

# **Project Report**

On

## **Frontend Web Development**

Submitted

In Partial Fulfillment of

### **BACHELOR OF COMPUTER APPLICATIONS (BCA)**

**Submitted by:**

Name: Yashveer Singh

Roll No: 24/SCA/BCA(AI&ML)/075

**Under the Supervision of:**

Dr. Sakshi Gupta

Assistant Professor, SCA



**School of Computer Applications**

**Manav Rachna International Institute of Research and Studies**

**(DEEMED TO BE UNIVERSITY)**

Sector-43, Aravalli Hills Faridabad

– 121001

**June 2025**

# Declaration

I do hereby declare that this project work entitled “Web Development” submitted by me for the partial fulfillment of the requirement for the award of BACHELOR OF COMPUTER APPLICATIONS is a record of my own work. The report embodies the findings based on my study and observation and has not been submitted earlier for the award of any degree or diploma to any Institute or University.

SIGNATURE

Name: Yashveer Singh

Roll No:24/SCA/BCA(AI&ML)/075

Date: 17.7.25

## **Certificate from the Guide**

This is to certify that the project report entitled “Web Development” submitted in partial fulfillment of the degree of **BACHELOR OF COMPUTER APPLICATIONS** to Manav Rachna International Institute of Research and Studies, Faridabad is carried out by Ms. Yashveer Singh(24/SCA/BCA(AI&ML)/075) under my guidance.

Signature of the Guide

Name: Dr. Sakshi Gupta

Date: 17.7.25

**Head of Department**

**Prof. Dr. Suhail Javed Quraishi**

# ACKNOWLEDGEMENT

I gratefully acknowledge for the assistance, cooperation, guidance and clarification during the development of Web Development website. My extreme gratitude to **Dr. Sakshi Gupta -Assistant Professor, SCA** who guided us throughout the project. Without her willing disposition, spirit accommodation, frankness, timely clarification and above all faith in us, this project could not have been completed in due time. Her readiness to discuss all important matters at work deserves special attention of. I would also like to thank all the faculty members of the computer application department for their cooperation and support. I would like to give special gratitude to **Dr. Raj Kumar–Associate Professor** for his guidance during the project.

I would like to extend my sincere gratitude to **Prof. Dr. Suhail Javed Quraishi – HOD** for his valuable teaching and advice. I would again like to thank all faculty members of the department for their cooperation and support. I would like to thank non-teaching staff of the department for their cooperation and support.

I would like to extend special thanks to Prof. **Dr. Brijesh Kumar, Dean - SCA** for her valuable insight and motivation.

I perceive this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

Name: Yashveer Singh

Roll No: 24/SCA/BCA(AI&ML)/075

Date: 17/7/25

# INDEX

Topic	Page No	Sign	Remark
Introduction	6		
System Study	7		
Feasibility Study	8-9		
Project Monitoring System	10-11		
System Analysis	12-20		
System Design	21		
Input/Output Form Design	22-24		
System Testing	25		
System Implementation	26		
Documentation	27-28		
Scope of the Project	29		
Bibliography	30		

## **INTRODUCTION**

### **(a) About Organization:**

CODTECH IT SOLUTIONS PVT. LTD is a leading provider of IT services and internship programs focused on nurturing technical skills among students and freshers. The company offers a real-world experience in development, design, and deployment of modern software systems through task-based learning environments.

### **(b) Aims & Objectives:**

The primary objective of this internship was to enhance practical frontend development skills by engaging in four structured projects, each reflecting a real-world problem. The internship aimed to improve knowledge of HTML, CSS, JavaScript, and React.js while working on responsive designs and interactive UIs.

### **(c) Manpower:**

The internship program was overseen by Dr. Sakshi Gupta, who guided the progress and quality of each task. Interns were grouped under project mentors and communication was maintained through online platforms including WhatsApp and GitHub.

# **SYSTEM STUDY**

## **A: Existing System Study**

In traditional learning and communication systems, users often rely on static platforms or manual setups. Portfolio showcases are done manually through PDFs or offline resumes. Real-time communication lacked web-based channels, and quiz assessments were conducted using paper formats.

## **B: Proposed System Design**

The new system includes four modern web-based applications that address these gaps:

- Quiz Application (JavaScript)
- Real-Time Chat Application (React.js)
- Personal Portfolio Website (HTML/CSS)
- E-learning Platform UI (React.js)

These systems are user-friendly, dynamic, and accessible via web browsers with mobile responsiveness.



# FEASIBILITY STUDY

## A: Technical Feasibility

The applications use web technologies like HTML, CSS, JavaScript, and React.js, which are supported on all major platforms. The developer was already familiar with these technologies, ensuring smooth development.

### 1. Programming Language Used

- HTML- For structure of web pages
- CSS- For styling and layout
- JavaScript- For basic interactivity (calculator logic)

### 2. Tools used

- Code editor: VS Code
- Browser: Google Chrome or any modern browser for testing
- No backend, no hosting tools used in basic version

### 3. Libraries & Framework

- None used in current versions
- All features implemented using HTML, CSS, and JavaScript

## **B. Behavioural Feasibility**

The interfaces are simple and easy to use, based on standard UX/UI practices. Positive feedback was received during testing and peer reviews

### **1. User Interaction**

- Buttons, links, and clickable elements are clearly visible and functional
- Basic but intuitive design with simple navigation

### **2. User Satisfaction**

- Suitable for beginners and basic users
- Clear Structure of content and interaction
- Portfolio is personalized and easy to follow

### **3. Ease of Use**

- Clean layout with organized sections
- Buttons and links clearly labelled
- Minimalist design helps user focus on content

## **C. Economic Feasibility**

All development was done using free and open-source tools. No additional costs were incurred for licenses, hosting, or deployment. Thus, the project is highly economical.

## PROJECT MONITORING SYSTEM

### A. Gantt chart

Week	Task
Week 1	Planning & Research, GitHub setup, Quiz App layout
Week 2	Quiz functionality, Chat UI layout
Week 3	HTML/CSS/JS Development, Chat features, Portfolio website development
Week 4	E-learning UI, Testing, Bug Fixing, Presentation, Final report

### Timeline Overview:

#### A. Planning and Research

##### 1. Project Scope and objectives

- The scope is to Complete Four Tasks:  
A **Quiz** app with Scoring and Dynamic questions loading  
A **Real-Time Chat App** interface with message history  
A **Portfolio Website** to showcase skills, resume and projects  
A **Elearning Platform** with styled and interactive multipage interface

- These projects are front-end only, built with HTML, CSS and JS
- No backend/database integration is involved

## **Objectives:**

- Built simple, functional, and visually clean web applications
- Improve front-end development skills and UI/UX understanding
- Ensure basic responsiveness and cross-device compatibility

## **2. User Requirements**

- Identify the target audience and their needs.
- Gather feedback through surveys or interviews.

## **B. Development Phase**

- **Front-End Development**
  - ☐ Develop the front-end using HTML, CSS, and JavaScript.
  - ☐ Implement features like the home page.
- **Back-End Development**
  - ☐ Set up the server and database using suitable technologies (e.g., Node.js, Express.js, MongoDB).
  - ☐ Implement user authentication, management, and streaming functionalities.
- **Testing and Debugging**
  - ☐ Perform initial testing and debugging to fix any issues in the code.

## **C. Testing Phase**

- **Functional Testing:**
  - ☐ Test all features to ensure they work as expected
- **Usability Testing:**
  - ☐ Conduct usability tests with real users to gather feedback.

- ☐ Identify any issues in the user experience and make improvements.

- **Performance Testing:**

- ☐ Test the website's performance, including loading times and responsiveness.
- ☐ Optimize code and resources for better performance.
- ☐ Deploy to the live environment.
- ☐ Gather feedback and plan for future updates and improvements.

# SYSTEM ANALYSIS

## A. Requirement Specification

Each application had the following key features:

- Quiz App: Dynamic question loading, score calculation, answer feedback
- Chat App: Real-time messaging, responsive layout, scrollable history
- Portfolio Website: Navigation bar, project sections, responsive design
- E-learning Platform: Course list, video embedding, progress tracking

## B. System Flowcharts

### 1. User Flowchart

Start --- Homepage --- Choose  
page(Quiz/Chatapp/Portfolio/E-learning) --- View content ---  
Interact (click buttons) --- End

### 2. Search Functionality Flowchart

Start --- User Inputs Keyword --- System searches static content ---  
Display matching section (Content, History, etc) --- End

## c. DFDs/ERDs (up to level 2):

### 1. Chat App

#### External Entities

- **User**: Sends messages, reads messages, logs in/out.
- **Server** (e.g. Firebase or backend): Stores/retrieves data.

#### Process 0: Chat-App System

- Receives input from User
- Sends data to Server
- Returns responses to User

### Level 1 DFD

#### Processes

1. **Login / Authentication**
2. **Send Message**
3. **Receive Message**
4. **User Status Update**

#### Data Flows

- **User** → (username/password) → *Login Process*
- **Login** → (auth token) → *Client UI*

- **Client** → (message content + token) → *Send Message Process*
- **Send Message** → (DB write) → *Server/DB*
- **Receive Message** → (notifications/flows) → *Client*
- **User Activity** → (status update) → *Server*

## Level 2

- **Client** → (message + auth) → *2.1 Verify Token* → if valid → *2.2 Format Message* → *2.3 Persist to Messages (DB)* → *2.4 Broadcast to Recipients* → **Client UI**

## Entities & Key Attributes:

- **User** (user\_id PK, username, password\_hash, status, profile\_info)
- **Chat** (chat\_id PK, is\_group\_flag)
- **ChatUser** (chat\_id FK, user\_id FK, last\_read\_timestamp)
- **Message** (message\_id PK, chat\_id FK, sender\_id FK, timestamp, content, optional\_media\_url)

## Relationships:

- *User* ↔ (many-to-many via ChatUser) ↔ *Chat*
- *Chat* 1—*Message*
- *User* 1— *Message* (as sender)



## **2. Quiz-App**

**Context / Level 0 DFD**

**External Entities:**

- **User**
- **Quiz-App Backend/API (e.g. Retrieves questions, stores scores)**

**Process 0: Quiz System**

**– User selects quiz → API serves questions → User answers → Results are returned & optionally stored.**

**Level 1 DFD**

**Processes:**

- 1. Load Quiz (fetch questions metadata)**
- 2. Start Quiz (fetch questions set)**
- 3. Submit Answer / Navigate**
- 4. Finish Quiz (calculate + return score, optionally save result)**

**Data Stores:**

- **D1: Questions (question\_id, text, options, correct\_answer, quiz\_id)**
- **D2: Quiz Sessions (session\_id, user\_id, quiz\_id, start\_time, end\_time, score)**

## Level 2 DFD (Submit Answer Flow)

1. User → (answer + session\_id) → 2.1 Validate Session → 2.2 Check Answer → increments score or logs attempt → 2.3 Advance Question or End Quiz → UI updated.

## ERD Sketch

- Quiz (quiz\_id PK, title, description)
- Question (question\_id PK, quiz\_id FK, text, correct\_option)
- Option (option\_id PK, question\_id FK, text)
- Session (session\_id PK, user\_id FK [optional], quiz\_id FK, start\_time, end\_time, score)
- Answer (answer\_id PK, session\_id FK, question\_id FK, selected\_option\_id, is\_correct)

## Relationships:

- Quiz 1– Question
- Question 1–Option
- Session 1– Answer
- Answer ↔ Question, Session

## 3. Portfolio

### Context / Level 0 DFD

### **External:**

- **Visitor (browses portfolio)**
- **Owner (updates content via GitHub CMS backend)**

### **Process 0: Portfolio System**

- **Visitor browses content pulled from CMS.**

### **Level 1 DFD**

#### **Processes:**

- 1. Fetch Projects**
- 2. Fetch Blog Posts**
- 3. Fetch About / Contact Info**

#### **Data Stores:**

- **D1: Projects (id, title, description, link, tech tags)**
- **D2: Blog (id, title, content, date, tags)**
- **D3: About Info (static content)**

### **Level 2 DFD (Fetch Projects)**

- **Visitor → Request Projects → 2.1 Query CMS/Repo → data returned → UI renders.**

### **ERD Sketch**

- **Project (project\_id PK, title, description, repo\_link, live\_link, created\_date)**

- **BlogPost** (post\_id PK, title, content, created\_date)
- **Tag** (tag\_id PK, name)
- **ProjectTag** (project\_id FK, tag\_id FK)
- **PostTag** (post\_id FK, tag\_id FK)

#### **Relationships:**

- **Project M-N Tag via ProjectTag**
- **BlogPost M-N Tag via PostTag**

## **4. E-learning**

### **Context / Level 0 DFD**

#### **External:**

- **Student** (browses courses, watches lessons, takes quizzes)
- **Instructor / Admin** (manages courses, content)

#### **Process 0: E-learning System**

– **Catalog courses → enroll → consume lessons → assess → track progress.**

### **Level 1 DFD**

#### **Processes:**

- 1. Browse Courses**
- 2. Enroll in Course**
- 3. Take Lesson**

#### **4. Take Quiz / Submit Assignment**

#### **5. Track Progress & Score**

#### **Data Stores:**

- **D1: Courses (course\_id, title, description, instructor\_id)**
- **D2: Lessons (lesson\_id, course\_id, title, content\_url)**
- **D3: Quizzes (quiz\_id, course\_id, lesson\_id)**
- **D4: QuizQuestions (question\_id, quiz\_id, text, correct\_answer)**
- **D5: Progress (progress\_id, student\_id, course\_id, lesson\_id, completed)**

#### **Level 2 DFD (Take Quiz)**

- **Student → selected answers → 2.1 Validate Enrollment → 2.2 Grade Quiz → 2.3 Update Progress/Score → return result.**

#### **ERD Sketch**

- **User (user\_id PK, name, email, role)**
- **Course (course\_id PK, title, description, instructor\_id FK)**
- **Lesson (lesson\_id PK, course\_id FK, title, content\_url)**
- **Quiz (quiz\_id PK, lesson\_id FK, title)**
- **Question (question\_id PK, quiz\_id FK, text, correct\_option)**

- **Option (option\_id PK, question\_id FK, text)**
- **Enrollment (enroll\_id PK, user\_id FK, course\_id FK, enroll\_date)**
- **Progress (progress\_id PK, user\_id FK, lesson\_id FK, completed\_at)**
- **QuizSubmission (submission\_id PK, user\_id FK, quiz\_id FK, score, submitted\_at)**
- **Answer (answer\_id PK, submission\_id FK, question\_id FK, selected\_option\_id, is\_correct)**

#### **Relationships:**

- **Course 1– Lesson**
- **Lesson 1– Quiz**
- **Quiz 1– Question**
- **Question 1– Option**
- **Course M-N User via Enrollment**
- **User-Lesson via Progress**
- **QuizSubmission logs results; has multiple Answers.**

# **System Design**

## **File/Data Design**

### **1. Quiz App**

- Local JSON question set
- Score variable

### **2.Chat App**

- State variables using React Hooks

### **3. Portfolio**

- Static HTML sections
- CSS for layout

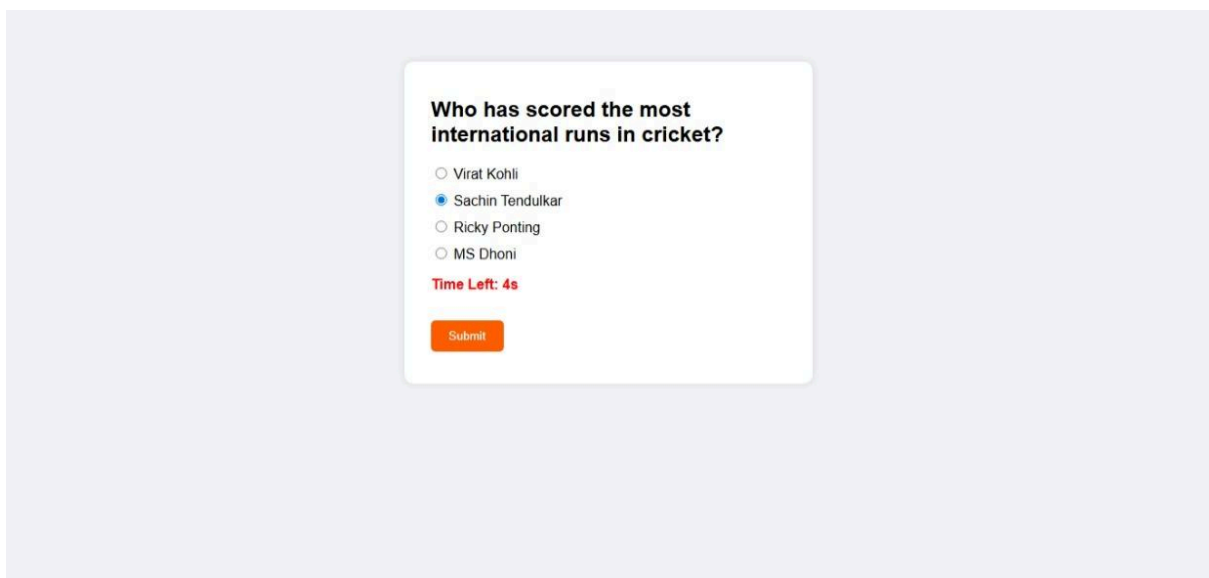
### **4. E-learning Platform**

- Components for course cards
- Progress bars
- Videos

## INPUT / OUTPUT FORM DESIGN

### (a) Screen Design

#### 1. Quiz App



The screenshot shows a quiz app interface on a light gray background. A white rounded rectangle contains the question and options. The question is "Who has scored the most international runs in cricket?". There are four radio button options: Virat Kohli, Sachin Tendulkar (selected), Ricky Ponting, and MS Dhoni. Below the options, it says "Time Left: 4s" in red. At the bottom of the white box is an orange "Submit" button.

**Who has scored the most international runs in cricket?**

☐ Virat Kohli

☒ Sachin Tendulkar

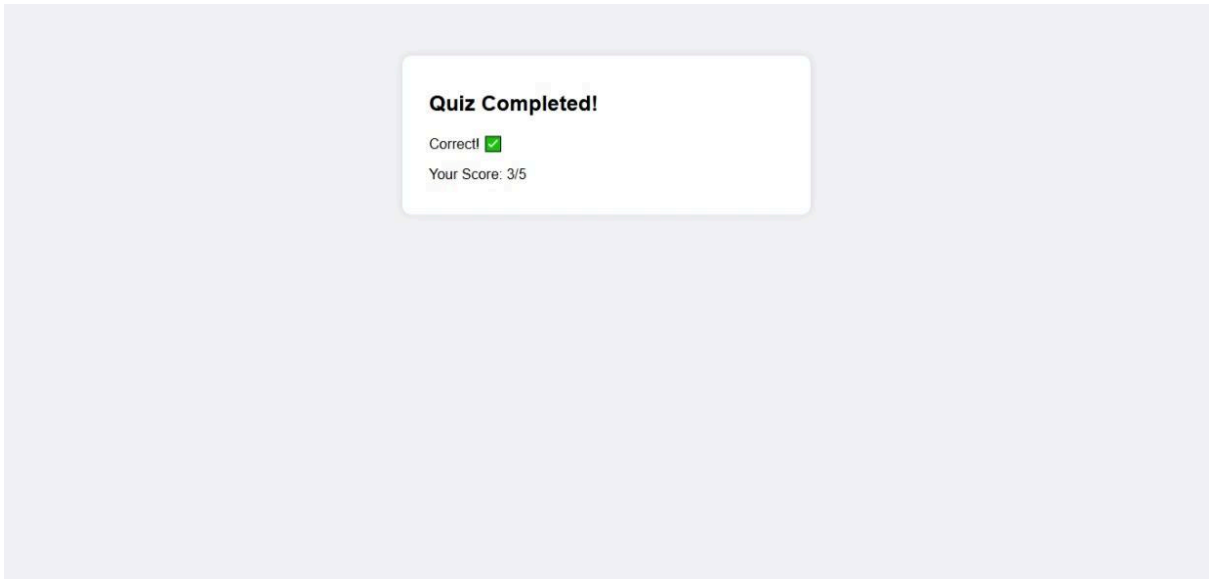
☐ Ricky Ponting

☐ MS Dhoni

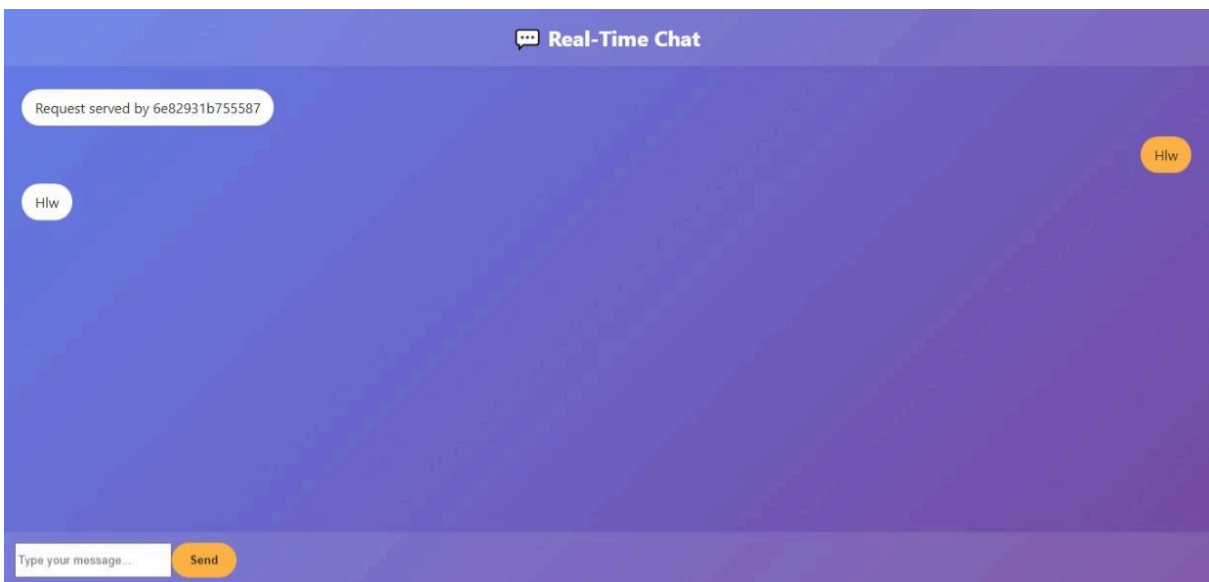
**Time Left: 4s**

**Submit**





## 2. Chat App



## 3. Portfolio



## 4. Elearning Platform

## Upgrade Your Skills Online

Interactive courses in programming, design, and more. Start learning today!

[Browse Courses](#)

### Popular Courses

#### HTML & CSS Basics

Learn how to build beautiful websites from scratch.

[View Course](#)

#### Python for Beginners

Start coding in Python with hands-on exercises.

[View Course](#)

### About Learnify

Learnify is a simple e-learning platform designed to help you learn new skills at your own pace. Our courses are interactive, beginner-friendly, and completely free.

### Contact Us

[Send Message](#)

# System Testing

## Preparation of Test Data:

- Dummy quiz questions
- Sample user messages
- Test project entries and course titles

## Testing with Live Data:

- All apps tested on Chrome and Edge
- Responsive design validated using browser dev tools

## Test Cases and Results:

Test Case	Input	Expected Output	Result
1.Quiz answer	Correct option	Score increases	Pass
2.Chat message	Text input	Message appears instantly	Pass
3.Portfolio nav	Click “Projects”	Scroll to project section	Pass
4.E-learning video	Play button	Video starts	Pass

# SYSTEM IMPLEMENTATION

## (a) System Requirements (Hardware/Software):

### 1. Hardware Requirements

- **Processor:** Minimum Dual-core 1.6 GHz or higher for smooth development.
- **RAM:** At least 2 GB to run code editor and browser simultaneously.
- **Storage:** Minimum 500 MB free space to store HTML, CSS, JS, and media files.
- **Display:** A 13-inch or larger screen for better visibility and layout testing.
- **Input Devices:** Standard keyboard and mouse for coding and navigation.
- **Internet Connection:** Needed for live testing, hosting, and feedback collection.

### 2. Software Requirements

- **Operating System:** Compatible with Windows 10/11, Linux, or macOS.
- **Code Editor:** Visual Studio Code or any lightweight text editor (e.g., Sublime Text).
- **Web Browser:** Google Chrome, Firefox, or Microsoft Edge (latest version preferred).
- **Deployment Platform (Optional):** GitHub Pages, Netlify, or Vencel for hosting.
- **File Type Support:** Works with .html, .CSS, .JS, .pdf, .jpg, and .PNG.

### 3. Optional Tools

- **Live Server Extension:** For real-time preview while editing in VS Code.
- **W3C Validator:** To check HTML and CSS code validity.
- **Google Page Speed Insights:** To test website performance and loading speed.

## Documentation

Each application was modularly developed. Comments were included to improve readability. GitHub repositories were maintained for version control and collaborative tracking.

Documentation included detailed reports of planning, system analysis, design, testing, and deployment. It covered project objectives, tools used, code structure, test cases, user feedback, and improvements. This helps future developers understand, maintain, and enhance the projects efficiently.

## Components

The web application clone application consists of the following key components:

- **Front-end:** HTML, CSS, JavaScript
- **Back-end:** Node.js

## Technology Stack

- **Front-end:** HTML5, CSS3, JavaScript (ES6+)
- **Back-end:** Node.js
- **Development Tools:** Visual Studio Code, Git/GitHub

## System Requirements

### Hardware Requirements

- **Server:** Dual-core processor, 4GB RAM, SSD storage
- **Network:** Stable internet connection

### Software Requirements

- **Operating System:** Linux/Windows Server
- **Programming Languages:** HTML, CSS, JavaScript (Node.js)

## **Text editor**

- **Visual Studio Code:** Integrated development environment for coding, debugging, and version control.

## **Version Control**

- **Git:** Distributed version control system for tracking changes, managing codebase, and facilitating collaboration.

## **System Design File/Data Design**

- **HTML Files:** Structured web pages for user interface elements.
- **CSS Files:** Stylesheets for visual design and layout.
- **JavaScript Files:** Client-side scripting for interactive features and user actions.

## **Implementation**

### **Development Environment Setup**

- Installation and configuration of necessary software components Node.js.
- Setup of development environment in Visual Studio Code.
- Integration with version control system (Git/GitHub) for collaborative development.

### **Coding Standards**

- Adherence to coding conventions and best practices for HTML, CSS, JavaScript, and Node.js.
- Documentation of coding standards to ensure consistency and maintainability of codebase.

### **Version Control Workflow**

- Branching strategy for feature development, bug fixes, and release management.
- Commit practices, pull request reviews, and merge strategies to maintain code quality and integrity.

## **SCOPE OF THE PROJECT**

This internship demonstrated the practical implementation of frontend development techniques. It provided experience with real-world tools and strengthened skills in UI/UX, JavaScript logic, React components, and GitHub versioning.

Each project is built using HTML, CSS, and JavaScript to demonstrate front-end skills. The focus is on clean UI design, responsive layouts, and functional features. The project serves educational purposes, showcasing beginner-level development, user interaction, and personal branding in a web environment.

## **1. Functional Scope**

- Perform basic Quiz App.
- E-learning with Videos.
- Showcase skills, projects in the portfolio.
- Enable navigation across website sections.

## **2. Technical Scope**

- Built using HTML, CSS, and JavaScript (no backend).
- Internal CSS and inline JavaScript for interactivity .
- Deployed using static hosting platforms like GitHub Pages .
- Developed using VS Code with optional Live Server extension.

## **3. User Experience (UX) Scope**

- Simple, intuitive interface with clean layout.
- Responsive design for desktop, tablet, and mobile users.
- Easy navigation with clear headings and well-structured content..



# BIBLIOGRAPHY

1. Yashveer976679. (2024). Chat App [Source code]. GitHub. <https://github.com/yashveer976679/Chat-app>
2. Yashveer976679. (2024). Quiz App [Source code]. GitHub. <https://github.com/yashveer976679/Quiz-app>
3. Yashveer976679. (2024). Portfolio Website [Source code]. GitHub. <https://github.com/yashveer976679/porfolio>
4. Yashveer976679. (2024). E-learning Platform [Source code]. GitHub. <https://github.com/yashveer976679/E-learning>
5. [React.js](https://reactjs.org)-<https://reactjs.org>
6. Mozilla-<https://developer.mozilla.org>
7. W3Schools-<https://www.w3schools.com>