

# Web Application Development using SpringBoot and Angular

Nithya Sai Teja Reddy Sangati

ComputerEngineeringandDataScience  
PresidencyUniversity  
Tadipatri, India  
[nithyasatejareddy99@gmail.com](mailto:nithyasatejareddy99@gmail.com)

Ms.Bhagya.K

Assistant Professor  
Computer Science and Engineering  
PresidencyUniversity  
Bangalore,India  
[bhagya.k@presidencyuniversity.in](mailto:bhagya.k@presidencyuniversity.in)

**Abstract**—There are two ways to create web pages, one way is to use a multi-Page application, and another way is to use Single Page Application. This research paper focuses on the complete creation of a Single Page Application (SPA) using an open-source java framework called Spring Boot for the back end and a JavaScript framework for the front end called Angular.

**Keywords**—Spring Boot, Dependency Injection, Angular, Bean.

## I. INTRODUCTION

Nowadays, all the corporate companies are using web applications or stand-alone applications to maintain their business and its functionality. These applications, save their time and money. If the apps are secured and web-enabled to collaborate with their employees and store customer details for future insights. Spring Framework is the core of brand creation. It was created as an alternative to programming applications that use EJB (Enterprise Java Beans) [1].

## II. IMPORTANCE OF WEB APPLICATIONS

Web apps are necessary for a variety of reasons. They have several advantages, one of which is that they conduct all of the required functions using a web browser rather than installed software. Web apps are also becoming a vital component of companies in today's increasing global needs due to their cloud-based capabilities. To suit their business needs, companies are embracing and developing web applications with the help of developers. Here are five compelling reasons to explore web applications.

## III. METHODOLOGY

For web application development, we are using MVC architecture. MVC is an architectural pattern, which contains three parts Model, View, and Controller.[2]

### A. Model

It is the lowest level in the architecture that is responsible for data management. It is connected to the database internally. Any changes in the model are reflected in the database. For any data, the controller will contact only the model and then the model communicates with the database and provides the data.

### B. View

View Component is responsible for data representation. It generates a user interface for the user. Views use HTML/CSS pages and include data from the model which are requested by the controller.

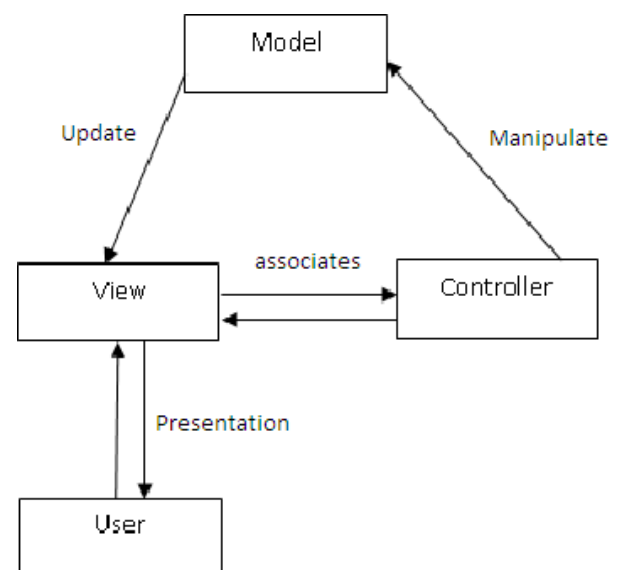


Fig. 1. MVC Architecture

### C. Controller

The controller is the main component that enables the communication between the other two components' view and model and acts as an intermediary. The controller doesn't dwell on handling data logic, the data logic is maintained by the model, and the obtained information from the model is then sent to view.

We can create as many as controllers based on our requirement and each controller will receive the request from the user and the process is based on the code. As in Fig 1. User events on the view will call the controller and these changes are applied to the model and the updated data is displayed on the User Interface.

### IV. WORKING FLOW

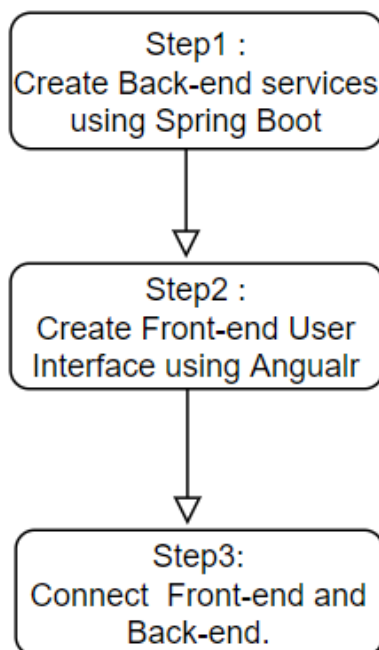


Fig.2.Working Flow

### A. Backend Service Creation

Spring Boot is a java based open-source framework which is used to create stand-alone, web-based applications. Spring Boot is built on the top of the conventional spring framework (an open-source lightweight framework used to develop enterprise applications.) used to create REST APIs. To use spring boot, we can directly bootstrap our application from a website called <https://start.spring.io/> and use it in building tools like maven. We can also create and use a

spring boot application in an Integrated Development Environment called Spring Tool Suite (STS).

Before setting up STS, we need to install Java JDK (Java Development Kit), which is crucial for the development. After installing JDK (preferably JAVA 8.0) add the path of the JDK file inside System Environment Variables and then install STS. After installing, open the tool, and it will prompt you to select the directory of the working location. In this location only, all the work is saved.

To create a new project in STS, go to File>New>Spring Starter Project. After that, we will be prompted with a window, which contains the name of the project, the java version, and other configurations for our project.

After successful completion of the project creation, we are provided an application with basic configurations [3]. The application contains many packages and some of them are very crucial. The package src/main/java contains all the java codes which are required for our project to work. The package src/main/resources contain JavaScript, HTML, CSS, and Images too, there is a file named application.properties, which contain configurations to run in different environments. There is another file named pom.xml, which adds the packages required for the application. For example, if we need to connect to a database from our code, we can use the previously available code instead of writing on our own. If we add a spring-boot-starter-jdbc dependency, then we can use it just by importing. Similarly, we can add other dependency codes based on the functionality requirement.

In the Spring Boot application, we use annotations. These are a kind of metadata, which provide information about a program. Main annotations are "ComponentScan", which scans a package to find all the beans (class in terms of java), "Configuration" is used by Spring Containers as a place for bean definitions, and "EnableAutoConfiguration" is used to automatically configure the beans present in the main code and to run the methods. All the above three annotations are replaced by "SpringBootApplication" [4]. We are using APIs,

which are protocols for building and integrating application software. We are using REST (Representational State Transfer) API for our application development. RESTful API transfers data in any one of several forms via JSON, HTML, or plain text. It doesn't store client information between getting requests and each request made by the user is separate and unconnected. [6]

### B. Front End (User Interface) Creation.

We may also utilize the Angular Framework to develop the front end. Angular is an open-source JavaScript framework based on TypeScript. It's operated by Google, and one of its main goals is to develop single-page apps. As a framework, Angular has obvious advantages, and it also provides a standard structure for developers to work with. It enables users to create large, easy-to-manage applications.

To work with Angular Framework, we need to install Node.js. Node.js is an environment that runs JavaScript on the server. Along with Node.js, Node Package Manager (NPM) will be installed. NPM is a large Software registry, which is open source. Now, we are installing the Angular command-line interface from this npm only.

After installing Angular CLI, we can create a new project with the command "ng new project\_name" in the command shell. While generating the project, a few questions are asked to configure the project, and give yes or no based on your requirements. This will generate the basic project. We need to edit the generated project file. Each part of the page is considered as a component in Angular. To generate a new component, use the command "ng generate component component\_name" [5]. After the submission of the command, it will generate a file that contains the HTML, CSS, and Component Class Files. We can design the component as per our requirements.

Based on the requirement, we can create as many as components required. To connect these components, we need to use a routing module, which helps in routing the request from one component to another. The routing module is created if you had given yes while project creation or else, we can create the routing module using the command "ng generate module module\_name --routing".

Angular create Single Page Applications. There are numerous individual single components in the paradigm of single-page web apps. Instead of loading the complete page for each user action, the components update on their own. As a result, it grows customer satisfaction, responsiveness, and interaction with the website [7].

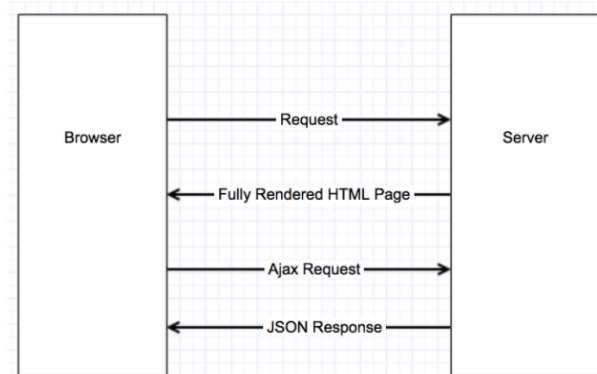


Fig. 3. SPA Working

In a multi-page application, whenever we click a link on a web page, it will move to a new page. Whereas in Single-page Applications, only one HTML page is rendered on the client-side. If the user clicks on the links or components, the only particular part will be fetched from the server, and the page will not be reloaded.

### C. Connecting Front End and Back End.

After the creation of the back end and front-end, we need to connect them to get data from the back-end to front end. We need to create a service in the angular project, this should be able to access the back end by sending the HTTP request. To send an HTTP request, we need to inject HttpClient into the constructor of the service [8].

There are four types of requests we can make:

1. GET
2. POST
3. DELETE
4. PUT

GET request is used to request data from the back-end. A POST request is used to send data to the back-end. DELETE request is used to delete resources in the back end. A PUT request is used to update a resource in the backend. All these requests are sent to the application, which was developed in spring boot, and the controllers in spring boot will execute these requests. Instead of PUT and DELETE, we can use POST to perform these operations [8].

## V. ADVANTAGES AND DISADVANTAGES

MVC architecture is easy to maintain, and the components can be reused. Testing can be done on the components independently. On the other hand, it has high complexity and is not fit for small applications. The View component does not have data access. It is hard to understand. However, it is not impossible to learn and code. It separates business logic and the presentation.

## VI. CONCLUSION

Through this paper, we have provided a process of creating a web application using spring boot and angular. Single-page applications will grow the user interactivity as they will respond in less time. It will take more time while loading for the first time but for the subsequent actions, the time taken is less than the Multi-Page Application. As we are using component-based architecture, it is easy to debug the errors and code reusability chances are also high.

## VII. REFERENCES

- [1] M. Gajewski and W. Zabierowski, "Analysis and Comparison of the Spring Framework and Play Framework Performance, Used to Create Web Applications in Java," 2019 IEEE XVth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH), 2019, pp. 170-173, doi: 10.1109/MEMSTECH.2019.8817390.
- [2] <https://towardsdatascience.com/everything-you-need-to-know-about-mvc-architecture-3c827930b4c1#:~:text=%2DMVC%20is%20an%20architectural%20pattern,the%20view%20whenever%20data%20changes>.
- [3] <https://spring.io/quickstart>
- [4] <https://www.javatpoint.com/spring-boot-annotations>
- [5] <https://angular.io/cli>
- [6] <https://www.redhat.com/en/topics/api/what-is-a-rest-api>
- [7] S. Deshmukh, D. Mane and A. Retawade, "Building a Single Page Application Web Front-end for E-Learning site," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), 2019, pp. 985-987, doi: 10.1109/ICCMC.2019.8819703.
- [8] <https://codingthesmartway.com/angular-6-mean-stack-crash-course-part-3-connecting-front-end-to-back-end/>