

BLOGFUSE

**A PROJECT REPORT
for
Major Project (KCA 451)**

Session (2023-24)

Submitted by

**TUSHAR NAGPAL
(2200290140165)**

**Submitted in partial fulfilment of the
Requirements for the Degree of**

MASTER OF COMPUTER APPLICATION

Under the Supervision of

**Dr. Akash Rajak
(Professor)**



Submitted to

**DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206
(May 2024)**

DECLARATION

I hereby declare that the work presented in this report entitled “**BlogFuse**”, was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma from any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, and results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name – Tushar Nagpal (2200290140165)

CERTIFICATE

Certified that **Tushar Nagpal** have carried out the project work having **BlogFuse (Major Project-KCA 451)** for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

Tushar Nagpal (220029014065)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Dr. Akash Rajak
Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Dr. Arun Kumar Tripathi
Professor & Head
Department of Computer Application
KIET Group of Institutions, Ghaziabad

ABSTRACT

Welcome to a digital haven where words dance, ideas collide, and perspectives intertwine. At BlogFuse, we believe in the power of expression, the beauty of storytelling, and the magic that happens when minds converge in the boundless realm of the internet.

Our blog is more than just a collection of articles; it's a vibrant community where individuals from all walks of life come together to explore, learn, and engage with a myriad of topics. From insightful think pieces to practical how-tos, from thought-provoking analyses to heartwarming narratives, our platform is a melting pot of diverse voices and perspectives.

Whether you're a seasoned writer looking for inspiration, a curious reader hungry for knowledge, or someone simply seeking a moment of reflection, you'll find something here to spark your interest and ignite your imagination.

Join us on this journey as we delve into the depths of human experience, celebrate the richness of cultural diversity, and embark on a quest for understanding in an ever-changing world. Together, let's explore, discover, and connect through the power of words.

ACKNOWLEDGEMENTS

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Dr. Akash Rajak** for his guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Arun Kumar Tripathi**, Professor and Head, the Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Tushar Nagpal

(2200290140165)

TABLE OF CONTENTS

i	Declaration.....	ii
ii	Certificate	iii
iii	Abstract.....	iv
iv	Acknowledgements.....	v
v	Table of Contents.....	vi-vii
1	Introduction	1-3
1.1	Objectives.....	1
1.2	Need for Blog Website	1
1.3	Modules.....	2
1.4	Functionalities	2
2	Literature Review	4-5
2.1	Challenges in Blog Website	4
2.2	Future directions and innovations	5
3	Feasibility Study	6-9
3.1	Technical Feasibility	6
3.2	Economical Feasibility	8
3.3	Operational Feasibility	9
4	Technology Used and Setup	10-15
4.1	Hardware Requirements	10
4.2	Software Requirements	11
4.3	Installation of Software Requirements	15
5	Results	16-22
5.1	Home Page	16
5.2	Registration Page.....	17
5.3	Login Page.....	18
5.4	Create Post Page.....	19
5.5	Delete Post Page.....	20
5.6	Comment Page	21
5.7	Update Post Page.....	22
6	Flowchart/E-R Diagram	23-30
6.1	E-R Diagram	23
6.2	Flowchart.....	25
6.3	User Authentication.....	23
6.4	Content Creation	24
6.5	Content Reading.....	28
6.6	Commenting and like	29
6.7	Category Reading.....	30

7	Testing	31-36
7.1	Unit Testing.....	31
7.1.1	Benefits of Unit Testing	31
7.2	Integration Testing	32
7.2.1	Big Bang.....	32
7.2.2	Top-Down and Bottom-Up	33
7.3	Black-Box Testing	33
7.4	White-Box Testing	34
7.5	System Testing	36
8	Conclusion.....	38-40
8.1	Conclusion.....	38
8.2	Future Scope.....	38

CHAPTER 1

INTRODUCTION

Welcome to a digital haven where words dance, ideas collide, and perspectives intertwine. At BlogFuse, we believe in the power of expression, the beauty of storytelling, and the magic that happens when minds converge in the boundless realm of the internet.

Our blog is more than just a collection of articles; it's a vibrant community where individuals from all walks of life come together to explore, learn, and engage with a myriad of topics. From insightful think pieces to practical how-tos, from thought-provoking analyses to heartwarming narratives, our platform is a melting pot of diverse voices and perspectives.

Whether you're a seasoned writer looking for inspiration, a curious reader hungry for knowledge, or someone simply seeking a moment of reflection, you'll find something here to spark your interest and ignite your imagination.

Join us on this journey as we delve into the depths of human experience, celebrate the richness of cultural diversity, and embark on a quest for understanding in an ever-changing world. Together, let's explore, discover, and connect through the power of word.

1.1 Objective

To empower developers with insights, tutorials, and best practices for leveraging Fuse to create exceptional native mobile apps efficiently."

This objective encapsulates the primary purpose of the blog: to provide value to developers who are interested in utilizing Fuse for their mobile app development projects. It emphasizes the blog's role in offering practical guidance, educational resources, and inspiration to help developers maximize their productivity and achieve success with Fuse.

By aligning the blog's objective with the needs and interests of the target audience—developers interested in Fuse—the content can effectively attract, engage, and retain readers, ultimately contributing to the growth and success of the Fuse platform within the developer community.

1.2 Need for Blog Website

In today's digital landscape, a blog website serves as an indispensable platform for individuals and businesses alike. It goes be

yond being merely an online diary, offering a multitude of advantages. Personally, blogs provide an avenue for self-expression and creativity, enabling individuals to share their experiences, thoughts, and expertise with a global audience. Professionally, they play a pivotal role in personal branding, helping professionals showcase their skills, attract career opportunities, and establish authority in their respective fields. Blogs are also invaluable for knowledge sharing, serving as hubs of information that offer insights, tutorials, and resources to readers.

1.3 Modules

User Registration and Login:

This module is crucial for creating a personalized experience for your users. It allows visitors to sign up for an account, log in securely, and manage their profiles. With a personalized account, users can keep track of their activity, save favorite posts, and receive tailored content recommendations.

Home Page

The home page serves as the central hub of your blog website. It typically features a curated list of recent posts, popular articles, and featured content. The home page should be designed to engage visitors from the moment they arrive, encouraging them to explore further and stay longer.

Blog Creation

Create Blog: This module allows users to create and publish their blog posts. It should provide an intuitive interface for writing, formatting, and uploading images or other media. Key features include a text editor, the ability to save drafts, and options for categorizing and tagging posts for better organization and discoverability.

Blog Management

Update Blog: Users need the ability to update their blog posts as necessary. This module allows them to edit content, update images, and revise tags or categories. It ensures that blog content remains current and accurate.

Delete Blog:

This module enables users to remove blog posts that are no longer relevant or needed. It's important for maintaining the quality and relevance of the content on the site.

Comments :

Comment: Engagement is a vital aspect of any blog website. The comment module allows readers to leave feedback, ask questions, and engage in discussions related to the blog posts. This interaction can help build a community around your blog and provide valuable insights and feedback from your audience.

1.4 Functionalities

User Registration

- Account creation form.
- Input fields: username, email, password, confirm password.
- Email verification.

Login

- Login form.
- Input fields: username/email, password.
- "Remember Me" option.

Home Page

- Content Display
- Recent blog posts.
- Featured/recommended articles.

Update Blog

- Edit existing posts.
- Update media and text.
- Change categories and tags.
- Modify SEO settings.

Delete Blog

- Permanently delete posts.
- Confirmation prompt.

Comments

- Commenting System:
- Comment form.
- Input fields: name, email, comment text.
- Basic formatting.
- Approve, edit, delete comments.

Security

- HTTPS.
- Password encryption.

CHAPTER 2

LITERATURE REVIEW

The research papers we considered while doing our analysis are listed below. A wireless meal ordering system was designed and implemented together with consumer feedback for a restaurant. It makes it simple for restaurant operators to change menu presentations and set up the system in a WIFI setting. The configurable wireless meal ordering system has linked a smart phone with real-time customer feedback implementation to enable real-time contact between patrons of restaurants and business owners. The goal was to investigate the variables that affect internet users' perceptions of online food ordering among university students in Turkey. Davis's Technology Acceptance Model (TAM), which he created in 1986, was used to analyse how the Web environment for ordering food was adopted. Along with TAM, three additional primary factors—Trust, Innovation, and External Influences—are included in the paradigm.

This research examines the initiatives made by restaurant owners to implement ICTs—such as PDAs, wireless LANs, and pricey multi-touch screens—to improve the dining experience. In order to address some of the drawbacks of the traditional paper-based and PDA-based food ordering systems, a low-cost touchscreen-based restaurant management system that uses an Android smartphone or tablet is suggested in this study.

The study's objective was to determine whether the application is user-centred and based on user requirements. This system developed all problems pertaining to every user that it includes. Almost anyone may use the program if they know how to use an Android smartphone. The various problems with the Mess service will be resolved by this system. The implementation of an online food ordering system is done to assist and resolve significant issues for consumers. Based on the application, it can be said that: This system makes placing orders simple; it gives customers the information they need to place orders.

The evolution of food ordering systems can be traced back to traditional methods such as phone orders and walk-in orders at restaurants. However, the emergence of the internet and mobile technologies has revolutionized the way people order food. Online food ordering platforms like Seamless, Grubhub, and Uber Eats have gained immense popularity, offering convenience and variety to consumers. These platforms utilize web and mobile applications to connect users with nearby restaurants, enabling them to browse menus, place orders, and track deliveries in real-time.

2.1 Challenges in Blog Website

Creating a blog website involves several challenges, including technical, content-related, and promotional aspects. Technically, you need to choose the right platform, ensure reliable web hosting, design an intuitive and mobile-responsive interface, and optimize for SEO and performance while maintaining robust security. Content-wise, generating high-quality, engaging, and consistent content can be demanding, requiring a clear strategy and regular updates. Additionally, promoting your blog to attract and retain readers involves effective marketing, social media engagement, and possibly leveraging SEO tactics. Balancing these elements requires a blend of technical expertise, creativity, and strategic planning.

Additionally, promoting your blog to attract and retain readers involves effective marketing, social media engagement, and possibly leveraging SEO tactics. This includes understanding and implementing effective keyword strategies, creating shareable content, and engaging with your audience through comments and social media. Balancing these elements requires a blend of technical expertise, creativity, strategic planning, and ongoing effort to adapt to changing trends and feedback.

2.1 Future Directions and Innovations

The future of blog websites is poised for exciting developments driven by advancements in technology and evolving user expectations. Artificial intelligence and machine learning are set to enhance content personalization, offering readers customized recommendations based on their reading habits and preferences, and even assisting in content creation and moderation. Voice search optimization will become increasingly important with the rise of voice-activated devices, necessitating the use of natural language processing to ensure content is discoverable through voice queries. Additionally, the integration of interactive elements like quizzes, polls, and infographics, along with multimedia content such as videos and podcasts, will make blogs more engaging. Augmented Reality (AR) and Virtual Reality (VR) could also provide immersive content experiences. Enhanced user experience (UX) will be a key focus, incorporating advanced design principles, faster loading times, and seamless navigation. Progressive Web Apps (PWAs) may offer app-like experiences on the web, improving accessibility and engagement. Moreover, blockchain technology could introduce innovations in content verification, digital ownership, and monetization, ensuring transparency and security in the blogging ecosystem.

CHAPTER 3

FEASIBILITY STUDY

A feasibility study is a detailed analysis that considers all of the critical aspects of a proposed project in order to determine the likelihood of it succeeding.

Success in business may be defined primarily by return on investment, meaning that the project will generate enough profit to justify the investment. However, many other important factors may be identified on the plus or minus side, such as community reaction and environmental impact.

A feasibility study is an important step in any project, including an emotion detection project. It helps to determine the technical, economic, operational, and legal feasibility of the project. Here are some key aspects to consider in a feasibility study for an emotion detection project.

Based on the results of the feasibility study, the project team can make informed decisions about the viability and scope of the emotion detection project. If the feasibility study indicates that the project is viable and has potential benefits, the team can proceed with the project planning and implementation. If the study indicates that the project is not feasible or has significant risks and limitations, the team can consider alternative approaches or abandon the project altogether.

Before starting the project, feasibility study is carried out to measure the viability of the system. Feasibility is necessary to determine if creating a new or improved system is friendly with the cost, benefits, operation, technology and time.

Feasibility studies are important for a communications service provider to determine whether your broadband project will succeed or not. It should be the first action taken when wanting to begin a new project. It is one, if not the most important factor in determining whether the project can and should move forward. Also, if you are applying for broadband loans and grants, a feasibility study is normally required.

Following feasibility is given below:

3.1 Technical Feasibility

The technical feasibility of the Zomato Clone project involves assessing the availability of resources, technology requirements, compatibility, scalability, and security considerations. Here are the key factors to consider:

Resource Availability:

- Evaluate the availability of skilled developers, UI/UX designers, and testers for web development.
- Assess the availability of infrastructure, servers, and hosting resources for the backend components.
- Consider the availability of devices and browsers for testing the website on various platforms.

Technology Requirements:

- Determine the tech stack: MongoDB for the database, Express.js for the backend, React for the front end, and Node.js for server-side scripting.
- Evaluate the compatibility of the chosen technologies with different web browsers.
- Consider the integration capabilities of the MERN stack with external services or APIs for features like location services.

Compatibility:

- Ensure the website is compatible with a wide range of web browsers (Chrome, Firefox, Safari, etc.).
- Perform testing on different devices and screen sizes to ensure a responsive and consistent user experience.
- Adhere to web standards and guidelines to ensure compatibility across platforms.
- Scalability and Performance:
 - Design the website architecture to handle a large number of users, restaurants, and menu items.
 - Optimize database queries and backend processes for efficient performance.
 - Conduct load testing to simulate a high number of concurrent users and ensure the website's responsiveness.

Security:

- Implement robust authentication and authorization mechanisms to protect user data and prevent unauthorized access.
- Use encryption methods for sensitive data, such as user credentials and payment information.
- Follow best practices for secure communication protocols (HTTPS) and data storage.
- Integration of different modules.
- Evaluate integration requirements with external services for features like maps, reviews, and payment gateways.
- Implement APIs or web services for seamless integration with external systems.
- Ensure compatibility with relevant APIs and adhere to their usage policies and limitations.

Testing and Quality Assurance:

- Develop a comprehensive testing strategy that includes functional testing, usability testing, and compatibility testing across different browsers.
- Perform rigorous testing to identify and fix any bugs or issues before deploying the website.
- Conduct security testing and vulnerability assessments to protect against potential threats.

Documentation and Training:

- Prepare technical documentation that outlines the website's architecture, features, and deployment instructions.
- Provide user manuals and guides for restaurant owners, customers, and administrators to understand and use the website effectively.
- Conduct training sessions or provide training resources to familiarize users with the website's functionalities.
- Considering these technical feasibility factors will ensure the successful development, deployment, and performance of the Zomato Clone and meeting requirements.

3.2 Economical Feasibility

For economic feasibility, Economic analysis or cost/benefits analysis is the most frequently used technique for the effectiveness of a proposed system. It is a procedure to determine the benefits and savings that are expected from the proposed system and compare them with cost. If the benefits outweigh the costs, a decision is taken to design and implement the system. Otherwise, further justification or alternative in the proposed system will have to be made if it is to have a chance of being approved. This is an ongoing effort that improves in accuracy at each phase of a system life cycle.

- **Cost-Benefit Analysis:** Assessing the potential costs involved in implementing a food ordering system against the anticipated benefits such as increased revenue, improved operational efficiency, and customer satisfaction.
- **Return on Investment (ROI):** Determining the expected return on investment over a specific period, considering factors like initial setup costs, ongoing maintenance expenses, and projected revenue growth.
- **Market Demand:** Analyzing the demand for online food ordering services in the target market, including factors like demographics, consumer preferences, and competitor analysis to gauge the revenue potential.
- **Scalability:** Evaluating the scalability of the food ordering system to accommodate future growth and expansion, while minimizing additional investment and operational costs.
- **Cost Reduction:** Identifying opportunities to reduce costs through automation, streamlining processes, and optimizing resource allocation within the food ordering system.
- **Revenue Generation:** Exploring various revenue streams such as transaction fees, subscription models, advertising, and partnerships to maximize revenue generation potential.
- **Risk Assessment:** Conduct a thorough risk assessment to identify potential economic risks such as market volatility, regulatory changes, competitive pressures, and technological disruptions, and develop mitigation strategies accordingly.

3.3 Operational Feasibility

No doubt the technically growing world needs more enhancement in technology, this app is very user friendly and all inputs to be taken all self-explanatory even to a layman. As far as our study is concerned, the clients will be comfortable and happy as the system has cut down their loads and brought the young generation to the same virtual world they are growing drastically.

Operational feasibility covers two aspects. one technical performance aspects and the other is acceptance within the organization.

Operation feasibility determine how the proposed system will fit in with the current operation and what needs to implement the system.

- **Alignment with Organizational Objectives:** The proposed system should align with the strategic goals and objectives of the organization to ensure its successful implementation.
- **Compatibility with Existing Processes:** The system should be compatible with the existing business processes, infrastructure, and technologies to minimize disruptions and facilitate integration.
- **Resource Availability:** Assess the availability of necessary resources such as financial, human, and technical resources required for system development, implementation, and maintenance.
- **Skills and Training Requirements:** Evaluate whether the organization has the necessary skills and expertise to develop, operate, and maintain the proposed system. Determine if additional training or hiring is required.
- **Acceptance by Stakeholders:** Consider the level of acceptance and support from key stakeholders, including management, employees, customers, and external partners, as their buy-in is essential for successful implementation.
- **Risk Assessment:** Identify potential risks and challenges associated with system implementation, such as technological barriers, resistance to change, and regulatory compliance issues. Develop mitigation strategies to address these risks effectively.
- **Scalability and Flexibility:** Assess the system's scalability and flexibility to accommodate future growth, changes in business requirements, and emerging technologies without significant disruptions or costly modifications.
- **Impact on Operations:** Analyze the potential impact of the new system on day-to-day operations, productivity, efficiency, and customer service. Minimize negative impacts through careful planning and stakeholder engagement.

CHAPTER 4

TECHNOLOGY USED AND SETUP

4.1 HARDWARE REQUIREMENTS

Hardware requirements for a project refer to the specific physical components or devices needed to support the project's objectives. These requirements can vary significantly depending on the nature of the project. These requirements include computing power, storage, network connectivity, memory etc.

The hardware requirements for accessing “BlogFuse” is enlisted in the table below:

Table 4.1: Hardware Requirements

S. No.	Description of system requirement
1	5 GB or more Hard disk.
2	8 GB RAM.
3	Core i5 7 th gen or above processor.
4	50 Mbps or more internet connectivity

5 GB or more Hard disk:

This specifies the storage requirement for the PC. It should have a hard disk with a capacity of 5 gigabytes (GB) or more. This is where you store your operating system, software applications, and data. A PC with a 5 GB or larger hard disk provides ample storage for the operating system, software applications, and user data. This ensures smooth performance, accommodates growing storage needs, and allows for data backups and future expansion.

8 GB RAM:

This sets the minimum random-access memory (RAM) requirement for the PC. It should have at least 8 gigabytes of RAM. RAM is essential for running applications and the operating system efficiently. Having 8 GB of RAM ensures smooth performance for running multiple applications simultaneously, faster operation of the operating system, seamless multitasking, and readiness for resource-intensive tasks like gaming or video editing.

Core i5 7th gen or above processor:

This specifies the processor requirement for the PC. It should have an Intel Core i3 processor or a more powerful one. The processor is a crucial component that determines the computer's overall speed and performance. A Core i5 7th gen or higher processor ensures strong performance and responsiveness for the PC. This Intel processor provides ample computing power for everyday tasks, multitasking, and even some demanding applications like photo or video editing. It offers a balance of speed, efficiency, and reliability, making it suitable for most users' needs.

50 Mbps or more internet connectivity:

A 50 Mbps or higher internet connection ensures fast and reliable access to online resources, data, and collaborative tools required for the project. With this speed, users can quickly download and upload files, stream multimedia content, participate in video conferences, and collaborate with team members in real-time. It provides a seamless online experience, reducing wait times and enhancing productivity, particularly for projects that rely heavily on cloud-based services, remote collaboration, or data-intensive tasks.

4.2 SOFTWARE REQUIREMENTS

Software requirements for a project outline the specific programs, platforms, and functionalities needed to achieve project goals. They detail essential software components such as operating systems, development tools, databases, and any specialized software necessary for project execution. These requirements serve as a roadmap for software selection, development, integration, and testing throughout the project lifecycle.

The software requirements of “Collablog” are enlisted in the following software requirements table:

Table 4.2 Software Requirements

S. No.	Description	Type
1	Operating System	Windows 10 or 11
2	Front End	Vite, React JS, JS, Tailwind CSS
3	Back End	Node JS
4	Database and Storage	MongoDB
5	IDE	VS Code
6	Browser	Chrome, Firefox, Edge

Operating System

The specified operating system requirement for the project development environment is either Windows 10 or 11, with the option to use a newer version if available. This ensures compatibility with the latest software development tools, libraries, and frameworks required for the project. Additionally, it provides a consistent and stable platform for developers to create, test, and deploy the project's software components. By standardizing the operating system environment, it facilitates collaboration among team members and simplifies software configuration management, version control, and troubleshooting processes throughout the project lifecycle.

Front End

The frontend of a project is what users interact with directly, including the interface, design, and user experience. The goal is to create an intuitive, visually appealing, and responsive interface that guides users seamlessly through the application.

React JS

React.js is a JavaScript library for building user interfaces. It simplifies UI development by breaking it down into reusable components and managing updates efficiently with a virtual DOM. Developers use JSX to write UI components, enabling a declarative approach to defining UIs based on application state

Tailwind CSS

Tailwind CSS is a utility-first CSS framework that streamlines frontend development by providing a set of pre-defined utility classes for styling HTML elements. It prioritizes simplicity, responsiveness, and customization, making it easy for developers to create modern and responsive user interfaces efficiently.

Vite

Vite is a fast build tool for modern web development, specifically designed to optimize the development experience for frontend projects. It leverages native ES modules in modern browsers to deliver instant server startup and rapid hot module replacement (HMR). With Vite, developers can build and serve frontend applications quickly, enhancing productivity and facilitating a smooth development workflow.

Back End

The backend of a project comprises server-side code, databases, and APIs that handle data processing, business logic, authentication, security, and communication with clients. It powers the functionality of the application, manages data storage, and ensures that users can interact with the system securely and efficiently.

Node JS

Node.js is a runtime environment that allows you to use JavaScript for server-side development. It efficiently handles many connections simultaneously due to its event-driven and asynchronous nature. This makes it ideal for building scalable applications like real-time chat apps, APIs, and microservices. Using the same language for both frontend and backend simplifies development, and the npm ecosystem provides a vast array of modules to speed up the development process. Node.js is built on the fast V8 JavaScript engine, ensuring high performance.

Database and Storage

The database in a project provides essential data management capabilities, allowing for the storage, retrieval, and organization of data. Integrating a database involves installing the necessary drivers, connecting to the database, defining schemas and models, and using these models in the application logic.

MongoDB

MongoDB is a NoSQL, document-oriented database designed for managing large volumes of data across distributed systems. It features a flexible, schema-less data model that allows data to be stored in JSON-like documents with dynamic schemas. This flexibility enables developers to store complex data structures and adapt quickly to changing requirements without needing to define a rigid schema upfront. MongoDB supports horizontal scaling through sharding, distributing data across multiple servers, and is optimized for high-performance read and write operations, making it suitable for high-throughput applications. Its rich query language supports ad hoc queries, indexing, and real-time aggregation, while replication ensures high availability and redundancy through replica sets.

IDE

An Integrated Development Environment (IDE) is a software tool that combines various features to facilitate programming tasks. It typically includes a code editor, compiler/interpreter, debugger, build automation tools, version control integration, and project management capabilities. IDEs increase developer productivity by providing a centralized environment for coding, debugging, and managing projects, ultimately leading to more efficient software development.

Visual Studio Code

Visual Studio Code (VS Code) is a highly popular and versatile integrated development environment (IDE) developed by Microsoft. It's favoured by developers across various platforms and programming languages due to its extensive features and flexibility. Visual Studio Code (VS Code) is the preferred integrated development environment for coding.

Browser

Accessing a project via a web browser allows users to interact with web-based applications or websites. It provides accessibility, cross-platform compatibility, user-friendly interfaces, security features, scalability, and easy updates, making it a crucial aspect of modern computing. Mozilla Firefox, Google Chrome, Microsoft Edge, any of the browsers can be used to access the software.

4.3 INSTALLATION OF SOFTWARE REQUIREMENTS

i. Visual Studio Code (VS Code)

To install Visual Studio Code (VS Code):

- **Download:** Go to the official VS Code website (<https://code.visualstudio.com>) and download the installer for your operating system (Windows, macOS, or Linux).
- **Run Installer:** Once the download is complete, run the installer executable file. After the installation completes, launch Visual Studio Code from your system's applications menu or desktop shortcut.

ii. Node JS

To install Node.js:

- Download: Visit the official Node.js website (<https://nodejs.org>) and download the appropriate installer for your operating system (Windows, macOS, or Linux).
- Run Installer: Open the downloaded installer file. Finish the installation process. The installer will automatically add Node.js and npm (Node Package Manager) to your system PATH.
- Verify Installation: Open a terminal or command prompt and run the following commands to verify the installation:
 - `node -v`
 - `npm -v`

iii. Vite

- To install Vite, execute the following lines on the terminal
 - `vite build`
 - `npm create vite@latest .`

iv. React JS

- To install React JS, execute the following lines on the terminal
 - `npm create vite@latest my-project -- --template react`
 - `cd my-project`
- To install various react components used in the project
 - `npm install moment`
 - `npm install react-quill`
 - `npm install react-tagsinput`
 - `npm install react-share`
 - `npm install react-router-dom react-icons`
 - `npm install --save react-tag-input`

v. Tailwind CSS

- To install Tailwind CSS, execute the following lines on the terminal
 - `npm install -D tailwindcss postcss autoprefixer`
 - `npx tailwindcss init -p`

vi. MongoDB

Download:

Go to the MongoDB Download Center and download the latest version of MongoDB for Windows.

Install:

Run the downloaded .msi installer.

Follow the setup instructions. Make sure to select "Complete" setup.

During the setup, check "Install MongoDB as a Service".

Set Up Environment:

After installation, add the MongoDB bin directory (e.g., C:\Program Files\MongoDB\Server\{version}\bin) to the PATH environment variable.

Run MongoDB:

Open Command Prompt and type mongo to start the MongoDB shell.

CHAPTER 5

RESULTS

5.1 Home page

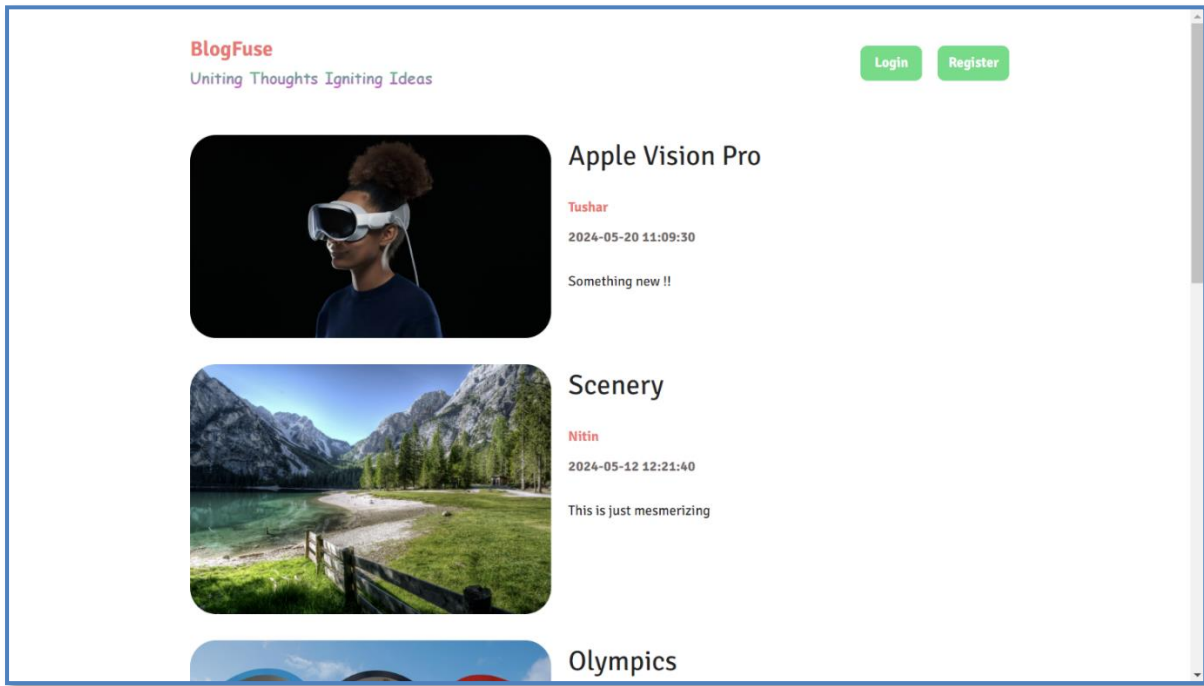


Fig 5.1 Home page

The image shows a webpage from a blog site named "BlogFuse," which has the tagline "Uniting Thoughts Igniting Ideas." The page displays several blog posts, each with a large image, title, author, date, time, and a brief description.

5.2 Registration Page

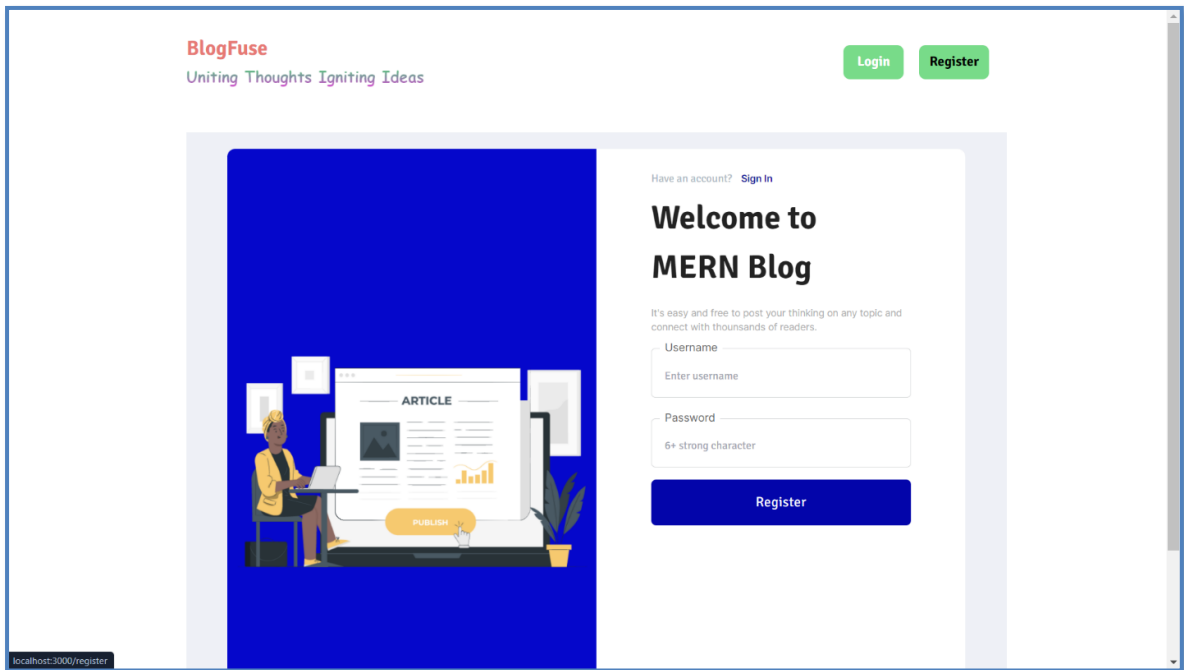
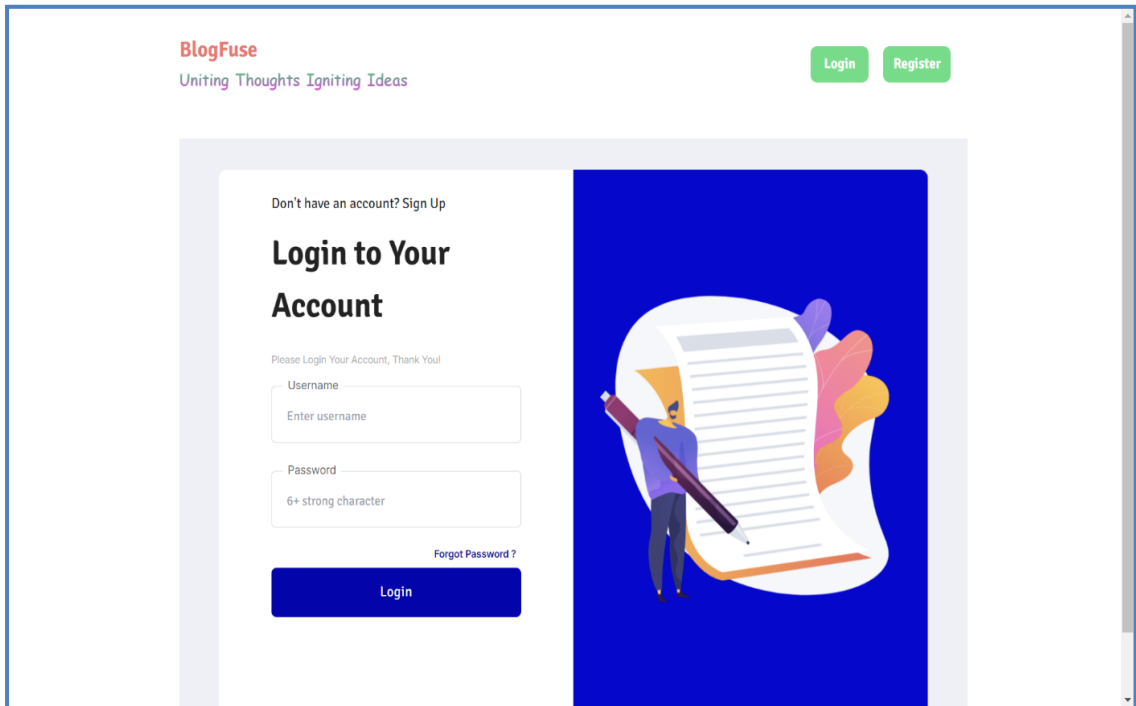


Fig 5.2 Registration page

The image displays a registration page from a blog site named "BlogFuse," which has the tagline "Uniting Thoughts Igniting Ideas." The page is titled "Welcome to MERN Blog," suggesting it is built using the MERN stack (MongoDB, Express.js, React, and Node.js). The right side of the page features a registration form with fields for entering a username and a password, accompanied by a "Register" button.

5.3 Login Page



The image shows a login page for a website named "BlogFuse". The page has a white background with a blue border. At the top left, the logo "BlogFuse" is displayed in red, with the tagline "Uniting Thoughts Igniting Ideas" in purple below it. At the top right, there are two green buttons labeled "Login" and "Register". The main content area is divided into two sections. On the left, there is a login form with the heading "Login to Your Account" and the subtext "Please Login Your Account, Thank You!". The form includes a "Username" field with the placeholder text "Enter username", a "Password" field with the placeholder text "6+ strong character", and a "Forgot Password?" link. A blue "Login" button is at the bottom of the form. On the right, there is a large blue rectangular area containing a white illustration of a person standing next to a large, rolled-up document, holding a pen.

Fig 5.3 Login Page

The image shows a login page from a blog site named "BlogFuse," with the tagline "Uniting Thoughts Igniting Ideas." The page is designed to allow users to log into their accounts. On the left side of the page, there's a login form with fields for entering a username and password, along with a "Login" button.

5.4 Create Post Page

BlogFuse
Uniting Thoughts Igniting Ideas

Create new post Logout (Tushar)

Title

Summary

Choose File No file chosen

Normal B I U S " | E E E E | | I

Create post

Fig 5.4 Create a post page

This is a "Create Post" page where users can craft a new blog entry by providing details such as the Title, Summary, and a photo related to the blog. Users can then add the main content of the blog, elaborating on it clearly and concisely.

5.4 Delete post page

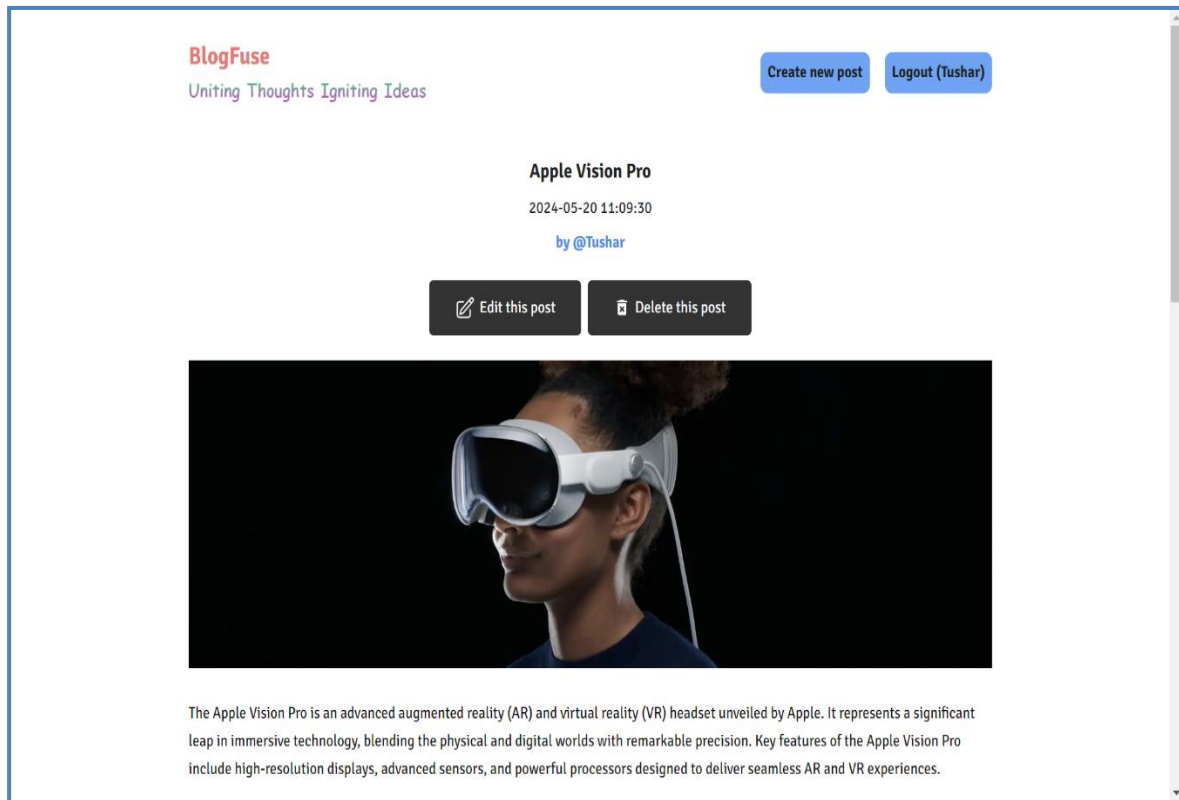


Fig 5.5 Delete Post Page

This page on the BlogFuse platform displays a detailed view of a blog post titled "Apple Vision Pro," It includes:

A header with the platform name, tagline, and buttons for creating a new post and logging out. The post title, publication date, and author link. Option to delete the post. A brief description of the Apple Vision Pro, highlighting its advanced AR and VR capabilities.

5.5 Comment Page

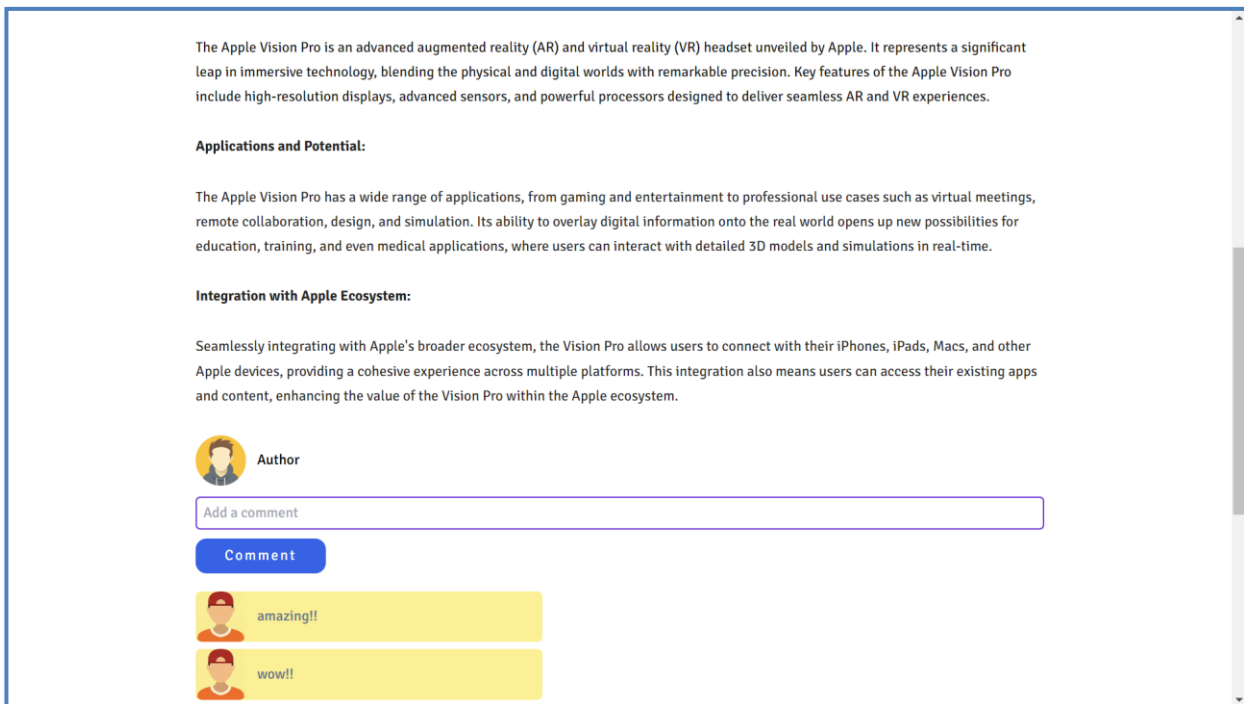


Fig 5.6 Comment Page screenshot

A comment page where user can share their views about the blog and can also give feedbacks.

5.7 Update Post Page

BlogFuse
Uniting Thoughts Igniting Ideas

Create new post Logout (Tushar)

Apple Vision Pro

Something awesome

Choose File No file chosen

Normal B I U G " ☰ ☷ ☹ ☶ 🔗 🖼️ ↵

The Apple Vision Pro represents a significant leap forward in the realm of augmented reality (AR) and virtual reality (VR) technology, marking Apple's ambitious entry into the mixed reality market. Unveiled as a cutting-edge device, the Vision Pro integrates seamlessly into Apple's ecosystem, promising to redefine how users interact with digital content and their physical environment.

Update Post

Fig 5.7 Update Post Page Screenshot

In Update post page user can simply update the post by filling given details.

CHAPTER 6

E-R DIAGRAM/FLOWCHART

E-R DIAGRAM

Entity-Relationship (ER) diagrams are used to model the logical structure of databases by visually representing entities, their attributes, and the relationships between them. Entities represent objects or concepts, such as "User" or "BlogPost," while attributes describe properties of entities, like "Username" or "PostID." Relationships depict how entities interact, such as a "User" writing multiple "BlogPost"s. ER diagrams help in designing databases by clarifying the data requirements and ensuring proper structure and organization. They are essential for database design, providing a blueprint for creating and maintaining a relational database, ensuring data integrity, and facilitating efficient data management.

Entities:

Represent objects or concepts like "Customer" or "Product."
Depicted as rectangles.

Attributes:

Describe properties of entities, such as "CustomerName," "Email," "ProductID," or "Price."
Depicted as ovals connected to their respective entities.

Relationships:

Show how entities interact, like a "Customer" placing an "Order."
Depicted as diamonds or lines connecting entities.

Primary Key (PK):

A unique identifier for each entity instance.
Ensures each record is uniquely identifiable.

Cardinality:

Indicates the number of instances in one entity related to instances in another (e.g., one-to-many, many-to-many).
Helps in understanding the relationship's nature and constraints.

Normalization:

ER diagrams help in organizing data to reduce

6.2.1 E-R Diagram

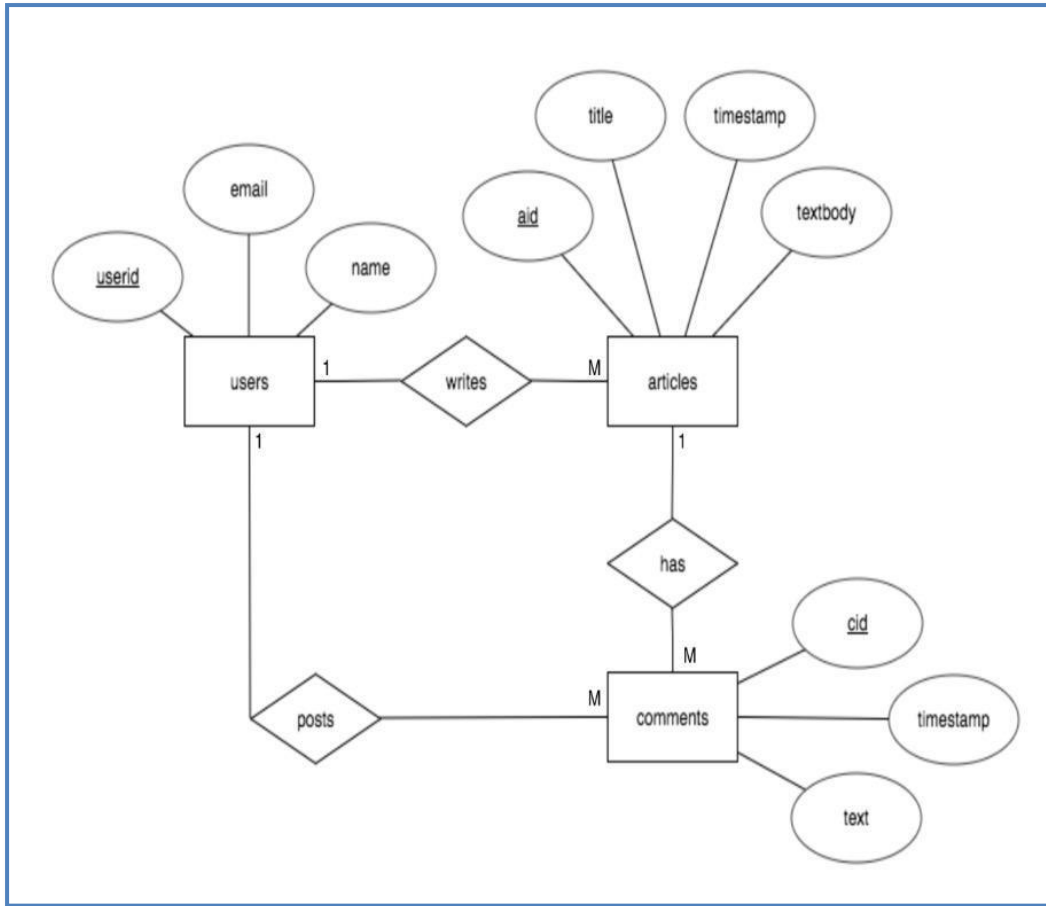


Fig 6.1 Diagram for E-R Diagram

FLOWCHARTS

Flowcharts are graphical representations of processes or systems, consisting of various symbols connected by arrows to illustrate the flow of steps or actions.

They are visual tools that represent the steps in a process or system using symbols such as ovals, rectangles, diamonds, and arrows. They are designed to simplify complex processes, making them easier to understand and communicate.

Here's more content about flowcharts:

Visual Representation: Flowcharts are diagrams that visually represent processes or workflows using symbols and arrows.

Symbolic Language: They use standardized symbols like rectangles for processes, diamonds for decisions, and arrows for flow direction.

Process Mapping: Flowcharts map out the steps of a process from start to finish, showing the sequence of actions.

Decision Points: They highlight decision points where the flow can take different paths based on conditions.

Easy Understanding: Flowcharts simplify complex processes, making them easier to understand and follow.

Problem-Solving Tool: They help identify bottlenecks, redundancies, and inefficiencies in a process, aiding in problem-solving and optimization.

Applications: Used in various fields including business, software development, project management, education, and engineering.

Standardization: Flowcharts provide a standardized way to document and analyze processes, ensuring clarity and consistency.

Software Tools: There are many software tools available for creating flowcharts, offering templates and drag-and-drop functionality.

Communication: Flowcharts facilitate communication by providing a visual representation of processes, making it easier to convey information to others.

6.2.2 User Authentication

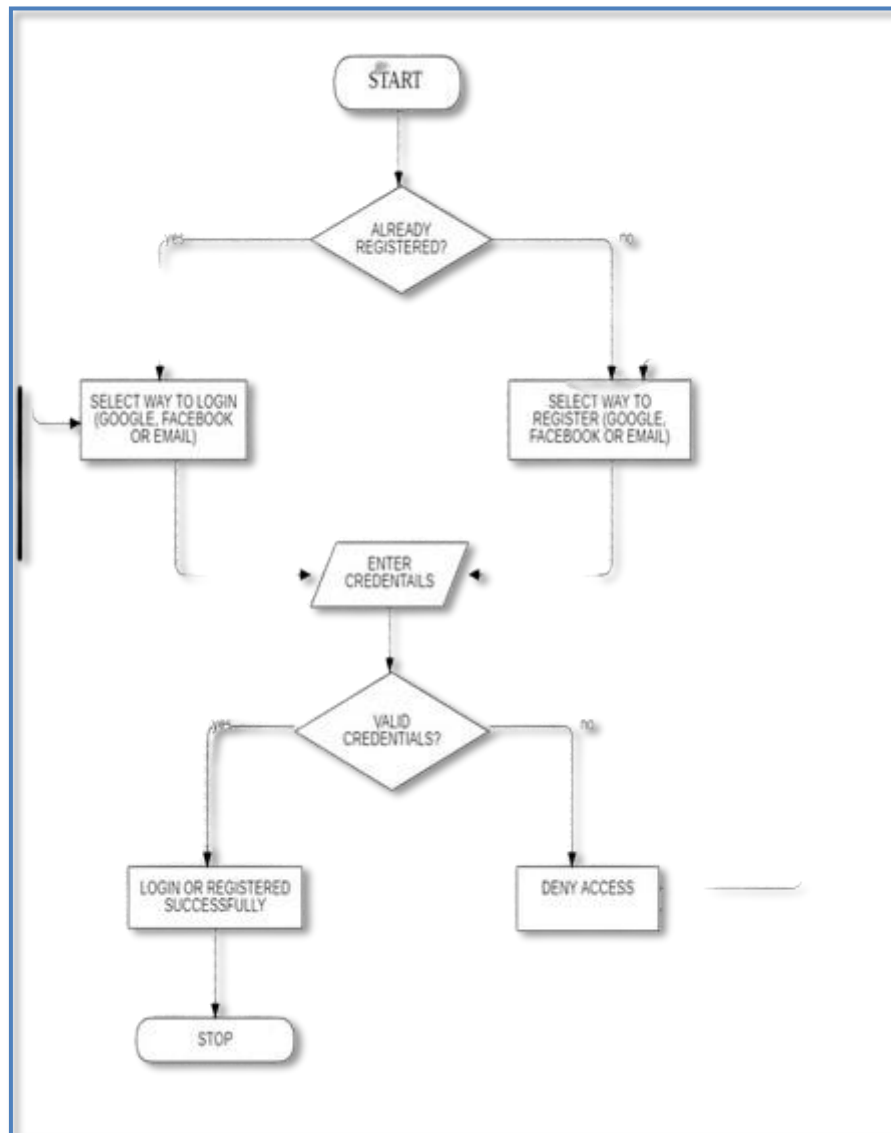


Fig 6.2 : Flowchart of User Authentication Module

6.2.3 Content Creation

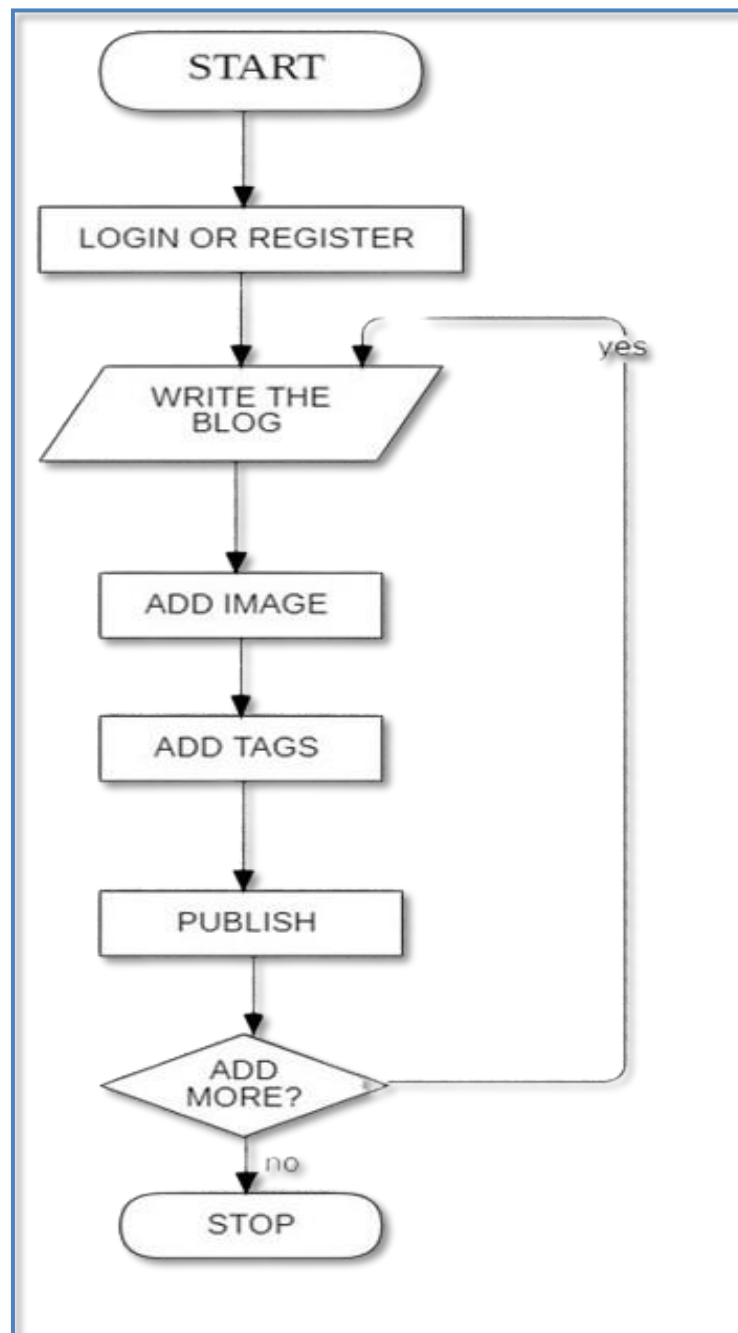


Fig 6.3: Flowchart of Content Creation Module

6.2.4 Content Reading

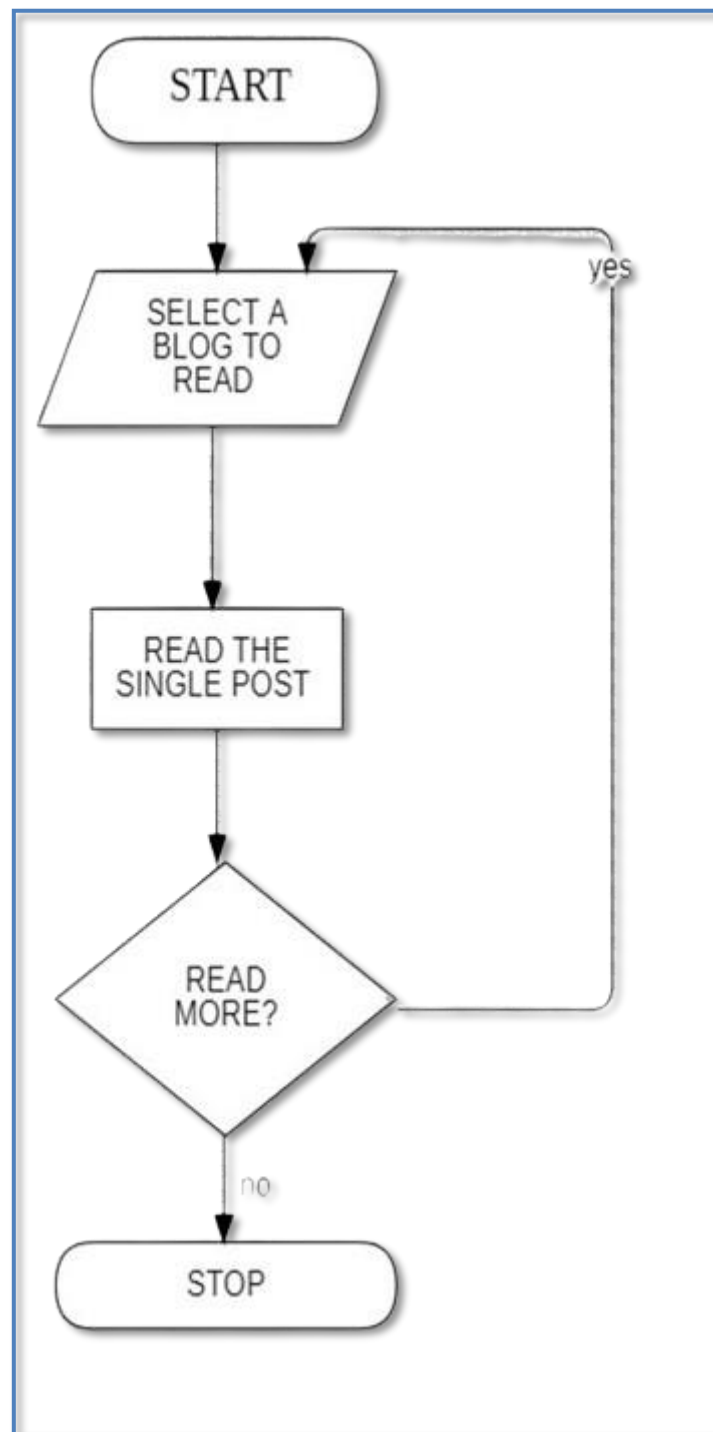


Fig 6.4 Flowchart for content Reading

6.2.5 Commenting and like

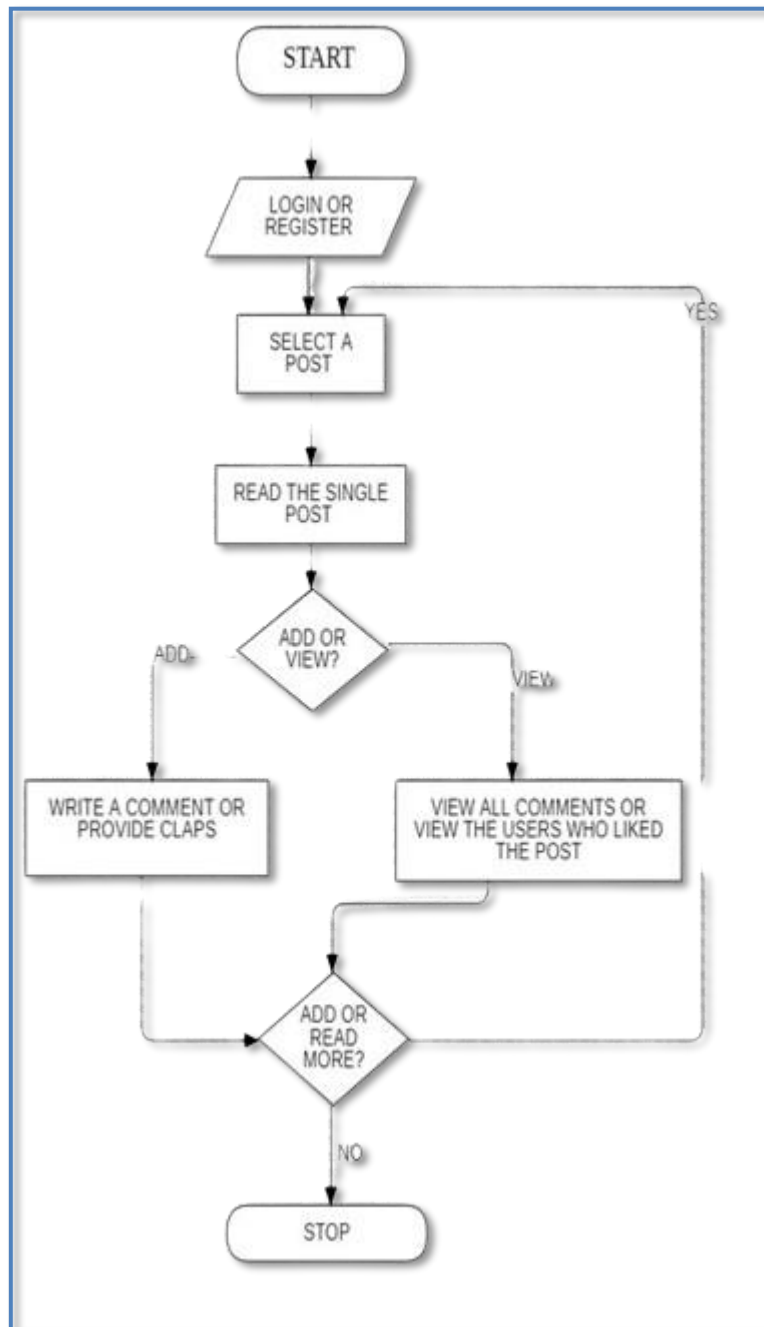


Fig 6.5 Figure for commenting and like

6.2.6 Category Reading

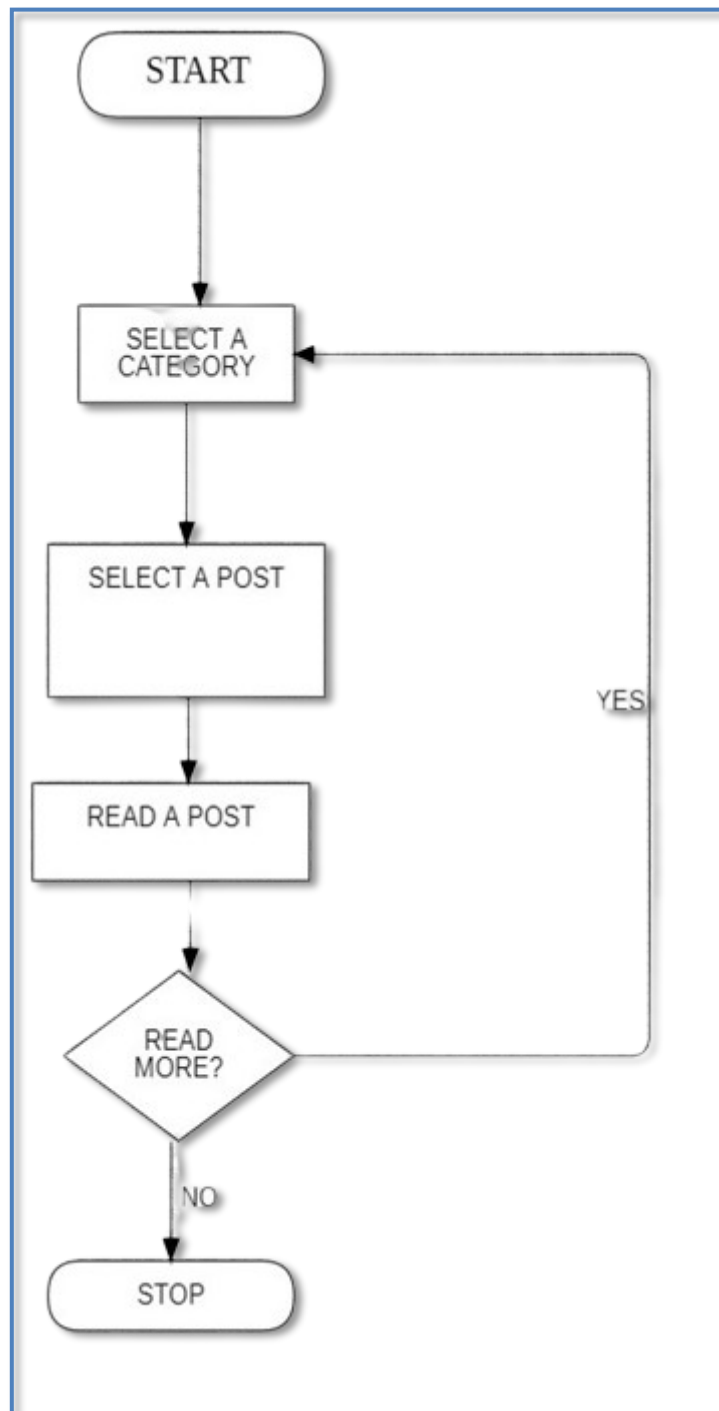


Fig 6.6 Figure for category Reading

CHAPTER 7

TESTING

7.1 Unit Testing

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but it could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist in testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended.

Benefits of Unit Testing

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it offers several benefits.

Find Problems Early:

Unit testing finds problems early in the development cycle. In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build.

The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, it is still early in the development process.

Facilitates Change:

Unit testing allows the programmer to refactor code or upgrade system libraries later, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes that may break a design contract.

Documentation:

Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface (API). Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate

appropriate/inappropriate use of a unit as well as negative behaviours that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in the development unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

7.2 Integration Testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

The purpose of integration testing is to verify the functional, performance, and reliability requirements placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using black-box testing, with success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface.

Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests. Some different types of integration testing are big-bang, top- down, and bottom-up, mixed (sandwich) and risky- hardest. Other Integration Patterns are collaboration integration, backbone integration, layer integration, client-server integration, distributed services integration and high-frequency integration.

7.2.1 Big Bang

In the big-bang approach, most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing.

This method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing. A type of big-bang integration testing is called "usage model testing" which can be used in both software and hardware integration testing. The basis behind this type of integration testing is to run user-like workloads in integrated user-like environments. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. Usage Model testing takes an optimistic approach to testing because it expects to have few problems with the individual components.

The strategy relies heavily on the component developers to do the isolated unit testing for their product. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the components in the environment.

To be more efficient and accurate, care must be used in defining the user-like workloads for creating realistic scenarios in exercising the environment. This gives confidence that the integrated environment will work as expected for the target customers.

7.2.2 Top-Down and Bottom-Up

Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher-level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower-level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready.

This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage. Top-down testing is an approach to integrated testing

where the top integrated modules are tested, and the branch of the module is tested step by step until the end of the related module. Sandwich testing is an approach to combine top-down testing with bottom-up testing.

7.3 Black-Box Testing

Black box testing is a technique used to test the functionality of a software application without having knowledge of its internal structure or implementation details. It focuses on the inputs and outputs of the system and verifies if the expected outputs match the desired results. Here are some examples of black box testing techniques that can be applied to the KIET Event Management App:

Equivalence Partitioning:

Identify different categories of inputs for the app, such as valid and invalid inputs, and divide them into equivalence classes.

- Test representative values from each equivalence class to ensure the app behaves consistently within each class.

Boundary Value Analysis:

- Identify the boundaries or limits for inputs in the app, such as minimum and maximum values, and test values at those boundaries.
- Test values just above and below the boundaries to verify the app's behaviour at critical points.

Decision Table Testing:

- Identify the different conditions and rules that govern the behaviour of the app.
- Create a decision table with combinations of conditions and corresponding expected results.

- Test different combinations of conditions to validate the app's decision-making process.

State Transition Testing:

- Identify the different states that the app can transition between.
- Define the valid and invalid transitions between states.
- Test different sequences of state transitions to verify the app's behaviour.

Error Guessing:

- Use experience and intuition to guess potential errors or issues in the app.
- Create test cases based on those guesses to verify if the app handles the errors correctly.

Compatibility Testing:

- Test the app on different platforms, browsers, or devices to ensure compatibility.
- Verify that the app functions correctly and displays appropriately across different environments.

Usability Testing:

- Evaluate the app's user interface and interactions from the perspective of an end-user.
- Test common user scenarios and assess the app's ease of use, intuitiveness, and overall user experience.

Security Testing:

- Test the app for potential security vulnerabilities or weaknesses.
- Verify if the app handles user authentication, data encryption, and access control appropriately.

Performance Testing:

- Test the app's performance under different load conditions, such as a high number of concurrent users or large data sets.
- Verify if the app responds within acceptable time limits and performs efficiently.

During black box testing, test cases are designed based on the app's specifications, requirements, and user expectations. The focus is on validating the functionality, user interactions, and expected outputs without considering the internal implementation details of the app.

7.4 White-Box Testing

White box testing, also known as structural testing or glass box testing, is a software testing technique that examines the internal structure and implementation details of the application. It aims to ensure that the code functions as intended and covers all possible execution paths. Here are some examples of white box testing techniques that can be applied to the KIET Event Management App:

Unit Testing:

- Test individual units or components of the app, such as functions or methods, to verify their correctness.
- Use techniques like code coverage analysis (e.g., statement coverage, branch coverage) to ensure that all code paths are exercised.

Integration Testing:

- Test the interaction between different components or modules of the app to ensure they work together seamlessly.
- Verify the flow of data and control between the modules and check for any integration issues or errors.

Path Testing:

- Identify and test different paths or execution flows through the app, including both positive and negative scenarios.
- Execute test cases that cover all possible paths within the code to ensure complete coverage.

Decision Coverage:

- Ensure that every decision point in the code (e.g., if statements, switch cases) is tested for both true and false conditions.
- Validate that the app makes the correct decisions based on the specified conditions.

Code Review:

- Analyse the code and its structure to identify any potential issues or vulnerabilities.
- Review the adherence to coding standards, best practices, and potential optimizations.

Performance Testing:

- Assess the app's performance from a code perspective, such as identifying any bottlenecks or inefficient algorithms.
- Measure the execution time of critical code sections and evaluate resource usage.

Security Testing:

- Review the code for potential security vulnerabilities, such as SQL injection, cross-site scripting (XSS), or authentication weaknesses.
- Verify the implementation of secure coding practices, data encryption, and access control mechanisms.

Error Handling Testing:

- Test how the app handles and recovers from unexpected errors or exceptions.
- Validate that error messages are clear, meaningful, and do not expose sensitive

information.

Code Coverage Analysis:

- Use tools to measure the code coverage achieved by the tests, such as statement coverage, branch coverage, or path coverage.
- Aim for high code coverage to ensure that all parts of the code are exercised.

During white box testing, the tester has access to the application's internal code, allowing for amore detailed examination of its behavior.

7.5 System Testing

System testing is a level of software testing that evaluates the complete system as a whole, rather than focusing on individual components or modules. It ensures that all components of the KIET Event Management App work together seamlessly and meet the specified requirements. Here are some examples of system testing techniques that can be applied to the app:

Functional Testing:

- Verify that all functional requirements of the app are met.
- Test various functionalities such as event creation, registration, club directory search, user log in and registration, event notifications, etc.
- Validate that the app behaves as expected and produces the correct outputs based on different inputs.

User Interface Testing:

- Test the graphical user interface (GUI) of the app for usability, consistency, and responsiveness.
- Check the layout, navigation, buttons, forms, and other UI elements to ensure they are visually appealing and intuitive.
- Validate that the app adheres to the design guidelines and provides a seamless user experience.

Performance Testing:

- Evaluate the performance of the app under different load conditions.
- Measure response times, throughput, and resource utilization to ensure the app can handle the expected user load without significant degradation.
- Identify and address any performance bottlenecks or scalability issues.

Compatibility Testing:

- Test the app on different devices, platforms, and browsers to ensure compatibility.
- Verify that the app works correctly on various operating systems (e.g., iOS, Android) and different screen sizes.
- Validate that the app functions properly on different web browsers (if applicable).

Security Testing:

- Assess the app's security measures to protect user data and prevent unauthorised access.
- Perform vulnerability scanning, penetration testing, and authentication testing to identify and address any security vulnerabilities.
- Test the app's resilience against common security threats, such as cross-site scripting (XSS) and SQL injection.

Integration Testing:

- Test the integration of the app with external systems, such as databases, mapping services, or notification services.
- Validate that data is exchanged correctly between the app and external systems.
- Verify that the app's functionality remains intact when integrated with other systems.

Recovery Testing:

- Simulate system failures or interruptions and evaluate the app's ability to recover and resume normal operation.
- Test scenarios such as unexpected shutdowns, network failures, or interrupted database connections.
- Ensure that the app can gracefully handle such situations and recover without data loss or integrity issues.

Regression Testing:

- Re-test previously tested features and functionalities to ensure that recent changes or additions did not introduce new bugs or regressions.
- Execute a set of comprehensive test cases to cover critical areas of the app and ensure that no existing functionality is compromised.

CHAPTER 8

CONCLUSION

8.1 Conclusion

In conclusion, developing a blog website has been a rewarding project that combines creativity, technical skills, and strategic thinking. By integrating a user-friendly interface, responsive design, and robust content management system, the website not only enhances user engagement but also ensures a seamless browsing experience across various devices. The implementation of SEO best practices and social media integration further amplifies the website's reach, driving organic traffic and fostering a growing community of readers. This project has underscored the importance of meticulous planning, continuous improvement, and adaptability to evolving digital trends, setting a solid foundation for future enhancements and expansion. Overall, the blog website stands as a dynamic platform for sharing ideas, connecting with audiences, and building a meaningful online presence.

In conclusion, developing a blog website has been a rewarding project that combines creativity, technical skills, and strategic thinking. By integrating a user-friendly interface, responsive design, and a robust content management system, the website not only enhances user engagement but also ensures a seamless browsing experience across various devices. The implementation of SEO best practices and social media integration further amplifies the website's reach, driving organic traffic and fostering a growing community of readers.

Moreover, the project has highlighted the importance of meticulous planning, collaboration, and adaptability to evolving digital trends. Working through challenges such as cross-browser compatibility and load time optimization has not only enhanced technical proficiency but also underscored the value of resilience and problem-solving in web development.

Throughout the project, significant emphasis was placed on creating compelling content that resonates with the target audience, utilizing multimedia elements such as images, videos, and infographics to enrich the user experience.

8.2 Future Scope

The future scope of the blog website is both expansive and promising, offering numerous opportunities for growth, innovation, and deeper audience engagement. Here are some potential areas for future development:

Personalized Content Recommendations:

Leveraging machine learning algorithms to analyze user behaviour and preferences can help deliver personalized content suggestions. This customization can enhance user experience by making content discovery more relevant and engaging.

Enhanced Multimedia Integration:

Expanding beyond traditional blog posts, incorporating rich multimedia content such as podcasts, webinars, and interactive infographics can cater to diverse audience preferences and make the platform more dynamic.

Community Building Features:

Adding features like user profiles, discussion forums, and comment sections can foster a sense of community among readers. Enabling user-generated content and contributions can further enhance engagement and loyalty.

Advanced SEO and Analytics Tools:

Implementing more sophisticated SEO tools and analytics dashboards can help track content performance more effectively, providing deeper insights into user behavior and content impact. This data-driven approach can guide future content strategies and marketing efforts.

Monetization Strategies:

Exploring various monetization avenues such as affiliate marketing, sponsored posts, premium content subscriptions, and e-commerce integrations can generate revenue streams and sustain the website's growth.

Mobile App Development:

Creating a dedicated mobile app can enhance accessibility and user experience, catering to the growing number of users who access content on the go. Features like push notifications can keep readers engaged with the latest updates.

Internationalization and Localization:

Expanding the blog's reach by offering content in multiple languages and tailoring it to different cultural contexts can attract a global audience. This involves not only translation but also adapting content to resonate with local audiences.

Integration with Emerging Technologies:

Staying at the forefront of technology by integrating features such as voice search, augmented reality (AR), and virtual reality (VR) can provide immersive and innovative experiences for users.

BIBLIOGRAPHY

- "Full-Stack React Projects" by Shama Hoque
- "Learning React: Modern Patterns for Developing React Apps" by Alex Banks and Eve Porcello
- "MongoDB: The Definitive Guide" by Shannon Bradshaw, Eoin Brazil, and Kristina Chodorow
- "Express in Action: Writing, building, and testing Node.js applications" by Evan Hahn
- "Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node" by Vasani Subramanian

Documentation

- MongoDB Documentation
- Express.js Documentation
- React Documentation
- Node.js Documentation