Movie Ticket Booking Application Developer’s guide

CONTENTS

[1. Objective 3](#_Toc519867706)

[2. Target Audience 4](#_Toc519867707)

[3. Architecture 5](#_Toc519867708)

[3.1 Security 5](#_Toc519867709)

[3.2 Database 5](#_Toc519867710)

[3.3 Deployment 6](#_Toc519867711)

[4. Development 7](#_Toc519867712)

[4.1 Prerequisites 7](#_Toc519867713)

[4.2 Maven POM 7](#_Toc519867714)

[4.3 Application Properties 7](#_Toc519867715)

[5. Deployment 8](#_Toc519867716)

[5.1 Stand Alone Tomcat 8](#_Toc519867717)

[5.2 Embedded Tomcat 8](#_Toc519867718)

[6. References 9](#_Toc519867719)

# Objective

This document provides detailed information about the Movie Ticket Booking application in technical terms to allow developers to extend the application and understand various functionalities of the app in detail.

# Target Audience

This document is targeted at anyone who is interested in using this application and modify its features.

# Architecture

The application uses Spring Boot 1.3.5 along with some other open source libraries as the base framework. The application till now is built with the focus on the following areas.

* Security – Default Spring Security and related validation.
* Database – MySQL
* Deployment – (Standalone) Tomcat
* (CI/CD) Jenkins
* (Static code analysis) Sonar

The application has a Maven parent POM. The parent POM defines a minimum set of required and security scanned libraries for building Spring Rest services .

## Security

Spring Security is used to satisfy the authentication requirement of this phase. The required dependencies should be added to the pom.xml file as mentioned in the later section.

Since we are using spring boot framework the lines of code required to use spring security is drastically reduced.

## Database

The database in this project is used for storing user details and his booking history. During registration user details are added to the database. For every subsequent login, the user needs to provide username and password.

The username field needs to be unique. mysql dependency needs to be added in pom.xml file

Name of the database is testdb; alternatively to use your database change the following line in application.properties file.

* spring.datasource.url=jdbc:mysql://localhost:3306/<database\_name>

The username of database used is "suprija" and password "suprijarao", alternatively these lines in application.properties can be changed.

* spring.datasource.username=<mysql-username>
* spring.datasource.password=<mysql-password>

Refer application.properties for more details regarding database configuration.

## Deployment

**Tomcat**

Tomcat needs to be installed to deploy the war file. Alternatively , spring boot’s embedded Tomcat can be used. Deployment will be covered in detail later

**Jenkins**

Jenkins is used for CI/CD. It provides various plugins to support building, automation and deployment. Jenkins war file needs to be downloaded and kept in web-apps for of tomcat. Now once tomcat is started Jenkins is available at tomcathome/jenkins

**Sonar**

Sonar is used for static code analysis. It scans the static code for any code smells.

# Development

## Prerequisites

* JDK 1.8 or later
* Maven 3 or later
* MySQL 5.6 or later
* Tomcat 8.x or later

## Maven POM

Apache Maven has been chosen as the build and dependency management tool to develop Movie Ticket Booking web application. As described in Section 3, a Maven parent POM that contains all necessary libraries for extending the application has been constructed to allow developers to create functionalities with concentration on business logic rather than worrying about the plumbing underneath.

## Application Properties

The following lines of code correspond to view displayed. This indicates that jsp pages are used for views

spring.mvc.view.prefix: /

spring.mvc.view.suffix: .jsp

Java Persistence API related configuration is specifies in the following lines.

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

MySQL related configuration is specified in the following lines

spring.datasource.url=jdbc:mysql://localhost:3306/testdb

spring.datasource.username=suprija

spring.datasource.password=suprijarao

spring.datasource.driver-class-name=com.mysql.jdbc.Driver

Validation related configuration is specified

spring.messages.basename=validation

# Deployment

## Stand Alone Tomcat

The war file can be deployed on tomcat . To generate the war file

mvn clean install

And place this file in webapps folder to deploy it on default port

For registration : <http://localhost:8080/Movie/registration>

For login : [http://localhost:8080/Movie/l](http://localhost:8080/Movie/registration)ogin

## Embedded Tomcat

The project can be imported in Spring Tool Suite as an existing Maven project and can be run in its embedded Tomcat by *choosing to run as Spring boot App.*

* For the register end point, use <http://localhost:8083/registration>
* For the login end point, use <http://localhost:8083/login>

# References

<https://hellokoding.com/registration-and-login-example-with-spring-security-spring-boot-spring-data-jpa-hsql-jsp/>