

**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**Flash: “Typing Speed Calculator”**

**A Project Report Submitted to**

**Department of Computer Application**

**Kathmandu Business Campus**

**In partial fulfillment of the requirements for**

**Bachelor’s degree in computer application**

**Submitted By:**

**Adarsha Shakya**

TU Roll No: 6-2-1219-03-2020

072025

**Under the Supervision**

**of**

**Bigyan Pandey**



**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**Kathmandu Business Campus**

# SUPERVISOR’S RECOMMENDATION

We hereby recommend that this project prepared under our supervision by Adarsha Shakya entitled “**Flash: Typing Speed Calculator**” in partial fulfillment of the requirements for a degree of Bachelor in Computer Application is recommended for the final evaluation.

**SIGNATURE**

**Bigyan Pandey**

SUPERVISOR

Faculty Member

Department of Computer Application

Kathmandu Business Campus



**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**Kathmandu Business Campus**

# LETTER OF APPROVAL

This is to certify that this project prepared by Adarsha Shakya entitled “**Flash: Typing Speed Calculator**” in partial fulfillment of the requirements for degree of Bachelor of Computer Application has been evaluated. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

|  |  |
| --- | --- |
| **Signature of Supervisor**  **Bigyan Pandey**  **Faculty Member**  **Department of Computer Application**  **Kathmandu Business Campus,**  **Balaju, Kathmandu** | **Signature of HOD / Coordinator**  **Stella Koirala**  **Faculty Member**  **Department of Computer Application**  **Kathmandu Business Campus**  **Balaju, Kathmandu** |
| **Signature of Internal Examiner** | **Signature of External Examiner** |

# Abstract

Flash is a Typing Speed Calculator that helps in skill development by helping a user to enhance their typing skills. Flash also motivates a user by offering various levels with time limit and a bunch of sentences that a user can type. It shows a detailed result on users typing by providing Word Per Minute (WPM), Character Per Minute (CPM), and Mistakes made by user during typing. User can create an account and save their progress which might help to compare their performance thoroughly. User can edit their profile along with changing their passwords as well. There is a feature to delete the saved results also. User can download their results which can come in handy for comparing their progress. If user doesn’t want to create an account, user can use it without an account as well but they will be limited to the features such as changing levels, saving progress, viewing results and downloading the results.

Keyword: Typing Speed Calculator, Web app, Skill development

# Acknowledgement

We would like to express our special thanks of gratitude to our supervisor **Mr. Bigyan Pandey** who gave us the golden opportunity to do this wonderful project on the topic Flash a Typing Speed Calculator, which also helped us in doing a lot of research and we came to know about so many new tools and technologies.

We would like to express our special gratitude and thanks to our BCA Program Coordinator **Mrs. Stella Koirala** for his support and help for our personnel development and mainly for the completion of this Project.

We are highly indebted to Kathmandu Business Campus for their guidance and constant supervision as well as for providing necessary information regarding the Project and support in the completion.

We would also like to thank our parents and friends who helped us a lot in finalizing this project within the limited time frame.

Finally, we would like to thank Tribhuvan University for giving us this opportunity via the course of Computer Application to help us understand the project ethics at this stage and helped us to evaluate our knowledge and expand it a little more.

# Table of Contents

[SUPERVISOR’S RECOMMENDATION 1](#_Toc209037172)

[LETTER OF APPROVAL 2](#_Toc209037173)

[Abstract 3](#_Toc209037174)

[Acknowledgement 4](#_Toc209037175)

[Table of Contents 5](#_Toc209037176)

[List of Abbreviations 7](#_Toc209037177)

[List of Figures 8](#_Toc209037178)

[List of Tables 9](#_Toc209037179)

[Chapter 1: Introduction 1](#_Toc209037180)

[1.1. Introduction 1](#_Toc209037181)

[1.2. Problem Statement 2](#_Toc209037182)

[1.3. Objective 2](#_Toc209037183)

[1.4. Scopes and Limitation 2](#_Toc209037184)

[1.4.1. Scope 2](#_Toc209037185)

[1.4.2. Limitation 3](#_Toc209037186)

[1.5. Methodology 3](#_Toc209037187)

[1.6. Report Organization 4](#_Toc209037188)

[1.6.1. Chapter 1: Introduction 4](#_Toc209037189)

[1.6.2. Chapter 2: Background Study and Literature Review 4](#_Toc209037190)

[1.6.3. Chapter 3: System Analysis and Design 4](#_Toc209037191)

[1.6.4. Chapter 4: Implementation and Testing 4](#_Toc209037192)

[1.6.5. Chapter 5: Conclusion and Future Recommendation 4](#_Toc209037193)

[Chapter 2: Background Study and Literature Review 5](#_Toc209037194)

[2.1. Background Study 5](#_Toc209037196)

[2.2. Literature Review 6](#_Toc209037197)

[Chapter 3: System Analysis and Design 7](#_Toc209037198)

[3.1. System Analysis 7](#_Toc209037200)

[3.1.1. Requirement Analysis 7](#_Toc209037201)

[3.1.2. Feasibility Analysis 9](#_Toc209037202)

[3.1.3. Object Modelling using Class and Object Diagrams 11](#_Toc209037203)

[3.1.4. Dynamic Modelling using State and Sequence Diagrams 14](#_Toc209037204)

[3.1.5. Process Modelling using Activity Diagrams 17](#_Toc209037205)

[3.2. System Design 19](#_Toc209037206)

[3.2.1. Refinement of Class, Object, State, Sequence and Activity diagrams 19](#_Toc209037207)

[3.2.2. Component Diagrams 23](#_Toc209037208)

[3.2.3. Deployment Diagrams 24](#_Toc209037209)

[3.3. Algorithm Details 25](#_Toc209037210)

[Chapter 4: Implementation and Testing 28](#_Toc209037211)

[4.1. Implementation 28](#_Toc209037213)

[4.1.1. Tools Used 28](#_Toc209037214)

[4.1.2. Implementation details of modules (Description of Procedures/Functions) 31](#_Toc209037215)

[4.1.3. References 36](#_Toc209037216)

# List of Abbreviations

|  |  |
| --- | --- |
| CPM | Character Per Minute |
| CRUD | Create, Read, Update and Delete |
| CSS | Cascading Style Sheet |
| DFD | Data Flow Diagram |
| ER | Entity relation |
| HTML | Hyper Text Markup Language |
| JS | Java Script |
| MS | Microsoft |
| PHP | Hypertext Preprocessor |
| RDBMS | Relational Database Management System |
| SDLC | System Development Life Cycle |
| SQL | Structured Query Language |
| UI | User Interface |
| WPM | Word Per Minute |

# List of Figures

[Figure 1: Waterfall Methodology 3](#_Toc209037102)

[Figure 2: Use Case Diagram of Flash 8](#_Toc209037103)

[Figure 3: Gantt Chart for Flash 10](#_Toc209037104)

[Figure 4: Class Diagram of Flash 12](#_Toc209037105)

[Figure 5: Class Diagram of Flash 13](#_Toc209037106)

[Figure 6: State Diagram of Flash 14](#_Toc209037107)

[Figure 7: Sequence Diagram of Flash 16](#_Toc209037108)

[Figure 8: Activity diagram of Flash 18](#_Toc209037109)

[Figure 9: Refined Class Diagram of Flash 19](#_Toc209037110)

[Figure 10: Refined Class Diagram of Flash 20](#_Toc209037111)

[Figure 11: Refined State diagram of Flash 20](#_Toc209037112)

[Figure 12: Refined Sequence diagram of Flash 21](#_Toc209037113)

[Figure 13: Refined Activity diagram of Flash 22](#_Toc209037114)

[Figure 14: Component diagram of Flash 23](#_Toc209037115)

[Figure 15: Deployment Diagram of Flash 24](#_Toc209037116)

[Figure 16; Levenstein distance implementation 26](#_Toc209037117)

[Figure 17: Code snippet for login 31](#_Toc209037118)

[Figure 18: Code snippet for Register 32](#_Toc209037119)

[Figure 19: Code snippet for Level Selection 32](#_Toc209037120)

[Figure 20: Code snippet for Paragraph Generator 33](#_Toc209037121)

[Figure 21: Code snippet for Levenshtein Distance 33](#_Toc209037122)

[Figure 22: Code snippet for Result 34](#_Toc209037123)

[Figure 23: Code snippet for Result 34](#_Toc209037124)

[Figure 24: Code snippet for Download 35](#_Toc209037125)

[Figure 25: Code snippet for Download 35](#_Toc209037126)

# List of Tables

[Table 1: Gantt Chart for Flash 10](#_Toc209037070)

# Chapter 1: Introduction

## Introduction

Current generation is full of Advanced Technology in which a typing skill has been one of the most essential skills among all. Typing is almost used for completing every single task in this tech world; The faster and more accurately a person can type, the more beneficial they can be in completing specific tasks. So, the ability to type quickly has become a must have skill for all people.

A typing speed calculator is a tool that measures how fast someone can type. It typically calculates the number of words typed per minute (WPM); is a measure of words processed in a minute, often used as a measurement of the speed of typing. It is calculated based on the text entered and the time taken to type it. Typing speed calculators also provide additional information, such as accuracy and error rate. Typing speed calculators are often used for training and improving typing skills, as well as for recruitment purposes by companies looking for employees who can type quickly and accurately.

Now talking about the project, the name of the project is Flash a typing speed calculator, word “Flash” has its own meaning that is to move quickly. Flash is a skill development platform that helps a user to enhance their typing skills. Flash also motivates a user by offering various levels with bunch of sentences that a user can type and shows a detailed result on users typing. Users can create an account and save their progress which might help to compare their performance thoroughly.

Flash can be used by anyone who wants to improve their typing skills. The website offers user friendly interface so that users have no trouble using it. Users get test typing where they don’t have to login, they can create an account only if they want to access extra features such as save progress, change the levels, see their records in their profile. Users can delete records, download records and also edit their information. Overall, Flash is a great way to enhance one's typing skills and increase productivity in various situations, such as school, work, or personal use.

## Problem Statement

In today’s Digital era, many jobs have shifted from traditional to computer-based work which requires computer skills including typing skill. Not only jobs but schools have also started to adopt technologies. Every job contains some sort of computer usage and lack of typing skills is creating a path for unemployment. Poor typing skills make it difficult for a person to complete a task on time which lowers their opportunities. This might make people feel less confident and reduce their ability to use technology. Slow typing can cause failure to submit work within deadlines and errors in typing can add more workload which reduces productivity [1].

Freelancing has become one of the prior picked jobs by the youths and the fresh graduates. Most of the jobs are based on typing such as content writing, re-typing, presentation creation, web development, etc. All these jobs have to be completed in certain given time. So, this can create low opportunities for people who have skills but lack to deliver it on time. This is why people should have accurate and efficient typing skills to showcase their work within the given deadline. It will also motivate a person to do work and build up confidence. In today's digital world, typing has become an essential skill for many professions. With the increasing demand for accuracy and speed in typing, it is crucial to have a tool that can measure and improve typing skills [1]. So, people must have accurate and quick typing skills to increase their chances to get more opportunities.

## Objective

* To help user develop their typing skills.
* To display results so that users can track their progress.

## Scopes and Limitation

There are some scopes and limitations of this project. They are:

### Scope

* **Words per minute (WPM):** The most basic scope of a typing speed calculator is to measure the number of words an individual can type in given time.
* **Accuracy:** Another important scope of a typing speed calculator is accuracy. The system measures the number of errors made during the typing test, which helps the user identify areas of improvement.
* **Timed tests:** Typing speed calculators can offer timed tests that measure an individual's typing speed and accuracy over a specific duration.

### Limitation

* It does not focus on physical aspect while calculating. It does not check the hand placement, body posture etc.
* It can only measure speed over a short duration, such as one minute.

## Methodology

Flash is a small-scale project with well-defined project requirements and a linear approach. There is less probability of changing the requirements during project execution. Under such development circumstances, Waterfall model [3] is the best choice. Waterfall model is best for projects which have well-understood requirements and development process needs to be strictly managed and controlled.

Waterfall approach was first SDLC Model [4] to be used widely in Software Engineering to ensure success of the project. In The Waterfall approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

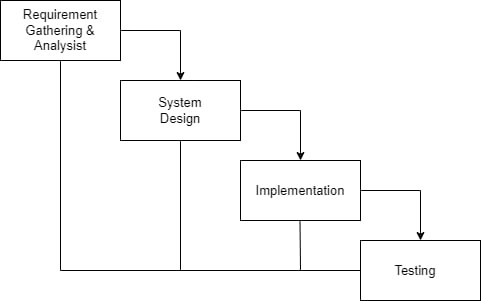
The following illustration is a representation of the different phases of the Waterfall Model.

Figure 1: Waterfall Methodology

## Report Organization

### Chapter 1: Introduction

This chapter deals with the introduction of the system with its objectives and limitations along with the reason why the system is made.

### Chapter 2: Background Study and Literature Review

This chapter summarizes the work that has been carried out in analyzing data and also describes the features of some existing areas of application.

### Chapter 3: System Analysis and Design

This chapter focuses on the different requirements of the system i.e., functional, non- functional, feasibility analysis, Entity Relational diagram, Data Flow Diagram, design of the system with system architecture, database schema, and interface design.

### Chapter 4: Implementation and Testing

This chapter highlights tools used in system development, implementing details and result of test performed.

### Chapter 5: Conclusion and Future Recommendation

This chapter highlights summary of outcome and conclusion of the whole project. It also explains what has been done and what further improvements could be made.

# Chapter 2: Background Study and Literature Review



## Background Study

The background study of A typing speed calculator is a tool that tracks and measures how quickly and accurately a person types. It calculates the number of words typed per minute (WPM) and character per minute (CPM) and provides a specific time limit to complete the given paragraph. Typing speed calculators also provide additional information, such as accuracy and error rate. Typing speed calculators are used for training and improving typing skills, as well as for recruitment purposes by companies looking for employees who can type quickly and accurately.

The name of the project is Flash a typing speed calculator, word “Flash” has its own meaning that is to move quickly. Flash is a skill development platform that helps a user to enhance their typing skills. Flash gives a user various levels of sentences that a user can type and shows a detailed result on users typing. Users can create an account and save and download their progress which might help to compare their performance thoroughly.

Flash can be used by anyone who wants to improve their typing skills. The website offers user friendly interface so that users have no trouble using it. Users get test typing where they don’t have to login, they can create an account only if they want to access extra features such as save progress, change the levels, see their records in their profile. Users can delete records, download records and also edit their information. Overall, Flash is a great way to enhance one's typing skills and increase productivity in various situations, such as school, work, or personal use.

## Literature Review

For this project, we conducted an analysis of several relevant websites, i.e., livechat [3], typingtyping [4], TypingTest [5], and typing [6]. These websites offer a similar feature set, presenting users with a series of sentences to type within a specified time limit, followed by the generation of a result. As part of our research, we actively engaged with these websites as a user to identify their areas of weakness and their strength. By assuming a user's perspective, we found out valuable understandings of the expectations and requirements for a high-quality typing speed calculator.

Typing speed calculators serve as valuable tools for assessing and measuring the speed and accuracy of an individual’s typing skills. Their applications extend across various areas including education, employment, and personal development [1]. This literature review aims to explore existing research and studies on typing speed calculators, specifically evaluating their use and understanding their effect on improving the typing skills.

Furthermore, this literature review will enable us to identify gaps in the current understanding of typing speed calculators and highlight areas for further research. Ultimately, the findings from this review will guide the development of our own website, ensuring that it effectively addresses the identified limitations of existing platforms and provides an optimized user experience for individuals seeking to enhance their typing skills.

By examining the available literature, we aim to identify the key features and functionalities that contribute to an efficient and user-friendly typing speed calculator. Additionally, we will explore the methodologies used in previous studies to measure typing speed, accuracy, and skill improvement. By conducting this complete literature review, we will extend our understanding of typing speed calculators, allowing us to develop our website with valuable understandings. Our goal is to overcome the limitations found in existing platforms and provide improved user experience.

# Chapter 3: System Analysis and Design



## System Analysis

System analysis is a crucial and complex phase in the development of a typing speed calculator. It plays a fundamental role in understanding the requirements, functionalities, and components of the system. Thus, it helps in ensuring its successful design and implementation of a system.

During this phase, the purpose of the typing speed calculator system is identified, which mainly focuses on measuring and evaluating user’s typing speed and accuracy. To achieve this objective effectively, it is essential to thoroughly analyze various aspects, including the user's expectations, features, and user interface preferences. Furthermore, system analysis involves researching the technical requirements and components of the typing speed calculator. This includes selecting the appropriate programming languages and databases to build an efficient system. This phase also involves identifying potential challenges and risks associated with the development and implementation of the typing speed calculator [3]. This may include security concerns, such as protecting user data and preventing unauthorized access.

In summary, system analysis plays an essential role in shaping the development of a typing speed calculator. It includes understanding user expectations, liked features, and user interface preferences while considering requirements and challenges.

### Requirement Analysis

Requirements for a system are identified through personal experience, analyzing and documenting the needs and expectations of the user. The analysis focuses on understanding the specific requirements that a typing speed calculator should meet.

A typing speed calculator should be reliable; it should deliver exact and reliable results, ensuring that it measures the user's typing speed. System should be able to give real-time feedback; It should offer immediate feedback to users, providing them with information on their typing speed and errors as they type. System should contain user-friendly interface; The interface should be intuitive and easy to navigate, allowing users to enter text effortlessly. System should be compatible so that users can easily use keyboard or touchscreen to enter the text.

By conducting a complete requirement analysis that considers reliability, real-time feedback, user-friendly interface, and compatibility, it helps to gather the necessary information to design and implement a typing speed calculator that meets user expectations and delivers an effective system.

#### Functional Requirement

* The site should be able to allow users to start, save and even reset progress.
* The site should provide results on accuracy and the number of errors made during typing.
* The site should allow users to adjust test difficulty level (1 as easy, 2 as medium and 3 as hard).
* The site should be able to allow users to save data, such as typing speed progress and error rates, so that users can track their performance over time.
* The site should be able to track a user's progress over time by providing accurate timing results according to its levels.

**Use Case Diagram**

This Section shows High level system design, behavior and interaction with external entities.

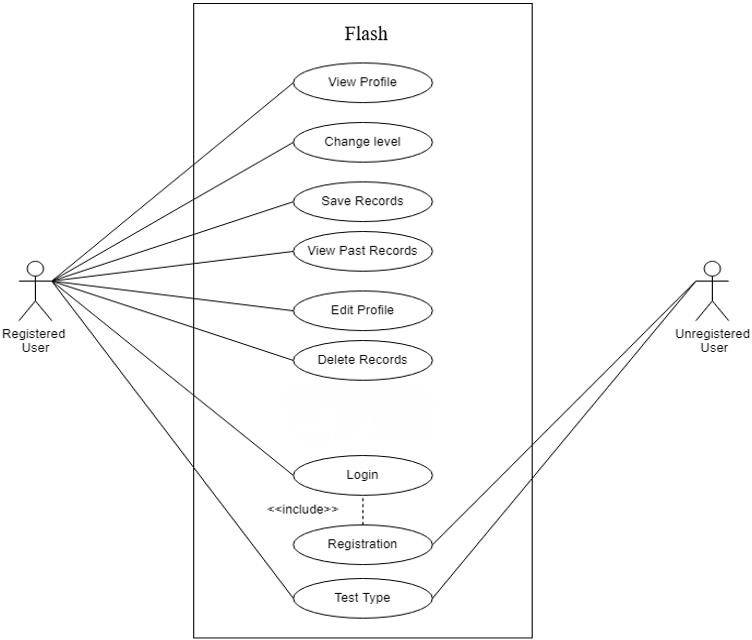


Figure 2: Use Case Diagram of Flash

Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

In this use-case diagram there are two actors who are represented as unregistered users and register users. The different roles assigned for actors are unregistered users who can test type in our website in form of demo without registering while unregister can register and create an account. Now registered users can login in, view their profile, change their levels, save their records, view past records, edit his/her profile, delete records and they can also download their records.

#### Non-Functional Requirements

* **Usability**: The site should have a user-friendly interface that is easy to navigate and understand.
* **Reliability**: The site should be accurate and reliable; it should be able to give feedback and response.
* **Security**: The site should ensure that users’ data is protected and secure, with appropriate measures to prevent unauthorized access such as using one-way hashing algorithm.
* **Scalability**: The calculator should be able to accommodate increasing numbers of users and perform well as usage scales up.
* **Performance**: The system should provide accurate and real-time calculation of typing speed and accuracy metrics, with minimal latency and response times.

### Feasibility Analysis

A feasibility Study is done to determine whether the project is feasible to persuade forward or not. It is done to identify any potential issues or obstacles that may arise during its implementation. A feasibility study for a typing speed calculator would assess whether the development and implementation of such site is capable of succeeding or not.

#### Technical Feasibility

This project meets technical feasibility as it will be developed with the use of existing technologies like frontend language: HTML, CSS, JavaScript, MySQL database, PHP script running on a cloud server or local computer. Development of this project will be done in free software and platforms as Visual Studio Code. So, there will not be any technical problem to build this project.

#### Operational Feasibility

An operational feasibility study would assess whether the calculator can be integrated and operated efficiently and effectively. It is easy to maintain and operate, so there will not be any operational problem.

#### Economic Feasibility

To complete this project, existing free of cost technologies (hardware, software, internet) will be used. So, there will not be any risks and challenges due to economic feasibility.

#### Schedule Feasibility

The System is completed within the scheduled time, and it does not exceed the scheduled time.

Table 1: Gantt Chart for Flash

|  |  |
| --- | --- |
| Task Name | Duration (days) |
| Requirement Identification | 7 |
| Planning | 9 |
| System Design | 11 |
| Implementation | 32 |
| Testing | 9 |

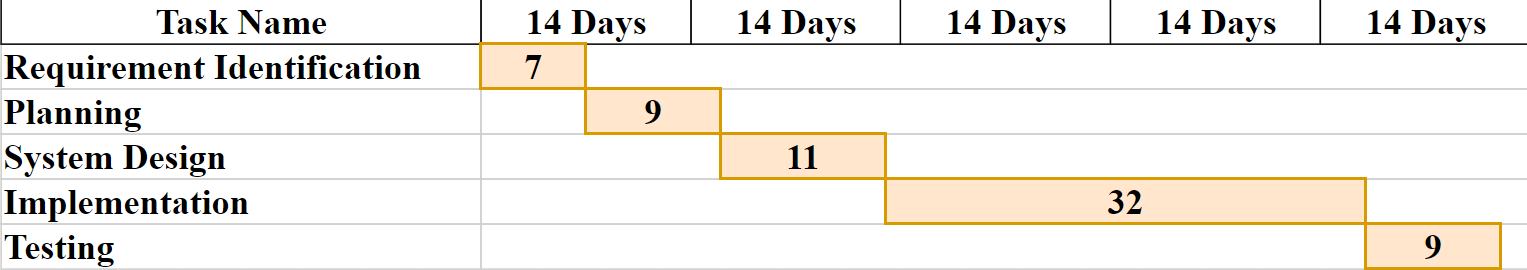


Figure 3: Gantt Chart for Flash

A Gantt chart is a visual representation of a project schedule that displays the start and end dates of individual tasks or activities in a horizontal bar chart format. The chart provides a timeline view of the project so that the development can go according to the schedule. It is commonly used in project management to plan and track project progress. They provide a clear and concise way to visualize the project timeline, identify potential scheduling conflicts or delays, and adjust the project plan as needed. The horizontal axis of the Gantt chart represents the timeline of the project, with the leftmost end showing the project start date and the rightmost end showing the project end date. Each task is represented by a horizontal bar that spans the duration of the task, starting from the task start date and ending on the task end date. The length of the bar represents the duration of the task, and its position on the chart shows its scheduled start and end dates.

### Object Modelling using Class and Object Diagrams

**Class Diagram**

This class diagram illustrates the core object-oriented structure of the Flash typing test platform, showing the essential entities and their relationships in a clean, vertical layout.

**Diagram Overview**

The diagram presents 5 fundamental classes that capture the platform's core functionality:

* User - Manages user authentication and result history access
* TypingTest - The core engine that conducts typing speed tests with timer and metric calculations
* Difficulty Level - An enumeration defining three test complexity levels (Easy, Medium, Hard)
* Test Result - Stores performance metrics (WPM, CPM, accuracy, mistakes) for progress tracking
* Database Manager - Singleton pattern implementation for centralized data persistence

**Key Relationships**

User → TestResult: One-to-many relationship showing users can have multiple test results

TypingTest → TestResult: Each test generates a performance result

TypingTest → DifficultyLevel: Tests use difficulty configuration for time limits

All Classes → DatabaseManager: Centralized data storage access

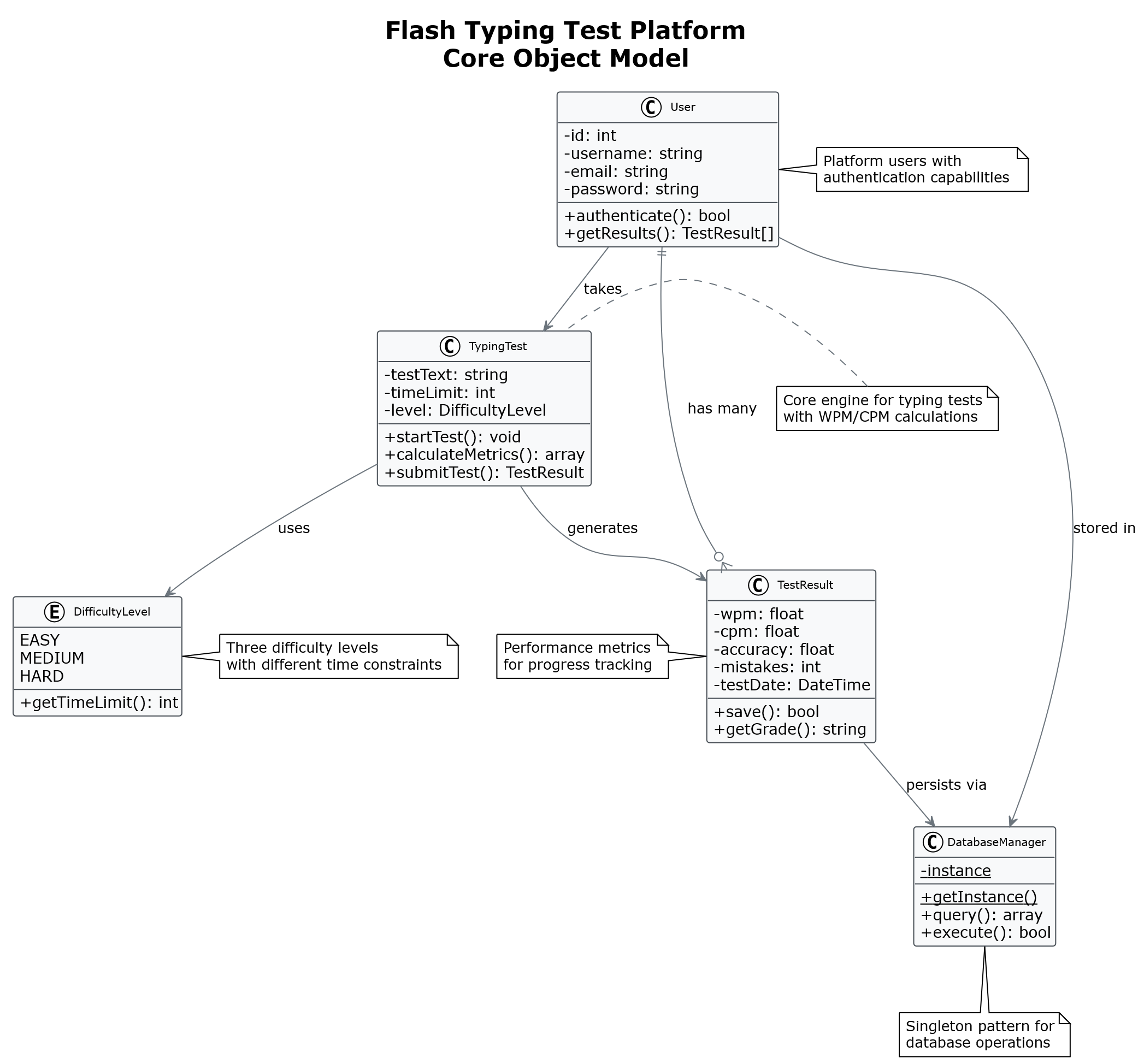


Figure 4: Class Diagram of Flash

**Object Diagram**

This object diagram illustrates a runtime snapshot of the Flash typing test platform, showing specific instances of the core classes and their actual relationships during a typical user session. Unlike class diagrams that show structure, this object diagram displays concrete objects with real data at a specific moment in time.

**Diagram Overview**

The diagram captures the state when user "john\_doe" completes a Medium difficulty typing test, showing:

* Active user instance with real profile data
* Current typing test session with specific text and timing
* Generated test result with actual performance metrics
* Database connection managing the persistence

This snapshot demonstrates the object collaboration during the core business process: a registered user taking a typing test and generating performance results that are stored for progress tracking.

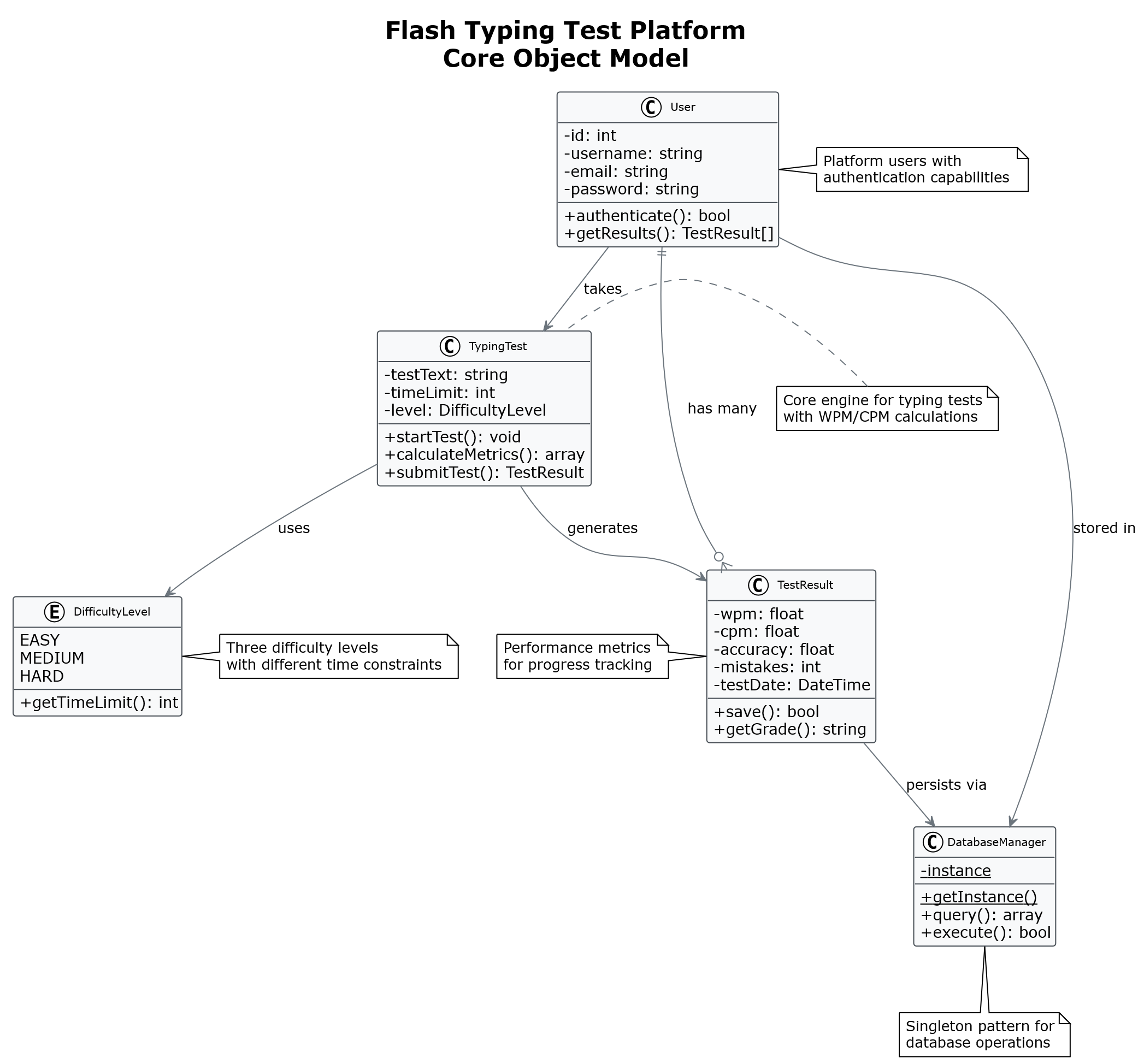


Figure 5: Class Diagram of Flash

### Dynamic Modelling using State and Sequence Diagrams

**State Diagram**

This state diagram models the user session lifecycle in the Flash typing test platform, showing how users transition between different states from initial visit to test completion.

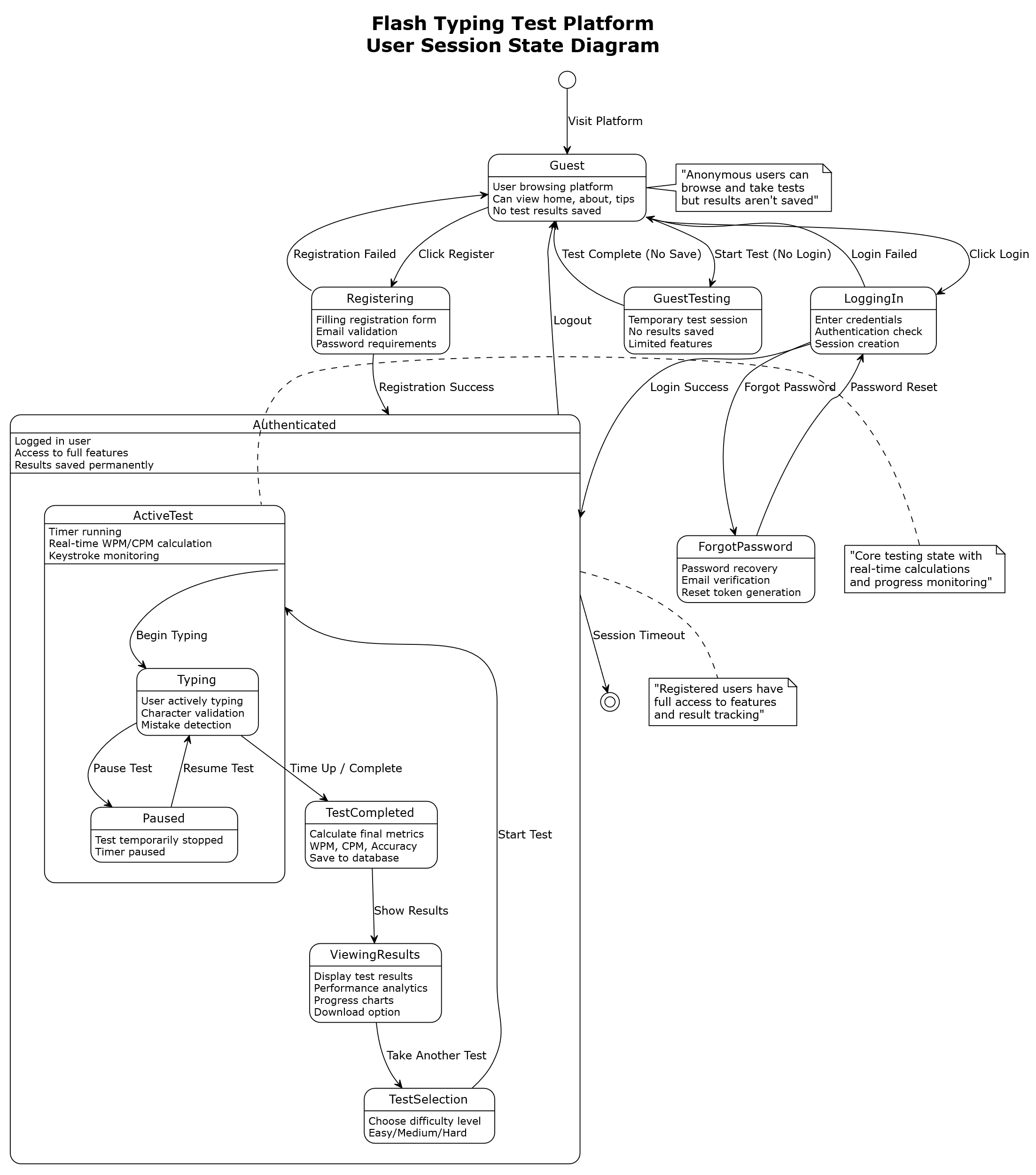


Figure 6: State Diagram of Flash

**Sequence Diagram**

This state diagram illustrates the complete user journey through the Flash typing test platform, capturing all possible user states and transitions.

Key States:

* Guest: Anonymous browsing with limited functionality
* Authenticated: Full platform access with saved results
* ActiveTest: Core typing test execution with real-time monitoring
* TestCompleted: Result calculation and persistence

Critical Transitions:

* Guest → Authenticated (via login/registration)
* TestSelection → ActiveTest → TestCompleted (core test flow)
* Typing ↔ Paused (test control)
* Session timeout handling

Sequence Diagram - Typing Test Execution Flow

This sequence diagram shows the interaction flow between system components during a typical typing test session for an authenticated user.

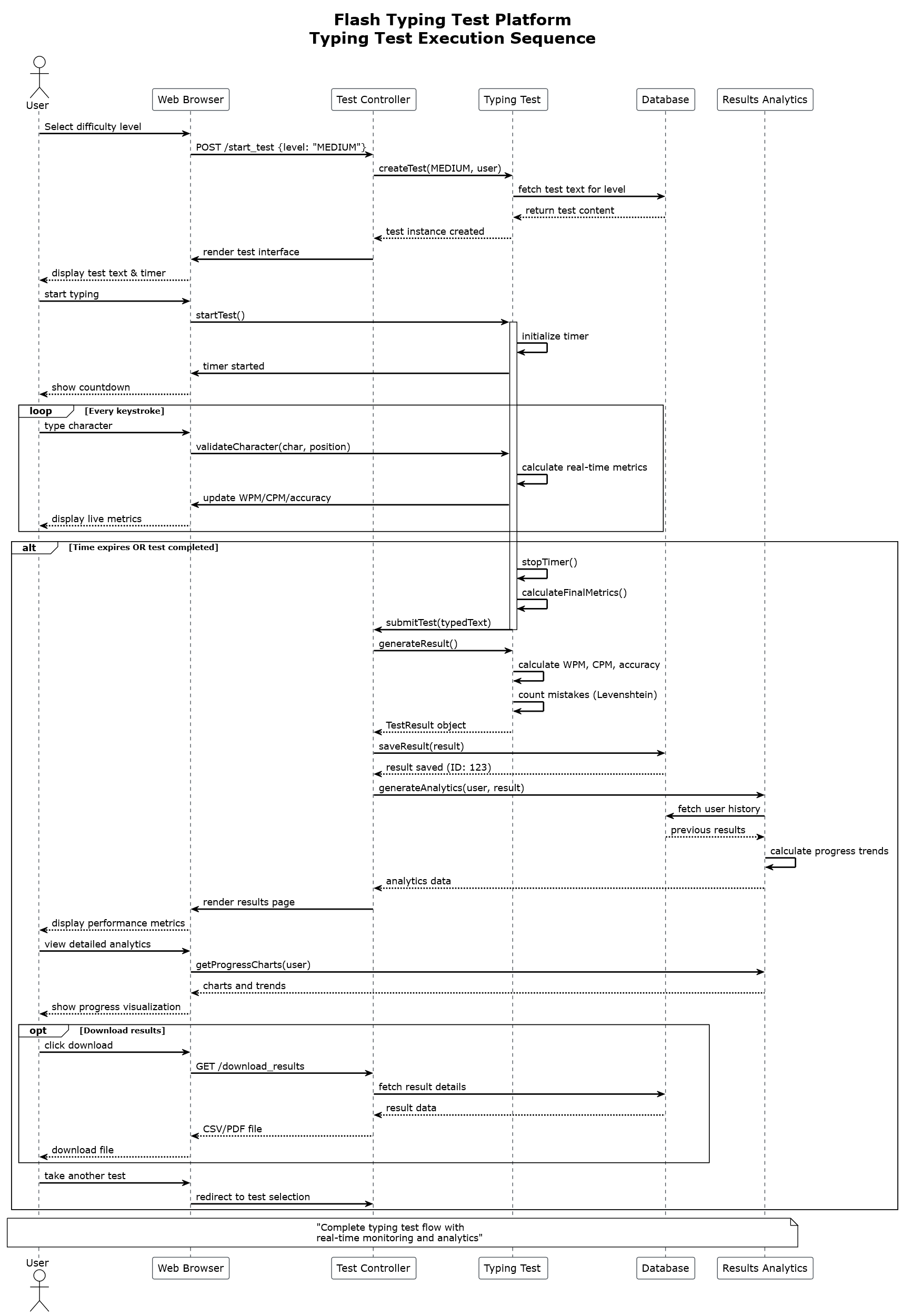


Figure 7: Sequence Diagram of Flash

### Process Modelling using Activity Diagrams

**Activity Diagram**

This activity diagram illustrates the streamlined user workflow for the Flash typing test platform, showing the core business process from platform entry to test completion.

Key Process Flow

1. **Platform Entry**: Users can either login for full features or continue as guests for immediate testing

2. **Test Setup**: Selection of difficulty level (Easy/Medium/Hard) followed by test initialization with parallel setup of timer, text display, and virtual keyboard

3. **Core Testing**: Real-time typing process with continuous keystroke processing and live metric updates (WPM/CPM/accuracy) until timer expires

4. **Results Processing**: Conditional handling based on user type - authenticated users get saved results with optional analytics and download capabilities, while guests receive temporary results only

5. **Session Management**: Option to repeat testing or end session

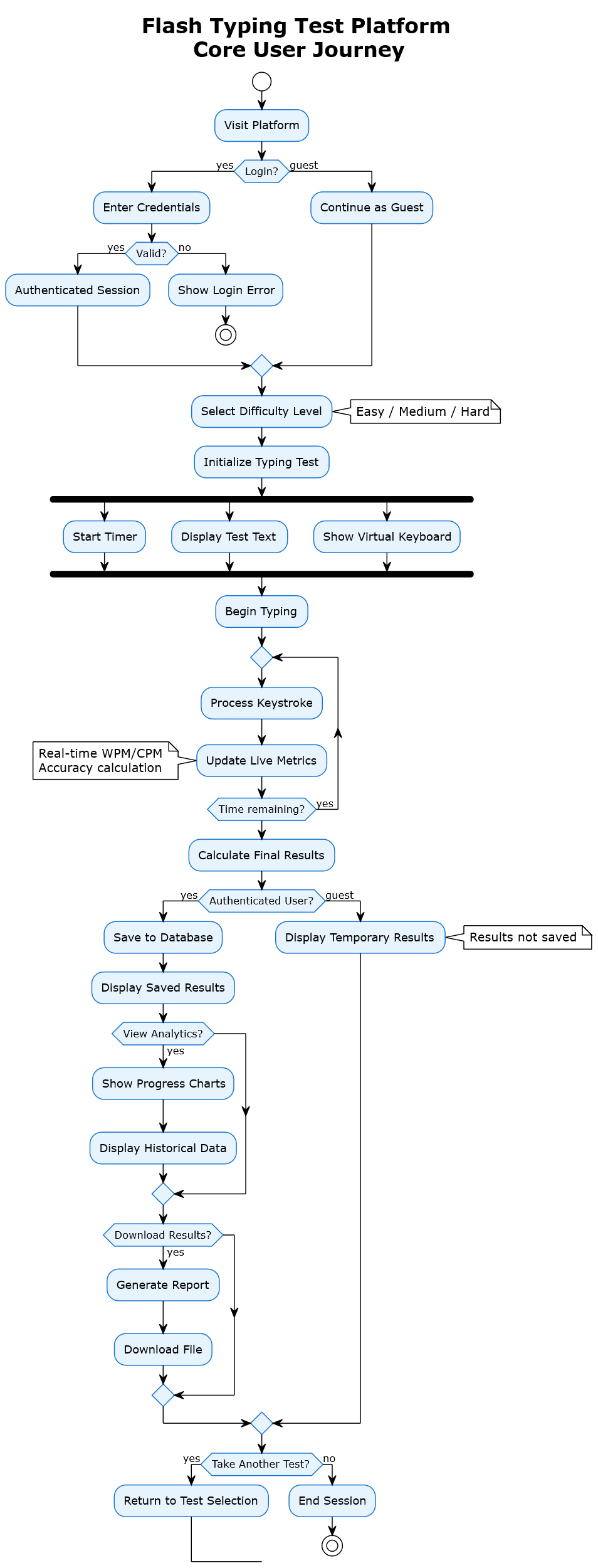
The workflow shows how Flash balances accessibility (immediate guest access) with value-added features (analytics and progress tracking for registered users), creating a smooth user experience that encourages both trial usage and long-term engagement. 

Figure 8: Activity diagram of Flash

## System Design

System design involves planning and defining the architecture, components, and interactions of the system. It is defined as a process of creating an architecture for different components, interfaces, and modules of the system and providing data that will be helpful in implementing such elements in systems.

### Refinement of Class, Object, State, Sequence and Activity diagrams

**Refined Class Diagram**

Refined class structure focusing on core entities - User, TypingTest, TestResult, and DifficultyLevel with essential TestService for business logic. Removes complexity while maintaining key relationships and functionality.

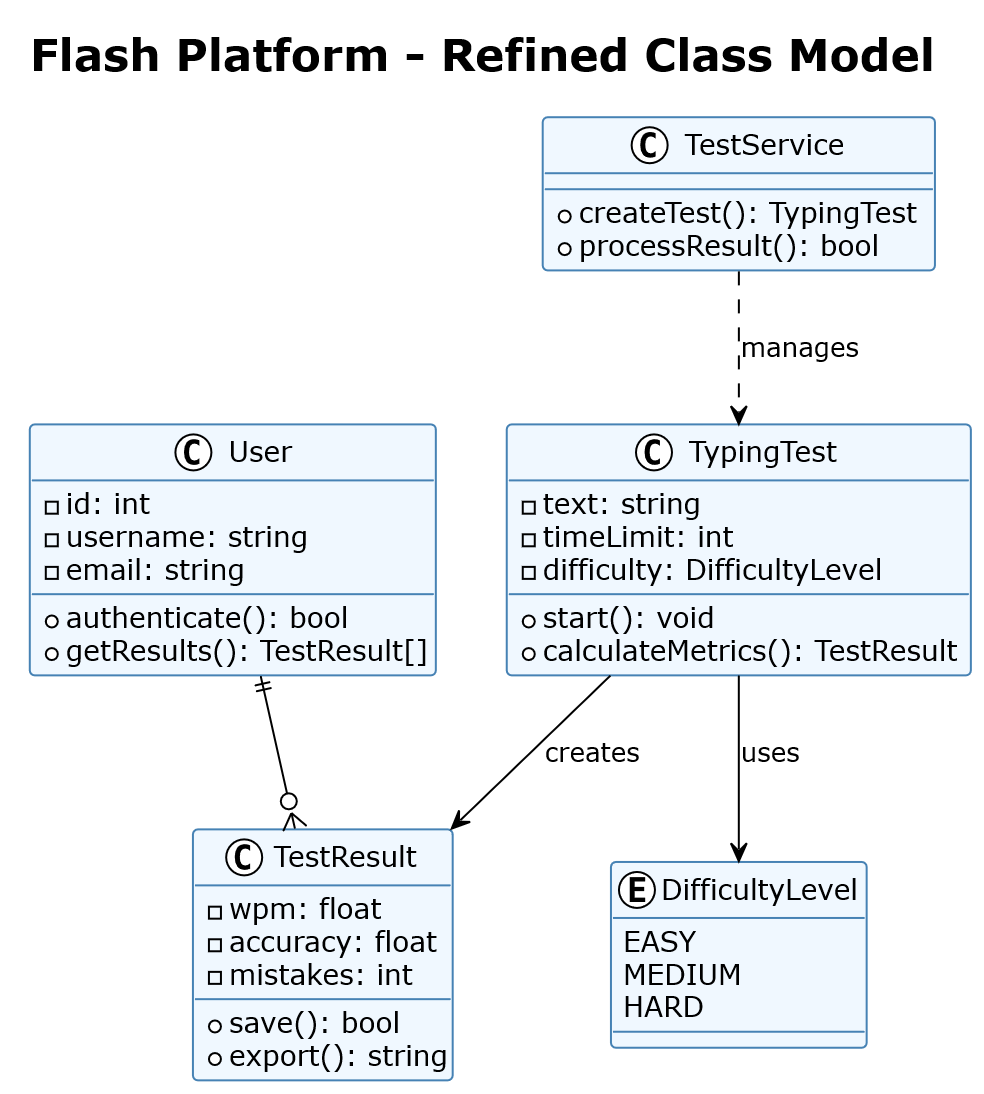


Figure 9: Refined Class Diagram of Flash

**Refined Object Diagram**

The refined object diagram represents runtime snapshot showing user "alice\_smith" completing an Easy-level test with 45.2 WPM and 92.5% accuracy. Demonstrates object relationships during actual system execution.

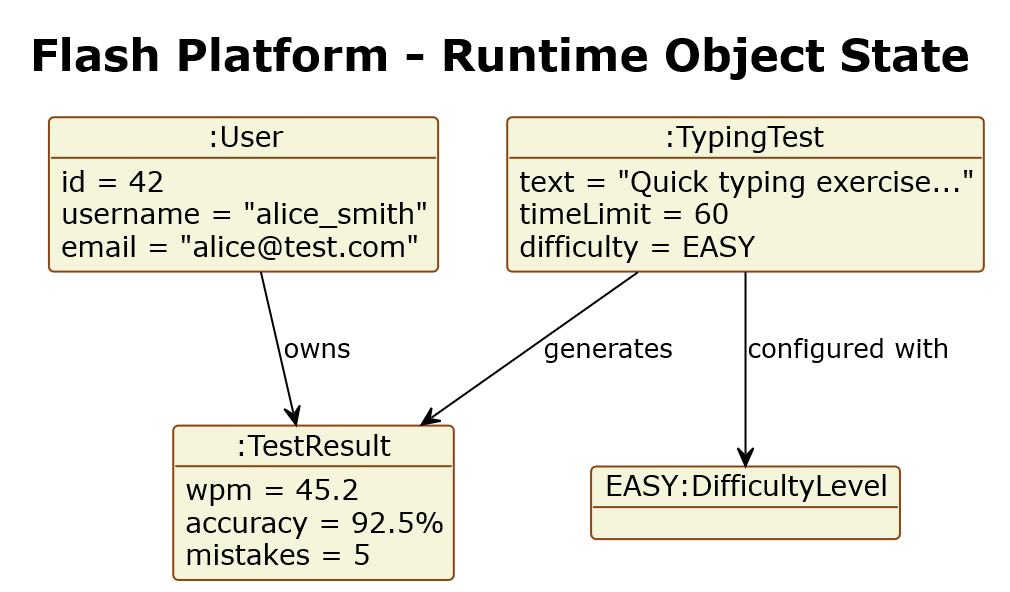


Figure 10: Refined Class Diagram of Flash

**Refined State Diagram**

This refined state model showing three core states - Idle (ready), Testing (active), and Completed (finished). Captures the essential test lifecycle with clear transitions

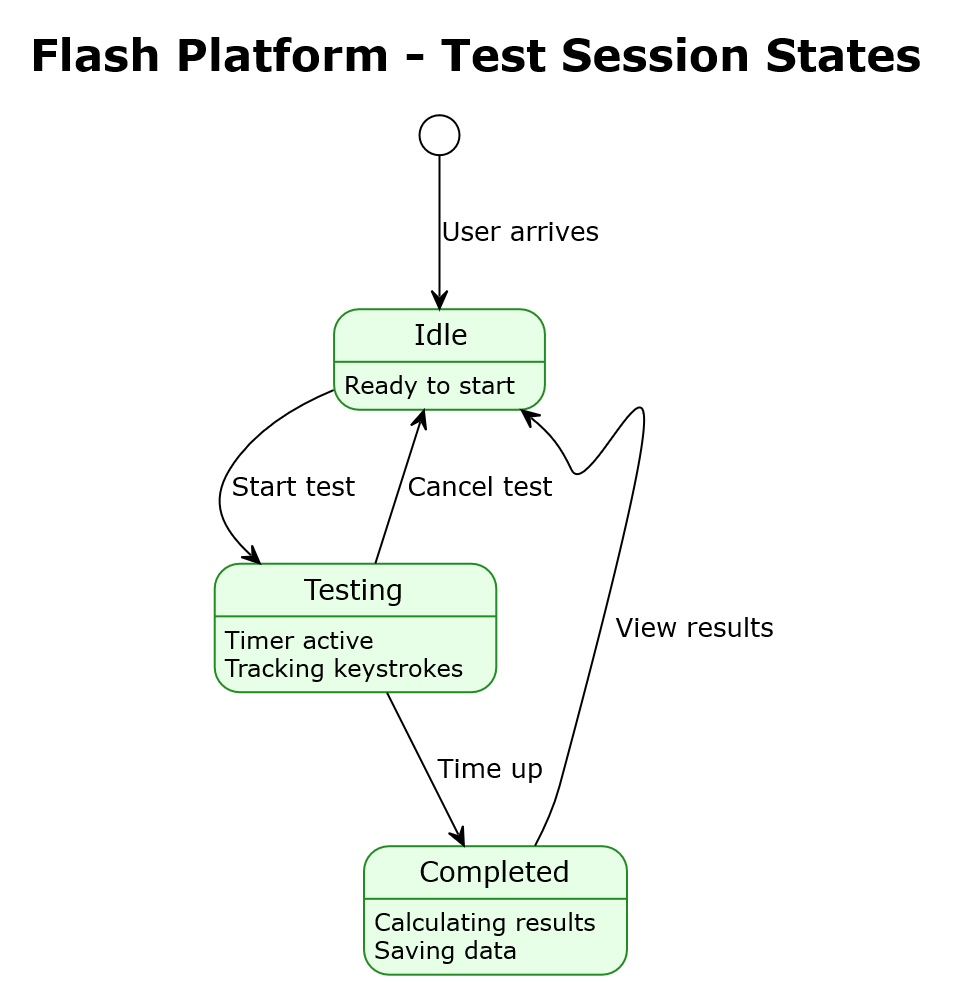


Figure 11: Refined State diagram of Flash

**Refined Sequence Diagram**

This refined sequence diagram shows core interaction flow showing test creation, real-time metric calculation, and result persistence. Demonstrates the key system interactions during a typing test session.

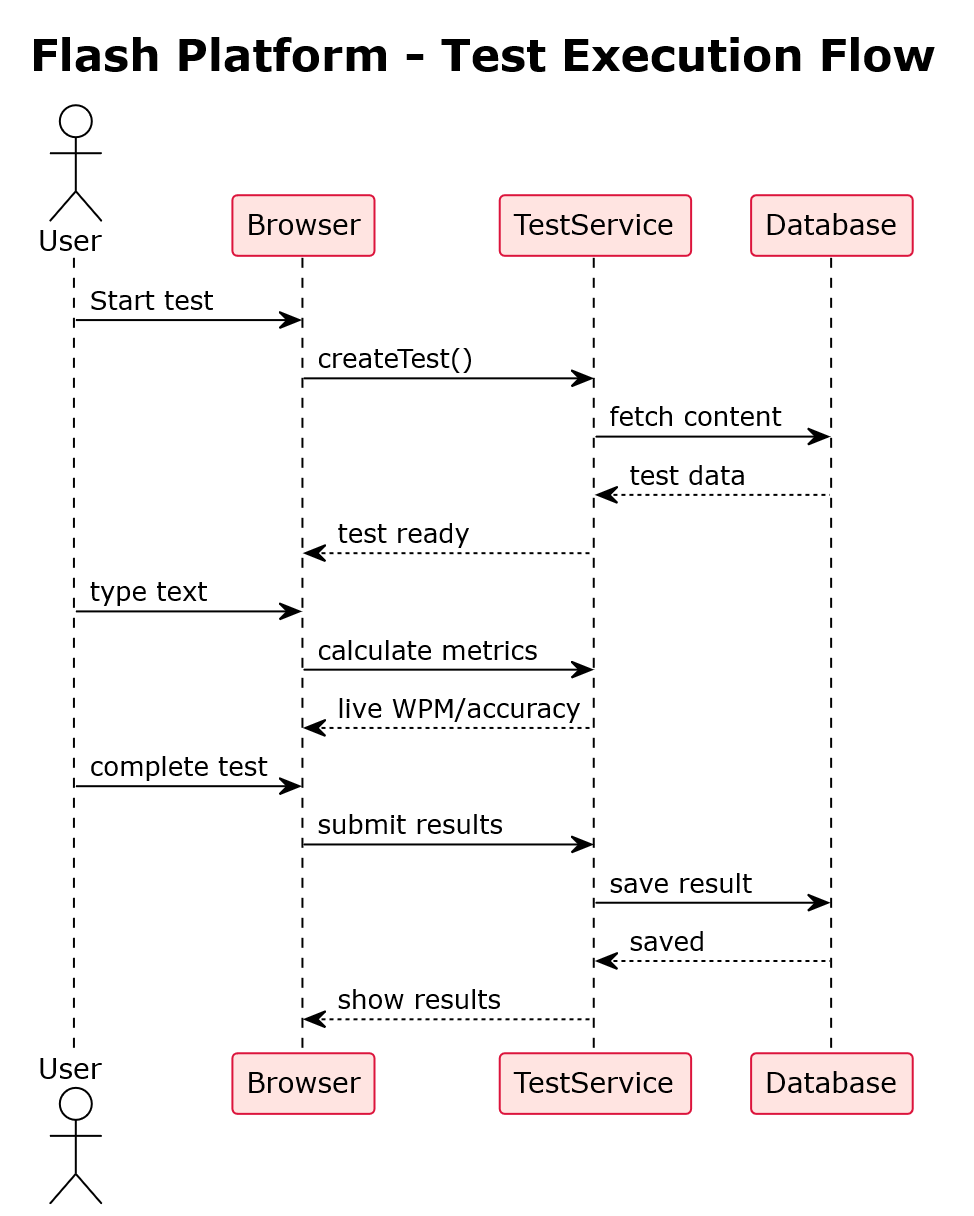


Figure 12: Refined Sequence diagram of Flash

**Refined Activity Diagram**

This refined activity diagram shows the streamlined workflow showing the core test process from platform entry to result display. Highlights the dual path for authenticated vs. guest users with parallel processing during testing.

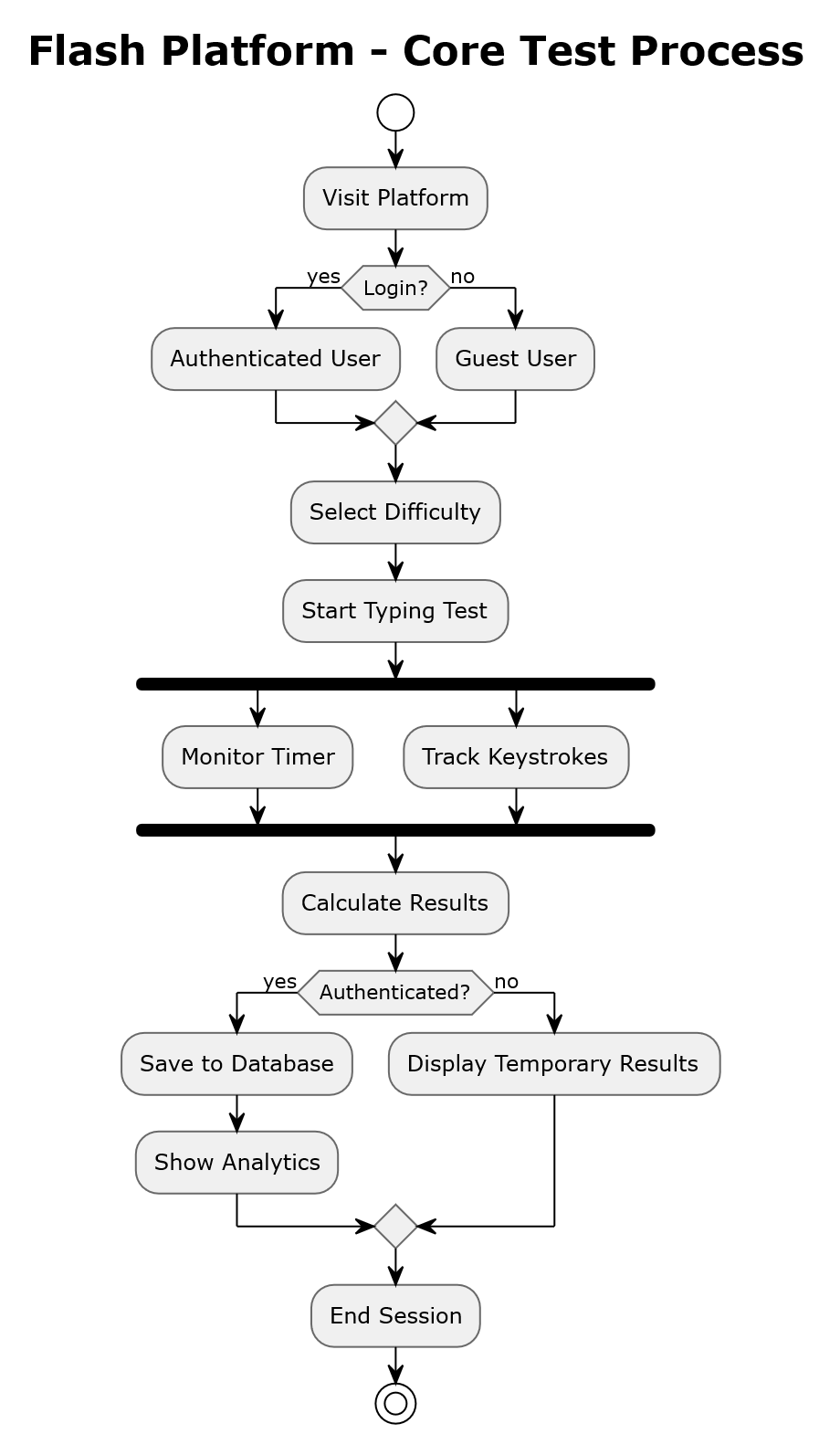


Figure 13: Refined Activity diagram of Flash

### Component Diagrams

This component diagram illustrates the high-level architecture of the Flash typing test platform, showing how major system components interact to deliver the typing speed testing functionality.

The Flash typing test platform uses a five-layer modular architecture for scalability and easy maintenance:

* Presentation Layer – Responsive UI and JavaScript engine for real-time interactions.
* Application Layer – Core business modules: Authentication, Test Controller, Results Analytics, and File Manager.
* Core Engine Layer – Typing Test Engine, Metrics Calculator (WPM/CPM with Levenshtein), and Virtual Keyboard.
* Data Layer – Database Manager for MySQL and Session Handler for authentication state.
* Static Assets – CSS and media files for styling and responsiveness.

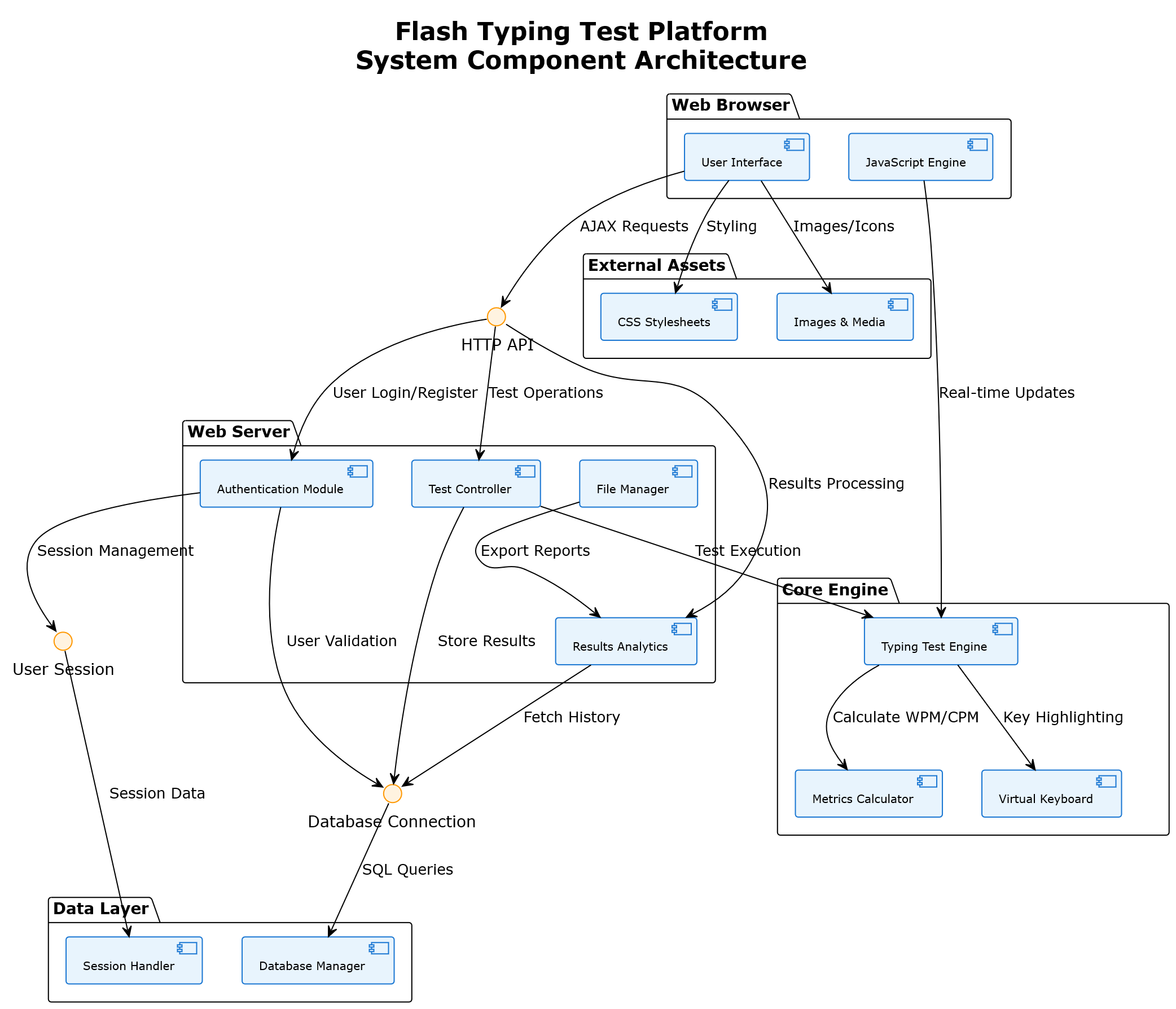


Figure 14: Component diagram of Flash

### Deployment Diagrams

This deployment diagram shows the physical deployment architecture of the Flash typing test platform across different hardware and software environments.

The Flash typing test platform uses a **three-tier deployment**:

* **Client Machine** – Web browsers (Chrome, Firefox, Safari) access the app via HTTP/HTTPS.
* **Web Server** – Apache/Nginx with PHP hosts the application (core PHP pages and static assets).
* **Database Server** – MySQL 5.7+ stores user, result, and download data.

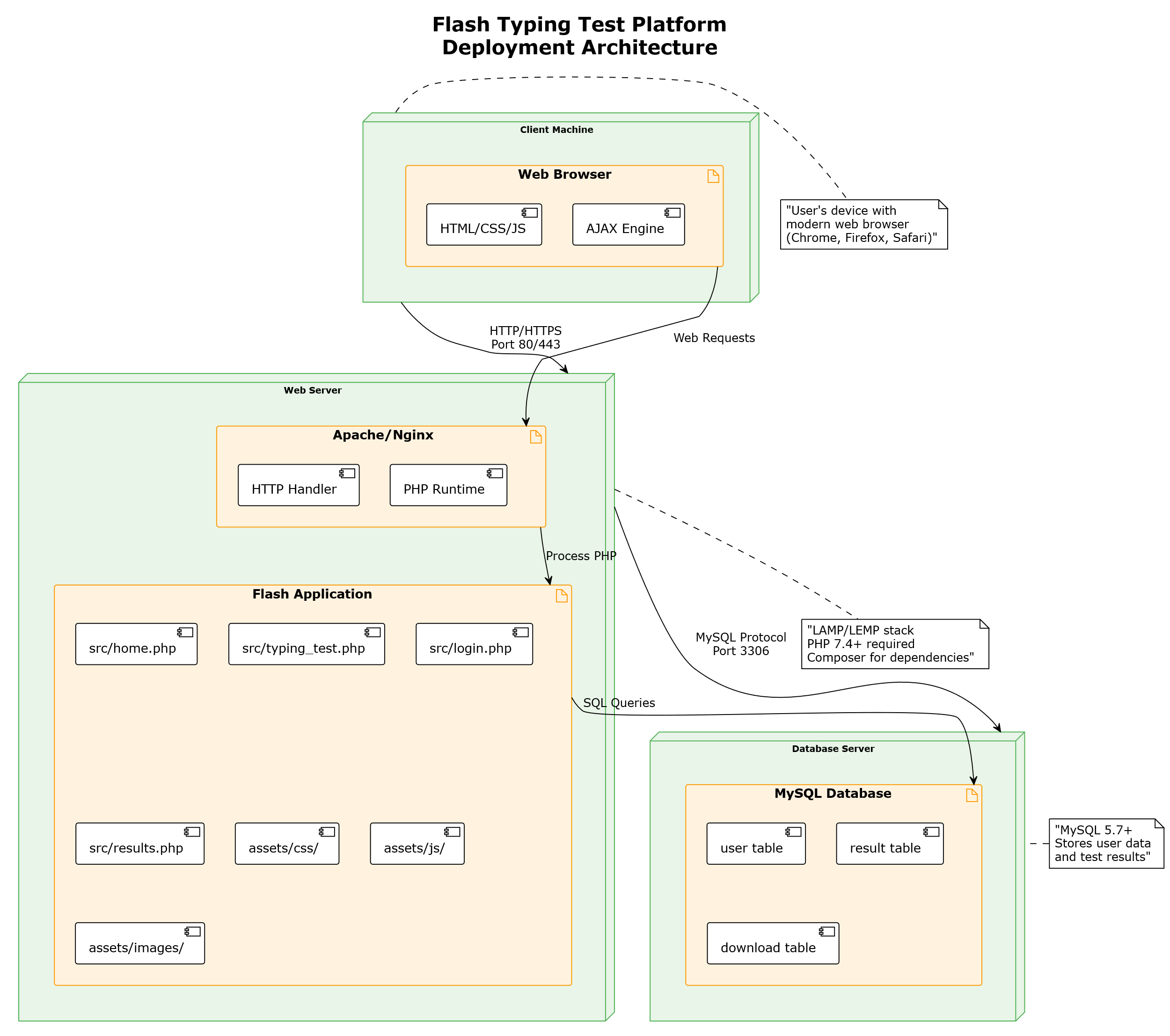


Figure 15: Deployment Diagram of Flash

## Algorithm Details

An algorithm is a procedure used for solving a problem or performing a computation. Algorithms act as an exact list of instructions that conduct specified actions step by step in either hardware- or software-based routines.

Algorithms are widely used throughout all areas of IT. In mathematics, computer programming and computer science, an algorithm usually refers to a small procedure that solves a recurrent problem.

I have used two different algorithms -

#### Levenshtein Distance:

Levenshtein distance is a measure of the similarity between two strings, which takes into account the number of insertion, deletion and substitution operations needed to transform one string into the other.

#### Operations in Levenshtein Distance are:

* **Insertion:** Adding a character to string A.
* **Deletion:** Removing a character from string A.
* **Replacement:** Replacing a character in string A with another character.

Let’s see an example that there is String A: “kitten” which need to be converted in String B: “sitting” so we need to determine the minimum operation required

* **k**itten → **s**itten (substitution of “s” for “k”).
* sitt**e**n → sitt**i**n (substitution of “i” for ????”).
* sittin → sittin**g** (insertion of “g” at the end).

In this case it took three operation do this, so the levenshtein distance will be 3.

There are formulas for calculating various metrics in our typing test. Let's break down each metric:

#### Mistake Calculation (Using Levenstein Distance)

In our implementation, we use the Levenstein distance to calculate the number of mistakes. The Levenstein distance is a measure of the difference between two strings. It represents the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into another.

**Levenshtein Distance Formula:**

Let a and b be two strings of length |a| and |b| respectively. The Levenstein distance between a and b is given by lev(a,b), where:

lev(a,b) = max(|a|, |b|)

lev(a,b) = min{ lev(tail(a), b) + 1, lev(a, tail(b)) + 1, lev(tail(a), tail(b)) + cost}

where cost = 0 if a[0] = b[0], and 1 otherwise and tail(x) is x with the first character removed

In our code, this is implemented in the Levenstein Distance function:



Figure 16; Levenstein distance implementation

The number of mistakes is then equal to the Levenstein distance between the typed text and the original text.

#### Merge Sort Algorithm

Merge sort is a sorting algorithm that follows the divide-and-conquer approach. It works by recursively dividing the input array into smaller subarrays and sorting those subarrays then merging them back together to obtain the sorted array.

In simple terms, we can say that the process of merge sort is to divide the array into two halves, sort each half, and then merge the sorted halves back together. This process is repeated until the entire array is sorted.

**Working Mechanism –**

Here’s a step-by-step explanation of how merge sort works:

**Divide:** Divide the list or array recursively into two halves until it can no more be divided.

**Conquer:** Each subarray is sorted individually using the merge sort algorithm.

**Merge:** The sorted subarrays are merged back together in sorted order. The process continues until all elements from both subarrays have been merged.

# Chapter 4: Implementation and Testing



## Implementation

### Tools Used

Following are the tools used for the accomplishment of this project:

#### Front End Tools

* HTML

In Flash, we used HTML for creating the user interface and structuring the contentof the webpage. It is used to organize content into sections that enhances the readability and accessibility of the webpage. We added HTML elements such as input fields, buttons, labels, links, images and text which helped Flash to become interactive and user-friendly.

#### CSS

In Flash, we used CSS to style the appearance and layout of HTML elements. It is used to design and define different components by the help of universal, id and class selector. We included CSS in different ways such as: inline, internal and external. It is used to controls attributes like colors, fonts, spacing, and positioning which helped to enhance the user interface.

#### JavaScript

In Flash, we used JS to create dynamic contents which provides responses to user actions such as clicks, input and interactive features. We also used JS in client-side validation which helps to validate data before it gets send to the server. It is used to provide the functionality to the webpage and add special effects which made the webpage seem more alive.

#### Back End Tools

* PHP

In Flash, PHP is used for server-side scripting which is used to create dynamic content and backend purpose. It is used for responding to user actions such as form submissions and user authentication. It is used for interacting with database and performing operations such as data retrieval, storage, and manipulation the data.

#### Server

#### Apache Server

In Flash, Apache server is used to run the PHP files and create the dynamic contentof the website.

#### Database

* MySQL

MySQL is an open-source Relational Database Management system (RDBMS) which is widely used for managing and organizing structured data. It is used for storing all the information required to database in Flash. It is used for performing CRUD operation such as create, read, update and delete as requested by users.

#### Documentation Tools

* MS Office

MS word is used to create and edit the documentation of Flash. The tools of MS word such as spell checking and grammar correction was used to enhance the quality of documented content. It is used for tracking and managing the document changes over time.

#### Draw.io

Draw.io is a diagramming tool which is used for creating several types of diagrams, charts, and visual representations. In Flash, all the diagrams were created using components which were freely available in draw.io.

#### Figma

Figma is a design tool which we used for designing the wireframe of the project. In Flash, we created our UI design using various elements which are freely available in Figma.

### Implementation details of modules (Description of Procedures/Functions)

Different modules of the system are described below:

#### Users Module

1. Login Module



Figure 17: Code snippet for login

The output for above mentioned logic is demonstrated in appendix Figure.

#### Register Module

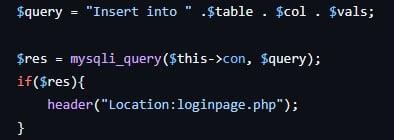
The output for above mentioned logic is demonstrated in appendix Figure.

Figure 18: Code snippet for Register

#### Typing Module

#### Level Selection

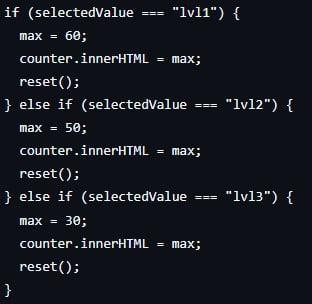
The output for above mentioned logic is demonstrated in appendix Figure.

Figure 19: Code snippet for Level Selection

#### Paragraphed Generator

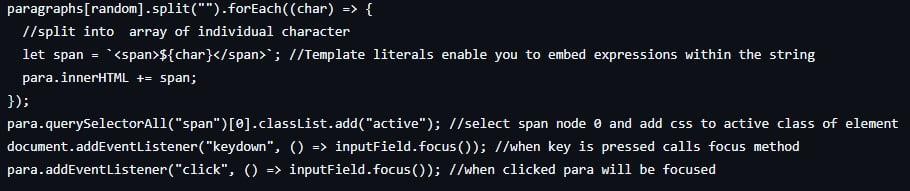
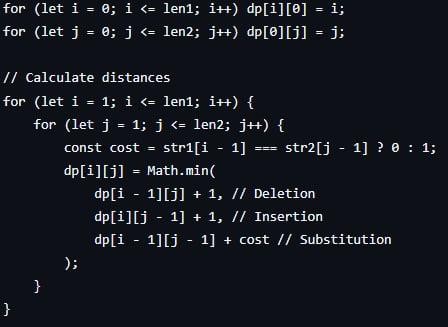


Figure 20: Code snippet for Paragraph Generator

The output for above mentioned logic is demonstrated in appendix Figure.

#### Levenshtein Distance

Figure 21: Code snippet for Levenshtein Distance

The output for above mentioned logic is demonstrated in appendix Figure.

#### Result Module

1. Result

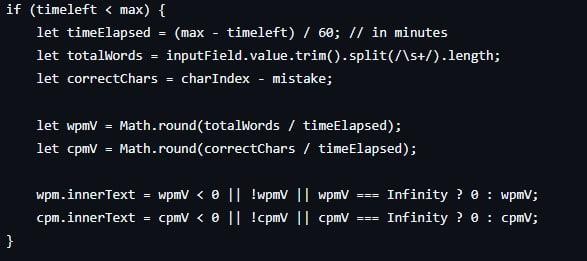
The output for above mentioned logic is demonstrated in appendix Figure.

Figure 22: Code snippet for Result

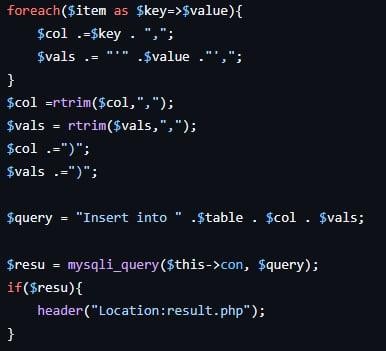


Figure 23: Code snippet for Result

The output for above mentioned logic is demonstrated in appendix Figure.

#### Download Module

i. Download

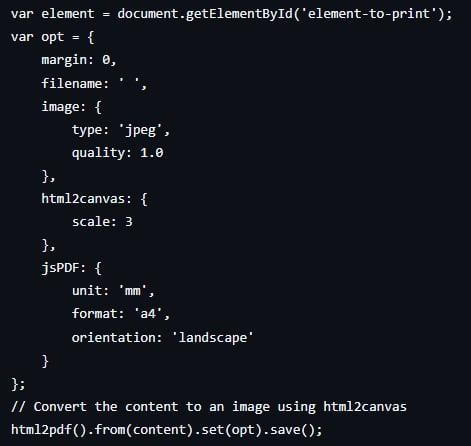


Figure 24: Code snippet for Download

The output for above mentioned logic is demonstrated in appendix Figure

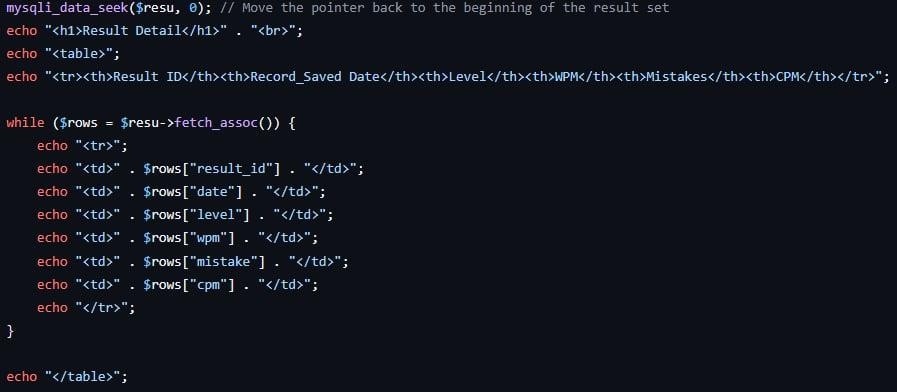


Figure 25: Code snippet for Download

The output for above mentioned logic is demonstrated in appendix Figure

### References

|  |  |
| --- | --- |
| [1] | E. Gondre and Grabowski, "Problems of Computer Keyboarding in anEFL Context," *Writing Research,* pp. 27-52, 2008. |
| [2] | "Words per Minute," [Online]. Available: https://en.wikipedia.org/wiki/Words\_per\_minute. |
| [3] | "Lucidchart," [Online]. Available: https[://www.lucidchart.com/pages/](http://www.lucidchart.com/pages/uml-use-case-diagram)uml[-use-case-diagram.](http://www.lucidchart.com/pages/uml-use-case-diagram) |
| [4] | "Lucid Chart," [Online]. Available: https[://www.lucidchart.com/pages/](http://www.lucidchart.com/pages/data-flow-diagram/data-flow-)d[ata](http://www.lucidchart.com/pages/data-flow-diagram/data-flow-)-[flow](http://www.lucidchart.com/pages/data-flow-diagram/data-flow-)-[diagram/data-flow-](http://www.lucidchart.com/pages/data-flow-diagram/data-flow-)diagram- symbols. [Accessed 02 April 2023]. |
| [5] | "Smart Draw," [Online]. Available: https[://www.s](http://www.smartdraw.com/entity-)mar[tdraw.com/entity-](http://www.smartdraw.com/entity-) relationship-diagram/. [Accessed 01 April 2023]. |
| [6] | "LiveChat," [Online]. Available: https[://www.livechat.com/typing](http://www.livechat.com/typing-)-speed- test/#/. |
| [7] | "TypingTyping.com," [Online]. Available: https[://www.typingtyping.com/wpm](http://www.typingtyping.com/wpm-calculator/)-[calculator](http://www.typingtyping.com/wpm-calculator/)/. |
| [8] | "Typing Test," [Online]. Available: https[://www.typing.com/student/tes](http://www.typing.com/student/tests)ts. |
| [9] | "TypingTest," [Online]. Available: https[://www.typingtest.com/.](http://www.typingtest.com/) |
| [10] | "draw.io," [Online]. Available: https://app.diagrams.net/. |