IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

Volume 2, Issue 5, June 2022

Impact Factor: 6.252

Spring Boot Backend Development

Ritik Singhal

Student, B.Tech, Department of Computer Science and Engineering

Dronacharya College of Engineering, Gurugram, Haryana, India

Abstract: *Every web application must be able to save and update HTTP-accessible data. This information must be kept safe. Spring Boot provides a good platform for Java developers to develop a stand-alone and production-grade spring application that you can just run. The backend may be more difficult to get started with because seeing the results of your efforts is more difficult. In this article, we'll go over what backend development is all about, as well as backend programming approaches. Java is an excellent choice for any application.*

Keywords: Spring Boot, Spring JPA, Java, H2 Database

I. INTRODUCTION

In the field of Web development, data management and updating is critical, as is data security. Applications are constantly evolving, with additional features and enhancements getting implemented all the time. Many applications are built with Spring Boot, a Java framework. It is easy to construct and deploy a standalone spring application with minimal spring configurations. Spring MicroService Architecture can be implemented in java for any web application. Regardless of the fact that NoSql is rising in popularity, RDBMS has a numerous advantages, one being that the the data is structured. We will be using H2 database and JPA to connect to databases in our research. Spring JPA helps in code development. It minimizes the amount of time and energy necessary for development and maintenance. You can make custom finder methods like findById() and interfaces for JPA repositories, and spring boot will take care of the implementation. The next part shows how to use h2 database with spring data JPA. Spring Boot enables developers to easily build apps that use standard RDBMS or embedded databases.

Section I contains an introduction to using Spring data JPA with H2 database, Section II includes related works and technologies, Section III describes the methodology, Section IV explains the implementation, Section V contains the results and discusses data storage, and Section VI concludes the study with research recommendations.

II. LITERATURE REVIEW

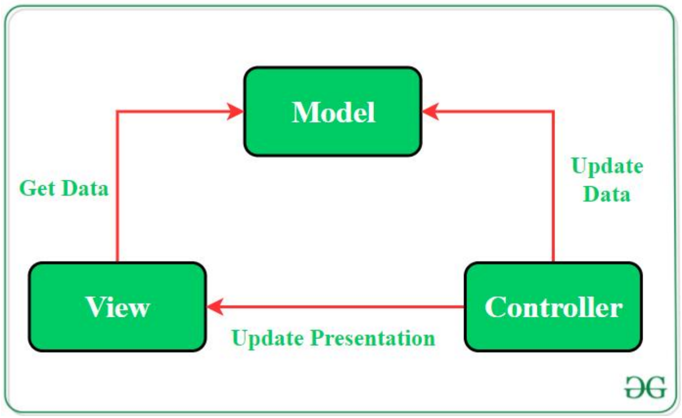
The Spring Boot Framework is a well-known and widely used framework for web and corporate application development. Spring boot uses dependency injection that allows you to create beans via XML, annotations, and Java code. It's a promising platform for creating business and web apps. JPA (Java Persistence API) is a specialised technology that allows you to interface with relational databases. It serves as a bridge between SQL Tables and Java objects. The persistence layer is required for real-world projects that require data storage and retrieval. The author presents how framework integration can speed up system development and significantly cut on coding time. It describes the design and implementation of the Spring and Spring Boot frameworks in the development of Web applications, which reduces the development process and burden of the system, allowing for easier growth and deployment.

III. METHODOLOGY

This paper's goal is to design and test a backend for a web application. We will use the following methodologies to achieve this: Spring Boot is a Java-based open source microservices framework. Micro Services is a type of architecture that allows developers to build and deploy services independently. Each operating service has its own process, resulting in a business application support paradigm that is lightweight.

The Java-based Spring Web MVC framework provides a Model-View-Controller (MVC) architecture and components that can be used to create flexible and loosely coupled web applications. The MVC architecture isolates the application's input logic, business logic, and user interface logic while allowing for loose coupling.

- The Model layer wraps the web application data, which is commonly POJO.
- The View layer is in charge of presenting the model data and producing HTML output that the client's browser can understand.
- The third layer, the Controller, is in charge of handling user requests. This layer generates a model, which is then rendered by the view layer.



Spring Data JPA, which is part of the larger Spring Data family, simplifies the creation of JPA-based repositories. This module focuses on improving JPA-based data access layer support. It simplifies the development of Spring-based apps that use data access methods.

For a long time, developing a new data access layer for an application has been difficult. To perform simple searches, pagination, and audits, far too much boilerplate code is required. Spring Data JPA aims to significantly simplify data access layer implementation by reducing the amount of effort required to the bare minimum. You write your repository interfaces, including custom locator methods, as a developer, and Spring handles the implementation.

Spring Tool Suite IDEA, an integrated development environment (IDE) for creating spring boot microservices, is used throughout the process. It includes a pre-configured workspace as well as a plug-in framework for additional customization.

IV. IMPLEMENTATION

In this paper, we will concentrate on developing a backend using the Spring Boot framework and fully configuring the JPA persistence layer. H2, a relational database, will be used for data storage. Java is used throughout the implementation. This article walks you through the process of configuring the Spring context using Java-based settings and the project's standard Maven pom. The following steps are followed:

To begin, we must create a project in start.spring.io and add the necessary dependencies such as JPA, H2, JDBC and so on.

The next step is to configure Spring JPA in the application. properties file of the project.

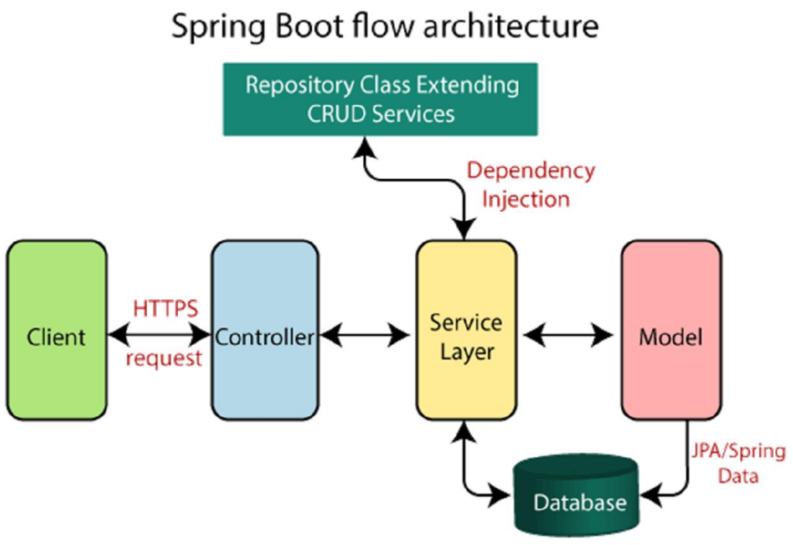
We must create a DataModel class under the Entity model. The annotation @Entity indicates that this class is an Entity. For example, if there is a class sSudent and its attributes are first name and last name, an automatic Student table with id, first name, and last name columns will be generated.

The following annotations are being used:

- @Entity: This annotation indicates that this class is a hibernate entity that will be mapped to a database table.
- @TableName: The name of the table to which H2 database should map.
- @Id: This specifies that this particular attribute is the table's primary key.
- @GeneratedValue denotes the primary key's value generation strategy.
- @Column: This specifies the name of the column to which the Java class attribute is mapping.

We must create an interface for the package repository that allows us to perform all CRUD operations on the datamodel class. The interface is an extension of CrudRepository and will be autowired in RestController to support repository and custom finder functions.

We specify the type of REST API call in the Controller layer by using annotations such as @GetMapping, @PostMapping, @PutMapping, and @DeleteMapping. We've also specified the web service's endpoint. This layer receives the input as a request object and forwards it to the next layer, the Service layer. Request mapping methods available in the RestController include save, findAll, findById, and findByFirstName.



To accomplish this, we'll build a package controller that accepts client requests, updates/gets data from a repository, and returns results. Make an H2 database that will be mapped to our entity.

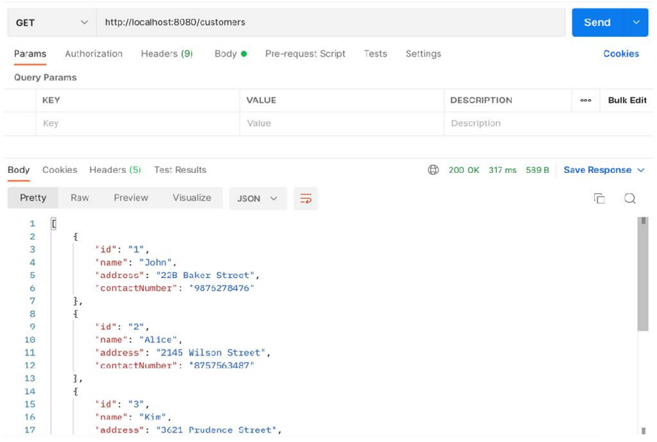
We used autowired repository methods that implement the interface CrudRepository in RestController methods annotated with @RequestMapping. The final step is to run the spring boot project and test the results. In this section, we have written logic to set the result data list to our response object, which is in JSON format.

When we send a request for a resource with the HTTP method, URI, and request body specified, we receive a response in the form of JSON along with HTTP status codes. The four HTTP methods listed below are commonly used.

- The GET method is used to access a read-only resource.
- A new resource is created and added using the POST method.
- The DELETE method is used to delete an existing resource.
- The PUT method is used to update an existing resource.

V. RESULTS AND DISCUSSION

The response for our request to the server is in JSON format.



```
http://localhost:8080/customers
```

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
200 OK 317 ms 539 B
```

```
[
    {
        "id": "1",
        "name": "John",
        "address": "22B Baker Street",
        "contactNumber": "9876278476"
    },
    {
        "id": "2",
        "name": "Alice",
        "address": "2145 Wilson Street",
        "contactNumber": "8757563487"
    },
    {
        "id": "3",
        "name": "Kim",
        "address": "3621 Prudence Street",
```

VI. CONCLUSION

The main conclusion of this paper is that it shows how to use Java to implement backend services Spring JPA with H2 Database. We used RDBMS for implementation of database and mapping of database with the entity.

BIOGRAPHY

My name is Ritik Singhal, and I live in New Delhi. I am a B.Tech student at Dronacharya College of Engineering, and I am currently working as an application developer. I am currently assigned in the field of spring boot development, which piqued my interest in how spring boot evolved past the spring framework, which led me to this research paper.