IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 6, June 2024

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.423

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306239|

Modern Web Development: The Role and Evolution of MVC Architecture

Abhishek Amit Damle, Dr. Shiv Kumar Goel

Department of MCA, Vivekanand Education Society's Institute of Technology, Mumbai, India

Project Guide, Department of MCA, Vivekanand Education Society's Institute of Technology, Mumbai, India

ABSTRACT: The function and importance of MVC (Model-View-Controller) architecture in modern web development are examined in this study paper. The study looks at how MVC continues to influence contemporary online applications, highlighting its progression from desktop programs to web frameworks. The article examines the advantages, difficulties, and real-world uses of MVC in the present development environment, with a focus on contemporary trends and methodologies. The study sheds light on the adaptability of MVC and its value in creating scalable and maintainable online applications by examining important elements such state management, cloud integration, and frontend and backend frameworks. This article provides developers and researchers negotiating the complexity of contemporary web development with useful insights through a thorough assessment of the literature and real-world experiences.

I. INTRODUCTION

One architectural pattern that is frequently utilized in web-based applications is Model-View-Controller, or MVC. This popular design pattern offers three primary levels: controller, view, and model. Three distinct class types are used by MVC to organize the application into a tidy framework:
Model: Uses the database to extract, add, or change data in order to carry out data domain logic.
View: Sets up the application's user interface so that people may interact with it.
Controller: Acts in response to user requests by carrying out tasks, choosing the suitable view based on user input, and starting tasks.
The application characteristics are divided into three tiers by the MVC architecture. User interface logic is implemented at the third layer, while business logic is managed by the second tier and user input logic by the first. MVC defines the locations of each logic component inside the application and allows for loose communication between these layers.

II. CURRENT STATE OF MVC ARCHITECTURE

One of the most important architectural patterns for creating web-based applications is still Model-View-Controller (MVC). Its capacity to divide an application into three interdependent components (Controller, View, and Model) keeps it a strong foundation for code organization and maintainability enhancement. This is a thorough examination of MVC architecture's present condition, which reflects its acceptance, development, and contemporary use in web development.

A. Modern Adaptations and Evolution:

1. **Conventional MVC vs Contemporary Implementations:**

Conventional MVC: Originally intended for desktop programs, MVC's concepts were later applied to web development as a way to control the complexity of online applications. MVC is closely adhered to by traditional web frameworks such as ASP.NET MVC, Ruby on Rails, and Django.

Modern MVC: Single Page Applications (SPAs) built using frameworks like Angular, React, and Vue.js are a common feature of modern web development. In order to better manage state and components, these frameworks modify the MVC pattern and frequently incorporate it into more intricate designs like MVVM (Model-View-ViewModel) or Flux architecture.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306239|

2. **Architecture Based on Components:**
   The display and controller functionality are contained within components that may nevertheless interface with a centralized model or state management system, thanks to the widespread usage of component-based architecture in modern front-end frameworks.

B. Advantages of MVC in Contemporary Development:

1. **Differing Issues:** MVC makes code easier to maintain and test by breaking the program up into discrete pieces that allow developers to concentrate on one area at a time.
2. **Modularity and Reusability**: The program's components might be utilized again in other projects or even in various sections of the application. Additionally, because of its modularity, many teams may work on the controller, view, and model at the same time in parallel development.
3. **Sustainability:** The code is easier to maintain and update when there is a clear separation. The user interface (view) is unaffected by changes made to the business logic (model), and vice versa.

C. Cons and Difficulties Complexity in Wide-Scale Applications:

1. **Complexity in Large Applications**: Strict MVC maintenance may get tiresome for really big systems, and other patterns and practices may be needed to handle complexity.
2. **Learning Curve:** It can be difficult for novices to comprehend and apply MVC appropriately, particularly when combining it with contemporary JavaScript frameworks that introduce additional layers of abstraction.

D. Current MVC Rendering Trends:

1. **Server-Sidevs.Client-Side:**
   Django and Rails, two classic MVC frameworks, concentrate on server-side rendering. But more and more contemporary web apps are depending on client-side rendering thanks to frameworks like React and Vue.js, which manage a large portion of the display and controller functionality within the browser.
2. **Whole-Stack Creation:** Full-stack development frameworks (e.g., Next.js for React, Nuxt.js for Vue.js) that offer end-to-end solutions combining both MVC concepts with contemporary JavaScript capabilities are becoming more and more popular.
3. **Connectivity to Cloud Services Integration:** In order to provide hosting, storage, and other features, modern MVC applications frequently make use of cloud services

III. PROPOSED SOLUTIONS

A. When to Use MVC Pattern Architecture?
An application's Model-View-Controller (MVC) architecture is composed of these three elements. This division of labour guarantees that every part performs a specific task: the Model handles data and business logic; the View shows information to the user; and the Controller receives user input and modifies the Model and View accordingly. This separation of issues facilitates testing because each component may be assessed separately. It also makes test-driven development (TDD) easier by allowing automated tests to be built for each component before the relevant code is written. This ensures that the program functions as planned from the beginning. The overall quality, scalability, and maintainability of software development projects are enhanced by the structured MVC technique.

a. Asynchronous Communication on the Backend:
MVC can handle asynchronous interactions well when the application has to handle them, such retrieving data from a server without requiring a page reload. When necessary, the controller can update the model and refresh the view asynchronously in response to requests and responses.

b. Preventing Full-Page Reloads:
Modern online apps often demand dynamic updates that don't involve a page refresh. The MVC architecture makes this feasible by allowing controllers to govern user actions (such filling out a form or clicking a button) and update views only when required, all without requiring a page refresh.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306239|

C. Client-Side Data Manipulation:
When most data manipulation takes place on the client side, MVC is still applicable (for example, via JavaScript frameworks like React or Angular). The model can describe the data structure and business logic, while the view presents the user with the data as it is at that moment. Controllers enable updates and communication between the view and the model, which guarantees consistent data processing.

d. Flexible Data Presentation:
When the same data needs to be shown in multiple formats or views on the same page (e.g., displaying user details in a pop-up modal and a profile sidebar), MVC's separation of responsibilities enables developers to manage these varied presentations in applications effectively. Without repeating business logic, the controller manages the retrieval and display of data across several views.

e. Several Data Modification Connections:
Applications requiring several ways to modify data (e.g., through buttons, switches, and forms) benefit from the structured approach of MVC. Controllers monitor these several input methods to make sure that every action updates the model correctly and reflects the changes in the view(s). They accomplish this without linking these exchanges too tightly.

B. MVC FEATURES:
The MVC architecture improves software application development and maintenance in a number of ways. The division of application logic into three discrete jobs facilitates the management and testing of each component independently: input logic, business logic, and interface logic. This division of responsibilities facilitates agile approaches such as test-driven development (TDD), which allows automated tests to be generated for every component prior to deploying the corresponding code. It also streamlines the testing process.

Because MVC frameworks are flexible, developers can use several unit testing frameworks that fit with the MVC framework they have chosen. This makes testing procedures quick and reliable. Furthermore, MVC frameworks are pluggable and adaptable, allowing developers to change or replace view engines, URL routing schemes, and serialization processes according to project requirements.

MVC designs allow a crucial technique called Dependency Injection (DI), which injects dependencies into classes instead of depending on class instantiation to promote loose coupling between components.

MVC frameworks offer URL mapping components that let developers design user-friendly, search engine-friendly URLs that improve both SEO optimization and usability. Because it guarantees distinct and meaningful URL patterns for resource addressing, this is especially advantageous for applications that need RESTful APIs.

Web application security, session management, and form authentication are just a few of the built-in features that frameworks like ASP.NET MVC offer. These features improve application security and ease typical development tasks. Additionally, they facilitate object-relational mapping (ORM), which streamlines database interactions and encourages effective data management techniques in applications.

C. ADVANTAGES AND DISADVANTAGES OF MVC ARCHITECTURE:
Advantages:
a. Control Complexity: An MVC application consists of three parts: the model, the view, and the controller. This makes it possible to control different aspects independently, including data handling, user interface, and application logic. Developers can maintain clarity and manage complexity by assigning specific responsibilities to each component, so creating a separation of tasks.

b. No Server-based Forms: MVC frameworks usually do not use server-based forms to regulate an application's behaviour. Instead, they emphasize the usage of AJAX (Asynchronous JavaScript and XML) for client-side interactions in order to enhance user experience and provide developers more control over how interactions are handled.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306239|

c. Support for Test-Driven Development (TDD): The MVC architecture facilitates Test-Driven Development (TDD) by allowing developers to write independent unit tests for the Controller, View, and Model components. This strategy ensures that functionality is validated early in the development cycle and promotes more reliable and high-quality code.

d. Front Controller Pattern: MVC employs the Front Controller pattern, which consists of a single controller handling all incoming requests to the application. This centralised control simplifies routing and request processing, reducing the frequency with which several controllers on the web server need to be configured. It improves maintainability and provides a uniform entry point for requests by merging request-handling logic.

e. Rich Routing Communications: The Front Controller in MVC frameworks often has robust routing capabilities. This enables developers to create clear and pertinent URL routes that match certain application resources or actions. Rich routing features improve the overall usability and accessibility of online applications while simplifying the organization of code.

Disadvantages:

a) **Understanding Flow:** Model-View-Controller (MVC) is divided into three separate components, and the way these components interact makes it difficult to understand at first. To properly create and manage apps, developers must comprehend the data flows between these elements.
b) **Complexity for Small Applications:** In smaller-scale applications where the delegation of responsibilities may result in overhead, MVC may appear unduly complex.
c) **Difficulty in Understanding:** Mastering MVC necessitates knowing its guiding principles and how to use them successfully. It may take some time for developers who are unfamiliar with architectural patterns to grasp this concept.
d) **Complexity of Deployment:** Compared to simpler architectures, MVC application deployment may need more configuration. Keeping concerns apart and making sure every part works properly in a production setting might lead to this complexity.
e) **Strict Guidelines for Methods:** MVC frameworks frequently enforce guidelines and standards for the organization and use of methods inside each component. While following these guidelines can be burdensome, doing so guarantees uniformity and maintainability.
f) **Use of jQuery, AJAX, and JavaScript:** For dynamic interactions and improved user experiences, MVC applications frequently make use of front-end technologies like jQuery, AJAX, and JavaScript. These technologies are strong, but they can also be complicated, particularly when handling client-side scripting and asynchronous activities.

D. DESCRIBING THE ARCHITECTURE

We have to decide which architecture type best meets our needs before we start the application development process. Examining our application's functional components and determining how they will communicate with one another is crucial. There are numerous design patterns available for developing applications. In this case, we'll use the MVC architecture pattern. The Model, View, and Controller are the three primary parts of our software that are separated based on the MVC paradigm. The model is how we communicate with our database. To extract data from the model and show it to the user, use view. Lastly, the controller establishes the application's behaviour. It responds to user interaction in some way.

Users usually use HTTP GET and HTTP POST to enter data into online applications.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306239|

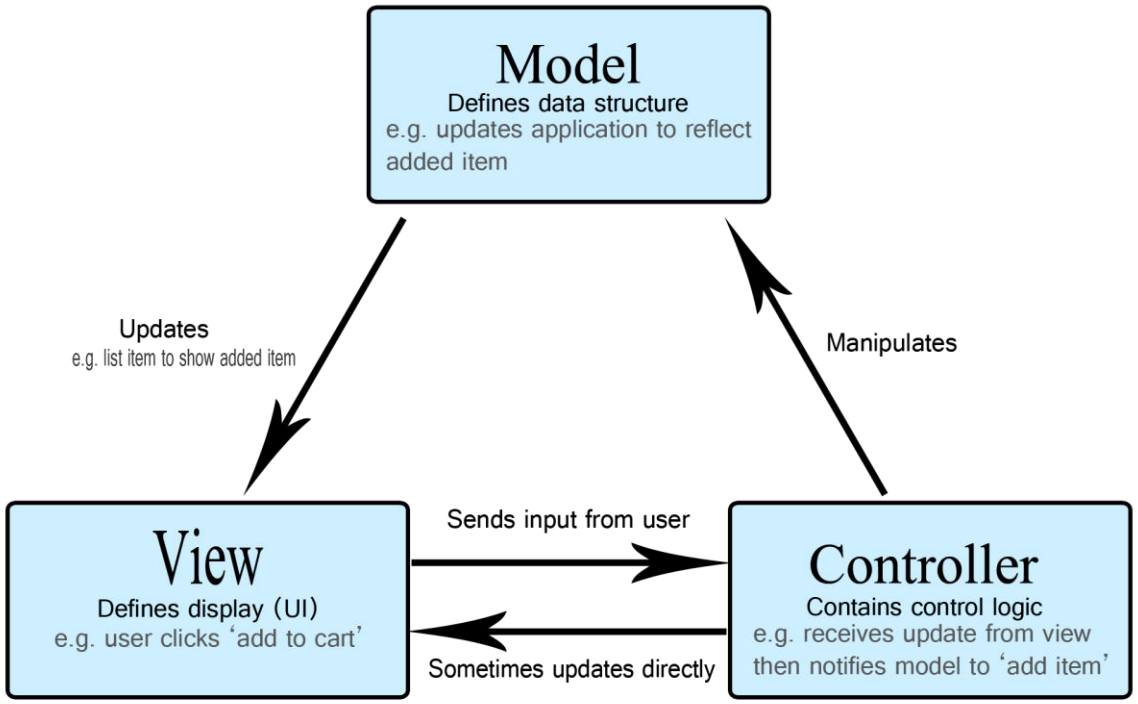

FIGURE 1 : MVC ARCHITECTURE

IV. FUTURE RESEARCH

The historical evolution of the Model-View-Controller (MVC) architecture is the first important topic to investigate in order to have a thorough grasp of the architecture and its present form. Originally intended for desktop software, MVC made the switch to web development and emerged as a key application complexity management tool. Conventional MVC frameworks like ASP.NET MVC, Ruby on Rails, and Django strictly follow the pattern's guidelines, but more recent adaptations—like Angular, React, and Vue.js—often combine MVC with other patterns like MVVM (Model-View-ViewModel) and Flux architecture.

Examining the advantages and disadvantages of Model-View-Controller (MVC) indicates that its main value is the distinct separation of concerns, which makes programs easier to maintain and scale. These components include model, view, and controller. This division makes maintenance easier and increases code modularity and reusability. Large applications, where rigorous adherence to MVC can become burdensome, and novices, who can find the learning curve high, present issues, though.

Modern MVC programming makes use of several tools and technologies. While frontend development frequently makes use of Angular, React, and Vue.js in conjunction with state management libraries like Redux or Vuex, backend frameworks like ASP.NET Core, Ruby on Rails, Django, and Laravel continue to be popular. Web application scalability and distribution are further improved by combining MVC with RESTful APIs and microservices architectures.

One of the current trends in MVC is the transition, fueled by the popularity of Single Page Applications (SPAs), from server-side to client-side rendering. MVC concepts are combined with contemporary JavaScript capabilities by full-

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306239|

stack development frameworks like as Next.js for React and Nuxt.js for Vue.js, providing end-to-end solutions. Furthermore, as MVC apps rely on AWS, Azure, and Google Cloud services for functionality and hosting, cloud integration has grown in importance.

Examining case studies and real-world instances of effective MVC implementations from a variety of sectors may be quite insightful for practical applications. These examples provide best practices for implementing MVC in web applications, both small and big.

A number of resources are suggested to aid with your investigation. Comprehensive information is provided by foundational publications like "Pro ASP.NET Core MVC" by Adam Freeman and "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma et al. Furthermore, scholarly publications, case studies, official documentation for well-known frameworks, and online courses from sites like Coursera, Udemy, and Pluralsight are priceless resources. A comprehensive grasp of the MVC architecture's current state and applications in contemporary web development can be obtained by utilizing a research methodology that includes literature review, comparative analysis, industry professional interviews, and hands-on practice with various MVC frameworks.

V. CONCLUSION

The MVC pattern design framework and integrated technologies such as JSP, Servlet, and EJB on the J2EE platform have made web application development easier. MVC architecture may be used to create online applications that are transparent, scalable, and portable. The MVC design, which separates the application logic into three tiers that any developer may work on concurrently on the same web application, makes the idea of parallel development practical. The model layer of the MVC architecture is where communication with the database takes place.

The logic layer of our program is mainly responsible for inserting, retrieving, and updating data from the database. The user interface, or how users interact with our application, is the focus of the second layer, known as view. it is shown to them—and is primarily utilized in front end development. The controller, the third layer of the MVC architecture, is where user input is obtained. It manages how the user's inputs change the model object state and the data that is displayed to them. Through the controller layer, the user can submit requests to our program and receive response.

REFERENCES

[1] MVC Architecture: A Detailed Insight to the Modern Web Applications Development Abdul Majeed1 and Ibtisam Rauf2 September 08, 2018; Published: September 26, 2018 Crimson Publishers
[2] International Research Journal of Engineering and Technology (IRJET) Mr. Siddharth Singh Volume: 07 Issue: 01 | Jan 2020 e-ISSN: 2395-0056 p-ISSN: 2395-0072

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details