INTERNATIONAL JOURNAL OF R&D IN ENGINEERING, SCIENCE AND MANAGEMENT
Vol.10, Issue 1, 2022, pp.5-10, ISSN 2393-865X

MVC MODEL: IMPLEMENTATION OF MODERN APPLICATION IN JAVA FRAMEWORK

Mudit Rajput[1], Satwik Bhasin[2]
[1,2]Department of Information Technology, Dr. Akhilesh Das Institute of technology and Management, Shastri Park, Delhi

Abstract

A Web application is a distributed application that runs on multiple computers and communicates with other computers over a network or server. Model, view and controller (MVC) is a well-known three-layer development architecture process used for web-based applications developments. Web programmers today face the challenge of working with ever-changing technologies and selecting the development technology to use. Model-View-Controller (MVC) could bring on the table be a very good solution for solving the problems of defining the user interface logic from the business logic that the developers found in this document. Web Applications serve the people by making their tasks easier. This research paper gives some details related to the MVC layers, advantages, disadvantages. We have stated the three layers of MVC in detail and their functions and features. The main objective of this study is to give overview on all the layer of the MVC and main functionalities using the Java application development and framework.

1. INTRODUCTION

Model view controller (MVC) is an architectural pattern used in web-based applications. It provides only 3 main layers; model, view, and controller. Most of the developers use MVC as a basic web-design pattern. It is a complete framework. Most of the languages like Java, PHP, Python, C#, etc. use this pattern to develop the applications. In Java, it is called and known as the Spring-MVC framework. MVC provide three types of classes:

1.1 Model- Model is an important layer of MVC application. It manages the information in the form of data which is can be used to represent the output with the help of views. It depicts the database records. It manages the user application data. It basically contains the application data, basic logic definition, function specification, business rules involvement. A model possibly is a single object or it is some composition of objects. This layer can manage the data and which will also allow the database communication i.e., insert, retrieve and update data in the database.

The domain model in application must follow certain rules:

- Contain the new domain data.
- Contain the basic logic for creating, modifying the domain data.
- Provides API that exposes the model data and operations on it.

1.2 View -Views are used to create the user-interface of our application. While using interface, users interact with the web pages. View shows the results of data that is contained and kept in model. A view has the responsibility of displaying all data of the model. It only shows the required features and attributes and hides the unnecessary attributes. It thus provides us the advantage of presentation. It is script-based like JSP, ASP and PHP.

The following rules should be followed by a view:

- Contain logic and markup required to present data to the user.

1.3 Controller- Controller layer is used to respond to the user's requests. Controller perform the users' requested actions. A controller is the connection between the user and the system. Controller handles both Model and View. It controls on how the data flow in model and always updates the view as soon as the data get altered. It is the separation wall between the Model and the View. The controller receives the data; it works on the data and from there it carries out the execution which alters the state of the data model. These component works with model layer and select the proper view that has been displayed to the user according to user requests.

Rules followed by controller are:

- Contain logic required to initialize the model data.
- Contain logic required by the view to present the data from the model.
- Contain logic required to change the model based on user interactions.

MVC pattern architecture is a three-layered architecture. It separates the characteristics of application. Its 1st layer is related to input logic, 2nd layer is related to the business logic and 3rd layer used to implement user interface logic. MVC offers extremely loose binding among these three layers. MVC pattern are used to define the location and logic in application. MVC patterns provide the facility of single and parallel development. It means that every layer of the application is not dependent on each other i.e., three developers will work on the one layer of application. One developer will be working on user input logic (controller logic), other developers will be working on the user interface logic (view) and third developer will be working on business logic (model) at the same time.

II. TIME TO USE MVC PATTERNS

MVC pattern architecture helps us to follow the concept of separation of concern, it helps to implement and execute the codes individually in the model, views and controller layer in the applications.

MVC makes it simple and reliable to test our application as relation among different parts of application is clearer, crisp and coherent.

MVC benefits us to implement the test-driven development approach, in which we implement automated test cases before we write the actual code. These unit test cases help us predefine the code and verify requirements of new code before writing it. If we are developing a web project with enough serious stimulating on the user side to go along with JavaScript alone. The some of the characteristics that will guide us if we need to use MVC architecture in our application or not:

- An asynchronous communication is always needed.
- The functionality of our web-application is not to retreated a full page for example viewing products in an online ecommerce platform.
- Manipulation of the data is mostly on the client side rather than server side.
- Same type of data is being delivered in different ways on a single page.
- When our application has many insignificant connections that are used to modify data

III. ADVANTAGES AND DISADVANTAGES OF MVC ARCHITECTURE

Advantages:

- MVC architecture helps us to control the complexness of application by dividing it into 3 parts i.e., model, view and controller.
- MVC does not use server-based forms, that's why it is perfect for those developers who want full control on their application behavior.
- MVC use front controller pattern. Front controller pattern handles all the multiple incoming requests using controller. Front controller provides decentralized control. We need to configure only one controller in web server instead of too many.
- Front controller provides support in communications to create our webbased application.

Disadvantages:

- Understanding flow of application is very hard.
- It is little bit complicated to understand and implement which make it unsuitable for small level applications.
- It is hard to understand MVC architecture.
- Its deployment is little bit hard.
- Should have strict rules on methods.
- There is much use of jQuery, AJAX, JavaScript making the programs complicated.
- MVC is difficult to learn, should have a deep knowledge of pattern.
- There is no built-in user control in web pages in MVC.

IV. STEPS TO IMPLEMENT THE MVC MODEL USING THE JAVA FRAMEWORK

```
public class LoginController {

  @FXML
```

```
    public Button loginB;
    @FXML
    public CheckBox showPwd;
    @FXML
    public TextField visiblePwd;
    @FXML
    public Button newUserB;
    @FXML
    public TextField uid;
    @FXML
    public PasswordField pwd;
    @FXML
    private final HomeController home = new HomeController();
    @FXML
    private final NewUserController newUser = new NewUserController();
    @FXML
    private final Utilities u = new Utilities();

    @FXML
    public void loginSystem() throws Exception {
        int i = 0;
       int flag = 0;

       ResultSet rs = Utilities.connection.createStatement().executeQuery("SELECT *
FROM USERS");
       while (rs.next()) {
          u.users.add(rs.getString(2));
```

```
        u.passwords.add(rs.getString(3));           if (uid.getText().equals(u.users.get(i))) {
            System.out.println("User found @ row" + (i + 1));
            Utilities.Name = rs.getString(1);
            Utilities.userID = rs.getInt(2);
            System.out.println("User Name is: " + Utilities.Name);
            if (pwd.getText().equals(u.passwords.get(i))) {                 flag = 1;
                Utilities.stage1.close();                 home.homePage();
                System.out.println("Login Successful!");

                break;                 } else {                 System.out.println("Incorrect Password!");
System.out.println("Login Unsuccessful!");                 break;                 }
        } else {
            System.out.println("User not found @ row" + (i + 1));                 flag = 0;
        }           i++;        }        rs.close();

    if (flag == 0) {
        Alert alert = new Alert(Alert.AlertType.ERROR);                 alert.setHeaderText("Login
Unsuccessful");           alert.setContentText("Incorrect User ID or Password");           alert.show();

    }
} //Takes the input from user and compares it with the database for logging the user in.

@FXML
public void showPwd() {        if (showPwd.isSelected()) {
        visiblePwd.textProperty().bindBidirectional(pwd.textProperty());
visiblePwd.toFront();
    } else {
        pwd.textProperty().bindBidirectional(visiblePwd.textProperty());           pwd.toFront();
    }
} //Shows User Password
```

```
@FXML
public void newUserPage() throws Exception {        newUser.newUserPage();
} }
```

V. LIMITATIONS OF OUR STUDY

In our research paper, we have discussed about MVC components and talked about its three major elements model, view and controller and also explained about communication of these components with each other. We didn't give enough detail and process about these elements individually. There are several other frameworks of MVC which can be used for the development of web-based application. We have not provided a proper code which are used in JAVA MVC and not provided a detailed and complete step to create application in JAVA MVC.

VI. CONCLUSION

The scalable, transparent, reliable and portable application can be created with the help of MVC architecture. MVC architecture help us to know the concept of single and parallel development as it divides the logic of application into three layers, so each different developer can work simultaneously on these three layers of the same application. The cost of setting up the MVC architecture are generally high. The processing time for small MVC project and the non-MVC projects are generally the same but, large project build in MVC architecture will be processed faster than nonMVC project.

REFERENCES

[1] Gupta P, Govil MC (2010) MVC Design pattern for the multi framework distributed applications using XML, spring and struts framework.
[2] Pro MVC 5 book by Adam freeman.
[3] www.interserver.net.
[4] www.tutorialspoint.com.
[5] www.tutorialteacher.com