

Spring Security JWT Authentication

A Spring Boot application demonstrating **JWT-based authentication and authorization** using **Spring Security**. This project is suitable for learning, internships, and backend practice.



Features

- User authentication with **JWT (JSON Web Token)**
- Secure REST APIs using **Spring Security**
- Password encryption
- Role-based authorization (if enabled)
- Clean project structure

Tech Stack

- Java
- Spring Boot
- Spring Security
- JWT
- Maven
- MySQL / H2 (configurable)

Project Structure

```
src/main/java/com/SpringSecurity/demo
├─ controller
├─ service
├─ repository
├─ security
├─ model
└─ DemoApplication.java
```

Configuration

Update `application.properties`:

```
spring.datasource.url=jdbc:mysql://localhost:3306/db_name
spring.datasource.username=root
spring.datasource.password=your_password

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

Authentication Flow

1. User logs in with username & password
2. Server validates credentials
3. JWT token is generated
4. Client sends token in `Authorization` header
5. Protected APIs validate JWT

Example header:

```
Authorization: Bearer <your_token_here>
```

Run the Project

```
mvn spring-boot:run
```

API Testing

Use **Postman** or **Swagger** to test APIs.

Future Improvements

- Refresh Token
- OAuth2 (Google / GitHub login)
- Swagger integration
- Docker support

Author

Nitish Gupta

GitHub: <https://github.com/Nitish-Gupta-NTV>

★ If you find this project useful, give it a star!