Web Application Development using MVC Framework

ROHITH.H
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING & INFORMATION SCIENCE
PRESIDENCY UNIVERSITY
rohith18102001@gmail.com

I. ***Abstract***— *This article focuses on the significance of utilizing MVC architecture in conjunction with the .NET Framework for web application development. It explores various components, including routing, controllers, views, data models, and validation strategies, while emphasizing best practices. Additionally, it discusses the integration of the .NET Framework with popular databases, with a specific focus on SQL Server interaction. This integration ensures data integrity, efficient storage, recovery, and administration. By leveraging MVC and the .NET framework, developers can create scalable and robust web applications. The objective of this article is to equip developers with the knowledge and understanding needed to leverage MVC and the .NET framework effectively, enabling them to build versatile and user-centric web applications.*
Keywords: MVC Architecture, .NET Framework, Web Application Development, Routing, Controller, Views, Data Model, Integration, SQL Server, Data Integrity, Scalability, Robustness.

II. INTRODUCTION

Developed by Microsoft, the.NET Framework is a software development framework that offers a runtime environment, libraries, and tools for creating and running programs on Windows operating systems. The framework supports a range of application types, including desktop, web, mobile, and gaming apps, and several programming languages, including C#, F#, and Visual Basic. 1. The Common Language Runtime (CLR) and the.NET Framework Class Library are the two main parts of the.NET Framework. The class library offers a substantial collection of pre-built functions and classes that may be used to develop a wide range of applications, while the CLR is in charge of controlling the execution of code written in any of the supported languages.

At the core of the .NET Framework are two essential components: the Common Language Runtime (CLR) and the .NET Framework Class Library. The CLR serves as the execution engine, providing essential services like memory management, security, and exception handling. It acts as an intermediary between the

Application code and the operating system, ensuring that programs written in different languages can run seamlessly on the Windows platform.

III. EXISTING WORK

On MVC and.NET 1, there is a plethora of prior studies.

1." Pro ASP.NET MVC Framework" by Steven Sanderson:

This book offers a comprehensive analysis of ASP.NET MVC, covering all facets of the MVC design, such as controllers, views, models, and data entry. Additionally, it explores complex subjects like security, caching, and testing.

2. "ASP.NET MVC in Action" by Ben Scheirman, Jimmy Bogard, and Jeffrey Palermo: To create ASP.NET MVC applications, this book offers useful instructions. Control, visualization, modeling, routing, and unit testing are some of the subjects it covers. Additionally, it offers practical illustrations and advice on how to create maintainable, tested programs.

3. Philip Japikse, Kevin Grossnicklaus, and Ben's "Building Web Applications with Visual Studio 2017: Using.NET Core and Modern JavaScript Frameworks".

IV. PROBLEM AND MOTIVATION.

I. The complexity of web applications has increased as they have developed into more sophisticated systems.
Significantly, modern web programs connect multiple types of components in a variety of ways, making it challenging to comprehend their architectural formalism. The engineering procedures of web-based systems are negatively impacted by this environment. Numerous concepts and artifacts are accompanied by extra learning curves, as well as their benefits and drawbacks, to address the linked problems and enhance web engineering. However, the Client-Server (C-S) model continues to serve as the cornerstone of increasingly sophisticated web applications.

V. METHODOLOGY

To gather expertise in the fields of web applications, software architecture and architectural styles, conceptual abstraction, and TTS, a literature review was done. Second, methodology The domains of web applications, software architecture and architectural styles, conceptual abstraction and TTS independence, MVC, and the TTS used to construct MVC-based online applications were the focus of a literature review. The absence of literature on TTS independence and concept abstraction was highlighted. Therefore, some tests were carried out to learn about the use of available TTS in MVC-based web construction and to gather empirical evidence. The experiments were incrementally carried out and prototype-based. Early iterations examined facts from the literature to discover bottlenecks and problems, while later rounds tested remedies to the problems that were found.

1. MVC modules

FRAMEWORK: MVC
MODEL, VIEW, CONTROLLER (MVC) IS AN ARCHITECTURAL/DESIGN FRAMEWORK THAT

MODEL, VIEW, AND CONTROLLER ARE THE THREE BASIC LOGICAL COMPONENTS THAT MAKE UP THIS ARCHITECTURE FOR DIVIDING APPLICATIONS. EACH ARCHITECTURAL ELEMENT IS DESIGNED TO MANAGE PARTICULAR APPLICATION DEVELOPMENT FACETS. IT SEPARATES THE PRESENTATION LAYER FROM THE BUSINESS LOGIC LAYER. ONE OF THE MOST POPULAR AND WIDELY ACCEPTED WEB DEVELOPMENT FRAMEWORKS FOR EXPANDABLE APPLICATIONS IS MVC. THE MVC FRAMEWORK'S PRIMARY BENEFIT IS THAT IT OFFERS A CLEAR SEPARATION OF CONCERNS BETWEEN THE VARIOUS PARTS OF THE APPLICATION. AS A RESULT, IT IS SIMPLER TO DESIGN, TEST, AND MAINTAIN THE PROGRAM BECAUSE EACH COMPONENT MAY BE DEVELOPED INDEPENDENTLY OF THE OTHERS. IT ALSO INCREASES THE APPLICATION'S EXTENSIBLE BECAUSE NEW FEATURES CAN BE ADDED WITHOUT AFFECTING THE EXISTING CODE.

The MVC framework includes the following 3 components:
1) Controller
2) Model
3) View

1) Controller*:*

The element that acts as the controller
provides the coordination or link between the Model and the View. The controller is in charge.
for managing user requests and directing communication between the application's Model and View components. When a user submits a request to the application, the controller receives it and decides what action needs to be taken depending on the parameters of the request and other contextual data.

2) Model:

The application's data and business logic are managed by the Model component. It could stand in for the data being transmitted between the View and Controller components or for any other data connected to the application's business logic. Interacting with the database or other data sources to retrieve and alter data is one of the main responsibilities of the Model component. This entails running queries, changes, and other database operations to get the data you want or

need when you need it. The Model component may also carry out data validation and other tasks to guarantee that the data is accurate and consistent.

There are 4 components of the Model in MVC :

I) Entity Model
II) View Model
III) View Data
V) Repository

I. *Entity Model:*

It includes a description of every class object for every instance made for the specified screen. The database tables are also mapped. By offering a high-level abstraction of the data, it is utilized to make accessing and modifying data in the data storage simpler.

II) *View Model:*

Between the Model and the View components, there is a ViewModel. Its goal is to keep the Model separate from the View while yet providing a condensed, domain-specific representation of the data that the View must display. In other words, the ViewModel is a specialized class that provides a mechanism for
the View to access the data without directly knowing about the Model and encapsulates the data that the View needs to present.

III*) View Data:*

The Controller component can send data to the View component via View Data. It gives the Controller a way to communicate with the View without directly exposing the Model to the View. Any type of data, including strings, numbers, objects, and collections, can be passed from the Controller to the View using the View Data method.

IV) *Repository:*

A design pattern called repository creates an abstraction layer between the data access layer and the remaining portions of the application. In online applications, the Repository pattern is frequently used to make it easier to access and manage data from databases and other sources of data. The Repository pattern's fundamental notion

is to consolidate the logic for data access into a single class or collection of classes. As a result, the complexity of data access is abstracted from the rest of the program, allowing it to function with a consistent and clear API.

3) View.

The View component is in charge of showing data to the user and gathering feedback from them. The user interface that the user interacts with is rendered through views. The Model and Controller components manage the business logic, which is handled by the View component. Views typically display data collected from the Model or user input recorded by the Controller instead of performing any data processing or manipulation. Depending on the requirements of the application, views can take many different forms. They can be simple web pages rendered in HTML, CSS, and JavaScript by a web server, windows from desktop programs, screens from mobile programs, or any other interface used to interact with the user.

VI. Contribution to the real world

Web application development has greatly benefited from the Model-View-Controller (MVC) architectural pattern's integration with the robust.NET Framework. Here, we list some of the essential benefits that MVC and the.NET Framework provide to programmers and companies:

1**. *Enhanced Development Productivity*:** MVC streamlines the development process and boosts productivity when combined with the extensive features of the .NET Framework.
MVC's separation of concerns encourages code modularity and reuse, allowing programmers to focus on particular components on their own. Developers may concentrate on application-specific logic and accelerate development thanks to the huge selection of pre-built components and libraries in the.NET

Framework, which removes the need to reinvent the wheel.

2. ***Scalable and maintainable applications****:* The MVC pattern makes it easier to create scalable and maintainable web applications when combined with the capabilities of the.NET Framework. Different parts of the program, such as data administration, business logic, and user interface, can be built, tested, and maintained independently thanks to the obvious separation of concerns. Because of this division, it is simpler to modify and expand the program as the company needs to change over time.

3.. ***IMPROVED USER EXPERIENCE:*** MVC AND THE.NET FRAMEWORK WORK TOGETHER TO MAKE IT POSSIBLE TO CREATE USER INTERFACES THAT ARE INTERESTING AND INTERACTIVE. THE ASP.NET MVC FRAMEWORK OFFERS A COMPREHENSIVE SET OF CAPABILITIES FOR DEVELOPING DYNAMIC UI COMPONENTS, PROCESSING USER INPUT, AND GENERATING VIEWS. WITH TOOLS LIKE CLIENT-SIDE LIBRARIES, AJAX SUPPORT, AND RESPONSIVE DESIGN CAPABILITIES, PROGRAMMERS CAN MAKE USER INTERFACES THAT ARE FLUID AND RESPONSIVE AND CAN ADJUST TO VARIOUS DEVICES AND SCREEN SIZES.

4. ***Security and Authentication*****:** The.NET Framework and MVC both provide strong security capabilities. Security is a crucial component of web applications. Built-in techniques for authentication, authorization, and defense against typical security threats are included in the.NET Framework.
6. The deployment of secure coding practices, input validation, and data sensitization strategies is made easier by MVC's separation of concerns, preserving the integrity and security of user data.

CONCLUSION:

In conclusion, two significant technologies that are revolutionizing Web development are MVC and. NET While MVC design offers a mechanism for creating apps,.NET offers developers a solid framework to create strong and adaptable applications. By offering more tools, flexibility, and quality work, current initiatives like ASP.NET Core, Entity Framework Core, and.NET 5 continue to enhance the web development process. To create fresh and effective web apps as technology develops, developers must stay current with the most recent MVC and .NET features.

REFERENCES

[1] https://www.geeksforgeeks.org/mvc-framework-introduction/
[2] https://www.javatpoint.com/net-framework.
[3] https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm
[4] https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet-framework
[5] https://learn.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022