FEST ORGANIZING SYSTEM

(FESTFUSION)

A Project Report

Submitted in partial fulfillment of
the requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Rahul Yadav (67)
Rohan Kangane (13)

Under the esteemed guidance of

Prof. Shraddha Parab

DEPARTMENT OF INFORMATION TECHNOLOGY

TOLANI COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MUMBAI, 400093
MAHARASHTA
2023 - 2024

TOLANI COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400093

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, " **Fest Organizing System**", is bonafied work of **Rahul Yadav** bearing roll no: **67** submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATIONTECHNOLOGY from University of Mumbai.

Internal Guide **Coordinator**

External Examiner

Date: **College Seal**

ABSTRACT

The Fest Fusion system is a comprehensive web application designed for organizing college fests. It caters to two key user roles: users and admin. Users can sign up and log in to the website, providing necessary details during registration and subsequently using their username and password for login. Upon accessing the dashboard, users can explore different college events and participate by filling out event-specific forms. After submitting these forms, users must await validation by the admin before receiving their event passes.

The system offers a seamless user experience, allowing students to easily sign up and log in. Users can browse and participate in college events through a user-friendly interface. Event forms capture essential participation details from users. Admins have a separate login system, requiring college admin login and password. Admins can create and manage events, access options like trophies, judges, sponsors, and merchandise. The admin can customize event pages, providing information of event types, fest details, sponsors and many more features. A dedicated user details form is available for admin use.

The Fest Fusion system simplifies college fest organization by providing a user-friendly platform for students to participate. Users sign up with their credentials and explore events, submitting participation forms. Admins manage events and access various features like trophies, judges, sponsors, etc. They also create customized event pages with event details and sponsor information. Users' participation is validated by admins, and event passes are issued once approved. This system streamlines fest management, ensuring efficient communication between participants and organizers, enhancing the overall fest experience.

ACKNOWLEDGEMENT

It takes great pleasure to me to present project report on **“Fest Organizing System”**.

I would like express my deepest thanks and gratitude to my project guide, **Prof. Shraddha Parab**, for her guidance and help during each step of the project.

I would like to express my thanks and gratitude to my parents and their utmost support during the academic year so that I can focus properly on my project. With proper coordination and full fledge cooperation among me and my guide, I was able to complete this project successfully.

I would like to express my gratitude and thanks to Co-Ordinator **Prof. Bandita Singh** and all the teaching staff of I.T Department for their utmost help and support in the project.

Finally, I wish to express profound thanks to all those who helped me in any way regarding the project.

Rahul Yadav

DECLARATION

I hereby declare that the project entitled, "**Fest Organizing System"**, done at Tolani College of Commerce, has not been in any case duplicated to submit to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum**.**

Signature of Student

Rahul Yadav

TABLE OF CONTENTS

INTRODUCTION

1.1 Introduction: -

In today's fast-paced and dynamic educational landscape, college fests have become a vibrant platform for students to showcase their talents, creativity, and competitive spirit. Organizing a successful college fest, however, involves intricate planning, meticulous coordination, and effective communication between organizers and participants. To address these challenges and optimize the fest management process, we introduce the Fest Organizing System, a cutting-edge web application designed to streamline fest registration and provide comprehensive information about events and activities.

The Fest Organizing System serves as a digital gateway for students from various colleges to participate in the fest of their choice. With its user-friendly interface, students can effortlessly register and gain access to a wealth of fest-related details, including event schedules, descriptions, rules, and participating teams or individuals. By centralizing all fest-related information in one place, the system aims to enhance engagement and create a seamless experience for both organizers and participants. Additionally, we will explore the impact of the Fest Organizing System on fostering efficient fest coordination, maximizing student engagement, and delivering an unforgettable fest experience.

The significance of the Fest Organizing System extends beyond mere practicality. Beyond utility, we are committed to uncovering the broader impact of this innovative tool on the festival landscape. Our aspiration is to measure how it enhances the efficiency of festival coordination, maximizes student engagement, and weaves the tapestry of an unforgettable festival experience. In this spirit of continual improvement, we pledge to refine the system further, adapting it to the ever-evolving needs of the educational landscape, ensuring that college festivals remain not just vibrant but also transformative sources of inspiration, growth, and community for all participants.

1.2 Background:-

Eventbrite is a widely recognized online event management and registration platform that has transformed the way events are organized and promoted. Founded in 2006 by Kevin Hartz, Renaud Visage, and Julia Hartz, the platform has since gained immense popularity for its user-friendly interface and powerful features.

Eventbrite has empowered event organizers, from small community gatherings to large-scale conferences, by providing them with the tools to create, customize, and manage event registrations effortlessly. Its ability to handle ticketing, promotion, and attendee communication has made it a go-to solution for individuals and organizations looking to streamline event planning and maximize attendance.

Eventbrite impact extends globally, with millions of events hosted and attended in over 170 countries, making it one of the largest event management platforms worldwide. The platform's intuitive event creation tools allow organizers to design custom event pages with branding, ticket types, and pricing options tailored to their specific needs.

Eventbrite seamless integration with social media platforms and email marketing services has revolutionized event promotion, helping organizers reach a wider audience and boost ticket sales.

Eventbrite commitment to innovation has led to the development of features like reserved seating, event access control, and fundraising capabilities, expanding its utility across various event types. The platform's customer support and resources, including webinars and guides, provide valuable assistance to event organizers, ensuring they get the most out of Eventbrite's offerings.

Eventbrite positive impact on the events industry is reflected in the diverse range of events it supports, from local charity fundraisers to international conferences, fostering a dynamic and inclusive event ecosystem. The company's continuous growth and evolution demonstrate its adaptability to changing market trends and the evolving needs of event organizers and attendees alike.

Over the years, Eventbrite has become a central hub for event discovery and registration, connecting event-goers with a diverse range of experiences.

1.3 Objective: -

The objective behind the Fest Organizing System is to simplify and enhance college fest organization. It aims to streamline participant registration, centralize fest information, optimize resource allocation, improve communication, and increase overall participant engagement. By leveraging technology, the system seeks to create a seamless and enjoyable fest experience for both organizers and students while promoting efficient coordination and improving the overall fest organization.

The Fest Organizing System is designed with a clear objective to revolutionize the organization of college fests. At its core, the system seeks to simplify the complex web of tasks associated with event planning. Its primary goal is to streamline the participant registration process, making it effortless for students to sign up for multiple events, reducing administrative burdens, and ensuring a user-friendly experience. By centralizing fest information, the system ensures that participants have easy access to event schedules, descriptions, and rules, thereby eliminating confusion and enhancing their fest experience.

Moreover, the Fest Organizing System is committed to optimizing resource allocation. Through the implementation of an online scheduler, it provides real-time insights into the availability of event venues and equipment, allowing organizers to make informed decisions and allocate resources efficiently. This feature not only enhances the overall fest planning efficiency but also ensures that resources are allocated where they are needed most, contributing to the success of each fest event.

Furthermore, the system places a strong emphasis on communication and engagement. By leveraging technology, it facilitates seamless interactions between organizers and participants. Timely notifications, a user-friendly design, and a feedback loop are integrated to ensure a positive and efficient experience for everyone involved. Ultimately, the Fest Organizing System's overarching aim is to enhance fest coordination, promote engagement, and elevate the overall quality of college fests, making them memorable and enjoyable for all.

1.4 Purpose, Scope and Applicability: -

1.4.1 Purpose: -

The fundamental purpose behind the development and implementation of a fest organizing system is to establish a centralized and highly efficient platform meticulously designed for the planning, coordination, and management of college festivals. At its core, this system endeavors to streamline and simplify the often intricate and multifaceted process of organizing events of such magnitude. It achieves this ambitious goal by providing a suite of indispensable features that encompass user-friendly registration processes, crystal-clear communication channels, and judicious resource allocation mechanisms.

In essence, the fest organizing system serves as the linchpin of event management, a digital sanctuary where the myriad elements and tasks required for a successful festival converge and harmonize. Its primary objective is to alleviate the considerable administrative burdens that often weigh down event organizers, allowing them to channel their energies and focus on the creative and logistical aspects that truly elevate a festival to greatness. By simplifying tasks like participant registration, event scheduling, and resource allocation, this system empowers organizers to work with unparalleled efficiency and precision, ultimately leading to the creation of memorable and seamlessly executed festivals.

However, the benefits of this system are not confined solely to the organizers. It extends its advantages to the heart of any festival—the participants themselves. With its intuitive design and user-friendly registration processes, the fest organizing system ensures that participants can navigate the festival landscape with ease. They gain swift and convenient access to crucial event information, including detailed schedules, event descriptions that vividly capture the essence of each competition, and comprehensive rules and regulations that govern their participation. This accessibility eliminates the frustrating hurdles that participants often encounter when engaging in festival activities, enhancing their overall fest experience.

In summation, the ultimate aim of the fest organizing system is to foster an environment where college festivals can thrive as vibrant and engaging celebrations of talent, creativity, and camaraderie. By reducing the administrative burdens on organizers and providing participants with easy access to event information, it aspires to create an enjoyable and immersive festival environment that resonates positively with all stakeholders. In doing so, this system plays a pivotal role in transforming college festivals into memorable and seamlessly executed experiences for everyone involved.

1.4.2 Scope:-

The scope of the Fest Organizing System is to provide a comprehensive and user-friendly platform for efficient fest planning and management. The system will focus on the following key aspects:

1. **User-Friendly Interface:** The fest organizing system places paramount importance on user-friendliness, striving to create an interface that caters to the needs of all users, be they event organizers or participants.

   This commitment translates into a design that prioritizes intuitive navigation, ensuring that users can effortlessly move through the platform's various features and functionalities. Event organizers, who shoulder the intricate task of orchestrating college festivals, will find the system's interface to be a valuable ally in simplifying their roles. It equips them with tools and options that streamline event logistics management, participant registration, and resource allocation, reducing administrative complexities. This liberates organizers to allocate their time and energy toward the creative and strategic aspects of festival planning, thereby enhancing their overall efficiency. For participants, the user-friendly interface ensures a frictionless experience, particularly during the registration process, sparing them from the frustrations of navigating convoluted systems.

2. **Efficiency and Time-Saving:** Efficiency and time-saving stand as pivotal pillars of the fest organizing system's design, aiming to revolutionize the way college festivals are managed and experienced. Through the implementation of automation, the system takes on the laborious tasks associated with registration, and communication, significantly streamlining these processes. For event organizers, this automation translates into a notable reduction in administrative burdens, liberating them to allocate their time and energy to more creative and strategic aspects of festival planning. The system expedites event management, allowing organizers to swiftly set up competitions and activities without getting bogged down by intricate logistics. Participants, too, reap the benefits of this time-saving approach. They can register effortlessly, bypassing the need to navigate through cumbersome registration procedures.

3. **User Satisfaction:** A fundamental objective of the fest organizing system is to prioritize user satisfaction by aligning its design closely with the specific requirements and needs of users, encompassing both organizers and participants. This user-centric approach ensures that the system functions as an effective tool, tailored to enhance the fest planning and execution stages, ultimately

contributing to a more enjoyable festival experience. By attentively meeting user requirements, the system fosters a sense of engagement and enthusiasm among all stakeholders involved in college fest activities. Organizers benefit from a platform finely tuned to their needs, reducing frustrations and inefficiencies in the planning process, and thereby enhancing their overall satisfaction. Similarly, participants find the fest organizing system to be a user-friendly and efficient means of engaging with festivals, leading to increased satisfaction and a more positive fest experience overall.

4. **Scalability:** The fest organizing system's scalability is a key feature, designed to empower it with the flexibility to expand and evolve, effectively aligning with the changing demands of fest organizers and participants. This crucial capability ensures that the system remains relevant and usable over the long term, capable of accommodating the dynamic landscape of college festivals. Scalability allows the system to gracefully handle an increasing volume of users, events, and data without compromising its performance or usability. As the needs of fest organizers and participants evolve, the system can seamlessly incorporate new features and functionalities, adapting to meet their ever-changing requirements. Ultimately, scalability is the system's insurance policy for longevity, guaranteeing that it remains a valuable and dependable resource for fest planning and execution, regardless of how the festival landscape evolves.

5. **Online Scheduling:** The fest organizing system incorporates a valuable online scheduling feature designed to efficiently manage the availability of essential fest-related resources. This feature serves as a dynamic tool that showcases the real-time availability of key assets, including event venues and equipment, allowing for optimized resource allocation. Organizers can easily access and visualize the schedules of these resources, enabling them to make informed decisions and avoid scheduling conflicts. Ultimately, the online scheduling feature enhances the overall efficiency of fest planning by ensuring that resources are allocated effectively and without disruptions, contributing to the seamless execution of college festivals.

6. **Data Security:** Data security stands as a paramount concern within the fest organizing system, signifying its unwavering commitment to safeguarding sensitive participant information and payment details. The system employs robust security measures, including encryption protocols and access controls, to protect against unauthorized access or potential breaches, ensuring the privacy of user data. Participant information, which may include personal details and financial data, is shielded within the system's fortified digital fortress, assuring users of their data's confidentiality. In essence, the fest

organizing system's dedication to data security forms a crucial foundation, fostering trust among users and upholding the integrity of the platform throughout the fest planning and execution stages.

7. **Social Integration:** The fest organizing system integrates seamlessly with social media platforms, enabling participants to forge connections, form teams, and invite friends. This social integration adds a dynamic dimension to the fest experience, fostering community engagement and interaction among participants. By facilitating these connections, the system enhances the social aspect of the fest, ensuring that participants can collaborate and share their festival journey with friends, ultimately enriching the overall fest atmosphere.

8. **Notification and Alerts:** The fest organizing system ensures that users stay informed through timely notifications and alerts, delivered via email or SMS. These notifications cover critical fest updates, approaching deadlines, and any changes in the event schedule, keeping users well-informed and engaged. By offering this streamlined communication channel, the system enhances user experience, ensuring that participants and organizers are always up-to-date with vital fest-related information.

9. **Data Generation:** The fest organizing system plays a crucial role in generating diverse data types, including event schedules, participant lists, and communication logs, offering valuable resources for a range of purposes. This generated data can be harnessed for activities such as event analysis, performance evaluation, and comprehensive reporting, enabling organizers to make data-driven decisions and enhance future festivals. By providing this data generation capability, the system empowers users to derive insights, track progress, and optimize their fest planning and execution processes, contributing to the success and growth of college festivals.

1.4.3 Applicability:-

1. It simplifies the registration process for participants, making it easy for them to sign up for multiple events, ensuring a less time-consuming and user-friendly experience.

2. The system offers a variety of features to meet different fest management needs, ensuring that organizers have all the necessary tools for a successful fest.

3. The user-friendly design of the system benefits both organizers and participants, creating a positive and efficient experience for everyone involved.

4. With an online scheduler, the system optimizes resource allocation by showing the availability of event venues and equipment, enhancing overall fest planning efficiency.

5. The system generates valuable data, such as event schedules and participant information, which can be used for various purposes, including event analysis and decision-making.

6. By automating many manual processes, the system reduces the need for paper-based forms, printing, and administrative overhead, leading to cost savings for both organizers and participants.

7. Participants and organizers can receive real-time updates on event changes, cancellations, or rescheduling, ensuring everyone stays informed.

8. The data generated by the system can be analyzed to make informed decisions about event planning, budget allocation, and marketing strategies.

9. Organizers spend less time on administrative tasks such as manual registration, event tracking, and communication, allowing them to focus on event quality and innovation.

10. The system ensures the security of participant data, including personal information and payment details, complying with data protection regulations.

1.5 Achievements: -

1. This web application has provided us with much information regarding the resources which will be used in the development of the project.

2. Asp.net has been the most compatible software for developing system applications. It is more user friendly as compared to other software.

3. The achievement of fest organizing system is marked by improved efficiency, increased revenue, reduce costs, enhanced customer satisfaction, and better overall management of fest.

SURVEY OF TECHNOLOGIES

Survey on Tools and Technologies Applicable For System Application Development

Abstract:-

In today's ever-evolving realm of software development, staying well-informed about the constantly changing landscape of tools and technologies is essential for the successful development of web applications. This survey takes a comprehensive look at the diverse array of tools and technologies at the disposal of developers, with the aim of simplifying the creation of efficient, scalable, and secure system applications. By offering insights into the current state of the field, this survey empowers developers, architects, and decision-makers to make informed choices when embarking on system application development projects.

In addition to exploring the technological landscape, this survey also highlights emerging trends, best practices, and potential challenges in system application development. It provides a valuable resource for those seeking to harness the power of the latest tools and technologies to build robust and cutting-edge system applications, ultimately fostering innovation and efficiency in the software development process.

Introduction:-

In today's digital era, there is an increasing demand for convenient and intelligent technological solutions to meet daily requirements. As a result, experts are diligently working on creating smart web applications that can be easily accessed through various devices. These web apps serve a multitude of purposes in our daily lives, whether it's paying utility bills online, shopping on platforms like Flipkart, streaming music via web-based services, or conducting online banking transactions.

Web applications span various categories, catering to entertainment, education, shopping, and more. To successfully design and develop these web applications, it's imperative to conduct a comprehensive survey of the tools and technologies that are prevalent in the realm of web application development. This survey paper focuses on shedding light on the diverse tools and technologies employed in the creation of web applications, with a particular emphasis on system applications.

The section 1: Introduces the survey, placing it within the context of web application development.

The section 2: Provides an in-depth analysis of the specific tools used for system web application development, offering a detailed comparison of their respective functionalities and applications.

The section 3: Presents findings from the survey regarding the utilization of .NET technology in the development of system web applications, providing insights into its role and contributions in this vital field of software engineering.

Tools Used To Develop Web Application:-

Introduction of Web Application:-

Web applications, often referred to as web-based software, constitute the vital framework of modern online environments. These essential software components serve as the digital bridge between users and web servers, facilitating seamless interaction and the efficient operation of online systems. Web applications are meticulously designed to enable a diverse range of critical functions, ensuring the smooth execution of tasks and the optimal utilization of resources within the digital realm.

Web applications significantly contribute to the stability and performance of online systems. They continuously monitor system health, manage error reporting, and implement mechanisms for fault tolerance and data recovery. Moreover, web applications optimize resource allocation, ensuring that online platforms run efficiently and that users can access services and information effortlessly.

In essence, web applications are the unsung heroes of the online world, diligently working behind the scenes to guarantee the reliability, security, and performance of digital systems. They form an integral part of the software landscape, bridging the gap between users and web servers, and play a pivotal role in delivering a dependable and efficient online experience.

Tools Information:-

For .NET web application development, a variety of tools and resources are instrumental in the design process:

1. **Visual Studio** – Visual Studio is an indispensable integrated development environment (IDE) for creating web applications on the .NET framework. Available for Windows and macOS, it streamlines the development of web-based, cloud, and mobile applications using languages like C#, F#, and Visual Basic.

2. **C# Language Resources** – C# serves as the primary programming language for .NET web application development. When building web applications, developers often leverage technologies like ASP.NET, ASP.NET Core, and Razor Pages for robust server-side functionality.

3. **ASP.NET** – ASP.NET is the core framework for creating web-based system applications. It provides essential features such as routing, authentication, and data access to build dynamic and scalable web

applications.

4. **ASP.NET Core -** ASP.NET Core is the modern incarnation of ASP.NET, designed to be cross-platform and highly modular. It offers improved performance and flexibility for web application development.

5. **.NET Framework Libraries** - The .NET framework offers a rich array of libraries that empower developers to construct powerful web applications. These libraries cover a wide spectrum of functionalities, from handling user interfaces to managing data access and security.

6. **Entity Framework** - Entity Framework simplifies database interactions by providing an object-relational mapping (ORM) framework. It allows developers to work with databases using .NET objects, reducing the need for complex SQL queries.

7. **Visual Studio Code** - While Visual Studio is the full-fledged IDE, Visual Studio Code (VS Code) is a lightweight, open-source code editor that's highly popular among developers. It supports various programming languages and offers numerous extensions for .NET development.

8. **SignalR** – SignalR is a real-time communication library for .NET that enables developers to add real-time functionality to web applications. It's useful for building chat applications, live dashboards, and collaborative tools.

i. **Base Class Library (BCL**): This is the fundamental library that provides core functionalities like data types, collections, file I/O, and more.

ii. **ADO.NET**: A library for data access, including database connectivity and manipulation using technologies like Entity Framework and LINQ.

iii. **Windows Presentation Foundation (WPF)**: A library for creating rich desktop applications with modern user interfaces. It supports XAML for UI design.

Technologies Used To Design and Develop .NET Web Application:-

1. **Language Used** - C# It serves as the core programming language for developing .NET web applications, providing a versatile and powerful foundation for web-based projects.

2. **Concepts Used** - ASP.NET and ASP.NET Core: ASP.NET and ASP.NET Core are fundamental concepts in .NET for building web applications. These frameworks enable the creation of dynamic, data-driven web pages, offering features like routing, authentication, and data access.

3. **IDE Used** - Visual Studio: Visual Studio is the primary integrated development environment (IDE) chosen for designing, coding, testing, and deploying .NET web applications. It provides a comprehensive set of tools for web development.

4. **Database Used** - Microsoft SQL Server: Microsoft SQL Server is a popular database management system (DBMS) used for storing and managing data in .NET web applications. It offers robust data management capabilities and integrates seamlessly with the .NET ecosystem.

5. **Web Frameworks** - Alongside ASP.NET and ASP.NET Core, various web frameworks can enhance web application development. Some popular ones include:

➢ MVC (Model-View-Controller): This architectural pattern is commonly used for building structured and maintainable web applications.

➢ Blazor: A framework for building interactive web applications using C# and .NET instead of JavaScript.

6. **Front-End Technologies –**

➢ HTML, CSS, and JavaScript: These are essential for designing and adding interactivity to web pages.

➢ JavaScript Libraries and Frameworks: Libraries like jQuery and front-end frameworks like Angular, React, or Vue.js can be integrated with .NET applications to enhance the user experience.

7. **Cloud Services** –

➢ Microsoft Azure: Offers a wide range of cloud services for hosting, scaling, and managing .NET web applications.

➢ AWS (Amazon Web Services) or Google Cloud: Alternatives to Azure for cloud hosting.

Advantages and Limitations of .NET Web Application:-

Advantages of .NET:-

1. **Robust and Secure**: .NET provides a robust and secure development environment. It includes features like code access security, encryption, and authentication to protect web applications from vulnerabilities and threats.

2. **Cross-Platform Development**: With .NET Core and .NET 5 (and later), you can build web applications that run on Windows, macOS, and Linux, increasing the reach of your applications.

3. **Rapid Development**: Visual Studio, the integrated development environment (IDE) for .NET, offers a wide range of tools and features that accelerate web application development, including templates, code generation, and debugging tools.

4. **Scalability**: .NET applications are highly scalable, making it easy to accommodate increased user loads and traffic as your web application grows.

5. **Code Reusability**: .NET promotes code reusability through libraries and components, reducing development time and effort.

6. **Open Source**: Many components of the .NET framework are open source, allowing developers to access and contribute to the codebase.

7. **Strong Support for Web Services**: .NET has excellent support for building web services, making it a great choice for developing web APIs, SOAP services, and RESTful services.

8. **Rich User Interface**: With technologies like ASP.NET and Blazor, .NET enables the creation of modern and feature-rich user interfaces for web applications.

9. **Easy Maintenance**: The structured and modular nature of .NET applications makes them easier to maintain and update over time, reducing the total cost of ownership.

Limitations of .NET :-

1. **Platform Dependency**: Historically, .NET was primarily associated with Windows, which limited cross-platform compatibility. Although .NET Core and .NET 5 (and later) have made significant strides in supporting multiple platforms, there may still be platform-specific challenges.

2. **Learning Curve**: Learning the intricacies of the .NET ecosystem, including C# and ASP.NET, may require time and effort, especially for developers who are new to the framework.

3. **Limited Support for Older Versions**: Microsoft often focuses on the latest versions of .NET, which means that older versions may receive limited support and updates. This can present challenges for maintaining legacy applications.

4. **Integration Complexity**: Integrating .NET applications with third-party tools and technologies can sometimes be more complex compared to platforms that are more open or have standardized interfaces.

5. **Resource Intensive**: .NET applications, especially when running on Windows, can be resource-intensive. This may require more memory and processing power compared to lighter-weight frameworks, impacting scalability and hosting costs.

6. **Limited Support for Non-Windows Environments**: While .NET Core and .NET 5 (and later) support cross-platform development, certain features and tools may still be more robust and polished on Windows, potentially affecting the development experience on other operating systems.

7. Deployment Complexity: The deployment process for .NET applications can be more complex compared to simpler scripting languages, especially when deploying to different environments or using containerization.

9. **Performance Tuning Complexity**: Achieving optimal performance in .NET applications may require in-depth knowledge and tuning, making it less accessible for developers without performance optimization expertise.

SQL Server:-

For developing the web applications as well as business applications, there is need to store the data and for this purpose most of the experts prefer the open source database applications which are also applicable while developing the mobile app. Therefore, this facility of the open source database has been provided by SQL. SQL SERVER is a fast, easy-to-use DBMS which is used for storing data generated through the small applications and big business applications. SQL SERVER is popular because of following reasons:

1. SQL SERVER is an open-source license so user has nothing to pay to use it. It handles large databases, so it will possible to organize 50 million rows or more in a table with default file size limit for a table is 4GB.
2. SQL SERVER handles a large subset of the functionality of the most expensive and powerful database packages. SQL SERVER uses a standard form SQL data language.
3. SQL SERVER works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
4. SQL SERVER offers excellent scalability options, allowing businesses to start with a small database and easily scale up as their data and application requirements grow. It can handle both small and large datasets efficiently.
5. SQL SERVER offers features like failover clustering, database mirroring, and Always On Availability Groups, ensuring high availability and minimal downtime in case of hardware failures or other issues.
6. SQL SERVER has a vast ecosystem of third-party tools, extensions, and community support, making it easier to extend its functionality and find solutions to common challenges.
7. SQL SERVER integrates well with Visual Studio, offering developer-friendly tools for creating, testing, and debugging database-driven applications.
8. SQL SERVER includes tools and features for performance optimization, including indexing, query optimization, and in-memory processing, which help applications run efficiently and respond quickly to user requests.
9. SQL SERVER has a large and active community of users and developers, as well as access to Microsoft's support resources, forums, and documentation.

Justification of selection of Technology:-

The choice of .NET technology is influenced by several factors, including project requirements, team preferences, and compatibility. Here are some compelling reasons why someone might opt for .NET technology:

1) **Cross-Platform Capability:**
   .NET Core, a part of the .NET framework, is cross-platform and supports Windows, macOS, and Linux. This versatility accommodates diverse development environments and teams.

2) **Versatile Development:**
   .NET encompasses a wide range of programming languages, including C#, F#, and Visual Basic. This versatility allows developers to choose the language that best suits their project's needs.

3) **Robust Ecosystem:**
   .NET offers a comprehensive ecosystem with libraries, frameworks, and tools for various application types, including web, desktop, mobile, cloud, and IoT.

4) **Strong Integration:**
   .NET integrates seamlessly with Microsoft technologies, such as Azure cloud services, SQL Server, and Visual Studio IDE, facilitating a unified development experience.

5) **Community and Resources:**
   The .NET community is active and well-supported. Developers can access extensive documentation, tutorials, and community forums for assistance and problem-solving.

6) **Modern Web Development:**
   .NET supports modern web development with technologies like ASP.NET Core, enabling the creation of high-performance web applications.

7) **Security and Reliability:**
   .NET emphasizes security and offers robust security features, making it a reliable choice for applications that prioritize data protection.

8) **Scalability:**

.NET applications can be easily scaled to accommodate increased workloads and user demands, making it suitable for both small-scale and large-scale projects.

9) **Enterprise-Ready:**

.NET technology is well-suited for enterprise-level applications, with features like scalability, performance optimization, and comprehensive support.

10) **Integrated Development Environment (IDE):**

Visual Studio, Microsoft's integrated development environment for .NET, provides powerful tools for code editing, debugging, testing, and collaboration.

11) **Support for Multiple Platforms:**

With .NET 5 and later, .NET provides support for a wide range of platforms, including desktop, web, mobile, cloud, and IoT, enabling developers to target diverse environments with a single codebase.

12) **Open Source:**

.NET is open source, allowing developers to use it without licensing costs and fostering a collaborative and transparent development community.

Conclusion:-

The conventional web application development scenario is changing with the usages of the smart devices .This results in opening of a door for development of web applications which will work on Smart Devices in the smarter way. With respect to development of the web application, the above survey is made. This concludes that, the developer can make the use freely available, open source tools and technologies which are user friendly.

SYSTEM ANALYSIS

3 .1 Problem Definition:- Spotting error message

1. **Runtime Exceptions**: .NET web applications may encounter runtime exceptions, such as 'System.TypeLoadException' or 'System.NullReferenceException,' which disrupt the normal execution flow. These exceptions occur when the application attempts to access resources or objects that are not present or properly initialized.

2. **File Not Found (FileNotFoundException)**: When working with file I/O operations, a common issue arises when the specified file is not found at the specified path. This exception can impede critical file-related operations within the web application.

3. **Infinite Loops**: The presence of infinite loops in a .NET web application without a defined exit condition can lead to application hangs, rendering it unresponsive. Identifying and rectifying such loops is essential for maintaining application stability.

4. **Build and Compilation Errors**: .NET web applications are susceptible to build and compilation errors that stem from issues like incorrect project settings, missing dependencies, or problems with the build path. These errors hinder the successful compilation and execution of the application.

5. **Missing Dependencies**: Specific to .NET, missing or improperly referenced dependencies can lead to errors like "Assembly not found" or "Type or namespace not found." These issues can disrupt the functionality of the web application.

The existing system has the following requirement:

- System need to store information about Events.
- System need to maintain quantity record.
- System need to keep the record of participants.
- System need to update and delete the Events after the events date.
- System also needs overall information about the college and different Events.

3.2 Modular Description:

1) **Login Info Module:** A Login, logging in or logging on is the entering of identifier information into a system by answer in order to access to that system.

2) **Dashboard Module:** Redirect you to admin page/module or user module. As per login info.

3) **College module:** In this module one can have a view on different college organizing the events**.**

4) **participant Module:** participant is the one that participate in the events, fill the event form, etc.

5) **Admin Module:** The admin works as the steering the web application as he is the one who can see the records of college and Events. The overall web application is handled by the admin from publishing different college to the Different Events in that Respective college . If the colleges wishes to update any of his Events details he has to intimate the admin. The admin has the rights to insert, update and delete any operation from the web application.

6) **User module:** In user module, the participant see the different college organizing the different events and the participants can fill the form of Events as per their interest .

REQUIREMENTS ANALYSIS

4.1 Functional Requirement: Authentication

1. **Login-** The user can login to the authentication system username and password.
2. **Logout-** The user can log out from the apps after complete their work.
3. **Login failure-** If the user does not match in the login phase or the user has not yet being authorized by the admin.

Non-Functional Requirement:

1. Performance Requirement

Performance requirements define the acceptable response times for system functions: -

1) The system shall take initial load time depending on the internet connection strength which also depends on the media from which the application is running.
2) The performance shall depend on the hardware components of the client.
3) The Fest Fusion website must be accessible and up and running 24 hours a day, 7 days a week and 365 days a year.
4) The project shall display clear human-readable error messages.

2. Maintainability Requirement

It should be easy to add, remove or modify modules in this website. Debugging should not be difficult.

3. Availability Requirement

The website should be available 24 x 7. Services should be provided to the customers as and when requested.

4.2 Software and Hardware Requirement:-

4.2.1 Hardware Requirement table:

For development In System:	
RAM	Minimum 4 GB
Processor	Corei3
Space Required	20 GB HD Space Minimum

4.2.2 Software Requirement table:-

For development of Application in Systems:	
Operating System	Windows 7 or Above, Android Studio
Front-End Language	C#, ASP.NET
Back-End Database	SQL

Preliminary Product Description:-

The first step in the system development life cycle is the preliminary description to determine the feasibility of the system. The purpose of the preliminary description is to evaluate project request. It is not a design study nor does it include the collection of details to describe the business system in all respect.

Rather, it's the collecting of information that helps committee members to evaluate the merit of the project request and make an informal judgement about the feasibility of the proposed project.

Analysts working on the preliminary investigation should accomplish the fallowing objectives:-

- Clarify and understand the project request.
- Determine the size of the project.
- Assess costs and benefits of alternative approaches.
- Report the findings to management, with recommendations outlining the Acceptance or rejection of the proposal.
- Determine the technical and operational feasibility of alternative approaches.

PLANNING AND SCHEDULING

Planning:-

The purpose of Project Planning is to identify the scope of the project, estimate the work involved, and create a project schedule. Project planning begins with requirements that define the software to be developed. The project plan is then developed to describe the tasks that will lead to completion.

Scheduling:-

The project schedule is the tool that communicates what work needs to be performed, which resources of the project members will perform the work and the timeframes in which that work needs to be performed. The project schedule should reflect all of the work associated with delivering the project on time.

Task No.	Task Name	Start Date	End Date	Duration
T1	Requirement Gathering	16-Aug-23	13-Sep-23	23
T2	Requirement Analysis	14-Sep-23	27-Sep-23	14
T3	Design	28-Sep-23	18-Oct-23	21
T4	Coding	19-Oct-23	20-Nov-23	36
T5	Testing	21-Nov-23	21-Dec-23	24
T6	Deployment	22-Dec-24	20-Jan-24	21
T7	Implementation	21-Jan-24	18-Feb-24	16

1.1. Planning and scheduling table

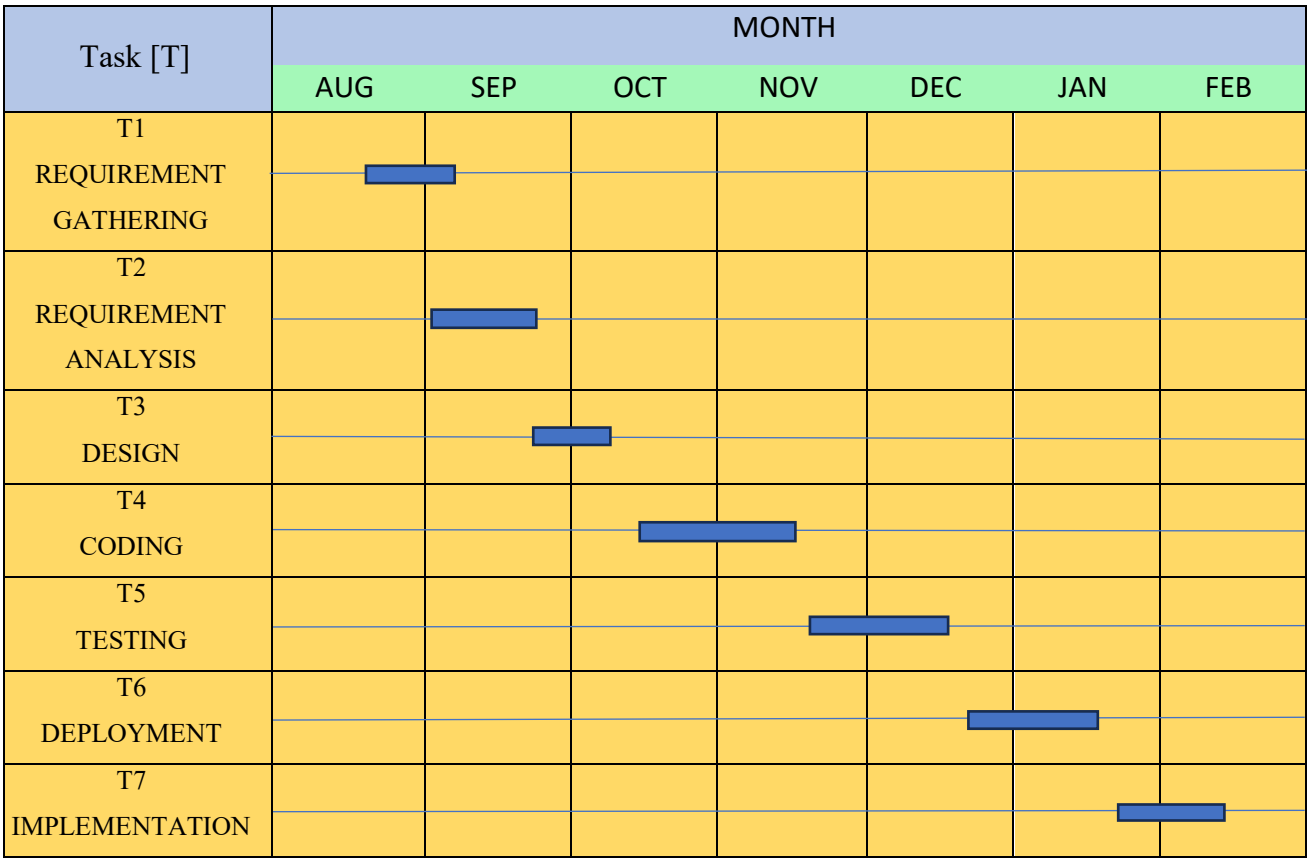
Gantt chart:

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale in month.

Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. This allows you to see at a glance.

- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to last
- Where activities overlap with other activities, and by how much
- The start and end date of the whole project

To summarize, a Gantt chart shows you what has to be done (the activities) and when (the schedule).



Gantt Chart for project Schedule Task against No of Days in Month

SYSTEM DESIGN

SYSTEM DESIGN:-

System design is the process of defining the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. It is meant to satisfy specific needs and requirements of a business or organization through the engineering of a coherent and well-running system.

Following diagrams are:

- Class Diagram
- Sequence Diagram
- Use case Diagram
- Activity Diagram
- ER Diagram

6.1 CLASS DIAGRAM:-

The class diagram of our Fest Fusion web application shows the three main classes: User, Admin, and Event.

1. User - represents users and includes attributes such as UserID, Username, Email, and more.

2. Admin - represents administrators and includes attributes like AdminID, CollegeUgcNo, and Password.

3. Event - represents events and includes attributes EventID, EventName, EventType, EventDetails, Members, and Sponsors.

There are two additional classes:

1. Pass - represents passes associated with users and events, with attributes like PassID and Valid.

2. Form - represents forms submitted by users for event participation, with attributes like FormID, Status, Activity, and EventID.

3. Relationships:

i Users can have multiple passes (1..*) for different events, as indicated by the relationship between User and Pass.

ii Users can submit multiple forms (1..*) for various events, as indicated by the relationship between User and Form.

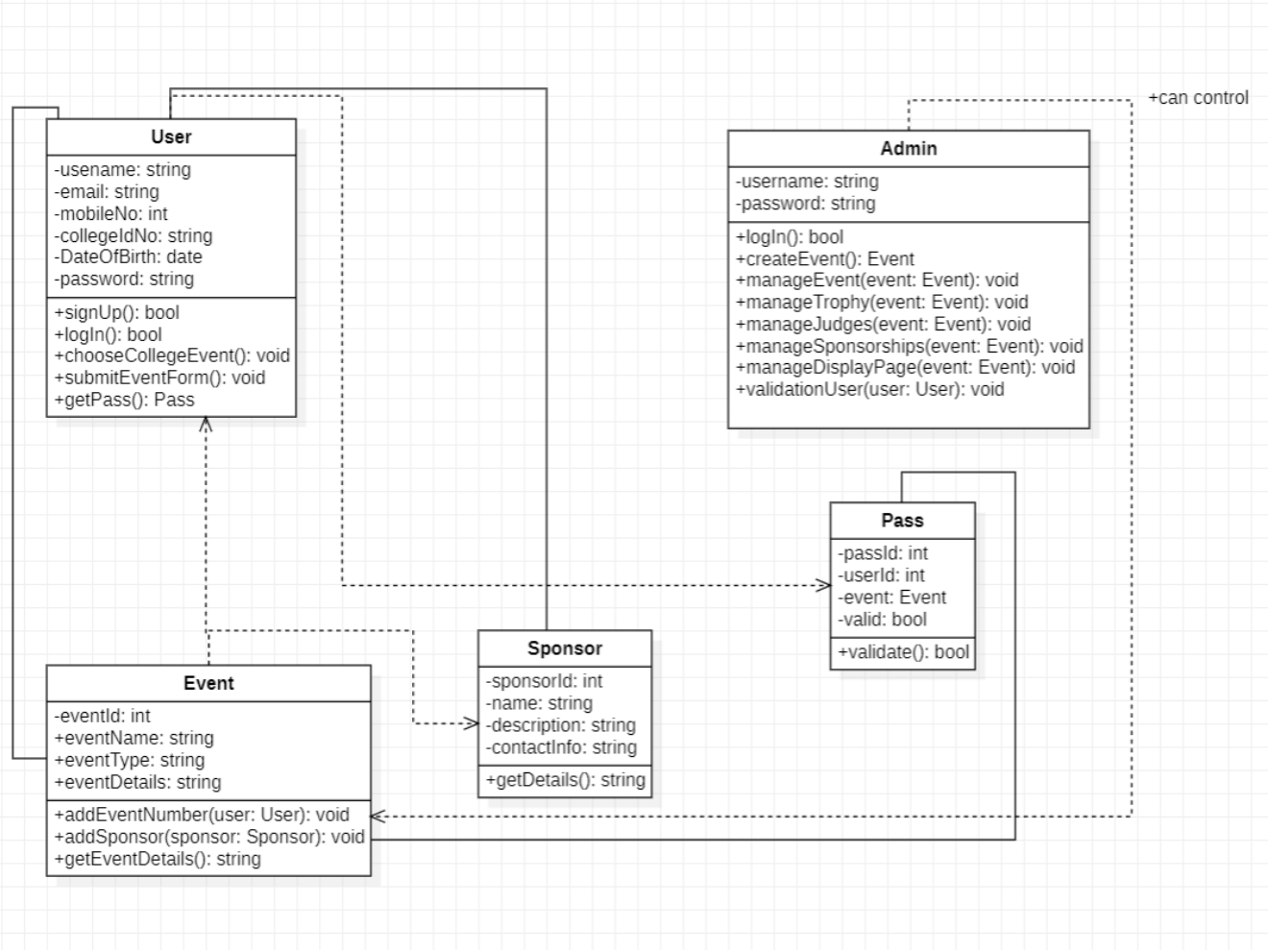


FIG:- 1 CLASS DIAGRAM OF FEST FUSION WEB APPLICATION

6.2 SEQUENCE DIAGRAM:-

The UML sequence of Diagram of fest fusion, where admin will be able to login to their account using their credentials. After login the user will be directly go to the home page. Where the user will find all the colleges list where the events are taking place. After selecting college the user can see the list of events which will going to take place in that college. The user can select the event and fill the form to take participate in the event.

After filling the form the user will get autogenerated pass through which the user can get the entry in that specific college on the event day.

The instance of class objects involved in this UML sequence diagram are as follows:

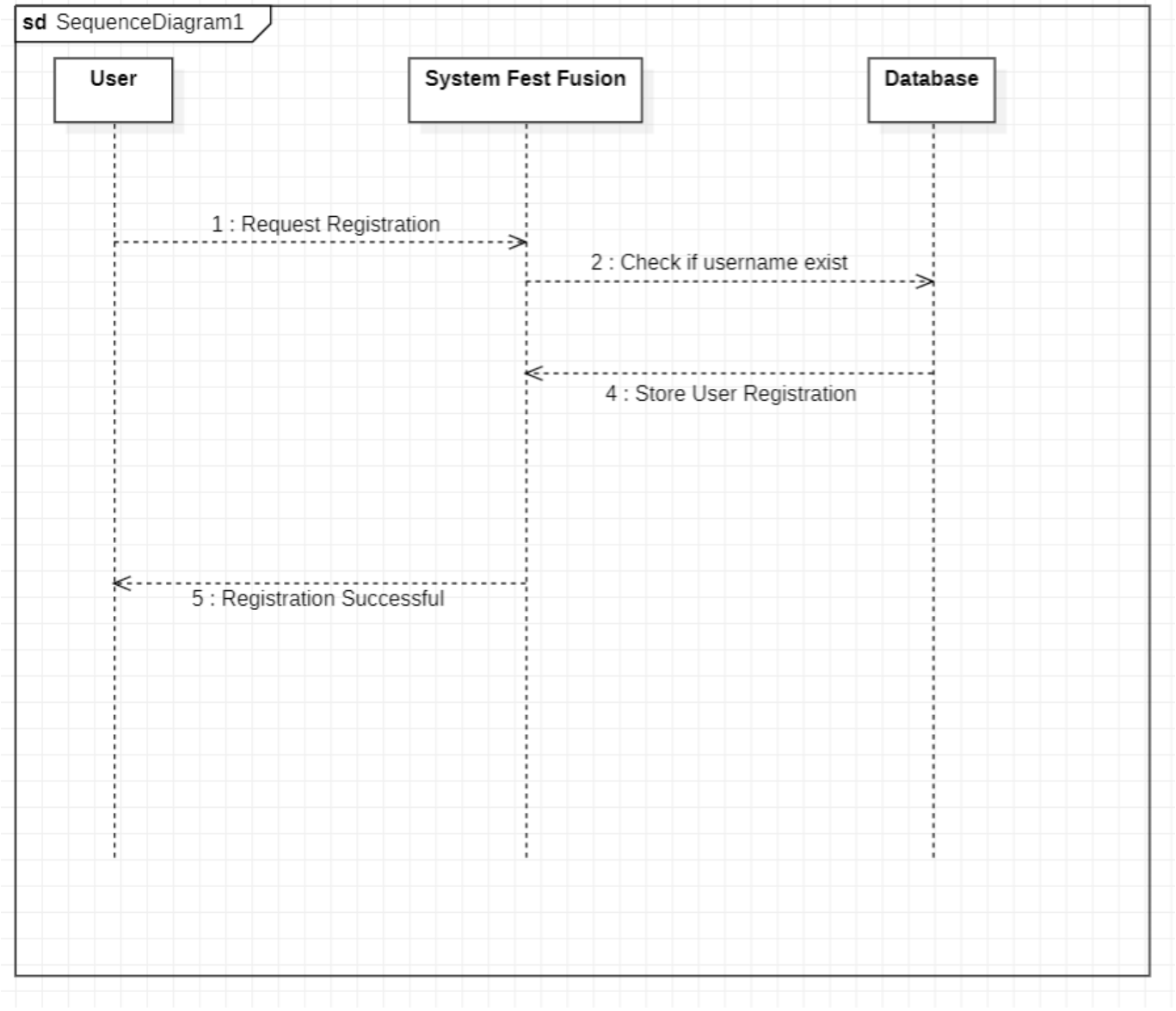


FIG:- 2 SEQUENCE DIAGRAM OF FEST FUSION WEB APPLICATION

6.3 USE CASE DIAGRAM:-

This Use Case Diagram is a graphic depiction of the interaction among the element of Doctor Appointment System. Some of the major elements are:

• Register: A user needs to register in order to access the application. The user needs to enter his/her username, password, age, and country. Those information will be stored in the database and will serve as the profile info.

• Login: Any app operation requires the user to be registered. Login is the first step before performing an action.

• Show colleges: After successful login the user can see the list of colleges on the dashboard where the event are going to take place.

• Show Events: After selecting college the user can see the list of events that are going to take place in that college.

• Registration form: The user can fill the form if he wants to take part in any college event.

•Validate user by detail: The admin will validate the user detail.

• Generate pass: After filling form and validation the user will get autogenerated pass.

• Checking details about events: The user can check all the detail about the events like time ,place ,etc.

• Create college event : The Admin have the right to add the colleges and remove the colleges after the fest.

• Add event detail: The Admin have the right to add the event details and manipulate it..

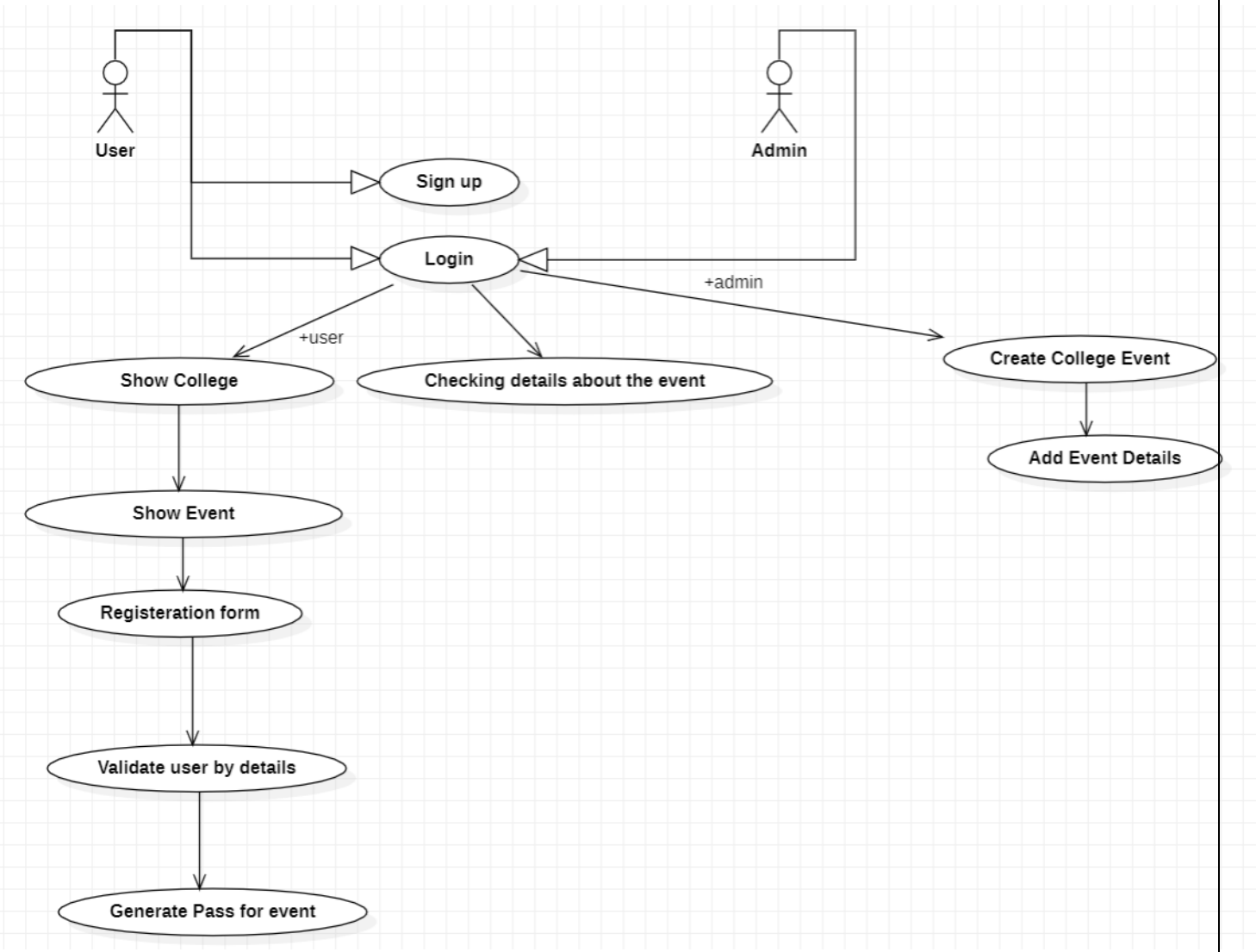


FIG:- 3 CASE DIAGRAM OF FEST FUSION WEB APPLICATION

6.4 ACTIVITY DIAGRAM:-

An activity diagram is a type of UML diagram used to model the flow of actions within a system, process, or use case. It visually represents the sequential and parallel activities performed by different actors (e.g., users, systems) and provides a high-level view of the system's behavior.

In an activity diagram:

1. Activities: Activities are represented as rounded rectangles and represent specific actions or steps in the process. They can range from simple tasks to complex operations.

2. Transitions: Transitions, represented by arrows, depict the flow of control from one activity to another. They show the order in which activities are executed.

3. Decision Nodes: Decision nodes are represented as diamonds and denote points where the flow branches based on certain conditions or decisions.

4. Forks and Joins: Forks split the flow into multiple concurrent activities, while joins bring them back together.

In this activity UML diagram of Recipe App it shows the flow between the activities as follows :

- Login/Sign-up Activity
- Create event Activity
- Choose college event Activity
- Manage event Activity
- Manage display page Activity

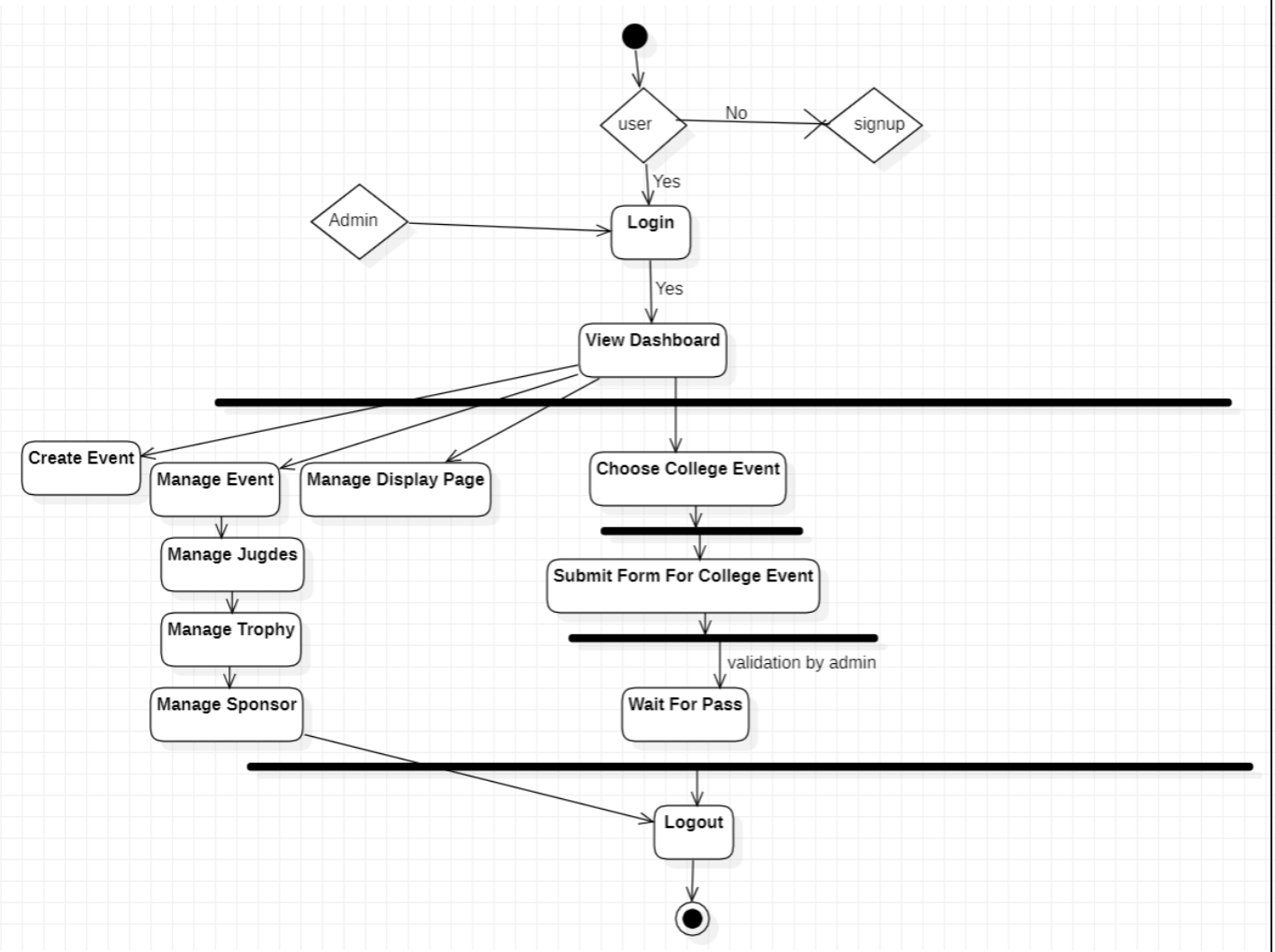


FIG:- 4 ACTIVITY DIAGRAM OF FEST FUSION WEB APPLICATION

6.5 ER DIAGRAM:-

An Entity-Relationship diagram is a visual representation of the data model that depicts the entities, attributes, and relationships among the entities in a database. It visually shows the entities (objects) in a database and the relationships between them. ER diagrams are widely used in database design and are an essential part of the Entity-Relationship Model.

In an ER diagram:

1. Entities: Entities represent the real-world objects or concepts that need to be stored in the database. Each entity is represented by a rectangle, and its attributes (properties) are listed within the rectangle.

2. Attributes: Attributes are the characteristics or properties of an entity. They provide more detailed information about the entities and are represented as ovals connected to the corresponding entity.

3. Relationships: Relationships illustrate the connections or associations between different entities. They show how entities are related to each other and are represented by diamond shapes. Relationships can be one-to-one, one-to-many, or many-to-many.

The main entities of Fest fusion are : User, Event form, Admin, Event.

• User Entity : The attributes are Name and Password ,etc.

• Event Form Entity : The attributes are ticket price and category event no of participation ,etc.

• Admin Entity : The attributes id and Password, college number,etc

• User Entity : The attributes are event type and event id ,event detail ,event name ,etc.

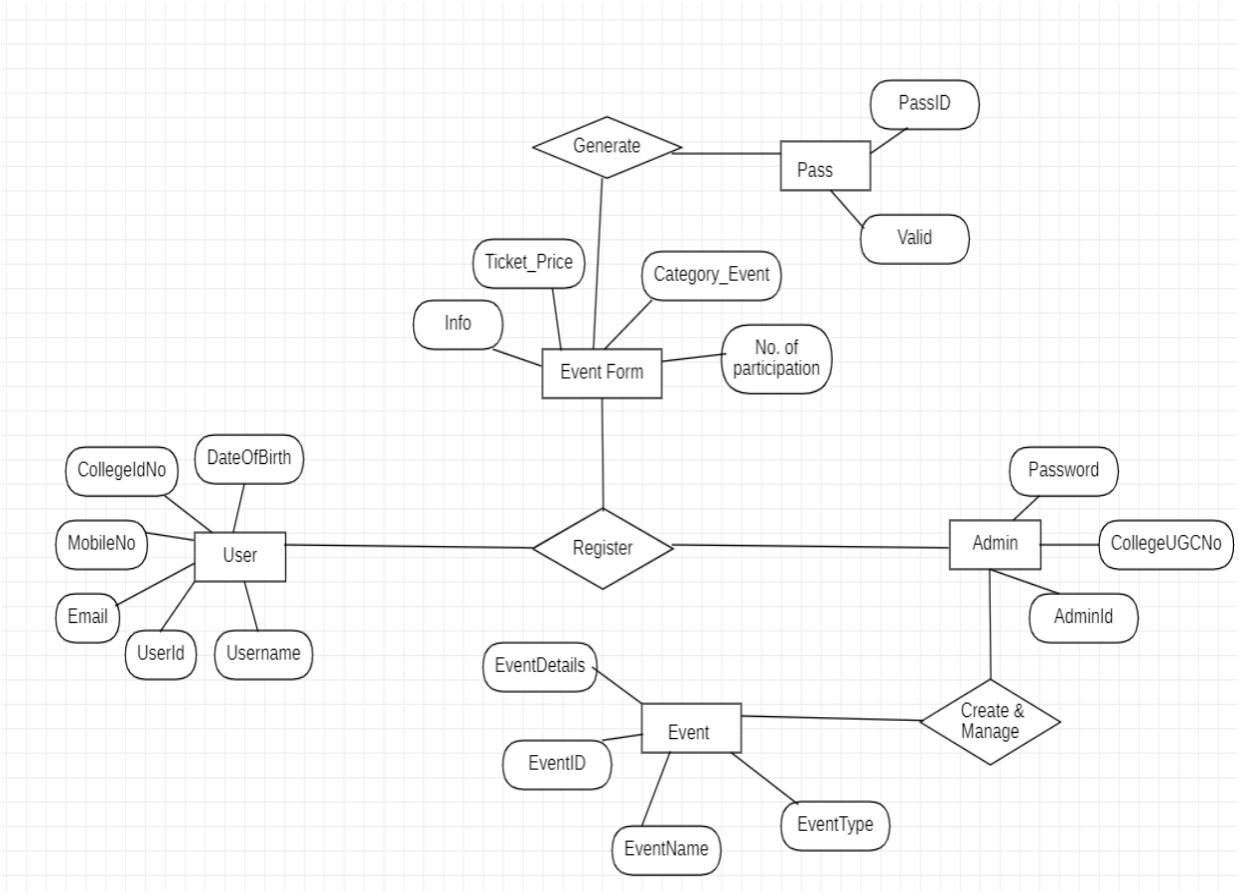


FIG:- 5 ER DIAGRAM OF FEST FUSION WEB APPLICATION

6.6 Data design:-

Data design involves the systematic planning and arrangement of data components within a system or application to enable effective storage, retrieval, and handling of information. It includes outlining the structure, format, and connections between data elements to guarantee precision, uniformity, and usability. This procedure usually entails recognizing data entities, attributes, and their relationships, as well as establishing guidelines for data storage, access management, and upkeep.

A well-executed data design provides a solid basis for efficient data management, analysis, and decision-making, allowing organizations to extract valuable insights and realize the potential of their data assets.

Data design entails the organized structuring and modeling of data to fulfill the distinct requirements and goals of an organization or project. It covers various phases, including conceptual design, logical structuring, and physical setup, all aimed at refining data configurations and workflows for improved efficiency and usability.

This approach frequently employs methods like entity-relationship modeling, normalization, and schema formulation, customized to meet the demands of the specific industry and application context. Through meticulous data architecture and schema design, businesses can enhance operational processes, elevate data integrity, and facilitate informed decision-making, ultimately fostering innovation and maintaining a competitive edge in the contemporary data-centric environment.

6.7 Data Integrity and Constraints:

Data integrity and constraints:

- Constraints
- GUI is only in English.
- Login and password is used for identification of user and there is no facility for guest.
- Data Integrity

1) Entity integrity
2) Referential integrity
3) Domain integrity

Definition of data integrity:

Data integrity pertains to the precision, uniformity, and dependability of data throughout its lifespan within a system or database. It ensures that the data remains unaltered and preserves its intended significance and validity over time, even in the presence of potential mistakes, corruption, or unauthorized changes.

Data integrity is upheld through various measures like data validation, error identification, and access management, which deter unauthorized alterations and maintain the reliability of the data. By ensuring data integrity, organizations can instill trust in their data assets, improve decision-making processes, and adhere to regulatory standards, thereby protecting the reliability and credibility of their operations and offerings.

1) Entity Integrity :

   Entity integrity guarantees that every row in a database table possesses a distinct identifier, usually achieved through primary keys, to avoid duplicate entries and uphold data uniformity.

2) Referential integrity :

   Referential integrity ensures that connections between tables in a relational database are preserved, often enforced by foreign key constraints, to ensure that links between associated tables stay accurate, preventing orphaned or incongruent data.

3) Domain integrity :

   Domain integrity ensures that data values stored in a database conform to predetermined rules or limitations, including data type, format, and range specifications, thereby ensuring the accuracy and uniformity of data at the attribute level.

6.7 Data integrity and constraints:

User Signup/ Login Detail:

User Detail	Data type	Constraints	Size
Id	Int	Primary key	16
Name	Varchar	Null	255
DOB	Date	Null	50
Username	Varchar	Null	255
Password	Varchar	Null	255
ConfirmPassword	Varchar	Null	255

Admin Signup/ Login Detail:

User Detail	Data type	Constraints	Size
ID	Int	Primary key	16
Password	Varchar	Null	255

Participant Detail:

User Detail	Data type	Constraints	Size
Id	Int	Primary key	16
Name	Nvarchar	Null	100
College_name	Nvarchar	Null	100
Game	Nvarchar	Null	100
Class	Nvarchar	Null	50
Class_year	Nvarchar	Null	50
Id_number	Nvarchar	Null	50
CreatedOn	Datetime	Null	50

Gamename Detail:

User Detail	Data type	Constraints	Size
ID	Int	Primary key	16
Game	Varchar	Null	50
price	Int	Null	50

Collegename Detail:

User Detail	Data type	Constraints	Size
ID	Int	Primary key	16
College	Varchar	Null	50

CollegeGame Detail:

User Detail	Data type	Constraints	Size
ID	Int	Primary key	16
College	Int	Null	50
GameId	Int	Null	50

Algorithms Design:-

STEP 1: Start

STEP 2: Login

STEP 3: Enter valid username & password

STEP 4: If password and username are correct then go to the Home Page Else If fail go back the login Page.

STEP 5: After successful authentication User can see the different Colleges which are organizing the events.

STEP 6: Click on college and event game in which user wants to participate

STEP 7: Fill the Event Form

STEP 8: Make the payment

STEP 9: Checkout the passes

STEP 10: Logout

Security Issues :

Security issues in system applications refer to vulnerabilities, weaknesses, or threats that can compromise the confidentiality, integrity, and availability of data and functionalities within software systems. These issues can arise due to various factors such as programming errors, design flaws, inadequate access controls, insecure configurations, and insufficient encryption. Common security issues in system applications include but are not limited to:

1. **Injection attacks:**
   Such as SQL injection or command injection, where attackers exploit vulnerabilities to execute malicious commands or scripts within the application's database or operating system.

2. **Authentication and authorization flaws:**
   Weak authentication mechanisms or improper authorization controls can allow unauthorized users to access sensitive data or perform unauthorized actions

3. Sensitive data exposure:

Failure to properly protect sensitive data, such as personal or financial information, can lead to data breaches and unauthorized access.

4. Cross-Site Scripting (XSS):

Vulnerabilities that allow attackers to inject malicious scripts into web pages viewed by other users, compromising their data or session information.

5. Insecure direct object references:

Allowing users to access resources directly via references like file names or database keys, without proper validation, can lead to unauthorized access to sensitive information.

6. Security misconfigurations:

Improperly configured security settings, permissions, or network protocols can create avenues for attackers to exploit vulnerabilities in the system.

7. Insufficient logging and monitoring:

Inadequate logging and monitoring capabilities can hinder the detection and response to security incidents, allowing attackers to operate undetected.

IMPLEMENTATION AND TESTING

Implementation Approaches :

Planning is the most vital aspect of this project. To implement the following project we had to gradually take a procedural approach. The entire project was highly dependent on the technique of each and every implementation aspect. The Project is basically implemented through 'ASP.NET' and the language used in it was 'HTML,CSS,JAVASCRIPT & C#'.

The Database Connectivity plays an important role in the back end of the project. This project is entirely based on the Web application and is made much convenient for the user to understand the website and its working procedures.

ADOPTED METHODOLOGY

Iterative Model :

In Iterative Model, you can start with some of the software specifications and develop the first version of the software. After the first version if there is a need to change the software, then a new version of the software is created with a new iteration. Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.

The Iterative Model allows the accessing earlier phases, in which the variations made respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

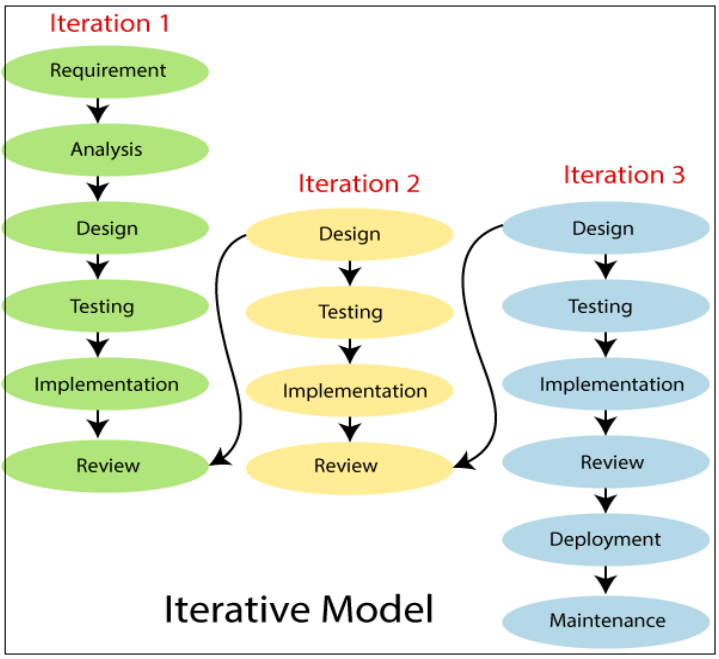


Fig. Iterative Model

What is iterative model?

The Iterative Waterfall Model is a software development approach that combines the sequential steps of the traditional Waterfall Model with the flexibility of iterative design. It allows for improvements and changes to be made at each stage of the development process, instead of waiting until the end of the project. The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.

This iterative process is flexible and adaptive, allowing for changes to be incorporated based on evolving requirements or new insights. By addressing high-risk elements early and continuously testing and refining the product, the Iterative Model helps manage project risks effectively. The Iterative Model offers flexibility, adaptability, and a focus on delivering value incrementally, making it suitable for projects with evolving requirements or uncertain conditions.

The various phases of Iterative model are as follows:

1. Requirement gathering: This is the first stage where the business owners and developers meet to discuss the goals and requirements of the website.

2. Design: In this stage, the developers create a preliminary design of the website based on the requirements gathered in stage 1.

3. Implementation: In this stage, the developers begin to build the website based on the design created in stage 2.

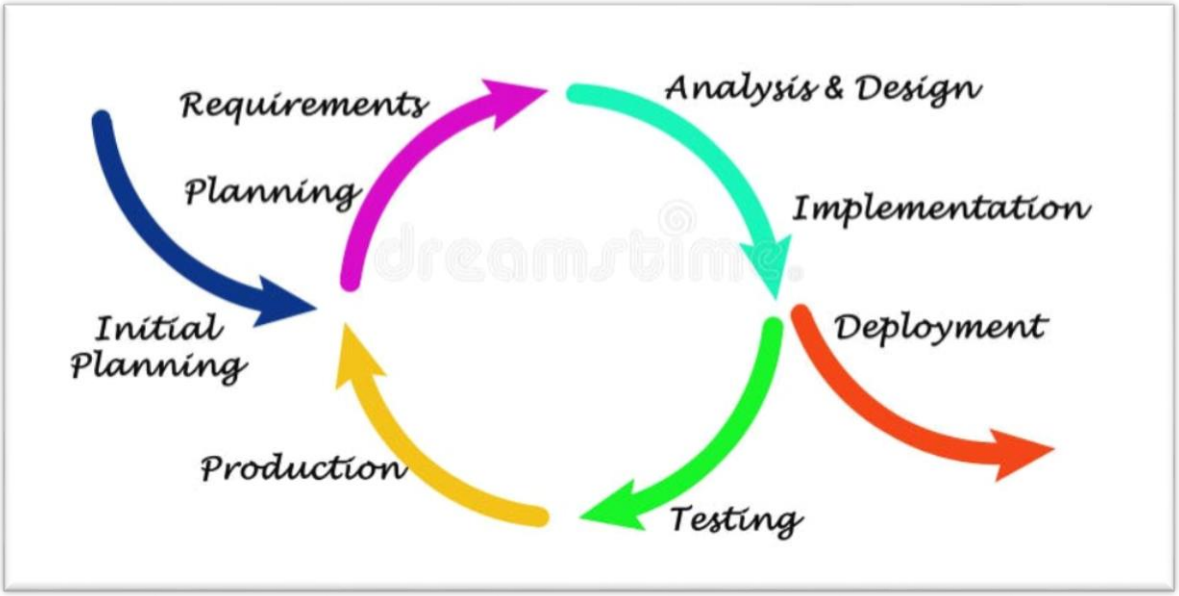
4. Testing: Once the website has been built, it is tested to ensure that it meets the requirements and functions properly.

5. Deployment: The website is then deployed and made live to the public.

6. Review & Improvements : After the website has been live for a while, the business owners and developers review its performance and make any necessary improvements.

This process is repeated until the website meets the needs and goals of the business. Each iteration builds upon the previous one, allowing for continuous improvement and iteration until the final product is complete.

7. Maintenance: In the maintenance phase, after deployment of the software in the working environment there may be some bugs, some errors or new updates are required. Maintenance involves debugging and new addition options.



When to use the Iterative Model?

1. The prerequisite of being well-defined and comprehended.
2. The development team is gaining knowledge about new technologies.
3. Certain characteristics and objectives carry a significant chance of failure in the future.

Advantage of Iterative Model:

1. Some working functionality can be developed quickly and early in the life cycle.
2. Parallel development can be planned.
3. Progress can be measured.
4. Testing and debugging during smaller iteration is easy.
5. Less costly to change the scope/requirements.
6. It supports changing requirements.

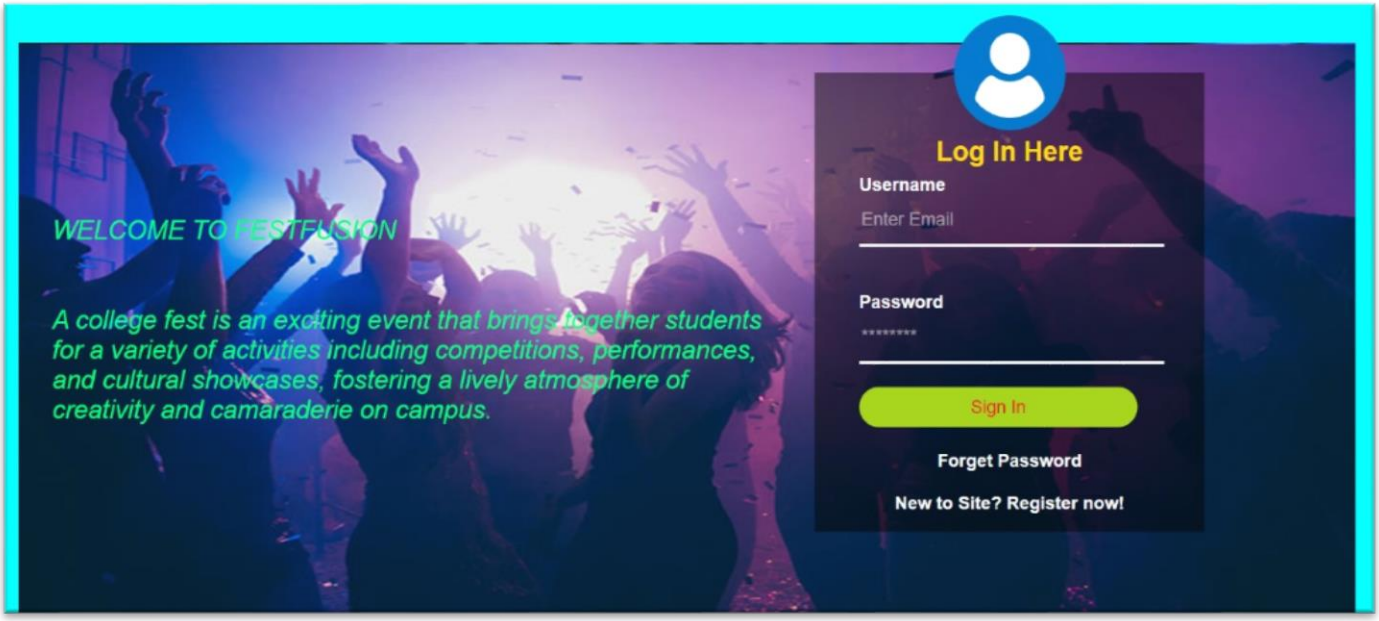
Disadvantage of Iterative Model:

1. More resources may be required.
2. Although cost of change is lesser but it is not very suitable for changing requirements.
3. Defining increments may require definition of the complete system.
4. Not suitable for smaller projects.
5. Management complexity is more.
6. End of project may not be known which is a risk.

7.1 Coding details and Code Efficiency

Source Code:-

Login.aspx.cs:

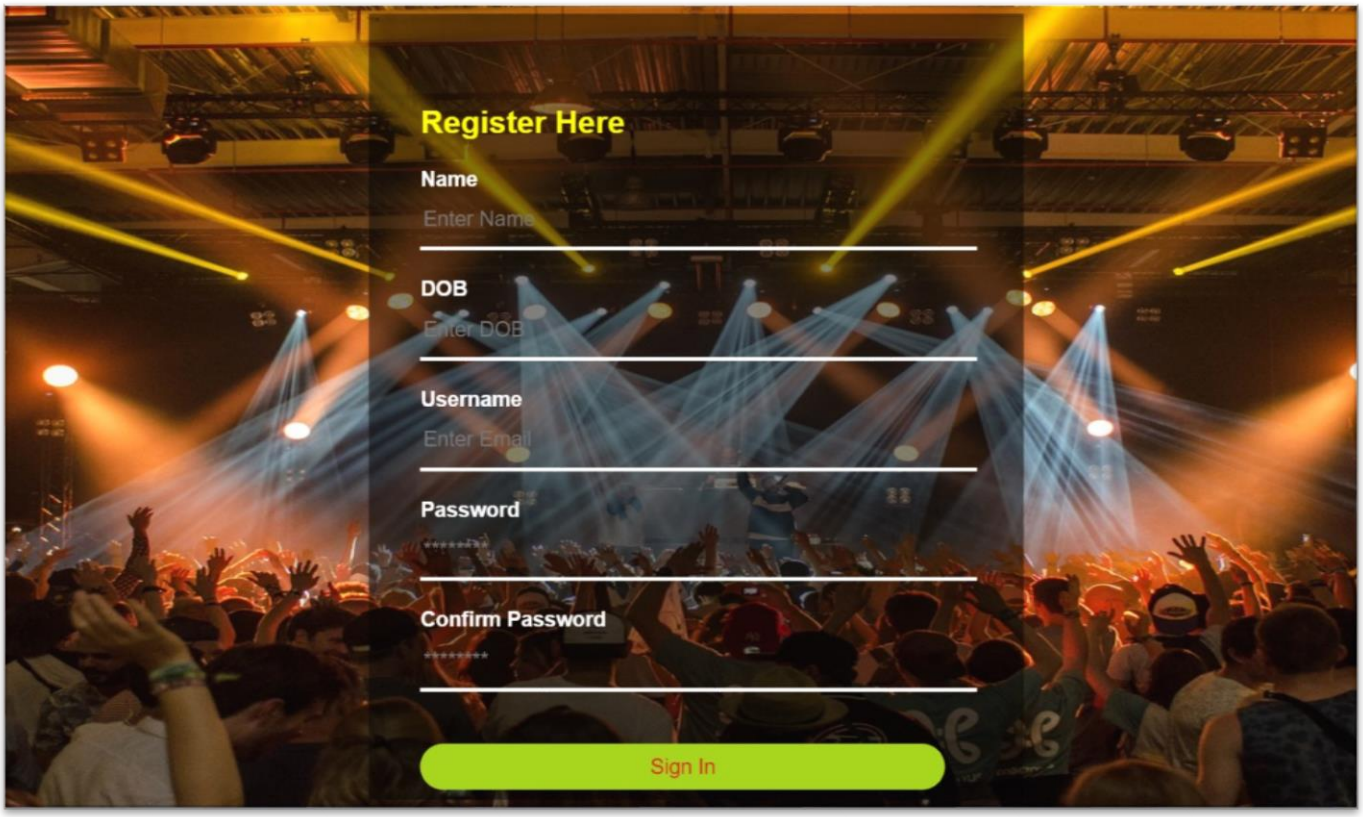



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data.SqlClient;

public partial class login : System.Web.UI.Page
{
    string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void button1_Click(object sender, EventArgs e)
```

```
    {
        string Username = TextBox1.Text.Trim();
        string Password = TextBox2.Text.Trim();
        using (SqlConnection con = new SqlConnection(connection))
        {
            try
            {
                con.Open();
                string query = "SELECT COUNT(*) FROM signup WHERE Username = @username AND
Password = @password";
                SqlCommand cmd = new SqlCommand(query, con);
                cmd.Parameters.AddWithValue("@username", Username);
                cmd.Parameters.AddWithValue("@password", Password);
               int result = (int)cmd.ExecuteScalar();
                if (result > 0)
                {
                    if (Username.ToLower() == "admin" && Password.ToLower() == "admin")
                    {
                        // Redirect to admin page
                        Response.Redirect("adminpage.aspx");
                    }
                    else
                    {
                        // Successful login for regular user
                        Session["Username"] = Username; // Store username in session
                        Response.Redirect("Homepagelogout.aspx"); // Redirect to dashboard or another page
                    }
                }
                else
                {
                    // Invalid login
```

```
                Label1.Text = "Invalid username or password";
            }
        }
        catch (Exception ex)
        {
            // Handle exception
            Label2.Text = "An error occurred: " + ex.Message;
        }
    }
}
protected void Unnamed4_Click(object sender, EventArgs e)
{
    Response.Redirect("signup.aspx");
}
}
```

Signup.aspx.cs :

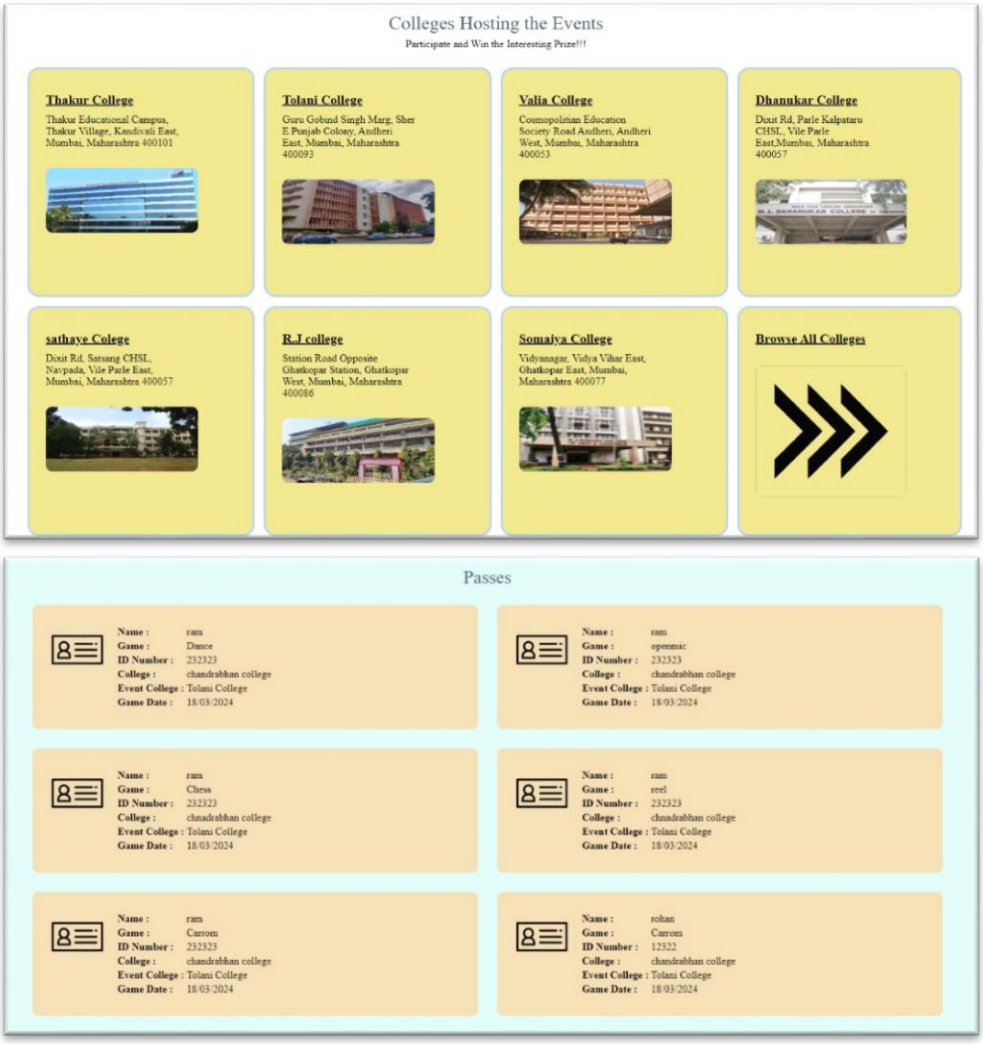



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data.SqlClient;

public partial class signup : System.Web.UI.Page
{
    string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
    protected void Page_Load(object sender, EventArgs e)
    {
    }
```

```
protected void button1_Click(object sender, EventArgs e)
{
    string fullName = TextBox3.Text;
    DateTime dob = DateTime.ParseExact(TextBox4.Text, "dd/MM/yyyy", null);
   // DateTime dob = Convert.ToDateTime(TextBox4.Text); // Changed data type to string
    string username = TextBox1.Text;
    string password = TextBox2.Text;
    string confirmPassword = TextBox5.Text;
    using (SqlConnection con = new SqlConnection(connection))
    {
      try
      {
        con.Open();
        string query = "INSERT INTO signup (Name, DOB, Username, Password, ConfirmPassword)
" +
                    "VALUES (@full_name, @dob, @username, @password, @confirm_password);";
        SqlCommand cmd = new SqlCommand(query, con);
        cmd.Parameters.AddWithValue("@full_name", fullName);
        cmd.Parameters.AddWithValue("@dob", dob);
        cmd.Parameters.AddWithValue("@username", username);
        cmd.Parameters.AddWithValue("@password", password);
        cmd.Parameters.AddWithValue("@confirm_password", confirmPassword);
        int rowsAffected = cmd.ExecuteNonQuery();
        if (rowsAffected > 0)
        {
          // Data inserted successfully
          Session["SuccessMessage"] = "Registration successful";
          Response.Redirect("login.aspx");
        }
        else
        {
```

```
            Response.Write("Registration failed.");
        }
    }
    catch (Exception ex)
    {
        // Handle exception
        Response.Write("An error occurred: " + ex.Message);
    }
  }
}
```

Homepagelogout.aspx.cs:




```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;

public partial class homepage : System.Web.UI.Page
{
```

```
string connectionString;
protected void Page_Load(object sender, EventArgs e)
{
    connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString; // Initialize
connectionString in Page_Load
    if (!IsPostBack)
    {
        FetchDataFromDatabase();
    }
}
private void FetchDataFromDatabase()
{
    string query = "SELECT Name, Game, ID_Number, EventCollege,  College_name,EventDate
FROM participantdetails order by Id desc";
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        DataTable dt = new DataTable();
        SqlCommand cmd = new SqlCommand(query, conn);
        conn.Open();
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        da.Fill(dt);
        if (dt.Rows.Count > 0)
        {
            if (dt.Rows.Count == 1)
            {
                lblNamed.Text = dt.Rows[0]["Name"].ToString();
                lblGamed.Text = dt.Rows[0]["Game"].ToString();
                lblIDNumber.Text = dt.Rows[0]["ID_Number"].ToString();
                Label52.Text = dt.Rows[0]["College_name"].ToString();
                Labelcollegename.Text = dt.Rows[0]["EventCollege"].ToString();
```

```
            lblGameDate.Text =
Convert.ToDateTime(dt.Rows[0]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            lblName.Visible = true;
            lblGame.Visible = true;
            lblID.Visible = true;
            lblcollname.Visible = true;
            lblgameda.Visible = true;
            Label51.Visible = true;
        }
        else if (dt.Rows.Count == 2)
        {
            lblNamed.Text = dt.Rows[0]["Name"].ToString();
            lblGamed.Text = dt.Rows[0]["Game"].ToString();
            lblIDNumber.Text = dt.Rows[0]["ID_Number"].ToString();
            Label52.Text = dt.Rows[0]["College_name"].ToString();
            Labelcollegename.Text = dt.Rows[0]["EventCollege"].ToString();
            lblGameDate.Text =
Convert.ToDateTime(dt.Rows[0]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            lblName.Visible = true;
            lblGame.Visible = true;
            lblID.Visible = true;
            lblcollname.Visible = true;
            lblgameda.Visible = true;
            Label51.Visible = true;
            Label2.Text = dt.Rows[1]["Name"].ToString();
            Label4.Text = dt.Rows[1]["Game"].ToString();
            Label6.Text = dt.Rows[1]["ID_Number"].ToString();
            Label54.Text = dt.Rows[1]["College_name"].ToString();
            Label8.Text = dt.Rows[1]["EventCollege"].ToString();
```

```
            Label10.Text = Convert.ToDateTime(dt.Rows[1]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            Label1.Visible = true;
            Label3.Visible = true;
            Label5.Visible = true;
            Label7.Visible = true;
            Label9.Visible = true;
            Label53.Visible = true;
        }
        else if (dt.Rows.Count == 3)
        {
            lblNamed.Text = dt.Rows[0]["Name"].ToString();
            lblGamed.Text = dt.Rows[0]["Game"].ToString();
            lblIDNumber.Text = dt.Rows[0]["ID_Number"].ToString();
            Label52.Text = dt.Rows[0]["College_name"].ToString();
            Labelcollegename.Text = dt.Rows[0]["EventCollege"].ToString();
            lblGameDate.Text =
Convert.ToDateTime(dt.Rows[0]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            lblName.Visible = true;
            lblGame.Visible = true;
            lblID.Visible = true;
            lblcollname.Visible = true;
            lblgameda.Visible = true;
            Label51.Visible = true;
            Label2.Text = dt.Rows[1]["Name"].ToString();
            Label4.Text = dt.Rows[1]["Game"].ToString();
            Label6.Text = dt.Rows[1]["ID_Number"].ToString();
            Label54.Text = dt.Rows[1]["College_name"].ToString();
            Label8.Text = dt.Rows[1]["EventCollege"].ToString();
            Label10.Text = Convert.ToDateTime(dt.Rows[1]["EventDate"]).ToString("dd/MM/yyyy");
```

```
            //  Set visibility to true for the labels
            Label1.Visible = true;
            Label3.Visible = true;
            Label5.Visible = true;
            Label7.Visible = true;
            Label9.Visible = true;
            Label53.Visible = true;
            Label12.Text = dt.Rows[2]["Name"].ToString();
            Label14.Text = dt.Rows[2]["Game"].ToString();
            Label16.Text = dt.Rows[2]["ID_Number"].ToString();
            Label56.Text = dt.Rows[2]["College_name"].ToString();
            Label18.Text = dt.Rows[2]["EventCollege"].ToString();
            Label20.Text = Convert.ToDateTime(dt.Rows[2]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            Label11.Visible = true;
            Label13.Visible = true;
            Label15.Visible = true;
            Label17.Visible = true;
            Label19.Visible = true;
            Label55.Visible = true;
        }
        else if (dt.Rows.Count == 4)
        {
            lblNamed.Text = dt.Rows[0]["Name"].ToString();
            lblGamed.Text = dt.Rows[0]["Game"].ToString();
            lblIDNumber.Text = dt.Rows[0]["ID_Number"].ToString();
            Label52.Text = dt.Rows[0]["College_name"].ToString();
            Labelcollegename.Text = dt.Rows[0]["EventCollege"].ToString();
            lblGameDate.Text =
Convert.ToDateTime(dt.Rows[0]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
```

```
lblName.Visible = true;
lblGame.Visible = true;
lblID.Visible = true;
lblcollname.Visible = true;
lblgameda.Visible = true;
Label51.Visible = true;
Label2.Text = dt.Rows[1]["Name"].ToString();
Label4.Text = dt.Rows[1]["Game"].ToString();
Label6.Text = dt.Rows[1]["ID_Number"].ToString();
Label54.Text = dt.Rows[1]["College_name"].ToString();
Label8.Text = dt.Rows[1]["EventCollege"].ToString();
Label10.Text = Convert.ToDateTime(dt.Rows[1]["EventDate"]).ToString("dd/MM/yyyy");
//  Set visibility to true for the labels
Label1.Visible = true;
Label3.Visible = true;
Label5.Visible = true;
Label7.Visible = true;
Label9.Visible = true;
Label53.Visible = true;
Label12.Text = dt.Rows[2]["Name"].ToString();
Label14.Text = dt.Rows[2]["Game"].ToString();
Label16.Text = dt.Rows[2]["ID_Number"].ToString();
Label56.Text = dt.Rows[2]["College_name"].ToString();
Label18.Text = dt.Rows[2]["EventCollege"].ToString();
Label20.Text = Convert.ToDateTime(dt.Rows[2]["EventDate"]).ToString("dd/MM/yyyy");
//  Set visibility to true for the labels
Label11.Visible = true;
Label13.Visible = true;
Label15.Visible = true;
Label17.Visible = true;
```

```
            Label19.Visible = true;
            Label55.Visible = true;
            Label22.Text = dt.Rows[3]["Name"].ToString();
            Label24.Text = dt.Rows[3]["Game"].ToString();
            Label26.Text = dt.Rows[3]["ID_Number"].ToString();
            Label58.Text = dt.Rows[3]["College_name"].ToString();
            Label28.Text = dt.Rows[3]["EventCollege"].ToString();
            Label30.Text = Convert.ToDateTime(dt.Rows[3]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            Label21.Visible = true;
            Label23.Visible = true;
            Label25.Visible = true;
            Label27.Visible = true;
            Label29.Visible = true;
            Label57.Visible = true;
        }
        else if (dt.Rows.Count == 5)
        {
            lblNamed.Text = dt.Rows[0]["Name"].ToString();
            lblGamed.Text = dt.Rows[0]["Game"].ToString();
            lblIDNumber.Text = dt.Rows[0]["ID_Number"].ToString();
            Label52.Text = dt.Rows[0]["College_name"].ToString();
            Labelcollegename.Text = dt.Rows[0]["EventCollege"].ToString();
            lblGameDate.Text =
Convert.ToDateTime(dt.Rows[0]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            lblName.Visible = true;
            lblGame.Visible = true;
            lblID.Visible = true;
            lblcollname.Visible = true;
            lblgameda.Visible = true;
```

```
Label51.Visible = true;
Label2.Text = dt.Rows[1]["Name"].ToString();
Label4.Text = dt.Rows[1]["Game"].ToString();
Label6.Text = dt.Rows[1]["ID_Number"].ToString();
Label54.Text = dt.Rows[1]["College_name"].ToString();
Label8.Text = dt.Rows[1]["EventCollege"].ToString();
Label10.Text = Convert.ToDateTime(dt.Rows[1]["EventDate"]).ToString("dd/MM/yyyy");
//  Set visibility to true for the labels
Label1.Visible = true;
Label3.Visible = true;
Label5.Visible = true;
Label7.Visible = true;
Label9.Visible = true;
Label53.Visible = true;
Label12.Text = dt.Rows[2]["Name"].ToString();
Label14.Text = dt.Rows[2]["Game"].ToString();
Label16.Text = dt.Rows[2]["ID_Number"].ToString();
Label56.Text = dt.Rows[2]["College_name"].ToString();
Label18.Text = dt.Rows[2]["EventCollege"].ToString();
Label20.Text = Convert.ToDateTime(dt.Rows[2]["EventDate"]).ToString("dd/MM/yyyy");
//  Set visibility to true for the labels
Label11.Visible = true;
Label13.Visible = true;
Label15.Visible = true;
Label17.Visible = true;
Label19.Visible = true;
Label55.Visible = true;
Label22.Text = dt.Rows[3]["Name"].ToString();
Label24.Text = dt.Rows[3]["Game"].ToString();
Label26.Text = dt.Rows[3]["ID_Number"].ToString();
```

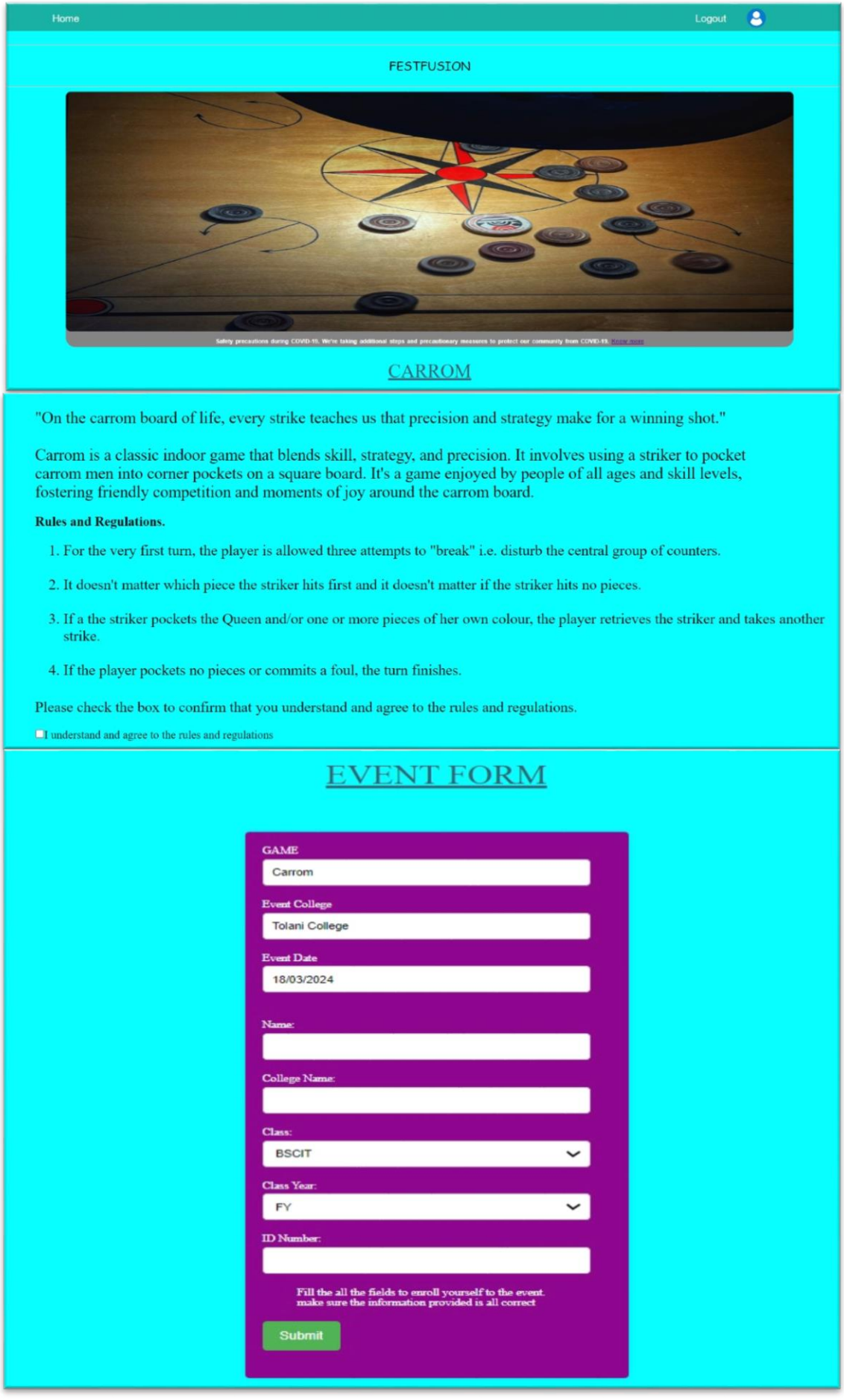
```
    Label58.Text = dt.Rows[3]["College_name"].ToString();
    Label28.Text = dt.Rows[3]["EventCollege"].ToString();
    Label30.Text = Convert.ToDateTime(dt.Rows[3]["EventDate"]).ToString("dd/MM/yyyy");
    //  Set visibility to true for the labels
    Label21.Visible = true;
    Label23.Visible = true;
    Label25.Visible = true;
    Label27.Visible = true;
    Label29.Visible = true;
    Label57.Visible = true;
    Label32.Text = dt.Rows[4]["Name"].ToString();
    Label34.Text = dt.Rows[4]["Game"].ToString();
    Label36.Text = dt.Rows[4]["ID_Number"].ToString();
    Label60.Text = dt.Rows[4]["College_name"].ToString();
    Label38.Text = dt.Rows[4]["EventCollege"].ToString();
    Label40.Text = Convert.ToDateTime(dt.Rows[4]["EventDate"]).ToString("dd/MM/yyyy");
    //  Set visibility to true for the labels
    Label31.Visible = true;
    Label33.Visible = true;
    Label35.Visible = true;
    Label37.Visible = true;
    Label39.Visible = true;
    Label59.Visible = true;
}
else
{
    lblNamed.Text = dt.Rows[0]["Name"].ToString();
    lblGamed.Text = dt.Rows[0]["Game"].ToString();
    lblIDNumber.Text = dt.Rows[0]["ID_Number"].ToString();
    Label52.Text = dt.Rows[0]["College_name"].ToString();
```

```
            Labelcollegename.Text = dt.Rows[0]["EventCollege"].ToString();
             lblGameDate.Text =
Convert.ToDateTime(dt.Rows[0]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            lblName.Visible = true;
            lblGame.Visible = true;
            lblID.Visible = true;
            lblcollname.Visible = true;
            lblgameda.Visible = true;
            Label51.Visible = true;
            Label2.Text = dt.Rows[1]["Name"].ToString();
            Label4.Text = dt.Rows[1]["Game"].ToString();
            Label6.Text = dt.Rows[1]["ID_Number"].ToString();
            Label54.Text = dt.Rows[1]["College_name"].ToString();
            Label8.Text = dt.Rows[1]["EventCollege"].ToString();
            Label10.Text = Convert.ToDateTime(dt.Rows[1]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            Label1.Visible = true;
            Label3.Visible = true;
            Label5.Visible = true;
            Label7.Visible = true;
            Label9.Visible = true;
            Label53.Visible = true;
            Label12.Text = dt.Rows[2]["Name"].ToString();
            Label14.Text = dt.Rows[2]["Game"].ToString();
            Label16.Text = dt.Rows[2]["ID_Number"].ToString();
            Label56.Text = dt.Rows[2]["College_name"].ToString();
            Label18.Text = dt.Rows[2]["EventCollege"].ToString();
            Label20.Text = Convert.ToDateTime(dt.Rows[2]["EventDate"]).ToString("dd/MM/yyyy");
            //  Set visibility to true for the labels
            Label11.Visible = true;
```

```
Label13.Visible = true;
Label15.Visible = true;
Label17.Visible = true;
Label19.Visible = true;
Label55.Visible = true;
Label22.Text = dt.Rows[3]["Name"].ToString();
Label24.Text = dt.Rows[3]["Game"].ToString();
Label26.Text = dt.Rows[3]["ID_Number"].ToString();
Label58.Text = dt.Rows[3]["College_name"].ToString();
Label28.Text = dt.Rows[3]["EventCollege"].ToString();
Label30.Text = Convert.ToDateTime(dt.Rows[3]["EventDate"]).ToString("dd/MM/yyyy");
//  Set visibility to true for the labels
Label21.Visible = true;
Label23.Visible = true;
Label25.Visible = true;
Label27.Visible = true;
Label29.Visible = true;
Label57.Visible = true;
Label32.Text = dt.Rows[4]["Name"].ToString();
Label34.Text = dt.Rows[4]["Game"].ToString();
Label36.Text = dt.Rows[4]["ID_Number"].ToString();
Label60.Text = dt.Rows[4]["College_name"].ToString();
Label38.Text = dt.Rows[4]["EventCollege"].ToString();
Label40.Text = Convert.ToDateTime(dt.Rows[4]["EventDate"]).ToString("dd/MM/yyyy");
// Set visibility to true for the labels
Label31.Visible = true;
Label33.Visible = true;
Label35.Visible = true;
Label37.Visible = true;
Label39.Visible = true;
```

```
                Label59.Visible = true;
                Label42.Text = dt.Rows[5]["Name"].ToString();
                Label44.Text = dt.Rows[5]["Game"].ToString();
                Label46.Text = dt.Rows[5]["ID_Number"].ToString();
                Label62.Text = dt.Rows[4]["College_name"].ToString();
                Label48.Text = dt.Rows[5]["EventCollege"].ToString();
                Label50.Text = Convert.ToDateTime(dt.Rows[5]["EventDate"]).ToString("dd/MM/yyyy");
                // Set visibility to true for the labels
                Label41.Visible = true;
                Label43.Visible = true;
                Label45.Visible = true;
                Label47.Visible = true;
                Label49.Visible = true;
                Label61.Visible = true;
            }
        }
        else
        {
            conn.Close();
        }
    }
  }
}
```

Form.aspx.cs


Home
Logout
FESTFUSION
Safety precautions during COVID-19. We're taking additional steps and precautionary measures to protect our community from COVID-19. Know more
CARROM
"On the carrom board of life, every strike teaches us that precision and strategy make for a winning shot."
Carrom is a classic indoor game that blends skill, strategy, and precision. It involves using a striker to pocket carrom men into corner pockets on a square board. It's a game enjoyed by people of all ages and skill levels, fostering friendly competition and moments of joy around the carrom board.
Rules and Regulations.
1. For the very first turn, the player is allowed three attempts to "break" i.e. disturb the central group of counters.
2. It doesn't matter which piece the striker hits first and it doesn't matter if the striker hits no pieces.
3. If a the striker pockets the Queen and/or one or more pieces of her own colour, the player retrieves the striker and takes another strike.
4. If the player pockets no pieces or commits a foul, the turn finishes.
Please check the box to confirm that you understand and agree to the rules and regulations.
I understand and agree to the rules and regulations
EVENT FORM
GAME
Carrom
Event College
Tolani College
Event Date
18/03/2024
Name:
College Name:
Class:
BSCIT
Class Year:
FY
ID Number:
Fill the all the fields to enroll yourself to the event.
make sure the information provided is all correct
Submit

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data.SqlClient;

public partial class carromform : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        {
            if (!IsPostBack)
            {
                // Fetch game name from the database and populate the input field
                string gameName = GetGameNameFromDatabase();
                txtGame.Value = gameName;
                string collegeName = GetCollegeNameFromDatabase();
                Txtcollege.Value = collegeName;

                string eventDate = GetEventDateFromDatabase();
                TxtEventDate.Value = eventDate;
            }
        }
    }
    // Method to fetch the game name from the database
    private string GetGameNameFromDatabase()
    {
```

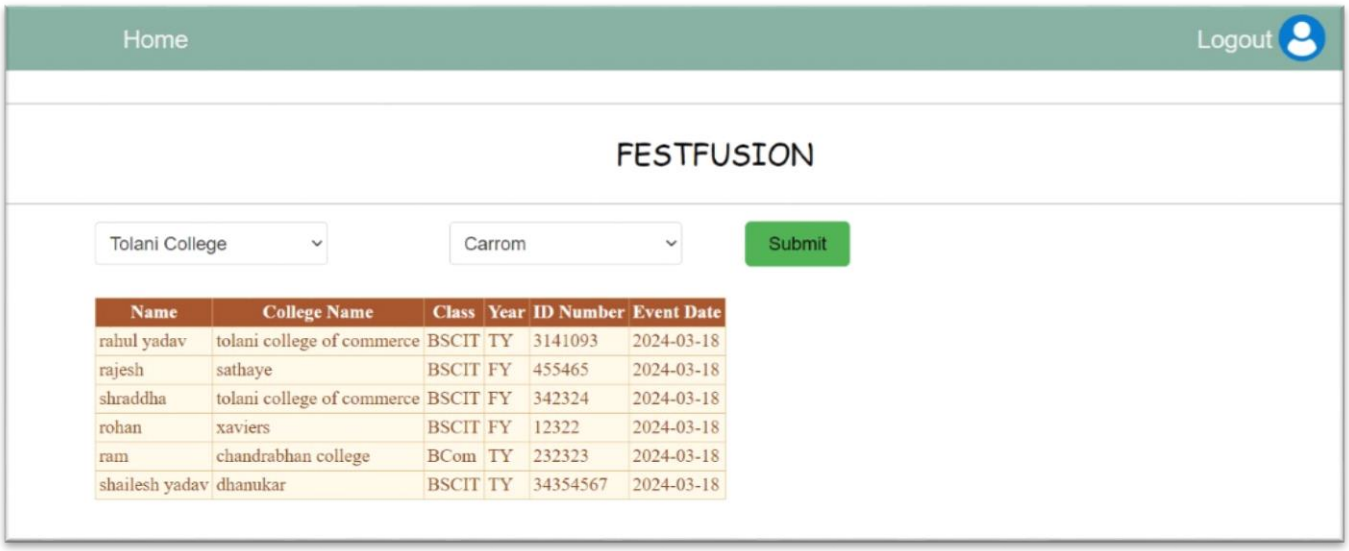
```
        string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
        string gameName = "";
        using (SqlConnection connection = new SqlConnection(connectionString))
        {
            SqlCommand command = new SqlCommand("SELECT Game FROM Gamename WHERE ID =
@ID", connection);
            command.Parameters.AddWithValue("@ID", 1); // Assuming game ID for "Carrom" is 1
            connection.Open();
            SqlDataReader reader = command.ExecuteReader();
            if (reader.Read())
            {
                gameName = reader["Game"].ToString();
            }
        }
        return gameName;
    }
    private string GetCollegeNameFromDatabase()
    {
        string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
        string collegeName = "";

        using (SqlConnection connection = new SqlConnection(connectionString))
        {
            SqlCommand command = new SqlCommand("SELECT College FROM Collegename WHERE
ID = @ID", connection);
            command.Parameters.AddWithValue("@ID", 2); // Assuming college ID for "Example College"
is 1
            connection.Open();
            SqlDataReader reader = command.ExecuteReader();
            if (reader.Read())
```

```
            {
                collegeName = reader["College"].ToString();
            }
        }
        return collegeName;
    }
    private string GetEventDateFromDatabase()
    {
        string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
        string eventDate = "";
        using (SqlConnection connection = new SqlConnection(connectionString))
        {
            SqlCommand command = new SqlCommand("SELECT EventDate FROM Collegename
WHERE ID = @ID", connection);
            command.Parameters.AddWithValue("@ID", 2); // Assuming the ID for the relevant record
            connection.Open();
            SqlDataReader reader = command.ExecuteReader();
            if (reader.Read())
            {
                eventDate = Convert.ToDateTime(reader["EventDate"]).ToString("dd/MM/yyyy");
            }
        }
        return eventDate;
    }
    private void ClearFormFields()
    {
        // Clear values of form fields
        txtName.Value = "";
        txtCollegeName.Value = "";
        ddlClass.SelectedIndex = 0;
```

```
        ddlClassYear.SelectedIndex = 0;
        txtIDNumber.Value = "";
    }
    protected void btnSubmit_Click(object sender, EventArgs e)
    {
        // This method is invoked when the submit button is clicked
        // Retrieve values entered by the user
        string game = txtGame.Value;
        string name = txtName.Value;
        string collegeName = txtCollegeName.Value;
        DateTime Eventdate = DateTime.ParseExact(TxtEventDate.Value, "dd/MM/yyyy", null);
       // string Eventdate =Convert.ToDateTime(TxtEventDate.Value).ToString("dd/MM/yyyy");
        string className = ddlClass.SelectedValue;
        string classYear = ddlClassYear.SelectedValue;
        string idNumber = txtIDNumber.Value;
        string eventcollege = Txtcollege.Value;
        // Your database connection string
        string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
        // Your SQL query to insert data into the database
        string query = @"INSERT INTO participantdetails (Game, Name, College_Name, Class,
Class_Year, ID_Number,EventDate,CreatedOn,EventCollege)
                        VALUES (@Game, @Name, @CollegeName, @ClassName, @ClassYear,
@IDNumber,  @eventdate, @Createdon,@eventcollege)";
        try
        {
            // Establish connection to the database and execute the query
            using (SqlConnection connection = new SqlConnection(connectionString))
            {
                SqlCommand command = new SqlCommand(query, connection);
                command.Parameters.AddWithValue("@Game", game);
```

```
        command.Parameters.AddWithValue("@Name", name);
        command.Parameters.AddWithValue("@CollegeName", collegeName);
        command.Parameters.AddWithValue("@ClassName", className);
        command.Parameters.AddWithValue("@ClassYear", classYear);
        command.Parameters.AddWithValue("@IDNumber", idNumber);
        command.Parameters.AddWithValue("@eventdate", Eventdate);
        command.Parameters.AddWithValue("@Createdon", DateTime.Now);
        command.Parameters.AddWithValue("@eventcollege", eventcollege);
        connection.Open();
        int rowsAffected = command.ExecuteNonQuery();
        if (rowsAffected > 0)
        {
            // Insert successful
            Response.Write("<script>alert('Data saved successfully!');</script>");
            ClearFormFields(); // Clear form fields after successful submission
        }
        else
        {
            // Insert failed
            Response.Write("<script>alert('Failed to save data. Please try again.');</script>");
        }
    }
}
catch (Exception ex)
{
    // Log the exception details for debugging purposes
    System.Diagnostics.Trace.WriteLine("An error occurred: " + ex.ToString());
    // Display the detailed error message to the user
    Response.Write("<script>alert('An unexpected error occurred: " + ex.Message + "');</script>");
} } }
```

Admin.aspx.cs



Name	College Name	Class	Year	ID Number	Event Date
rahul yadav	tolani college of commerce	BSCIT	TY	3141093	2024-03-18
rajesh	sathaye	BSCIT	FY	455465	2024-03-18
shraddha	tolani college of commerce	BSCIT	FY	342324	2024-03-18
rohan	xaviers	BSCIT	FY	12322	2024-03-18
ram	chandrabhan college	BCom	TY	232323	2024-03-18
shailesh yadav	dhanukar	BSCIT	TY	34354567	2024-03-18

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;

public partial class adminpage : System.Web.UI.Page
{
    string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            BindGrid();
```

```
    }
}
protected void ddlEventCollege_SelectedIndexChanged(object sender, EventArgs e)
{
    // You can add logic here if needed
}
protected void ddlGame_SelectedIndexChanged(object sender, EventArgs e)
{
    // You can add logic here if needed
}
protected void btnSubmit_Click(object sender, EventArgs e)
{
    BindGrid();
}
private void BindGrid()
{
    string college = ddlEventCollege.SelectedValue;
    string game = ddlGame.SelectedValue;
    if (!string.IsNullOrEmpty(college) && !string.IsNullOrEmpty(game))
    {
        using (SqlConnection connection = new SqlConnection(connectionString))
        {
            string query = "SELECT Name, College_name, Class, class_year, Id_number, EventDate " +
                           "FROM participantdetails " +
                           "WHERE EventCollege = @College AND Game = @Game";
            SqlCommand command = new SqlCommand(query, connection);
            command.Parameters.AddWithValue("@College", college);
            command.Parameters.AddWithValue("@Game", game);
            SqlDataAdapter adapter = new SqlDataAdapter(command);
            DataTable dataTable = new DataTable();
```

```
            try
            {
                connection.Open();
                adapter.Fill(dataTable);
                // Bind data to GridView
                GridView1.DataSource = dataTable;
                GridView1.DataBind();
                if (dataTable.Rows.Count == 0)
                {
                    lblMessage.Text = "No records found.";
                }
                else
                {
                    lblMessage.Text = string.Empty;
                }
            }
            catch (Exception ex)
            {
                Response.Write("An error occurred: " + ex.Message);
            }
        }
    }
}
```

Testing Approach :

Testing is a crucial phase in software development aimed at identifying defects, errors, or bugs in a software application. It involves executing the software system or its components with the intent of finding any discrepancies between expected and actual results. The primary goals of testing are to ensure the software meets its requirements, functions correctly, and performs reliably under various conditions.

There are several types of testing :-

1. **Unit Testing**: Unit testing verifies the correctness of individual units of code, like functions or classes, by isolating them and subjecting them to various inputs and scenarios. It helps in early detection of defects, facilitates code maintenance, and ensures reliable software development.

2. **Integration Testing**: Integration testing in software quality analysis examines the interaction between software modules to ensure they function together seamlessly, identifying and resolving any issues that arise from their integration.

3. **System Testing**: System testing verifies the entire software system's functionality, performance, and reliability to ensure it meets specified requirements and functions correctly in its intended environment.

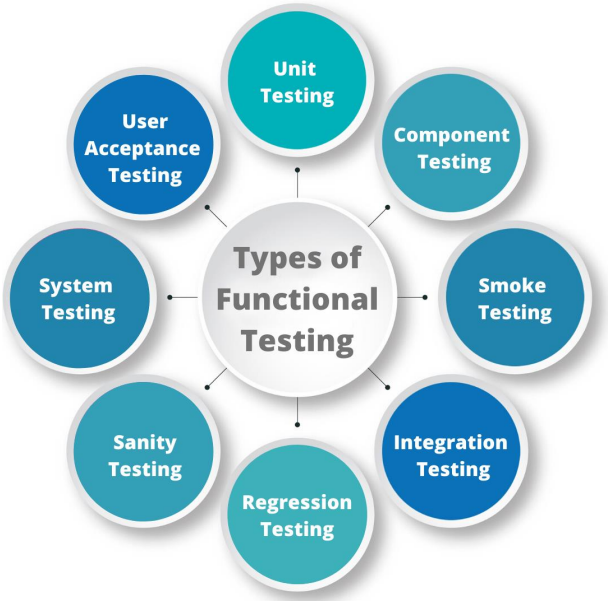
4. **Acceptance Testing**: Acceptance testing in software quality analysis verifies whether a system meets specified requirements and is suitable for deployment, typically conducted by end-users or clients to validate functionality and adherence to business needs prior to production release.

5. **Regression Testing**: Testing performed to ensure that recent changes or modifications to the software have not adversely affected existing functionality. Regression tests help detect and prevent regressions or unintended side effects introduced during development.

6. **Performance Testing**: Performance testing in software quality analysis refers to the evaluation of a system's responsiveness, scalability, and stability under varying workloads. It involves simulating real-world conditions to assess how efficiently a software application performs under stress, ensuring it meets expected performance criteria and can handle user demands effectively.

7. **Security Testing**: Security testing is the process of evaluating software systems to uncover vulnerabilities and weaknesses that could be exploited by malicious actors. It aims to ensure the confidentiality, integrity, and availability of data and resources within the system.

8. **Usability Testing**: Usability testing in software quality analysis assesses the ease with which users can interact with a software product. It involves observing real users performing tasks to identify usability issues and improve the overall user experience.



White box testing :

White box testing techniques analyse the internal structures the used data structures, internal design, code structure, and the working of the software rather than just the functionality as in black box testing. It is also called glass box testing or clear box testing or structural testing. White Box Testing is also known as transparent testing or open box testing.

White box testing is a software testing technique that involves testing the internal structure and workings of a software application. The tester has access to the source code and uses this knowledge to design test cases that can verify the correctness of the software at the code level.

White box testing is also known as structural testing or code-based testing, and it is used to test the software's internal logic, flow, and structure. The tester creates test cases to examine the code paths and logic flows to ensure they meet the specified requirements.

Process of White Box Testing

1. Input: Requirements, Functional specifications, design documents, source code.
2. Processing: Performing risk analysis to guide through the entire process.
3. Proper test planning: Designing test cases to cover the entire code. Execute rinse-repeat until error-free software is reached. Also, the results are communicated.
4. Output: Preparing final report of the entire testing process.

7.2 Levels Of Testing :

1. Unit Testing:

This level of testing involves testing individual units or components of the software in isolation, typically at the code level. Unit tests verify the correctness of small units of code, such as functions or methods, and are often automated to ensure rapid feedback during development.

Test ID	Test scenario	Value	Expected Result	Actual Result	Pass/fail
A1	Splash Screen	True	Working	Forward to next activity	Pass
A2	User Login	Username=Rahul Password = 12345	Working	Pass on to the Select options.	Pass
A3	Admin Login	Username= Rohan Password= 123	Working	Pass on to the Select options.	Pass
A4	Database Connectivity	Data Store	Data To Be Stored	Information Is Stored	Pass

- I have checked each and every unit thoroughly and resolved the errors which arrived during the development of the unit in the project. The login section is divided into admin phase and User phase. Each having their own characteristics.

2. Integrated Testing :

Integration testing is defined as the testing of combined parts of an application to determine if they function correctly.

Test Case ID	Test scenario	Value	Expected Result	Actual Result	Pass/fail
B1	College image button 1	To Pop message for login and redirect to login page.	Working	Landed on Login page Successfully	Pass
B2	College image button 2	To View college events details.	Working	Viewing Successfully	Pass
B3	Event hyperlink	To View the event details.	Working	Viewing Successfully.	Pass
B4	Form Submit button	To Submit user detail.	Working	Submit Successfully.	Pass
B5	Pass generate button	To Generate pass	Working	Pass Generate Successfully.	Pass
B6	Download Brochure button	To Download the College event brochure	Working	Brochure Downloaded Successfully.	Pass

- When all the units are combined and tested together, then the workflow of the project is understood. The login credentials when matched, navigates to the corresponding pages for both User as well as admin. The data searched by the User is fetched in the Database for getting the outcome.

Beta Testing :

Beta testing is a stage in software evaluation where a preliminary version of the software is provided to a chosen set of external users or clients, referred to as beta testers, for assessment and comments.

The main objective of beta testing is to collect practical feedback from users across various settings and usage conditions. This enables developers to pinpoint possible issues, acquire insights into usability, and implement required enhancements prior to the formal release. Beta testing is crucial for verifying the software's features, user-friendliness, and overall quality from the perspective of the end user, ensuring a more seamless and successful introduction of the final product.

Acceptance Testing :

Acceptance testing is a stage in software evaluation where the software is assessed to confirm if it aligns with the acceptance criteria and fulfills the requirements set by the stakeholders or users. This usually entails testing the software in an environment that mirrors production, using real-world data and scenarios, with an emphasis on validating that the software aligns with business goals and user expectations.

Acceptance testing can come in various formats, such as user acceptance testing (UAT), business acceptance testing (BAT), and operational acceptance testing (OAT), depending on the stakeholders and the specific objectives of the testing phase. The primary goal of acceptance testing is to guarantee that the software is prepared for deployment and caters to the requirements of its target users, ultimately securing their endorsement for release.

System Testing :

System testing is an extensive phase of software evaluation where the complete software system is assessed in its entirety to confirm its correct functioning and adherence to specified criteria. This entails testing the unified system against the established functional and non-functional requirements to authenticate its performance, reliability, and security.

System testing covers various facets, including functional testing to validate the system's features and functionalities, performance testing to gauge its capacity and responsiveness, security testing to pinpoint vulnerabilities and potential risks, and usability testing to assess the user experience. By examining the system as a whole, system testing strives to detect any defects or inconsistencies that might result from interactions among its components and to ensure the software's preparedness for deployment in its designated environment.

7.3 Test Cases

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Remark
TC1	Splash Screen					
TC2	Check Login of Admin	Enter admin Username, Password and Click on Login	Username = Rahul Password = 12345	Admin should be logged in	Login Successfully	Pass
TC3	Check Login of User	Enter Employee Username, Password and Click on Login	Username = Rohan Password = 123	User should be logged in	Login Successfully	Pass
TC4	SQL Query	Insert, Update, Delete	The data is taken.	Showing data.	Visible data successfully	Pass

COST ESTIMATION

Cost estimation models are used in various industries to predict the expenses associated with a project, product, or service. Overall, effective cost estimation is critical for project planning, budgeting, and decision-making, helping organizations allocate resources efficiently and manage risks effectively.

The Development Mode:

In the COCOMO (Constructive Cost Model) framework, which is a widely used model for estimating software development effort, duration, and cost, there are three modes of development: Organic, Semidetached, and Embedded. These modes represent different levels of complexity and risk associated with software projects. Each mode has its own set of characteristics, development processes, and cost estimating relationships. Here's a brief overview of each mode:

1. **Organic Mode:**

- This mode applies to projects that are relatively small, familiar, and well-understood by the development team. The team is cohesive, and communication is typically straightforward.
- Characteristics include a small team size, a low level of required innovation, and a well-defined set of requirements.
- Cost estimating relationships for projects in the Organic mode generally yield lower cost estimates compared to projects in the other modes.

2. **Semidetached Mode:**

- Semidetached mode is applicable to projects that fall between the Organic and Embedded modes in terms of complexity and risk.
- These projects may involve some elements of novelty or innovation, and the team may require some additional coordination and communication compared to Organic projects.
- Characteristics include moderate team size, moderate innovation, and requirements that may evolve over the course of the project.
- Cost estimating relationships for projects in the Semidetached mode typically result in higher cost estimates compared to Organic projects but lower than Embedded projects.

3. **Embedded Mode:**

- Embedded mode is suitable for projects that are large, complex, and involve significant technical challenges or constraints. These projects often require coordination with external entities and may have stringent regulatory or quality requirements.
- Characteristics include large team size, high innovation, and requirements that are often volatile and subject to frequent changes.
- Cost estimating relationships for projects in the Embedded mode generally yield the highest cost estimates due to the increased complexity, risk, and resource requirements associated with such projects.

Basic model:

The basic COCOMO equations take the form

Effort Applied (E) = ab(KLOC)bb**[person-months]**

Development Time (D) = cb(Effort Applied)db**[months]**

People Required (P) = Effort Applied / Development time **[count]**

1) **Compute the count-total which will be used to define the complexity of a project.**

Information Domain Values

Measurement Parameter	Count		Simple ○	Average ◉	Complex ○		Total
Number of user inputs	18	X	3	4	6	=	72.00
Number of user outputs	15	X	4	5	7	=	75.00
Number of user inquiries	0	X	3	4	6	=	.0
Number of files	0	X	7	10	15	=	.0
Number of external interfaces	0	X	5	7	10	=	.0
Count=Total							147.00

Count Total

2) **Find the complexity adjustment values based on responses to the questions.**

Complexity Weighting Factors

// heading of the second table Rate each factor on a scale of 0 to 5:

(0 = No influence, 1 = Incidental, 2 = Moderate, 3 = Average, 4 = Significant, 5 = Essential):

Question	0	1	2	3	4	5
1. Does the system require reliable backup and recovery?	○	○	○	○	○	◉
2. Are data communications required?	◉	○	○	○	○	○
3. Are there distributed processing functions?	○	○	○	○	◉	○
4. Is performance critical?	○	○	○	◉	○	○
5. Will the system run in an existing, heavily utilized operational environment?	◉	○	○	○	○	○
6. Does the system require on-line data entry?	○	○	○	○	◉	○
7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?	◉	○	○	○	○	○
8. Are the master file updated on-line?	○	○	◉	○	○	○
9. Are the inputs, outputs, files, or inquiries complex?	◉	○	○	○	○	○
10. Is the internal processing complex?	○	○	◉	○	○	○
11. In the code designed to be reusable?	○	○	○	○	◉	○
12. Are conversion and installation included in the design?	○	○	○	◉	○	○
13. Is the system designed for multiple installations in different organizations?	○	○	○	◉	○	○
14. Is the application designed to facilitate change and ease of use by the user?	○	○	○	○	◉	○
Total 34.00						

Show Total of weighting Factor

The Function Points is: Show Function Points 145.53

3) **Select a programming language used in the project.**

Programming Language	LOC/FP (average)	Select
Assembly Language	320	○
C	128	○
COBOL	105	○
Fortran	105	○
Pascal	90	○
Ada	70	○
Object-Oriented Languages	30	○
Fourth Generation Languages (4GLs)	20	○
Code Generators	15	○
Spreadsheets	6	○
Graphical Languages (icons)	4	◉

LOC/F P: Show LOC/FP 873.18

4) **Select complexity of the software project.**

Software Project	a_b	b_b	c_b	d_b	Select
Organic	2.4	1.05	2.5	0.38	◉
Semi-detached	3.0	1.12	2.5	0.35	○
Embedded	3.6	1.20	2.5	0.32	○

Calculate Effort and Duration

Effort (E) = $a_b(KLOC)^b_b$ = 1.36 Duration (D) = $c_b(E)^d_b$ = 2.81

Man-month = Unadjusted Function Point (UFP) / 18

= 145 / 18

= 8

Average Programmer is paid Rs. 5000 per month Total number of programmers: 2

Cost per month = Average Programmer cost * No of programmers

= 5000 * 2

= Rs.10,000 per month

Total cost of Project = Cost per month * man-month

= 10000 * 8

= Rs. 80,000

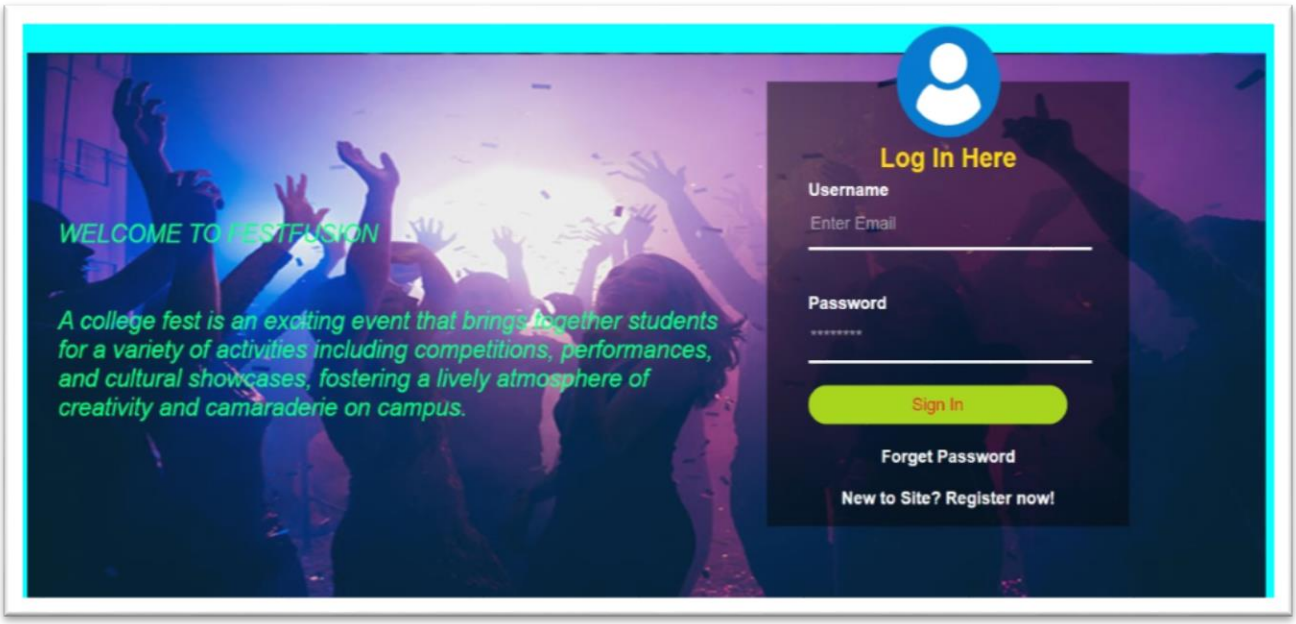
RESULT AND DISCUSSION

9.1 Test Report

Test cases are detailed scenarios or conditions under which a system application is tested to ensure that it functions correctly and meets its requirements. Each test case typically consists of inputs, expected outputs, and execution conditions. Test cases are designed to verify that the software behaves as expected under various circumstances and to identify any defects or bugs in the system.

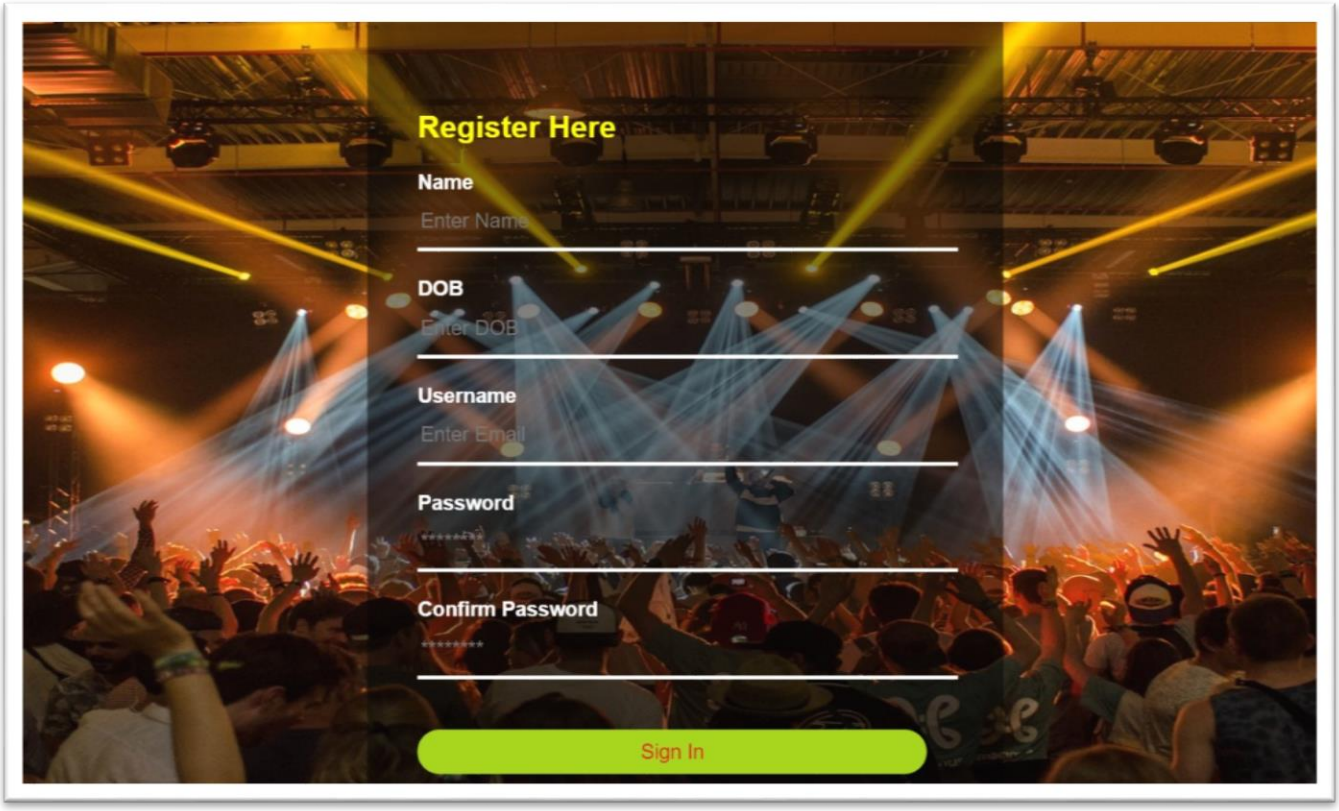
The Database is tested by adding, inserting, updating and deletion of the record. The Data fetching process is also considered during the applications.

A) Student and Admin Login:



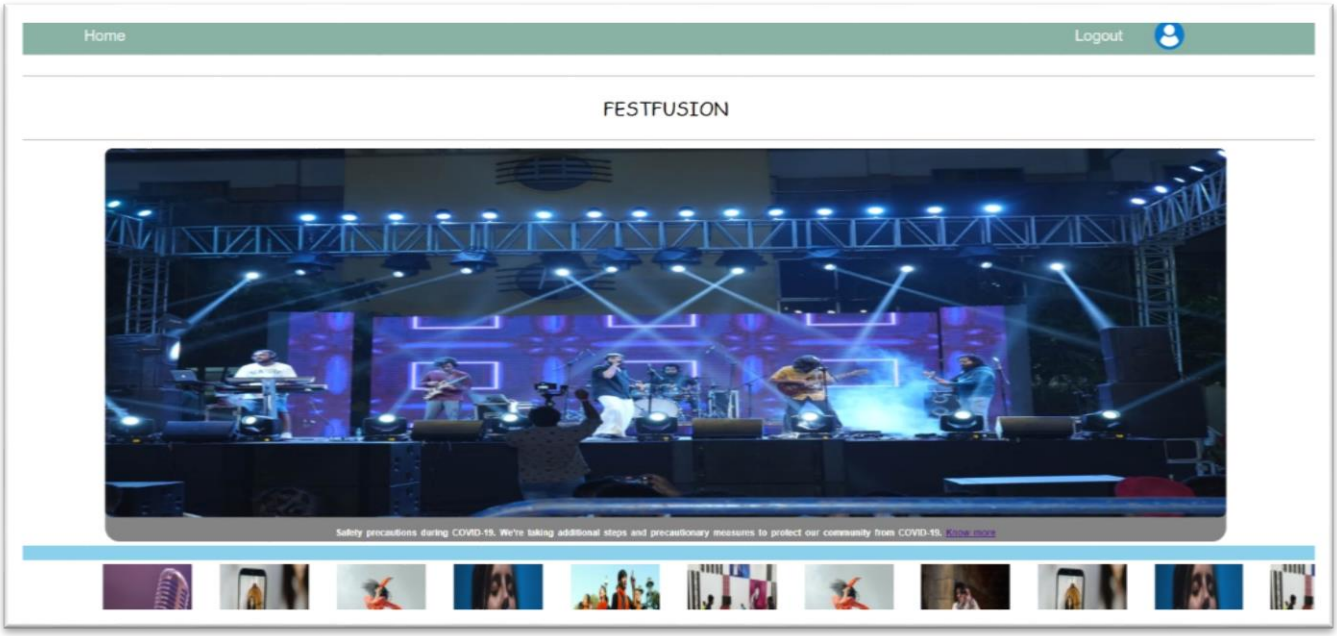
- This is the login page for both Participant and Admin where they have to entered their username and password.

B) Student Signup:



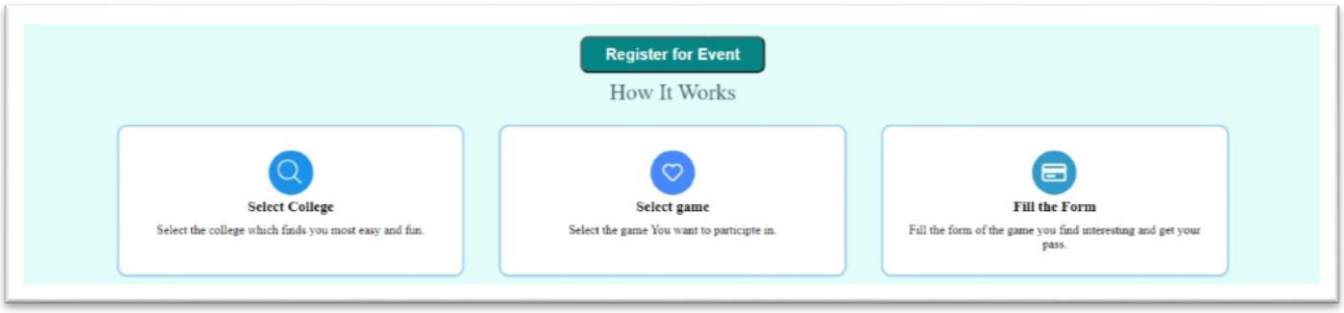
❖ This is the Signup page for Participants where they has to register themselve first by filling the details.

C) Home Page :



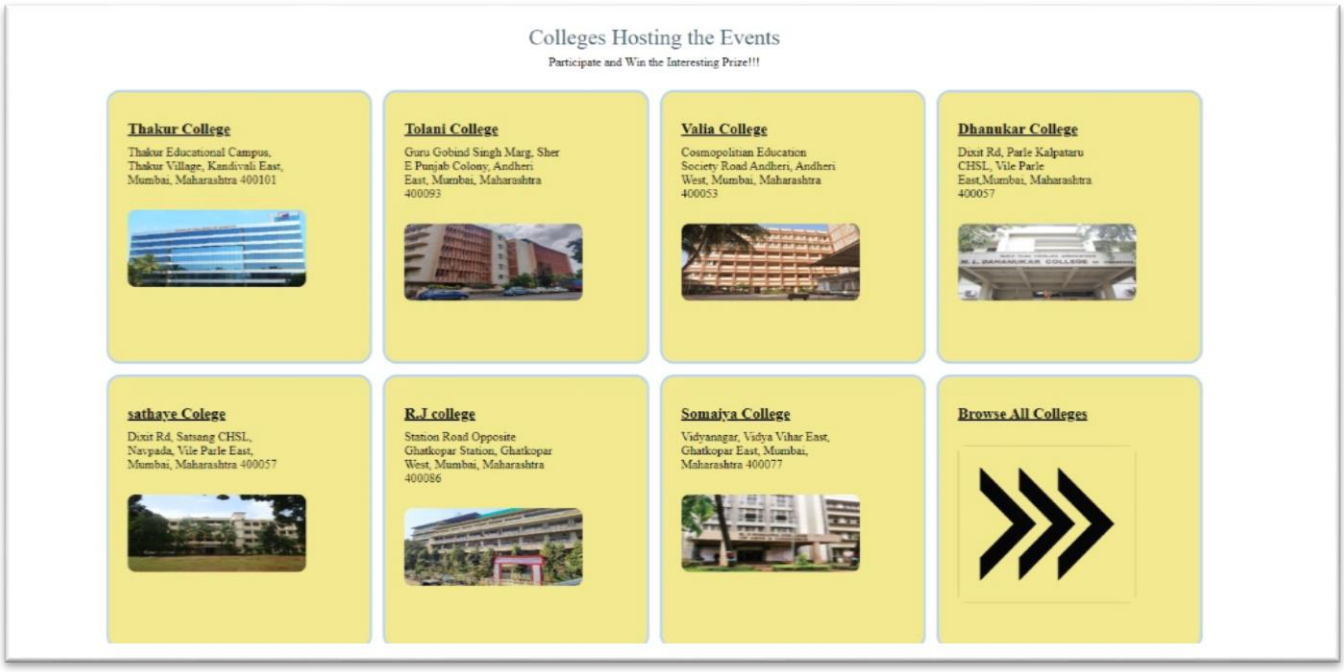
❖ This is our Home Page after entering correct Participant's Username and Password.

D) Addressing Participant Section :



❖ Here, a Partcipant can check the steps registering for an event.

E) College Details Section :



❖ We can view different colleges who are organizing their fest.

F) Generated Passes Section :

❖ After registering for an event the user can see the generated passes from here.

G) Review Section :

❖ We can see the reviews of participants who have visit the college fest.

H) Upcoming Event Section :

- Here we can see the Upcoming Events of different college.

I) Particular College Event Section :

- After clicking on particular college icon it will take you to this page. And you will find more details about the college and its fest.

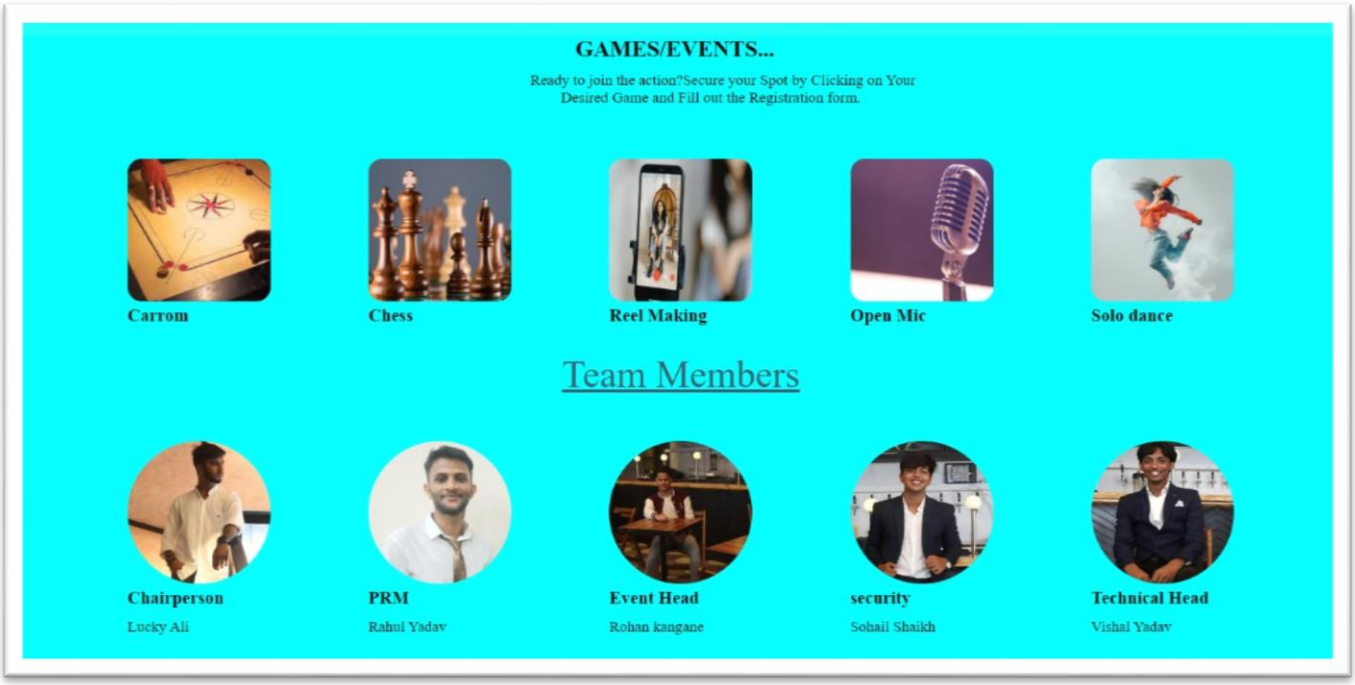
J) Fest Detail Section :

- You will find more details about the fest and you can download brochure from this section.

K) About College Section :

- In this Section, partcipants can see the detail of the college.

L) Games and Team Members Section :

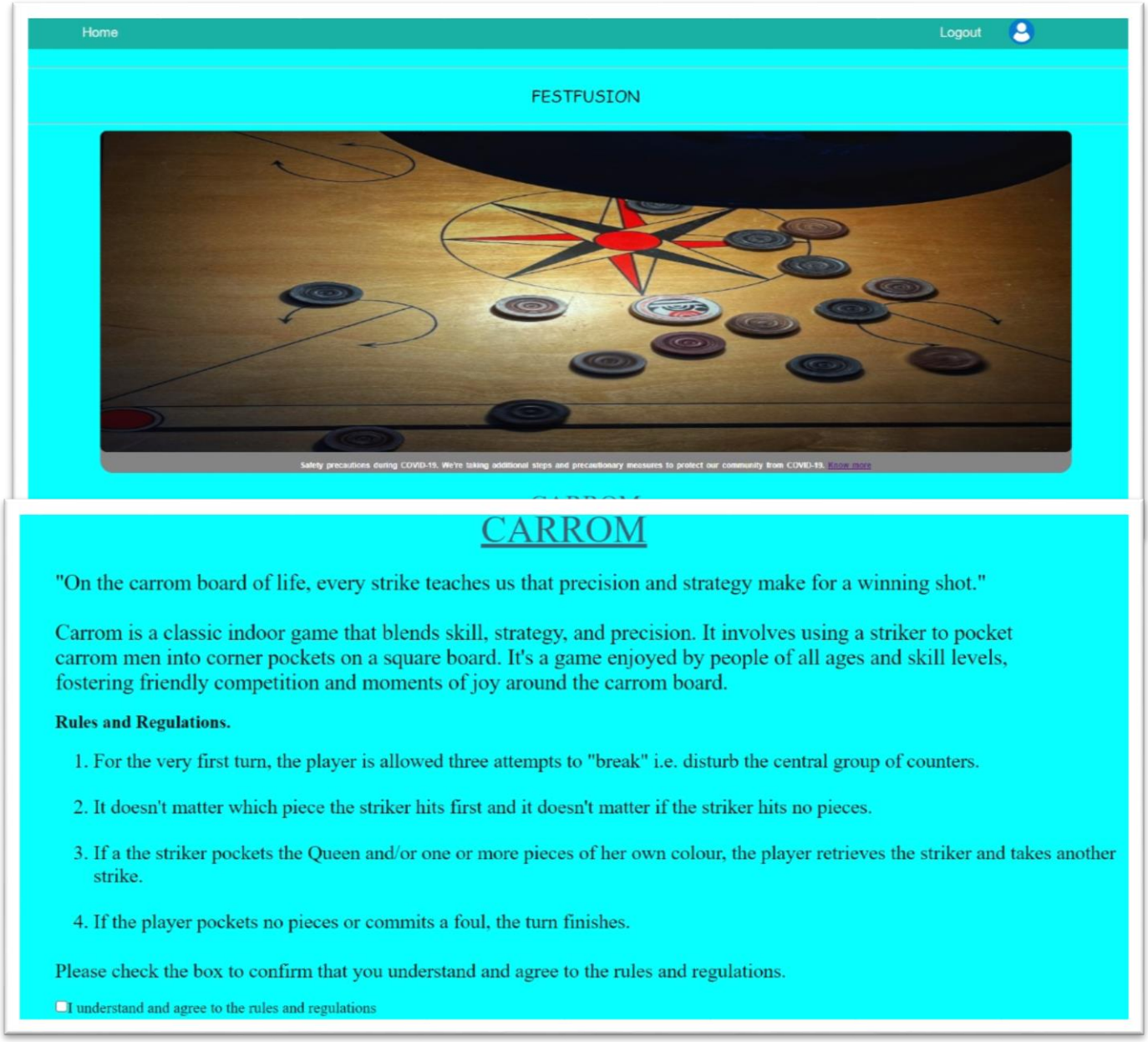


- In this section participant can see the different games of the event and the team member who are organizing the event.

M) Sponsors Section :

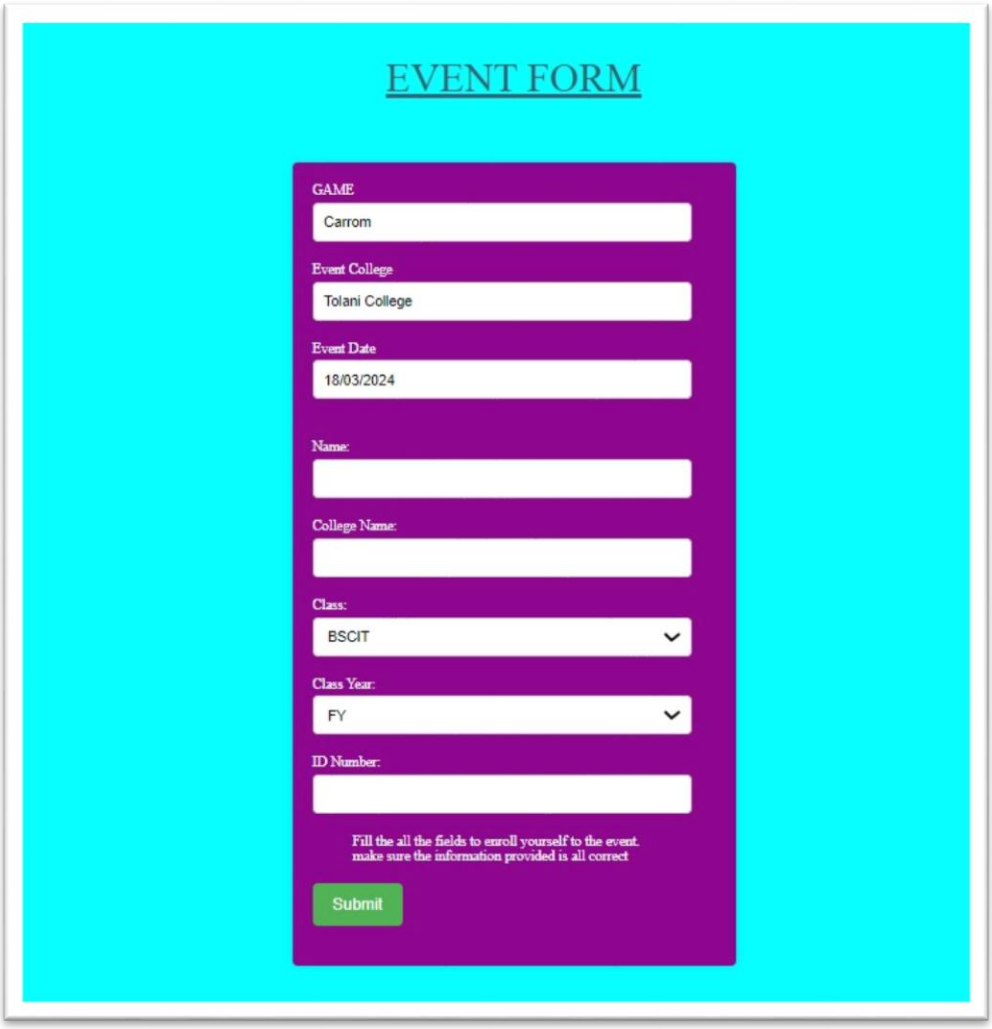
- In this section we can see the sponsor of the events.

N) Particular Game Detail Section :



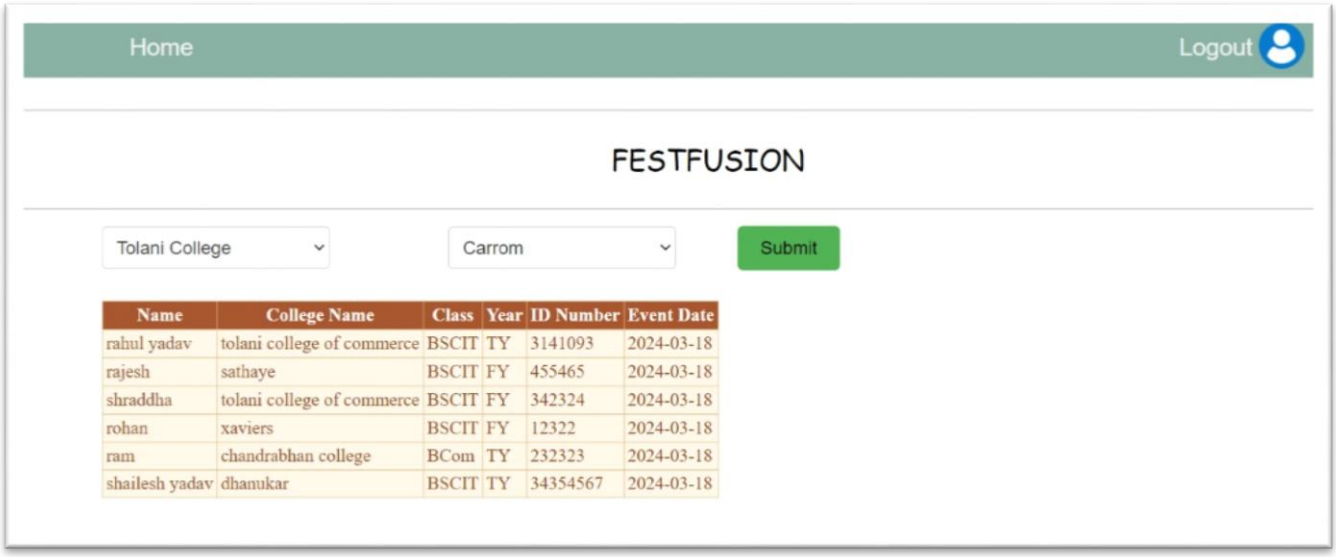
❖ Participants can view the whole details about a particular game in this section.

O) Game Form Section :



- Participant have to fill these form and submit to get registered for the event. Afterward they have to wait for the generating the pass.

P) Admin Section



Name	College Name	Class	Year	ID Number	Event Date
rahul yadav	tolani college of commerce	BSCIT	TY	3141093	2024-03-18
rajesh	sathaye	BSCIT	FY	455465	2024-03-18
shraddha	tolani college of commerce	BSCIT	FY	342324	2024-03-18
rohan	xaviers	BSCIT	FY	12322	2024-03-18
ram	chandrabhan college	BCom	TY	232323	2024-03-18
shailesh yadav	dhanukar	BSCIT	TY	34354567	2024-03-18

- This section is of admin where he can see all registered participants for an event by filtering the college name and game name accordingly.

CONCLUSION

The Fest Organizing System stands as a pivotal innovation in the realm of college fest organization, addressing the multifaceted challenges posed by today's dynamic educational landscape. College fests have evolved beyond mere extracurricular events they have become powerful platforms for students to express themselves, connect with peers from various institutions, and foster a sense of community. Nevertheless, the intricacies of fest planning, coordination, and efficient management often hinder the realization of these aspirations.

The Fest Organizing System, through its user-centric approach and technological prowess, surmounts these challenges with finesse. It not only simplifies the often cumbersome registration process but also centralizes fest-related information, optimizing resource allocation and enhancing overall efficiency. By offering a seamless, user-friendly experience to both organizers and participants, it strives to create an atmosphere that nurtures talent, encourages creativity, and fosters healthy competition. Beyond these immediate advantages, the system's potential extends to data-driven decision-making and event analysis, which can further elevate the fest experience.

In essence, the Fest Organizing System is more than just a tool for fest organization; it is a catalyst for forging memorable and meaningful connections among students, organizers, and sponsors. It stands as a testament to the transformative power of technology in enhancing the educational experience, demonstrating how innovation can simplify complexity, improve communication, and ultimately create an environment where the celebration of talent and the spirit of festivity can thrive. As we move forward in this ever-evolving educational landscape, the Fest Organizing System represents a beacon of efficiency, engagement, and enjoyment in the realm of college fests.

Significance of the System

1. Admin and User have different login activities for security.
2. Admin can add more Events if they want to Users.
3. The User know all the information about the Event.
4. Admin can configure the database and modify the event in it.

Limitations of the System

System Defects

The System possesses a great case of defects, there are various possibilities in which the system has defects. These System defects cause the program to lag or show errors while proceeding in its workflow. Several aspects of System Defects are as follows:

1. Storage Problem.
2. Database Connection Problem.
3. Battery Issues.
4. System Performance Problems.
5. Software Issues.

FUTURE SCOPE

Future Scope of the Project

There are also few features which can be integrated with this system to make it more flexible. Below list shows the future points to be consider :

1) The development of a dedicated mobile application would enable students to access fest information and register conveniently through their smartphones.
2) Integrating the system with popular social media platforms could amplify fest visibility by allowing participants to share event details and updates with their online networks.
3) Implementing data analytics tools would empower organizers with insights into participant preferences and event popularity, facilitating better resource allocation and planning.
4) Secure payment gateway integration can simplify financial transactions for paid events and merchandise, enhancing the overall fest experience.
5) Adapting the system for virtual or hybrid events would ensure flexibility in response to changing circumstances, such as pandemics or travel restrictions.
6) The introduction of gamification elements, such as point systems and rewards, could make the fest more engaging and interactive for participants.
7) Incorporating feedback and evaluation tools would allow for continuous improvement of the fest experience by gathering insights from participants and organizers.
8) AI-powered chatbots could streamline communication by providing instant responses to participant inquiries and event recommendations based on interests.

REFERENCES

Books:-

1. Beginning ASP.NET 4.5 in C# - Matthew MacDonald
2. ASP.NET 4.0 programming - J. Kanjilal
3. Beginning Visual C# 2010 - K. Watson, C. Nagel, J.H Padderson, J.D. Reid, M.Skinner

Website:-

1. https://www.eventbrite.com/
2. https://techbit4.netlify.app/#
3. https://www.scribd.com/document/359828916/12-College-Fest-Organizer-Abstract-docx

Glossary:-

1. User Interface (UI)
2. Real-Time Updates
3. Anticipation
4. Seamless Experience
5. GB-Giga Bytes
6. Mbps-Mega bits per second
7. HDD-Hard Disk Drive
8. UML-unified modeling language