Banking Application - Web App (V1) – Developers Guide

Version v1

Document history

Date	Version	Description	Author
07/15/2018	1.	Initial version	BALAJI P

Contents

1. Objective

This document provides detailed information about the Banking Application web app in technical terms to allow developers to extend the application and understand various functionalities of the app in detail. This document is based on the first version (v1) of the web application.

2. Target Audience

This document is targeted at all Web Application Developers, including the Business Services API developers, Architects, Engineering Leads, and Security personnel, who will be contributing to any part of web Application.

3 Architecture

The Banking Application Web app uses Spring Boot version 2.0.3 along with some other open source libraries as the base framework. The Web app in this phase (Phase I) concentrates on the areas of

- Security – Authentication using the Spring security.
- CICD : continuous integration using Jenkins.

The Web app consists of a utility library and a Maven parent POM. The utility library is intended to help developers to create Rest services. The parent POM defines a minimum set of required and security scanned libraries for building Spring Rest services.

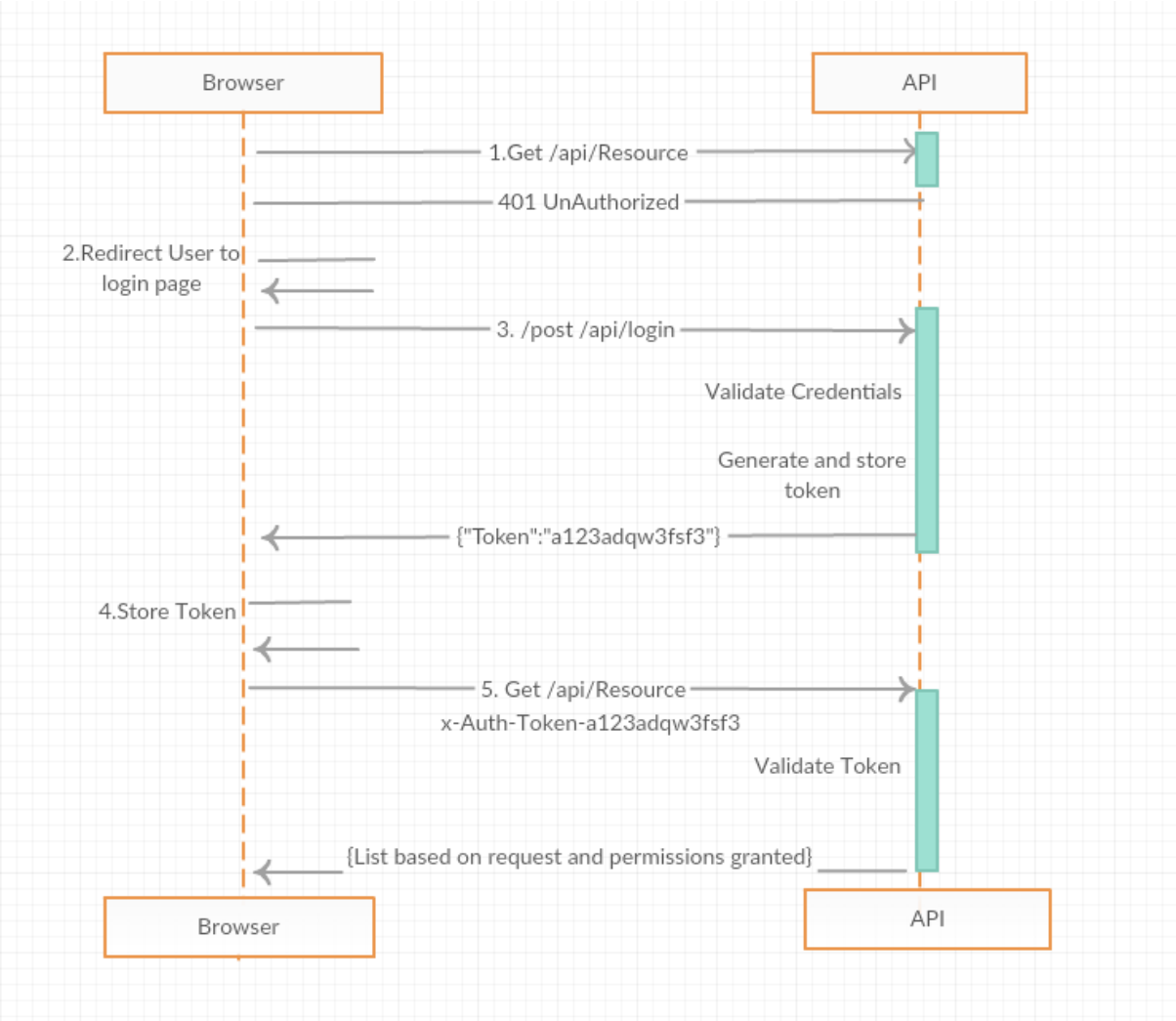
The first iteration of the framework supports Spring Boot applications deployed to stand alone Tomcat application servers. Only the users authenticated with spring security will be able to access the web application.

3.1 Security

In this release, the Framework uses Spring security as the authentication mechanism to validate user requests for the web application hosted in the Tomcat application servers. Spring Security is used to satisfy the authentication requirement of this phase. All the Security modules have been included in the *config package in the web app,* and there is no need for developers to directly deal with them.

As mentioned earlier, to access the secured web services deployed in a Tomcat application server, users must be valid registered users and have been authenticated with the Spring security . Figure 3 (next page) shows the authentication flow for a user to request a resource from a web service hosted in a Tomcat that is protected by the Spring security authentication scheme.

Fig .3



The steps of the sequence in Figure 3 are explained as follows:

1. Resource request before Authentication : The spring security tries to Authenticate the user using Jsession id that is not present in the server.
2. Redirect to login page : If the user tries to access the resources of the app before authentication , the spring security redirects the user to the Login page for Authentication.
3. User login – A User first logs into the website with his username and password.
4. Authentication with user credentials – The Workstation forwards the User's credentials to the Spring security.
5. Verify user credentials – The Custom Autentication of Spring secuirty service verifies the provided credentials with its MySQL database.

6. Authentication token returned – The Spring security returns the Authentication token to the Workstation if the User's credentials are matched with the credentials from the MySQL database
7. Cache token – The returned Authentication token is cached in the Workstation's local storage as JSESSION cookie.
8. Resource request – The User opens a Browser to request a resource from a URL provided by a Tomcat application server
9. Response Forwarding – Tomcat now forwards the response from Web App to Browser

4 Prerequisites

To develop web applications extending the Banking Web application, the developers' workstations will need:

- Maven 3.2+
- Eclipse Oxygen or Spring Tool Suite 3.9.1 with the m2e plugin
- JDK 1.8.0_101 or later
- Tomcat 8.0.36
- MySQL Database

4.1 Maven POM

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

4.3 Main Application Implementation

The main application class is implemented to instantiate all required Spring beans and to start the application with embedded Tomcat or to bind the servlet, filters, and beans from the application context to a stand-alone Tomcat server.

To implement the main application class, simply use the following code (replacing the class name with the preferred name):

```java
package com.balaji;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class MyapplicationApplication extends SpringBootServletInitializer {

	@Override
	protected SpringApplicationBuilder configure(SpringApplicationBuilder applica
	{

		return application.sources(MyapplicationApplication.class);
	}

    public static void main(String[] args) {
        SpringApplication.run(MyapplicationApplication.class, args);
    }
}
```

SpringBootServletInitializer is used here to allow the Spring Boot application to be deployed to a servlet container, such as a Tomcat application server.

4.4 Application Properties

Application properties should be defined in properties format. Use the *application.properties* file in the `src/main/resources` directory of the Banking web Application project as a starting point.

The following properties are required:

```
spring.mvc.view.prefix=/WEB-INF/view/
spring.mvc.view.suffix=.jsp



spring.datasource.url = jdbc:mysql://localhost:3306/assignment1?useSSL=false
spring.datasource.username = balaji
spring.datasource.password = balaji

spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
logging.level.org.hibernate.SQL=debug

server.servlet.context-path=/firstproject

```

Change the url,username and password of the SQL database with the that of your system. Also the first line is used to give the location of jsp files.

4.5 Build Process

Build The project: `mvn clean install`

5. Deployment

5.1 Stand Alone Tomcat

Copy the generated example-0.0.1-SNAPSHOT.war from the target folder of the Banking web application App to the web apps folder of the residing Tomcat Server in your system.

5.2 Embedded Tomcat

To run applications with embedded Tomcat ,

Right click on the app in the Eclipse and choose to run as Spring boot App.

To run applications with embedded Tomcat in a production environment, use the command

“java –war example-0.0.1-SNAPSHOT.war --server.port=8800 --server.servlet.context-path=/firstproject”

You can access various end points from an instance started with the second command above from a browser:

- For the login end point, use

  “http://localhost:8080/firstproject/login”

- For the register end point, use

  “http://localhost:8080/firstproject/register”