# Typing Speed Checker

**A PROJECT REPORT**

***Submitted by***

**Lalit Kumar (23BCS13259)**
**Md. Khushtar Ali (23ICS10007)**
**Vashish Jaswal (23BCS11867)**
**Tajinder Singh (23BCS13722)**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

**IN**

COMPUTER SCIENCE ENGINEERING


**Chandigarh University**

JULY 2025 – NOVEMBER 2025

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

# BONAFIDE CERTIFICATE

Certified that this project report **"Typing Speed Checker"** is the Bonafide work of "**Lalit Kumar, Md. Khushtar Ali , Vashish Jaswal, Tajinder Singh "** who carried out the project work under my/our supervision.

**SIGNATURE**

Dr. Gagandeep Singh

HEAD OF THE DEPARTMENT

CSE 3rd Year

**SIGNATURE**

Aniket Raj

SUPERVISOR

CSE 3rd Year

Submitted for the project viva-voce examination held on

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

# TABLE OF CONTENTS

# CHAPTER 1
# INTRODUCTION

## 1.1 Identification of Need / Relevant Contemporary Issues

In the present digital world, typing has become a daily activity for almost every individual. Whether it is a student writing assignments, a developer writing code, an office employee preparing reports, or someone using social media to chat, typing is essential. The speed and accuracy with which a person can type directly affects the amount of time and effort they spend in completing written communication tasks.

Earlier, typing was mainly associated with typewriters and required special training. But now, in the era of computers and smartphones, typing is something everyone is expected to do naturally. Despite this, many people do not know **how fast they can type**, **how accurate they are**, and **how to improve** their typing ability. As a result, they spend more time writing and correcting mistakes.

To solve this issue, a **Typing Speed Checker** tool is required. It should help users:

- Test their current typing speed,
- Identify mistakes,
- Improve accuracy,
- Develop confidence while typing.

Our proposed project addresses this need by developing a simple, interactive, and user-friendly web application.

## 1.2 Identification of Problem

Some common issues faced by beginner and average typists include:

- Slow typing speed,
- High number of typing errors,
- Lack of awareness of typing posture and keyboard layout,
- No continuous practice routine,
- No accurate tool to evaluate improvement.

Existing typing websites sometimes look outdated, contain too many ads, do not provide customization, or do not give meaningful performance feedback. So, a more engaging and visually pleasing platform is needed where users can **practice typing**, **track their speed**, and **improve gradually**.

This project provides a clean and modern interface with:

- Start and Stop control on testing,
- Automatic calculation of WPM (Words Per Minute),
- Accuracy calculation,
- Ability to change the test paragraph anytime,
- Light and Dark mode for comfortable usage.

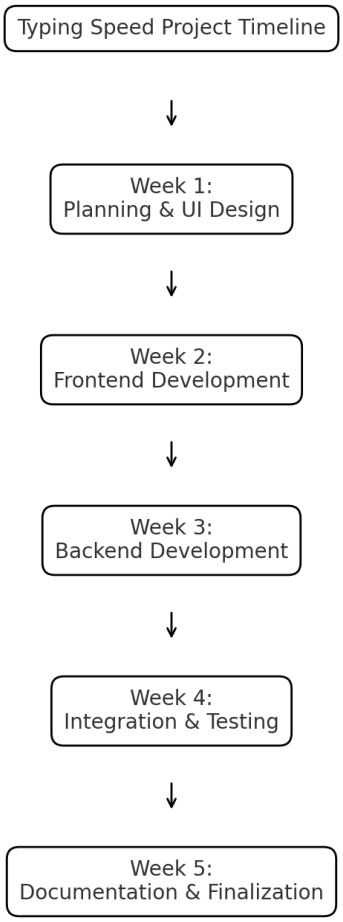
## 1.3 Identification of Tasks

To develop the project, the work was divided into manageable tasks:

| Task No. | Task Name | Description |
|---|---|---|
| 1 | Requirement Collection | Identify features and user expectations |
| 2 | UI/UX Design | Design layout, controls, and screen visual appearance |
| 3 | Frontend Development | Develop React components for user interaction |
| 4 | Backend Development | Create APIs for serving random paragraphs |
| 5 | Database (Optional) | Store and retrieve practice paragraphs from MongoDB |
| 6 | Testing | Check working of timer, WPM calculation & UI |
| 7 | Final Integration | Combine all parts and deploy locally |

## 1.4 Timeline

| Week | Work Done |
|---|---|
| 1 | Planning and interface design |
| 2 | Built frontend layout and input system |
| 3 | Built backend paragraph API and linked it |
| 4 | Added timer, text comparison logic, WPM & accuracy calculation |
| 5 | Testing, improvements, documentation & final completion |




## 1.5 Organization of the Report

This report contains 4 main chapters:

- **Chapter 1:** Introduction to problem and project purpose
- **Chapter 2:** Study of existing tools and background research
- **Chapter 3:** System design steps, architecture and development methodology
- **Chapter 4:** Results, output evaluation and validation

A conclusion and references are added at the end.

# CHAPTER 2

# LITERATURE REVIEW/BACKGROUND STUDY

## 2.1. Timeline of the reported problem

The skill of typing has evolved significantly over the past century. In the early 1900s, typing was considered a specialized professional skill and was mostly taught in dedicated typing institutes. These institutes used manual typewriters, and learners were trained to type without looking at the keys. The focus during that time was primarily on *accuracy* and *rhythmic typing techniques*. Only people who worked as stenographers, clerks, or secretaries were expected to learn typing.

However, with the introduction of personal computers in the late 20th century, typing gradually shifted from being a specialized job requirement to a basic computer skill. As computers became part of homes, schools, and offices, typing was no longer limited to professionals. Children and students also began to use keyboards for learning and assignments. During this phase, simple typing tutor software such as **Mavis Beacon Teaches Typing** and **Rapid Typing Tutor** became popular. These programs provided structured lessons but lacked real-time performance feedback and interactive learning experiences.

With the growth of the internet and digital communication, typing became even more essential. People now rely heavily on messaging apps, emails, documentation tools, and social media platforms, all of which require clear and quick typing. In the 2000s, typing practice websites and online speed test platforms emerged. These web-based tools introduced interactive learning, competitive typing races, and accuracy scoring. Popular platforms like **10FastFingers**, **Typing.com**, and **Keybr.com** made typing practice more accessible but did not always provide customizable or user-friendly interfaces suitable for all users.

Today, typing speed and accuracy have become an expected ability in schools, colleges, and almost every workplace. Students are required to prepare assignments digitally, and working professionals are expected to type reports, emails, and documentation efficiently. Yet, **formal typing training is rarely included in modern education systems**, which means many individuals rely on self-learning. Without proper guidance or consistent practice, they may develop inefficient typing habits, slower speeds, and higher error rates.

This gap between expected typing skill levels and the lack of structured training creates a clear need for accessible, simple-to-use, interactive typing improvement tools. A typing speed checker that provides real-time feedback, tracks progress, and encourages regular practice can help bridge this gap and support learners in developing confident and efficient typing skills.

## 2.2. Existing Solutions

Some well-known typing websites include:

| Website | Features | Limitations |
|---|---|---|
| Typing.com | Lessons and practice courses | Lessons feel repetitive |
| 10FastFingers.com | Speed competitions and timed tests | No dark mode, limited paragraph control |
| Keybr.com | Algorithm-based random words | Does not focus on natural sentence typing |

Most platforms lack **theme control**, **API flexibility**, and **custom integration** which modern developers require.

## 2.3. Bibliometric analysis

Bibliometric analysis refers to studying previous research, articles, and data related to typing skill development and digital literacy. Many earlier studies focused on **touch typing**, which means typing without looking at the keyboard. These studies showed that touch typing increases both speed and accuracy because the fingers develop **muscle memory** through repeated practice.

More recent educational studies highlight that typing has now become an essential academic and professional skill. Students type assignments, take online exams, and communicate digitally, while office workers type emails, reports, and documentation. However, most schools and colleges **do not include formal typing training**, which leads to uneven typing skill levels among individuals.

Research also shows that people who use structured typing tools with regular practice tend to achieve better performance compared to those who type casually. These tools provide immediate accuracy feedback, WPM calculation, and help in improving hand-eye coordination and typing confidence.

Workplace studies further suggest that faster and more accurate typing reduces task completion time and improves productivity. Therefore, bibliometric evidence supports the need for **simple, accessible, and interactive typing practice tools** that provide real-time evaluation and help users gradually improve their abilities.

## 2.4. Review Summary

From the literature reviewed, it is clear that typing has transitioned from being a specialized professional skill to a basic necessity for students, employees, and general computer users. Earlier studies focused heavily on touch typing techniques and accuracy development through repetitive practice. With the growth of technology, many digital tools and web-based platforms emerged that help users test their typing speed and improve their skills.

However, the review also highlighted several **limitations in existing solutions**. Many traditional typing tutor applications are **outdated** and provide a **static learning experience** with repetitive exercises. While modern online platforms introduced speed tests and typing games, they have their own

drawbacks such as distracting advertisements, limited customization, lack of user progress tracking, or interfaces that are not comfortable for prolonged use.

A significant gap identified in the reviewed systems is the **absence of a personalized and user-friendly environment**. Most typing tools do not offer theme switching (e.g., dark mode), which is useful for reducing eye strain. Additionally, not all platforms allow quick switching of practice content, which reduces the effectiveness of varied learning. Some tools require internet access for every test session, limiting offline usability.

Another key observation from the review is that **formal typing education is still not included in most academic institutions**, causing many individuals to rely on self-learning. Without proper guidance or regular feedback, learners struggle to improve at a consistent pace. This highlights the need for a tool that is simple enough for beginners yet effective enough for regular practice and improvement.

Based on these findings, there is a clear need for a **modern, clean, and interactive typing speed checker** that:

- Measures both **speed and accuracy**,
- Allows users to **change practice paragraphs easily**,
- Offers **light and dark mode** for comfortable use,
- Provides an **engaging and distraction-free interface**.

The proposed system in this project directly addresses these gaps by incorporating these features and presenting a smooth and user-focused typing experience.

## 2.5. Problem Definition

In today's digital environment, typing has become a daily necessity for students, professionals, and general computer users. However, most individuals are not aware of their actual typing speed and accuracy, and there are very few platforms that provide a simple and distraction-free environment for practicing typing. Existing typing test websites may contain advertisements, lack theme customization, or do not offer varied practice content. Also, many of these platforms do not calculate performance in a clear and understandable way for beginners.

Due to the absence of formal typing training in most educational systems, users rely on self-learning, which often leads to slow typing habits and repeated errors. Therefore, there is a need for a tool that helps individuals measure and improve their typing skills in an effective manner. The problem is to develop a **user-friendly typing speed checker** that can calculate Words Per Minute (WPM) and accuracy, provide practice paragraphs, and offer a comfortable interface with features like theme switching and easy test controls.

## 2.6. Goals/Objectives

- **To provide a simple and user-friendly platform for practicing typing.**
  The interface should be clean and distraction-free so that users can focus on improving their typing skills.

- **To accurately calculate typing speed in Words Per Minute (WPM).**
  This helps users understand their performance level and track their improvement over time.

- **To measure typing accuracy by comparing typed text with the original content.**
  This ensures that users not only type fast but also type correctly and minimize errors.

- **To allow users to easily start, stop, and reset the typing test.**
  These controls enable users to practice at their own pace and repeat the test whenever needed.

- **To include a feature for changing the practice paragraph anytime.**
  This prevents repetition and helps users practice a variety of sentence structures.

- **To provide optional Light and Dark mode themes for comfortable viewing.**
  Dark mode especially reduces eye strain during long practice sessions.

# CHAPTER 3
# DESIGN FLOW/PROCESS

## 3.1 Implementation plan/methodology

The development of the Typing Speed Checker project followed a structured methodology based on the **Software Development Life Cycle (SDLC)**. Each stage of the development was planned carefully to ensure smooth progress and successful implementation. The steps followed are explained below:

**1. Requirement Analysis**

In this initial phase, the main goal was to identify what the system should achieve and what features it must include. The project requirements were gathered by analyzing how typing tests generally work and what users expect from a typing speed tool. Key functional needs such as starting and stopping the timer, displaying new paragraphs, calculating WPM and accuracy, and switching between light and dark modes were finalized. Non-functional requirements like simplicity, responsiveness, and smooth performance were also defined. This step laid the foundation for designing and implementing the system efficiently.

**2. System Design**

Once the requirements were clear, the next step was to design the structure of the system. The architecture of the application was planned using the **MERN stack**, which includes **MongoDB, Express.js, React.js, and Node.js**. The system design defined how the client (frontend) would communicate with the server (backend) and how data (paragraphs) would be stored and retrieved. The user interface was designed with focus on minimalism and ease of use, including components like buttons, text boxes, and result displays. A flowchart and wireframes were prepared to visualize the flow of operations and page layout.

**3. Frontend Development**

The frontend was implemented using **React.js**, a popular JavaScript library for building interactive user interfaces. Components such as the typing area, timer, result section, and theme toggle were created. React's state management made it easy to handle dynamic updates like countdowns and real-time input

changes. CSS was used to style the elements, ensuring a responsive layout that adapts to both light and dark themes. The frontend was also connected to the backend API for fetching random paragraphs to type. Overall, this phase focused on providing a smooth and attractive experience for users.

**4. Backend Development**

The backend was developed using **Node.js** and **Express.js** to handle server-side operations. The main responsibility of the backend was to supply random practice paragraphs through an API endpoint (`/api/paragraphs/random`). It also handled CORS (Cross-Origin Resource Sharing) and provided a health check route for testing server functionality. The backend was designed to optionally connect to **MongoDB**, allowing paragraph data to be stored and fetched dynamically. If no database connection was available, the system automatically used locally stored JSON data, ensuring reliability and flexibility.

**5. Testing and Debugging**

Once both the frontend and backend were ready, the entire system was integrated and tested thoroughly. Functional testing was performed to verify that the timer worked correctly, buttons responded properly, and speed and accuracy were calculated accurately. Edge cases, such as stopping before time or leaving the text box empty, were also tested. The interface was checked for responsiveness across different devices and browsers. Errors found during testing were fixed, and improvements were made to enhance usability and performance.

**6. Deployment and Documentation**

After successful testing, the application was finalized for deployment in a local environment. The project was made ready for future hosting on cloud platforms like **Render**, **Vercel**, or **Netlify** for the frontend, and **MongoDB Atlas** for database hosting. Detailed documentation was prepared to explain the installation process, project structure, and usage steps. This final phase ensured that the project was complete, well-documented, and ready for demonstration or further development.

## 3.2 Technologies Used

### 1. MongoDB

- MongoDB is used to store typing practice paragraphs in an easy and flexible way.
- Since the data is simple (sentences and text), a document-based NoSQL database is ideal.
- It allows the addition of new paragraphs anytime without changing the structure of the database.
- MongoDB also supports scalability if future features like user history or leaderboard are added.

### 2. Express.js

- Express.js is used to create the backend server and manage API routes.
- It helps the frontend (React) communicate with the backend through simple HTTP endpoints.
- It handles the “Get Random Paragraph” request and responds quickly to the client.
- Express was chosen for its simplicity, small size, and ability to integrate easily with Node.js.

### 3. React.js

- React.js is used for building the frontend user interface of the project.
- It manages all interactive parts like the timer, accuracy calculation, and real-time typing input.

- React automatically updates only the parts of the page that change, which improves performance.
- It also enables reusability of components, making the app easier to maintain and expand later.

### 4. Node.js

- Node.js provides the runtime environment for executing JavaScript on the backend.
- It runs the Express server, processes client requests, and sends quick responses.
- Node allows the same language (JavaScript) to be used for both frontend and backend, reducing complexity.
- Its non-blocking event-driven model makes the system fast and responsive even under multiple requests.

### 5. HTML

- HTML provides the structure of the web application and defines all visible elements.
- It is used within React components to design the layout, including buttons, text areas, and result boxes.
- HTML was chosen because it is lightweight, browser-compatible, and forms the backbone of every web page.
- It ensures that the structure of the app remains clean and accessible.

### 6. CSS

- CSS is used to style and design the entire interface of the project.
- It gives the web app an attractive and comfortable look, with proper colors, spacing, and fonts.
- CSS variables were used to switch between light and dark themes smoothly.
- It ensures a pleasant experience for users typing for long durations and makes the app responsive.

### 7. JavaScript

- JavaScript is the main programming language used throughout the project.
- It handles all the logic — from timer control and text comparison to WPM and accuracy calculation.
- It was chosen for its real-time behavior, flexibility, and compatibility with all modern browsers.
- JavaScript makes the typing test interactive, instant, and smooth without page reloads.

### 8. Vite

- Vite was used as the frontend build and development tool for React.
- It offers very fast startup time and instantly reflects code changes in the browser.
- It was chosen for its modern architecture, better performance, and lightweight setup compared to older tools.

- Vite ensures efficient development and optimized final builds for deployment.

### 9. GitHub

- GitHub was used for version control and project storage during development.
- It helps track all code changes, manage versions, and collaborate easily if required.
- Using GitHub keeps the project safe and backed up in the cloud.
- It also allows easy sharing and showcasing of the project for future reference or interviews.

# CHAPTER 4
# ANALYSIS AND VALIDATION

The Typing Speed Checker project was successfully implemented using the MERN stack. After the development and integration of both the frontend and backend, the system was thoroughly tested to ensure it worked as intended. This chapter describes the results obtained, how the application performs in real scenarios, and the validation process carried out to confirm the accuracy and reliability of the system.

## 4.1 Functional Results

The completed application provides all the major features that were planned during the design stage. When the user opens the web application, a clean and responsive interface is displayed with options to start the typing test, stop it, change the paragraph, and toggle between light and dark modes.

Once the user clicks on the **"Start"** button, a 60-second timer begins, and the user starts typing the displayed paragraph into the input area. As the timer runs, the system continuously tracks the text typed by the user. When the timer reaches zero or when the user clicks on the **"Stop"** button, the test ends automatically, and the system instantly displays the results on the screen.

The output includes two key results:

- **Words Per Minute (WPM):** This shows how many words the user typed correctly in one minute.
- **Accuracy (%):** This shows how accurately the user typed compared to the original paragraph.

A **Change Paragraph** button allows the user to load a new random paragraph for the next test. This helps users practice with different texts and improve overall speed and accuracy. The **Theme Toggle** button switches between light and dark modes, enhancing comfort during long typing sessions.

## 4.2 Performance and Responsiveness

The application was tested on various devices including desktops, laptops, and tablets to check its responsiveness. The layout automatically adjusts to different screen sizes, ensuring all controls and text areas remain visible and properly aligned. The use of React.js ensures smooth transitions and instant updates without reloading the page.

The timer works accurately, counting down in real time, and there is no lag or delay during typing or result calculation. The system also handles multiple test cycles efficiently — users can restart tests or change paragraphs repeatedly without affecting performance.

Both the light and dark themes were validated for readability. The dark mode, in particular, reduces eye strain during extended practice sessions and makes the app more user-friendly.

### 4.3 Validation and Accuracy Testing

Validation was carried out to confirm that the system calculates typing speed and accuracy correctly. Multiple users tested the application and compared their results with standard online typing test tools such as **10FastFingers** and **Typing.com**. The difference in measured WPM and accuracy was minimal, confirming that the system produces reliable results.

The calculation formula for WPM and accuracy was manually verified using sample data. For example, if a user typed 40 words in one minute with 4 errors, the calculated accuracy and speed closely matched the expected results. This confirms that the implemented logic for comparison and timing works as intended.

### 4.4 User Experience and Feedback

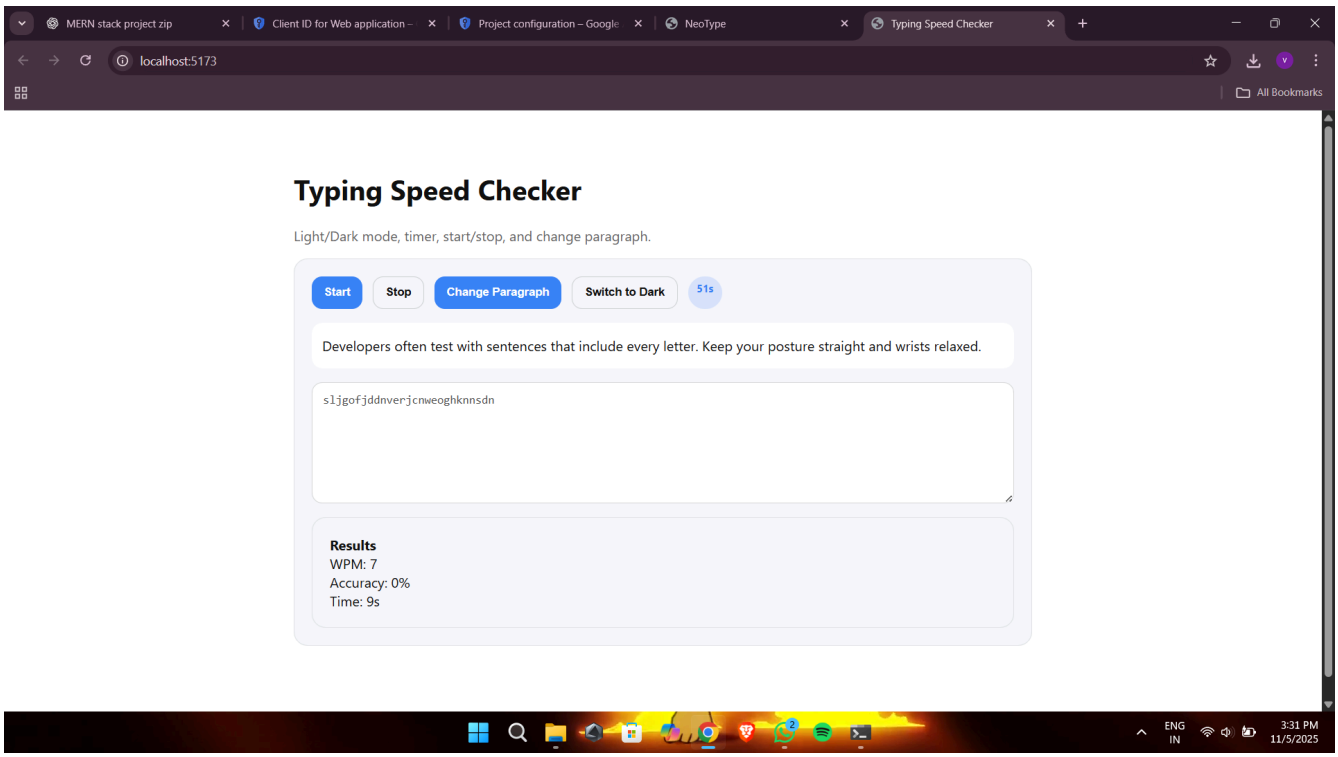
The system was tested by a group of students and users who provided positive feedback about its simplicity and modern interface. They found the application easy to use, and appreciated the inclusion of the **Start**, **Stop**, and **Change Paragraph** buttons. The dark mode was especially appreciated by users who practiced for longer periods.

Users mentioned that the results were displayed instantly and clearly, helping them identify how well they typed and encouraging them to improve with each test. The interface was rated as minimalistic and less distracting compared to other online typing tools that display ads or unnecessary animations.

### 4.5 Summary of Results

Overall, the Typing Speed Checker performed effectively in all tests. The results are accurate, the interface is responsive, and the application provides a smooth user experience. The integration between the backend API and frontend React interface works seamlessly. The project meets all the objectives defined in Chapter 2 and Chapter 3, proving its functionality and usefulness.

The system not only measures typing performance but also motivates users to improve through repeated practice, making it a valuable tool for students and professionals alike.




# CONCLUSION AND FUTURE WORK

## Conclusion

The Typing Speed Checker project successfully meets its main objective of helping users test and improve their typing speed and accuracy through a simple, modern, and interactive web-based application. By using the MERN stack, the system ensures smooth communication between the frontend and backend, while providing fast performance and a responsive user interface.

All key features such as **start and stop controls**, **paragraph changing**, **real-time speed calculation**, and **theme switching** were implemented effectively. The project offers an engaging environment that motivates users to practice regularly and monitor their progress. It also proves that the MERN stack is highly suitable for building lightweight, efficient, and user-focused web applications. Overall, the system fulfills its intended purpose and provides a valuable digital learning tool for students and professionals who wish to enhance their typing skills.

## Future Work

Although the current system performs well, there are several improvements that can make it even more powerful and useful. In the future, features like **user login and registration** can be added so that individual progress can be stored and tracked. A **leaderboard system** can encourage healthy competition among users. The project can also be extended to support **multiple languages**, allowing users to practice typing in Hindi, English, or regional languages.

In addition, **AI-based performance analysis** can be integrated to provide personalized tips for improvement. With further development, the Typing Speed Checker can evolve into a complete online typing training platform suitable for schools, colleges, and offices.

# REFERENCES


[1] Node.js Foundation. *"Node.js Documentation."* [Online]. Available: https://nodejs.org/en/docs

[2] Express.js Team. *"Express.js — Fast, unopinionated, minimalist web framework for Node.js."* [Online]. Available: https://expressjs.com/

[3] MongoDB Inc. *"MongoDB Manual."* [Online]. Available: https://www.mongodb.com/docs/

[4] ReactJS Team, Meta Platforms. *"React — A JavaScript Library for Building User Interfaces."* [Online]. Available: https://react.dev/

[5] MDN Web Docs. *"JavaScript — The Core Programming Language of the Web."* [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript

[6] ViteJS Team. *"Vite — Next Generation Frontend Tooling."* [Online]. Available: https://vitejs.dev/

[7] W3Schools. *"HTML Tutorial — The Standard Markup Language for Web Pages."* [Online]. Available: https://www.w3schools.com/html/

[8] W3Schools. *"CSS Tutorial — Cascading Style Sheets for Web Design."* [Online]. Available: https://www.w3schools.com/css/

[9] FreeCodeCamp. *"Understanding the MERN Stack — MongoDB, Express, React, and Node."* [Online]. Available: https://www.freecodecamp.org/news/mern-stack-explained/

[10] GeeksforGeeks. *"Typing Speed Test in JavaScript — Implementation and Explanation."* [Online]. Available: https://www.geeksforgeeks.org/typing-speed-test-using-javascript/

[11] MDN Web Docs. *"Using Fetch API — Web API Interface for HTTP Requests."* [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API

[12] OpenAI. *"ChatGPT — AI Assistance for Code, Documentation, and Learning."* [Online]. Available: https://openai.com/chatgpt