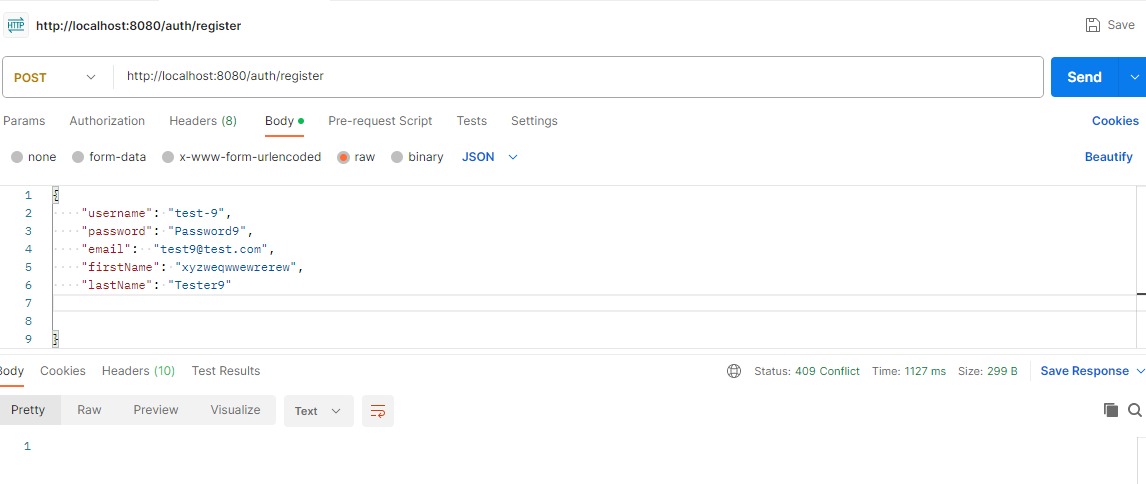


###### ERD Diagram

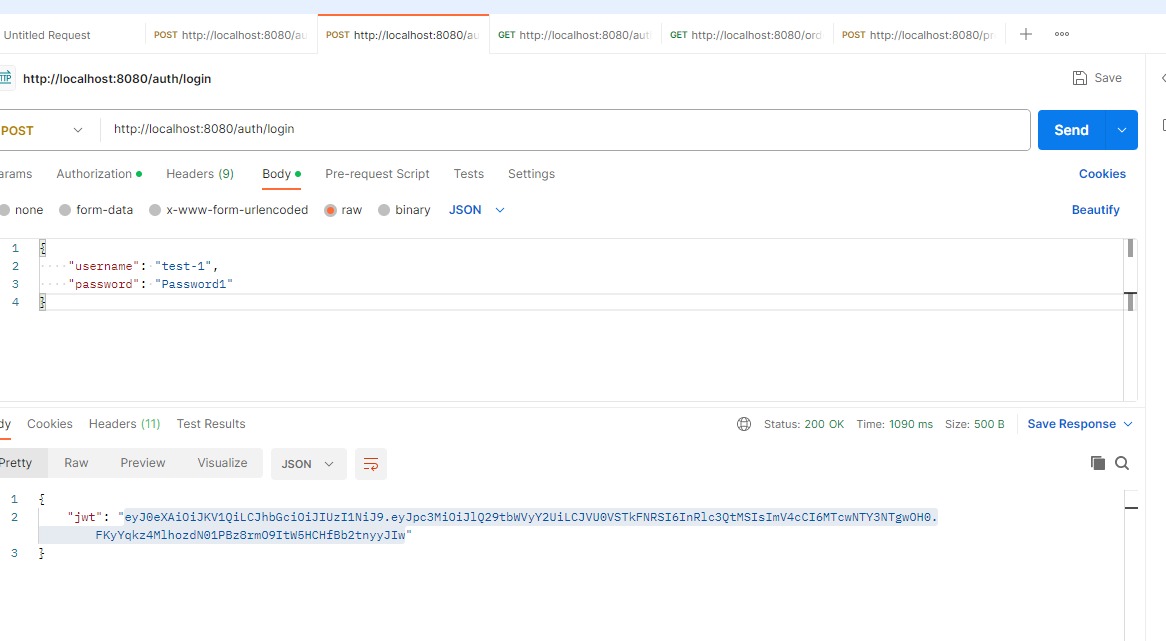


API:

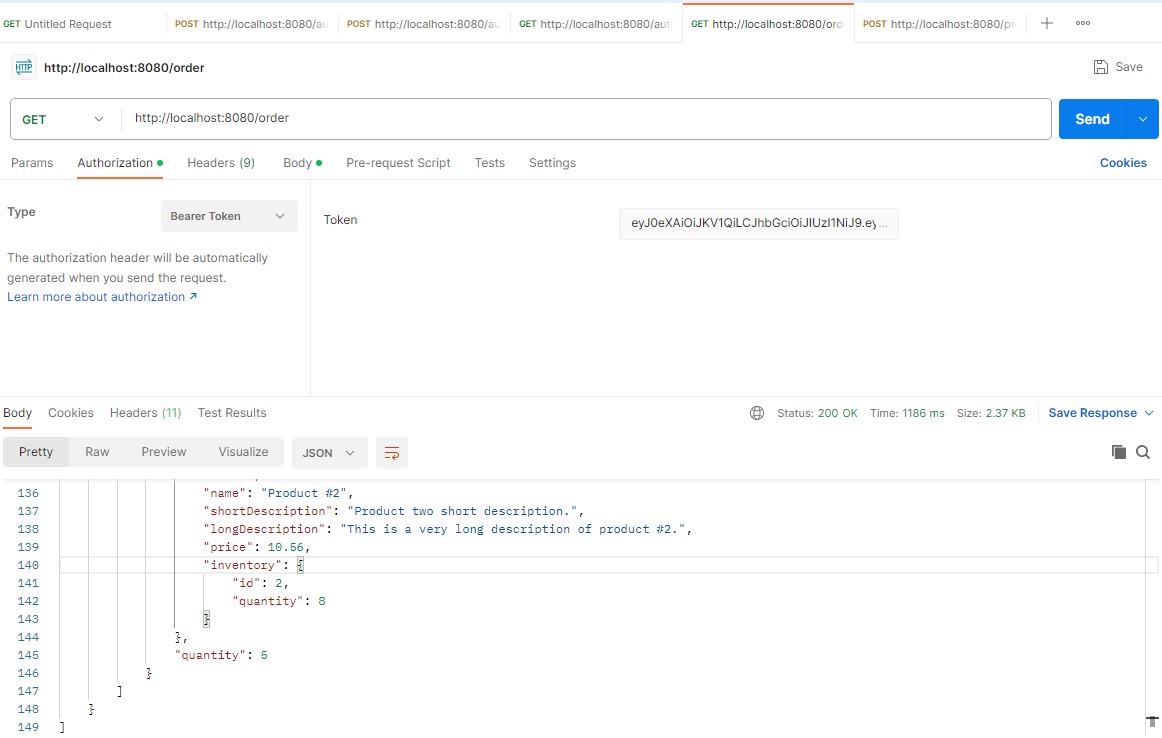
User Registrtaion API:



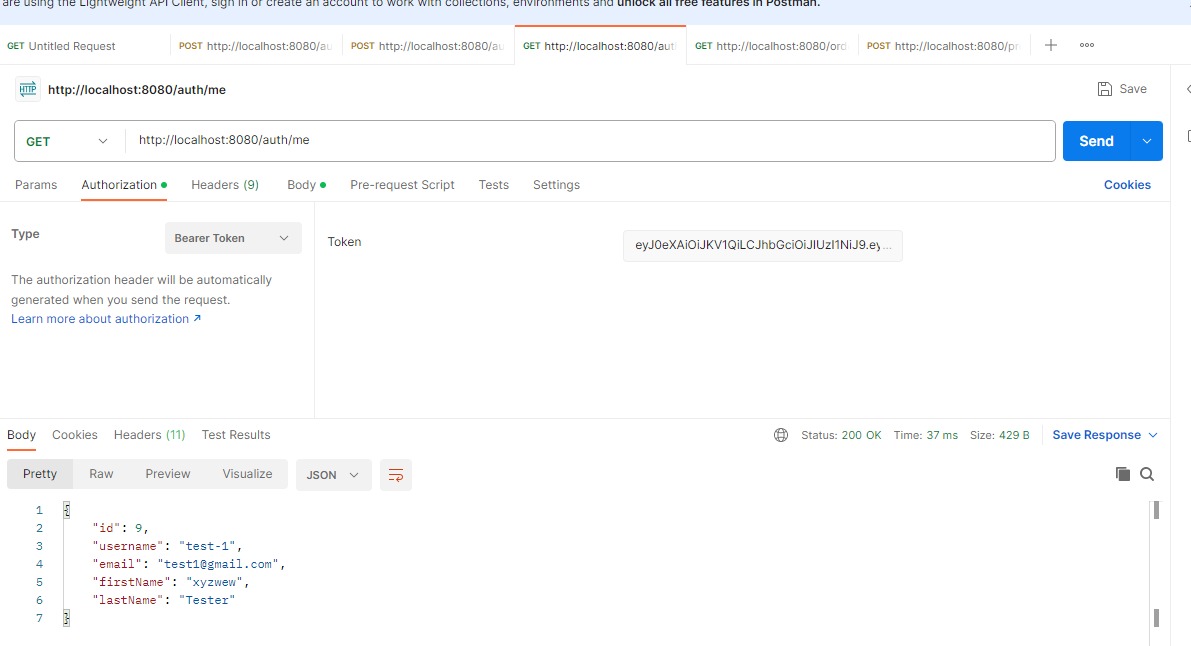
Login API:



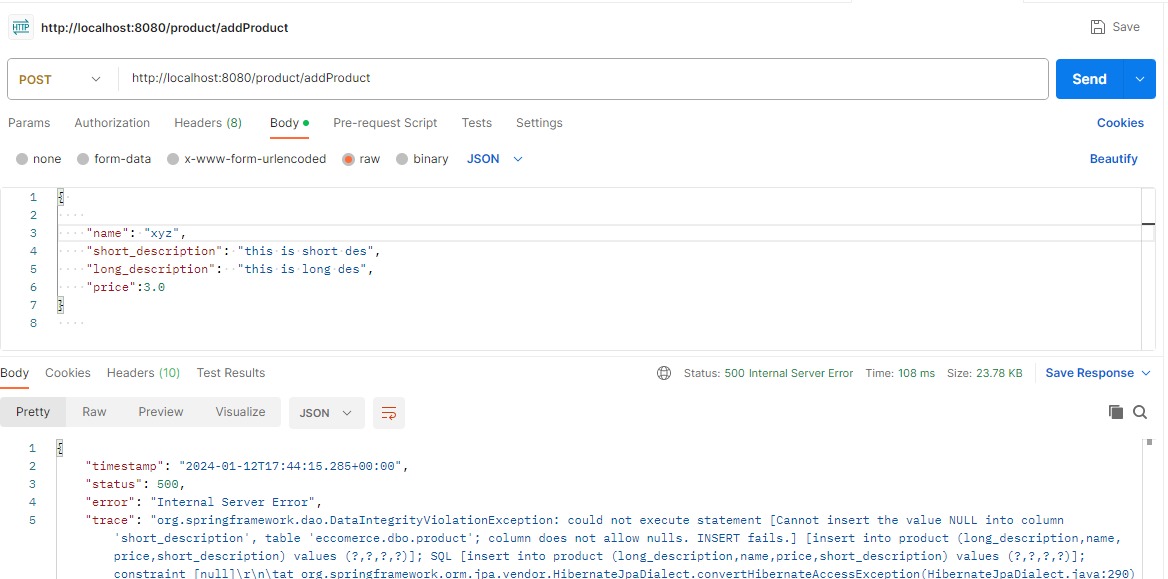
View Orders:



View Profile API:



Add Product API:

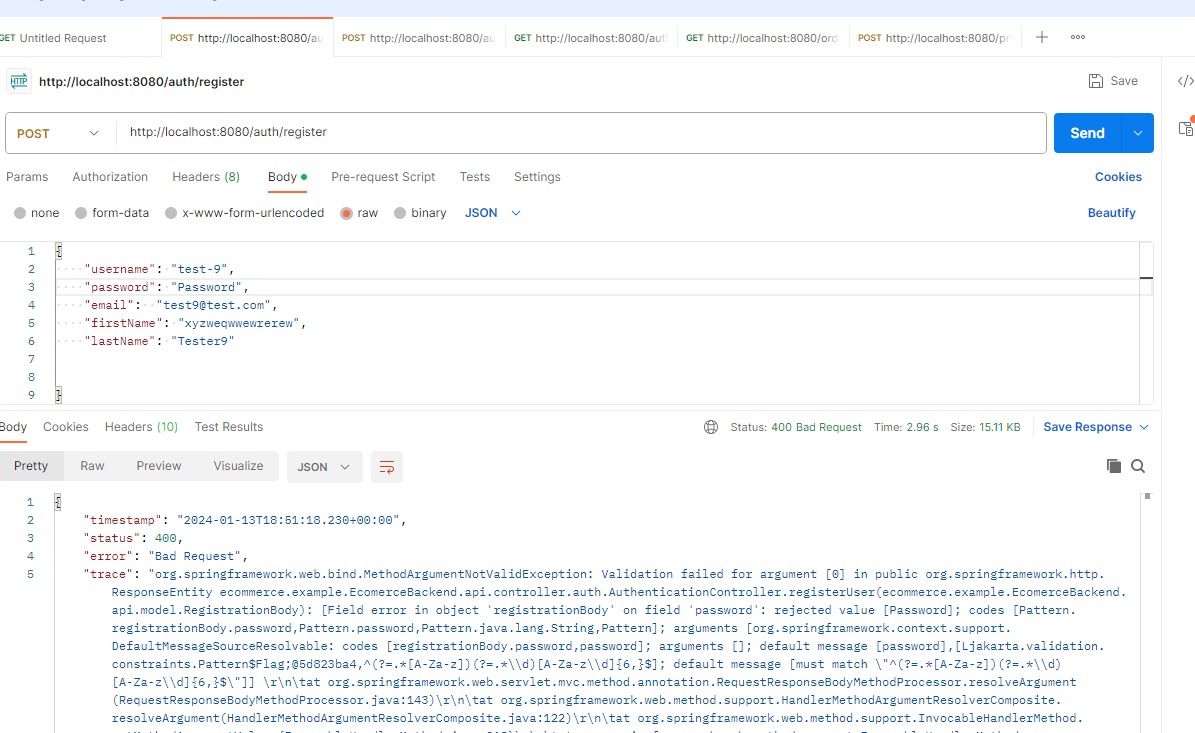


Testing:

User Registration API Testing with invalid password

* Registration with invalid password

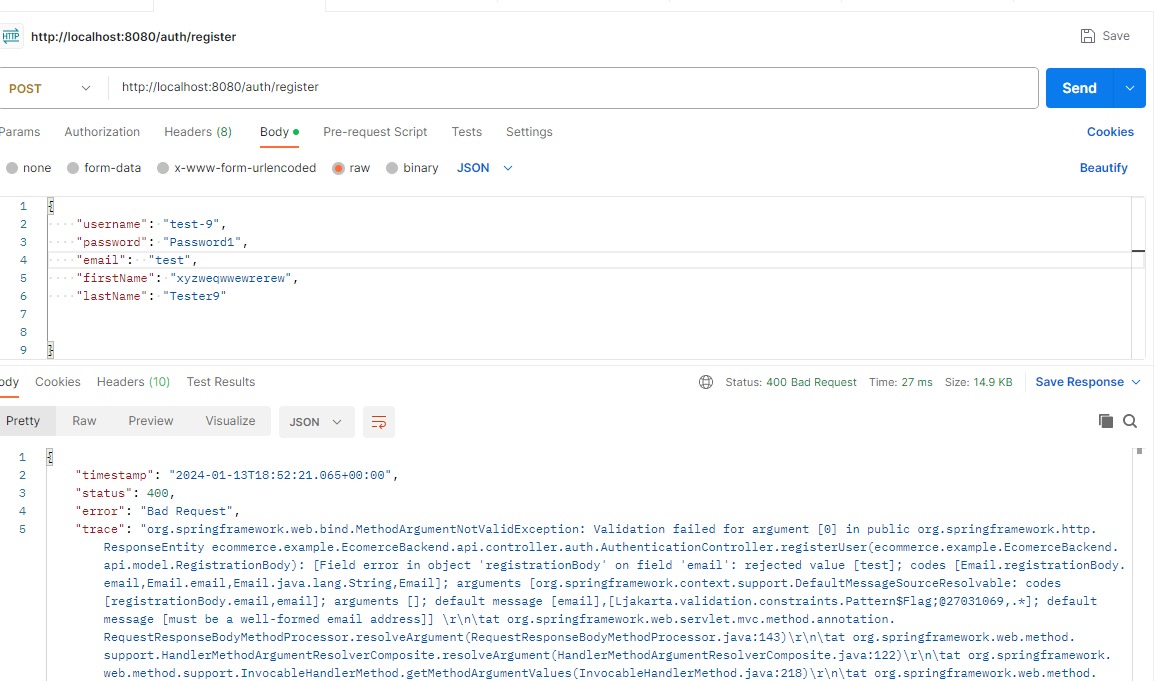
Result: pass



User Registration API Testing

* Registration with invalid usernmae

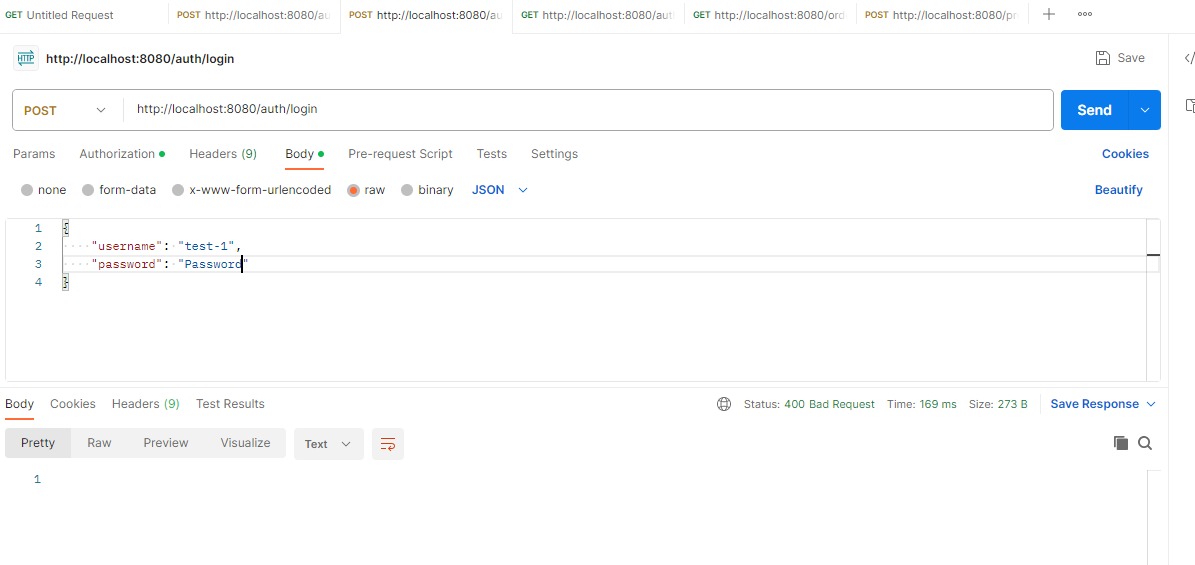
Result: pass



Login API testing:

* login with wrong password

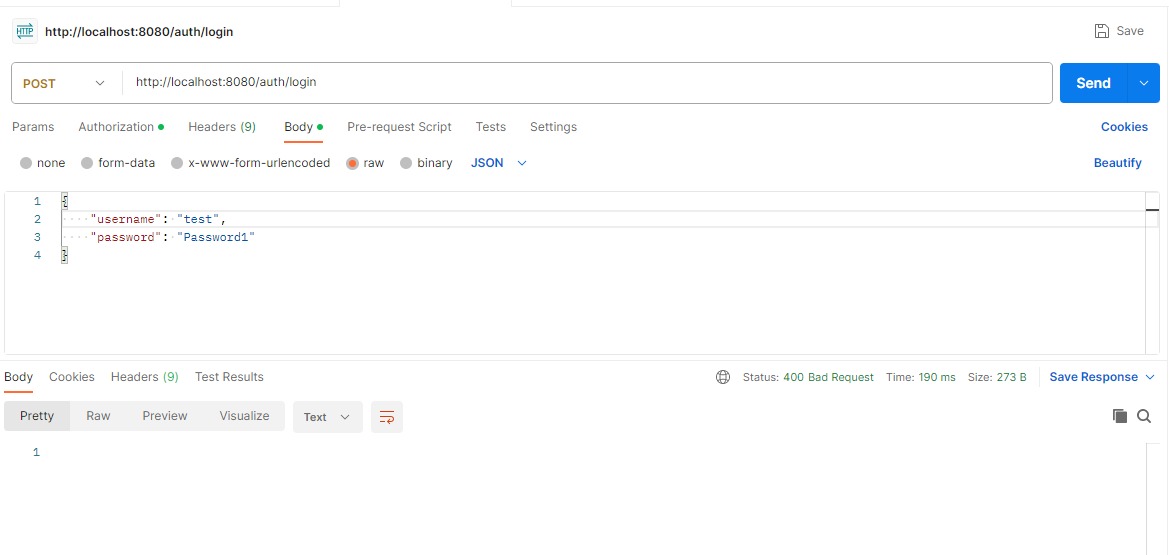
Result: pass



Login API testing:

* login with wrong username

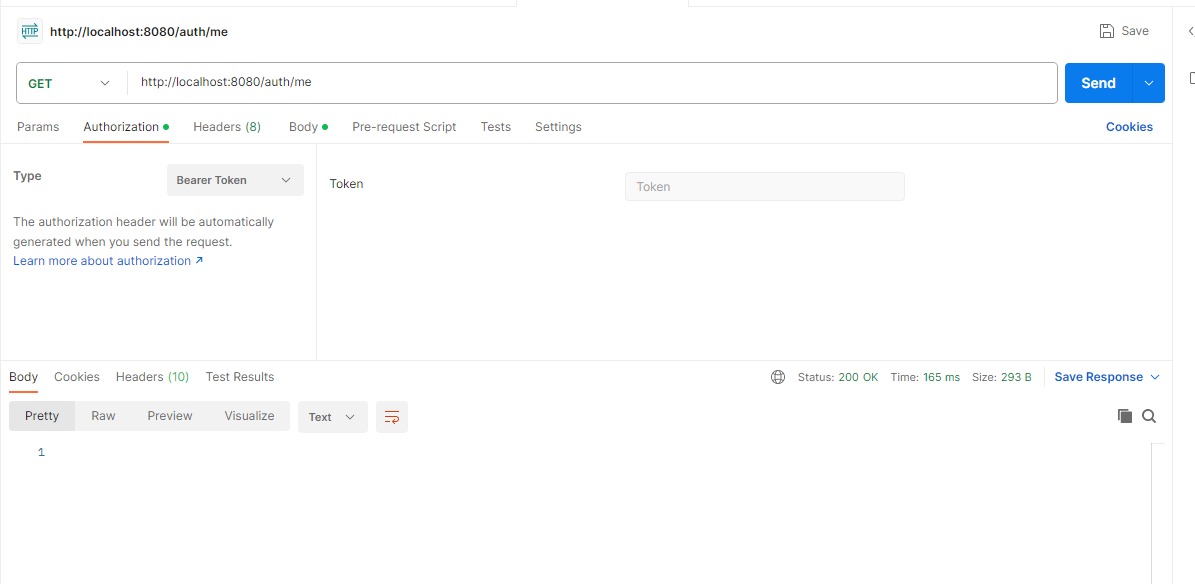
Result: pass



View Profile Testing:

* View profile without Token

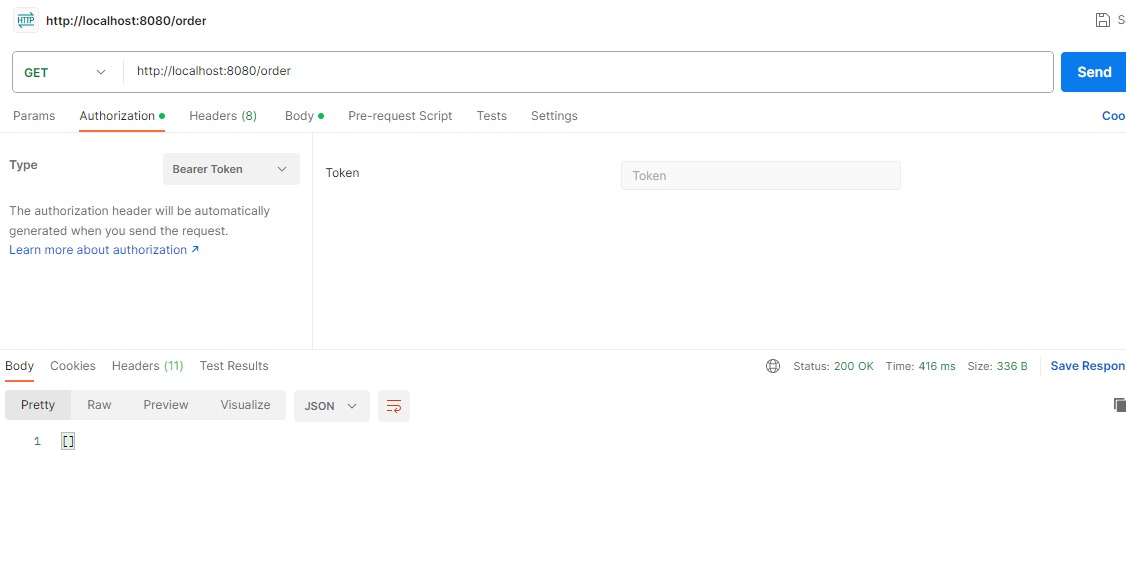
Result: pass



View Order Testing:

* View order without Token

Result: pass



Conclusion:

In summary, Spring Boot simplifies and accelerates the development of Java-based applications by providing a convention-over-configuration approach, smart defaults, a rich ecosystem, and support for modern development practices like microservices and cloud-native architectures. Its versatility makes it suitable for a wide range of applications, from small projects to large-scale enterprise systems.