

**TRIBHUVAN UNIVERSITY**

**FACULTY OF HUMANITIES AND SOCIAL SCIENCE**

**A Project Proposal**

**On**

**“KeyChha – A Typing Practice Web Application”**

**Submitted To**

**Department of Computer Application**

**National College of Computer Studies**

In partial fulfilment of the requirements of Bachelor Degree in Computer Application

Submitted By:

Sanil Maharjan

TU Registration Number: 6-2-551-75-2022

**TU Roll No:**55102141

# Table of Contents

[Table of Contents ii](#_Toc209011105)

[List of Figures iii](#_Toc209011106)

[1. Introduction to the Project 1](#_Toc209011107)

[2. Problem Statement 1](#_Toc209011108)

[3. Objective 1](#_Toc209011109)

[4. Methodology 2](#_Toc209011110)

[4.1. Requirement Identification 2](#_Toc209011111)

[4.1.1. Study of Existing System 2](#_Toc209011112)

[4.1.2. Literature Review 2](#_Toc209011113)

[4.1.3. Requirement Analysis 3](#_Toc209011114)

[4.2. Feasibility Analysis 5](#_Toc209011115)

[4.2.1. Technical Feasibility 5](#_Toc209011116)

[4.2.2. Operational Feasibility 5](#_Toc209011117)

[4.2.3. Economic Feasibility 6](#_Toc209011118)

[4.3. High-Level Design of System 6](#_Toc209011119)

[4.3.1. Methodology of the Proposed System 6](#_Toc209011120)

[4.3.2. Flow Chart 7](#_Toc209011121)

[4.3.3. Working Mechanism of the Proposed System 8](#_Toc209011122)

[4.3.4. Description of Algorithms 8](#_Toc209011123)

[5. Gantt chart 11](#_Toc209011124)

[6. Expected Outcome 12](#_Toc209011125)

[7. References 12](#_Toc209011126)

# List of Figures

[Figure 1: Use Case Diagram of KeyChha 4](#_Toc209011096)

[Figure 2: Waterfall Model 7](#_Toc209011097)

[Figure 3: System Flow for User 8](#_Toc209011098)

[Figure 4: Gantt chart of Project Activities 12](#_Toc209011099)

# Introduction to the Project

Typing is among the most essential computer skills in today's busy world, supporting learning, communication, and career growth. The majority of learners, however, struggle with sustaining speed, accuracy, and consistency during practice. This explains the existing demand for practical tools that not only provide structured practice but also motivate users to continue practicing to get better. Web-based platforms that offer convenience and personalization are increasingly playing important roles in the attempts to improve people's typing capacities in a fun and efficient way.

KeyChha is a typing practice software available online that is effective and enjoyable to learn to type with. It comes with a simple user interface supporting basic modules such as login, registration, personalized profiles, and detailed performance tracking. Users' progress can be monitored based on key parameters like words typed, mistakes made, accuracy, and words per minute (WPM). Along with simple practice, KeyChha is designed to accommodate each person's requirements, giving a customized experience which encourages progress at each step. In unifying personalization and accessibility, KeyChha aims to empower its users with confidence, precision, and speed in their typing experience.

# Problem Statement

Most users find typing hard to learn since practice materials tend to be too general or do not tailor themselves to the user. Traditional techniques do not point out individual weaknesses, give instant feedback, or maintain learners' attention, making improvement slow and frustrating. Without obvious monitoring of improvement, learners often become demotivated and continue repeating the same errors years later. What is actually needed is a user-friendly, interactive platform that renders practicing typing more engaging, gives personalized feedback, and helps to build up speed and accuracy in incremental steps while keeping the learners motivated in the process.

# Objective

1. The project aims to create an interactive, user-friendly typing practice platform that helps learners improve their speed, accuracy, and confidence through personalized feedback and progress tracking.
2. The selected algorithms will be used to analyze user performance and adapt practice content accordingly, ensuring that exercises match each learner’s weaknesses and gradually adjust difficulty to maximize improvement.

# Methodology

## Requirement Identification

The process of identifying and monitoring the precise requirements, objectives, and limitations that a system must meet is known as requirement identification. This covers learner demands including adaptive lessons, real-time feedback, error analysis, and progress tracking, as well as system-level objectives like usability, accessibility, and scalability. By precisely defining these needs, the platform is guaranteed to be both technically sound and user-focused, producing an entertaining tool for enhancing typing skills.

### Study of Existing System

A variety of typing practice platforms, including TypingMaster and Typesy, were evaluated as part of the system study. These platforms offer scheduled courses, real-time feedback, performance tracking, and a variety of other flexible and intuitive features. They improve the learning process by making tailored adjustments based on variables such as speed, accuracy, and error frequency. This evaluation of existing methods informed the development of "KeyChha," guaranteeing a fun, efficient, and customized typing practice tool.

### Literature Review

The area of typing practice software has been considered for decades in human-computer interaction and instructional technology. Early work considered typing as a motor skill and led to the development of training courses. Gentner et al. [1] compared novice and experienced typing movements, illustrating that systematic practice significantly enhances typing skill by stabilizing motor patterns. Their work led to the establishment of early practice programs with repetitive exercises and performance monitoring.

As computer-based intelligent tutoring systems developed, researchers began to add adaptivity to typing trainers. Lesniewicz et al. [2] designed and evaluated a computer typewriting tutor that adapted in real time based on user performance. Adaptive lesson planning lowered error rates and increased learner involvement compared to fixed drills, they demonstrated.

Secondly, keystroke-level data has been used to calibrate typewriting systems. Dhakal et al. [3] examined large-scale typing behaviour and depicted how typing expertise is not only determined by the frequency of use but also by smart feedback. These results legitimized the use of adaptive feedback mechanisms in typewriting software.

It has then been investigated with school students and language learners in applied classroom environments. Özyurt and Özyurt [4] looked at the use of a system for practicing typing to enhance touch-typing in students and reported quantifiable improvements to accuracy and speed of typing with systematic practice. In the same vein, a study by Shamir et al. [5] also identified how typing tutors boost not only motor speed but also writing fluency for students with disabilities and thus emphasized the overall educational function of such programs.

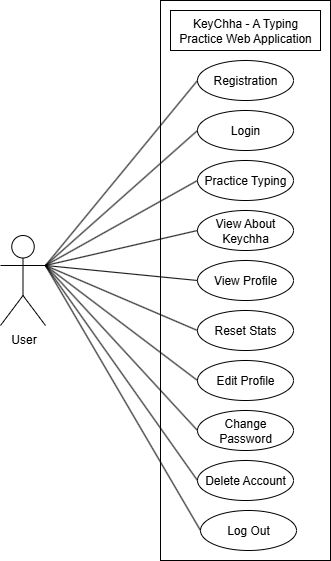
Combined, these works determine that effective typing practice software integrates structured and adaptive learning approaches, error-driven feedback, and long-term tracking, each of which informs the development of KeyChha.

### Requirement Analysis

The process of refining and evaluating requirements to make sure they are precise, workable, and in line with project objectives is known as requirement analysis. In addition to technological requirements like timeliness, scalability, and cross-platform compatibility, it takes into account user needs like adaptive practice, real-time feedback, and performance tracking. A well-organized set of functional and non-functional requirements that efficiently direct development and reduce misunderstandings is the end result.

* + - 1. **Functional Requirements**

A straightforward and efficient typing practice experience is guaranteed by the functional criteria. An About page containing platform information, a Home page for navigation, and account functionality like registration, login, and logout will all be included in the web application. Users have the option of creating an account to store their performance or practicing straight away without logging in, where progress is not kept. Users that are logged in can modify their personal information, examine or manage their statistics, or remove their account by going to their Profile page. As lessons advance from levels A to Z, the system will monitor key metrics such words typed, characters, errors, accuracy, and words per minute (WPM). Algorithm will also be employed to provide customized word sets to further increase typing efficiency.



**Figure 1: Use Case Diagram of KeyChha**

* Registration: New users can create an account to access KeyChha.
* Login: Registered users can log in securely to access their accounts.
* Practice Typing: Users can practice typing, track accuracy, errors, and speed.
* View About KeyChha: Users can read information about the KeyChha platform.
* View Profile: Logged-in users can view their profile details and typing statistics.
* Reset Stats: Users can clear their typing history and reset their performance data.
* Edit Profile: Users can update their username and email information.
* Change Password: Users can change their account password for security.
* Delete Account: Users can permanently delete their account and all associated data.
* Log Out: Logged-in users can log out of the system securely.
  + - 1. **Non-Functional Requirements**
* User-Friendly Interface: Provides a clean and intuitive design that allows learners to easily navigate pages like Home, About, Login/Register, and Profile.
* Scalability: Designed to handle a growing number of users and practice data without compromising speed or performance.
* Stability: Ensures consistent functionality so users can practice typing without interruptions or unexpected system failures.
* Security: Protects user data such as login credentials and profile information, allowing only authenticated users to access and manage personal statistics.
* Usability: Offers straightforward navigation and responsive layouts across devices, ensuring smooth practice sessions for both logged-in and guest users.
* Performance Efficiency: Calculates typing results (characters, errors, accuracy, and WPM) quickly and updates progress in real time.
* Dependability: Delivers reliable access to practice sessions, progress tracking, and account management, ensuring users can trust the system for continuous learning.

## Feasibility Analysis

The process of evaluating a proposed project to determine its ability to succeed, practicality, and value is known as feasibility analysis. To make sure the project can be developed and implemented successfully, it looks at crucial factors such technical competency, economic cost-effectiveness, operational usefulness, and overall viability.

### Technical Feasibility

KeyChha is a technically viable solution made with popular web technologies for ease of development and dependability. While HTML, CSS, and JavaScript are used in the frontend to create a responsive and user-friendly interface, PHP and MySQL are used in the backend to ensure safe data processing and effective storage. In order to improve typing efficiency, the system integrates algorithms that evaluate user statistics, including accuracy, errors, and typing speed, and produce appropriate word sets based on the data gathered. KeyChha's scalable design, lightweight technology, and strong database support allow for efficient development, deployment, and maintenance, guaranteeing stability and performance over the long run.

### Operational Feasibility

Because KeyChha can be easily accessible through any modern web browser without the need for installation or sophisticated hardware, and because it requires low resources for implementation, it is operationally practicable. People of various skill levels can easily explore pages, practice typing, and look over their statistics thanks to its clear and simple layout. Maintenance efforts are greatly decreased because the system is built to run autonomously without administrative control, guaranteeing seamless daily operations. With its user-friendly design, low resource requirements, and ease of accessibility, KeyChha provides a workable and sustainable long-term solution.

### Economic Feasibility

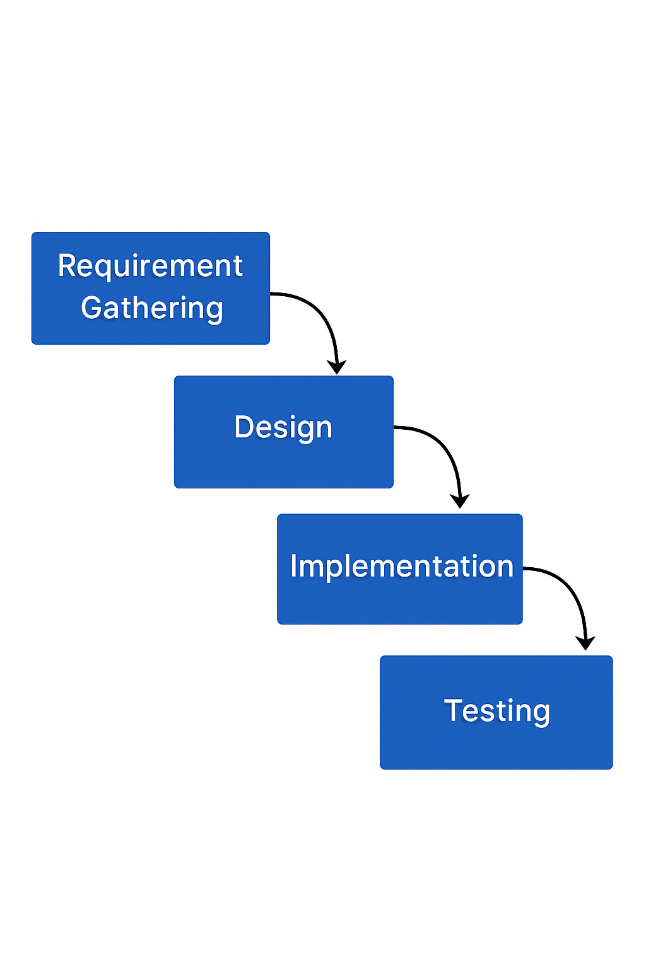
Since KeyChha is built with widely accessible and open-source technologies like PHP, MySQL, HTML, CSS, and JavaScript—all of which do not require expensive proprietary tools—it is economically viable. Users can avoid paying for extra software or expensive hardware because the application is web-based and simply requires a basic browser and an internet connection. The system's automated management and lightweight design keep maintenance costs low, and its scalability guarantees future development without requiring a large amount of reinvestment. All things considered, KeyChha is an affordable and sustainable way to provide typing practice due to its minimal development and operating costs.

## High-Level Design of System

The process of describing a system's overall architecture with an emphasis on its primary components, their connections, and the data flow between them is known as high-level design, or HLD. It serves as a guide for in-depth design and development by providing a clear blueprint of how the system will operate overall.

### Methodology of the Proposed System

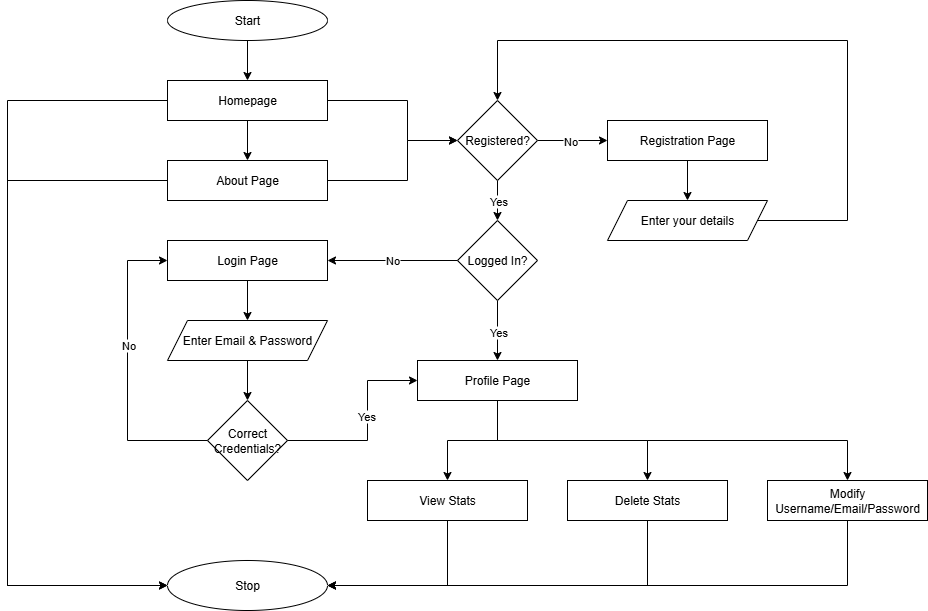
The proposed system has been developed using the Waterfall model, a linear software development approach in which each phase is completed in sequence. The process begins with requirement identification and analysis to define system objectives and user needs, followed by the design phase, which specifies the overall architecture, interface layout, and database model. The implementation stage then converts the design into a functioning web application, while testing ensures accuracy, usability, and performance. The Waterfall model was chosen for its simplicity, clarity, and suitability for projects with well-defined requirements, offering a structured and systematic progression. As an academic project, emphasis is placed on the main development phases, while deployment and long-term maintenance fall outside the current scope.

****

**Figure 2: Waterfall Model**

### Flow Chart

A flowchart is a visual diagram that represents the sequence of processes, decisions, and actions within a system using standardized symbols and connecting arrows. This flowchart explains how a user interacts with the typing practice website. When a user visits the site, they first arrive at the homepage where they can practice typing, but if they are not logged in, their progress is not saved. They also have the option to visit the About page to learn more about the platform. If the user wants to log in, they go to the login page and enter their email and password. If the details are wrong, they cannot continue, but if correct, they move forward. For new users who are not registered, they can register themselves on the registration page by entering their details and creating an account. Once registered and logged in, the user can access the profile page. From the profile page, they can view their typing statistics, delete stats, change their username, email, or password, and even delete their account if they wish. The process ends when the user logs out or leaves the site.



**Figure 3: System Flow for User**

### Working Mechanism of the Proposed System

The working mechanism of the system is designed to provide a smooth platform for practicing and improving typing skills. When a user accesses the system, they can either practice directly as a guest, where progress is not stored, or log in to track and manage their statistics. During each practice session, the system records user performance and calculates parameters such as total characters typed, words typed, correct entries, errors, accuracy, and words per minute (WPM). Algorithms are then used to analyze these statistics and generate suitable word sets that focus on improving efficiency and overcoming frequent errors. For registered users, performance data is stored in the database and accessible through the Profile page, where they can review or manage their progress as well as update account details. By automating performance tracking and adaptive practice, KeyChha ensures a user-centered and effective learning experience.

### Description of Algorithms

1. **Error Frequency Analysis (Rule-Based)**

This algorithm looks at how often users make a specific error. If an error appears more than a set threshold, the system applies a correction rule (for example, automatically fixing “teh” to “the”). The expression here basically checks the frequency of each error word against the threshold and decides whether to correct it.

**Mathematical Expression:**

If 𝐸(𝑤) = frequency of error word 𝑤, and T = threshold:

**Complexity:**

* Counting errors: O(N) for N words typed.
* Lookup/correction: O(1) with hash map.
* Overall: O(N)

1. **Markov Chains (Probability-Based Word Generation)**

In a Markov chain, the next word depends only on the current word. The expression describes the probability of one word following another, and the overall probability of a generated sequence is the product of these pairwise probabilities. In practice, this means if your training data shows “the” is often followed by “dog,” the generator will likely pick “dog” after “the.”

**Mathematical Expression:**

If Wt is the current word:

And generated sequence:

**Complexity:**

* Training (count transitions): O(N).
* Generation (walk length k): O(k).
* Overall: O(N+k)

1. **N-Gram Model (Adaptive Practice)**

The N-gram model predicts the next word based on the last n–1 words. The expression calculates the probability of a word given the previous ones, by dividing how many times a sequence appears with how many times the shorter prefix appears. In simpler terms: if “I like” is often followed by “apples,” the model learns to expect “apples” after “I like.”

**Mathematical Expression:**

**Complexity:**

* Training (build counts): O(N).
* Prediction lookup: O(1) with hash tables.
* Overall: O(N)

1. **Reinforcement Learning**

Here, an agent (like a typing practice app) learns by trial and error. The expression describes how the agent updates its estimate of how good an action is in a given state, based on the reward received and the expected future rewards. Over time, the agent refines its choices to maximize rewards. In typing, this might mean gradually adjusting which words to present so learners improve steadily.

**Mathematical Expression:**

Update rule:

where:

s = state, a = action

r = reward, α = learning rate, γ = discount factor

**Complexity:**

* Each update: O(1).
* Full learning: where N = states, A = actions.
* Overall: depends on state-action space (can be exponential in worst case).

1. **Personalized Typing Difficulty Curve**

This approach gradually changes the difficulty level for each learner based on their performance. The expression updates difficulty by comparing the learner’s accuracy with a target accuracy, then increasing or decreasing difficulty accordingly. In other words, if a learner types very accurately, the system makes things harder; if they struggle, the system eases up.

**Mathematical Expression:**

If D(t) = difficulty at time t:

Where: θ = target accuracy, β = adjustment rate.

**Complexity:**

* Updating after each session: O(1).
* Tracking errors across N words: O(N).
* Overall: O(N)

1. **Spaced Repetition Curve**

This model schedules reviews of difficult items at carefully chosen intervals. The expression represents how memory fades over time (forgetting curve), and how review sessions reset and strengthen memory. Each time a learner recalls a word correctly, the system increases the gap before the next review. This way, words that are easy show up rarely, while tricky words repeat more often until mastered.

**Mathematical Expression:**

Forgetting curve:

Where R(t) = retention, S = strength of memory.

Spaced repetition schedules reviews at:

where

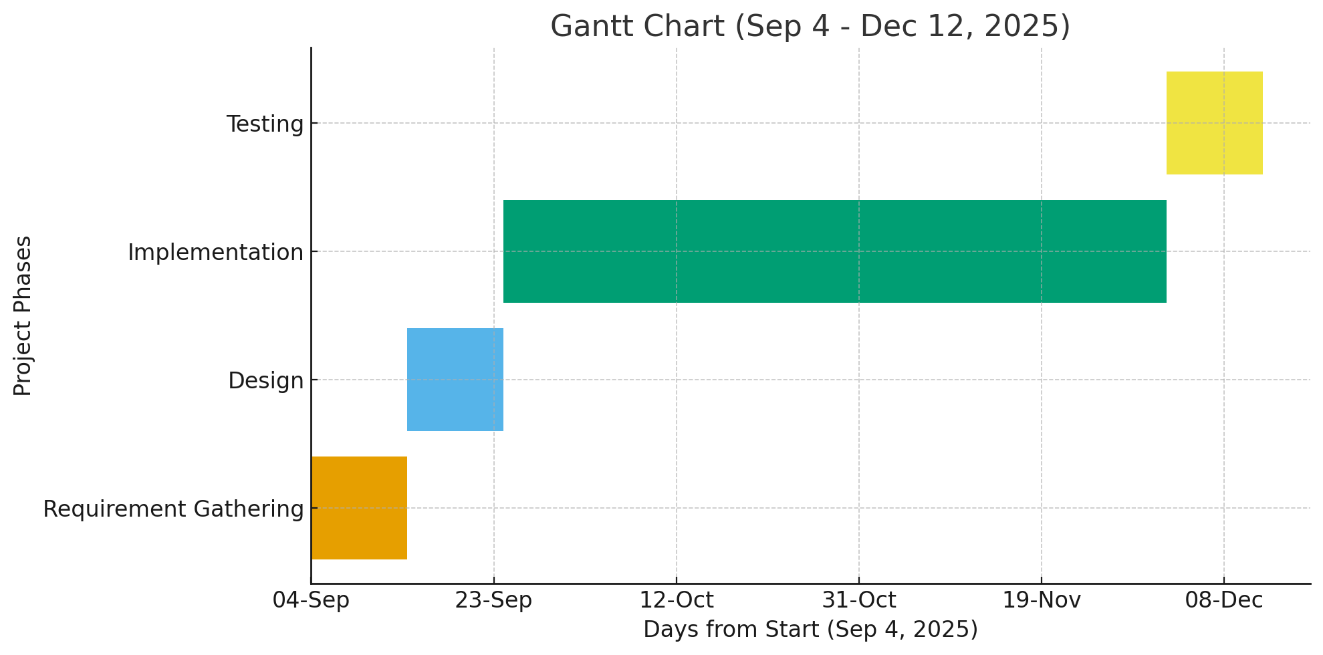
(commonly exponential: )

**Complexity:**

* Scheduling per word: O(1).
* Managing M words: O(M).
* Overall: O(M)

# Gantt chart

A Gantt chart is a project management tool that shows the start dates, durations, and end points of project activities visually over a timeline. The project's chart covers a total of 100 days, from September 4, 2025, to December 12, 2025. The requirements gathering phase is set for September 4–14 (10 days), while the design phase is set for September 15–24 (10 days). The longest and most crucial step is implementation, which starts on September 25 and lasts for 72 days, ending on December 5. Lastly, testing is scheduled for seven days, from December 6 to December 12. In accordance with the Waterfall model, this step-by-step arrangement guarantees that every phase is finished before the next one starts.



**Figure 4: Gantt chart of Project Activities**

# Expected Outcome

It is expected that the system will offer a simple and user-friendly platform that helps trainees' improvement of typing skills through organized practice and immediate feedback. While integrated algorithms evaluate performance data to produce customized word sets for more efficient practice, it will measure key metrics including characters typed, words per minute (WPM), accuracy, and errors. It is believed that the system will produce a customized and engaging learning experience that results in gradual improvements in speed, accuracy, and efficiency by enabling users to monitor and control their progress over time.

# References

[1] D. R. Gentner, D. W. Larochelle, and J. Grudin, "Lexical, sublexical, and peripheral effects in skilled typewriting," Cognitive Psychology, vol. 17, no. 3, pp. 280–313, 1985. doi: 10.1016/0010-0285(85)90010-2

[2] T. Lesniewicz, T. Richter, and J. Krüger, "An adaptive computer-based typewriting tutor," Computers & Education, vol. 23, no. 3, pp. 249–257, 1994. doi: 10.1016/0360-1315(94)90050-7

[3] V. Dhakal, A. Feit, P. Kristensson, and A. Oulasvirta, "Observations on typing from 136 million keystrokes," in Proc. CHI Conf. Human Factors in Computing Systems (CHI ’18), Montreal, Canada, 2018, pp. 1–12. doi: 10.1145/3173574.3174220

[4] Ö. Özyurt and H. Özyurt, "Implementation of a computer-based touch-typing training in education and its effects on students’ performance," Education and Information Technologies, vol. 18, no. 2, pp. 341–356, 2012. doi: 10.1007/s10639-012-9190-x

[5] A. Shamir, O. Mevarech, and A. Gida, "Computer-assisted teaching and typing skills: Implications for writing and learning," Journal of Computer Assisted Learning, vol. 25, no. 5, pp. 383–395, 2009. doi: 10.1111/j.1365-2729.2009.00318.x