**CEBU INSTITUTE OF TECHNOLOGY**

**UNIVERSITY**

COLLEGE OF COMPUTER STUDIES

Software Requirements Specifications

for

CyberKids

**CyberKids**

**­­**

Baguio, John Kenneth B.

Cultura, Jesson Chyd M.

Dedumo, Emmanuel A.

Vequiso, A­shley Josh V.

­­

Dr. Eugene Busico

Adviser

Consultation Schedule: BUSITUE200300

March 12, 2025

Change History

Table of Contents

Change History 2

Table of Contents 3

1. Introduction 4

1.1. Purpose 4

1.2. Scope 4

1.3. Definitions, Acronyms and Abbreviations 4

1.4. References 4

2. Overall Description 5

2.1. Product perspective 5

2.2. User characteristics 5

2.4. Constraints 5

2.5. Assumptions and dependencies 6

3. Specific Requirements 7

3.1. External interface requirements 7

3.1.1. Hardware interfaces 7

3.1.2. Software interfaces 7

3.1.3. Communications interfaces 7

3.2. Functional requirements 7

Module 1 7

Module 2 8

3.4 Non-functional requirements 8

Performance 8

Security 8

Reliability 8

# Introduction

## Purpose

CyberKids is an interactive puzzle and strategy game designed to immerse elementary school students in real-world cybersecurity scenarios. It aims to address the increasing vulnerability of children to online threats by providing an engaging and effective learning environment focused on password security, phishing awareness, and online privacy. The intended audience includes grades 5-6 elementary students, computer teacher, system administrators, and developers responsible for maintaining and improving the platform.

## Scope

CyberKids is a web-based educational game that helps students learn about cybersecurity best practices through a series of engaging missions. The game focuses on information sharing awareness, password security, and phishing scam detection.

**What the Software Will Do (**Core Functionalities**)**

1. **Single-Player Exploration**: Players control a character navigating a virtual world with different digital zones representing online platforms in each level.
2. **Mission-Based Learning:** Players complete three major cybersecurity challenges:
   * Data Leak Investigation – Identifying safe and unsafe personal information to share.
   * Password Fortress Defense – Constructing strong passwords to protect against hacking attempts.
   * Cyber Escape Room – Identifying phishing and scam attempts to escape a locked digital room.
3. **Gamification & Progress Tracking:** 
   * Players earn Cyber Points for correct decisions and lose points for unsafe actions.
   * A dynamic leaderboard ranks students based on performance.
4. **Teacher Dashboard**:

* Educators can track student progress and view mission completion rates.
* Teachers can assign specific missions and review student performance.

**What the Software Will NOT Do**

* No Real Social Media or Online Interaction: The game does not connect to actual social media platforms or online services. All interactions are within a simulated environment.
* No Multiplayer Features: CyberKids is a single-player game with a leaderboard for competition, but no real-time multiplayer interactions.
* No Real Data Collection: Players will never be required to enter real personal information—only fictional, simulated scenarios are used for learning.

## Definitions, Acronyms and Abbreviations

This section provides definitions of key terms, acronyms, and abbreviations relevant to the implementation of the CyberKids system.

**Acronyms:**

Acronyms used in this document shall be interpreted as follows:

* API – Application Programming Interface
* CORS – Cross-Origin Resource Sharing
* CRUD – Create, Read, Update, Delete
* CI/CD – Continuous Integration / Continuous Deployment
* JWT – JSON Web Token
* SQL – Non-Relational Database Query Language
* REST API – Representational State Transfer API
* UI – User Interface

**Abbreviations:**

Abbreviations used in this document shall be interpreted as follows:

* JS – JavaScript
* DB – Database
* UX – User Experience
* HTTP – HyperText Transfer Protocol
* SQL – Structured Query Language
* URL – Uniform Resource Locator

**Definitions:**

Definitions used in this document shall be interpreted as follows:

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Authentication | The process of officially recognizing an educational institution or program as meeting specific quality standards. |
| Authorization | The process of granting or denying access to specific resources or functionalities. |
| Backend | The server-side part of a software application, responsible for data processing, logic, and database interactions |
| Cybersecurity | The practice of protecting systems, networks, and programs from digital attacks. |
| Database | An organized collection of structured information, or data, typically stored electronically. |
| Encryption | The process of converting data into a secure format to prevent unauthorized access. |
| Front-end | The user-facing part of a web application that handles the visual presentation and interaction. |
| Git | A distributed version control system for tracking changes in source code. |
| GitHub | A web-based platform for version control and collaboration using Git |
| Malware | Malicious software designed to harm, exploit, or disrupt computer systems. |
| React.js | A JavaScript library for building user interfaces. |
| Scams | Fraudulent schemes used to deceive people into providing money, personal details, or access to sensitive data. |
| Spring Boot | A Java-based framework for building microservices and web applications. |
| Vite | A build tool for modern web development. |
| Wireframe | A visual guide that represents the skeletal framework of a website or application. |

## References

[1] IEEE Computer Society. (1998). IEEE 830-1998: Recommended Practice for Software Requirements Specifications. IEEE Std 830-1998. Source: IEEE Xplore Digital Library.

[2] React. (2019). Glossary. Source: <https://legacy.reactjs.org/docs/glossary.html>

[3] Axios Contributors. (2024). Axios API Reference. Source: <https://axios-http.com>

[4] National Institute of Standards and Technology (NIST). (2018). Cybersecurity Framework. Source: <https://www.nist.gov/cyberframework>

# Overall Description

## Product perspective

These modules ensure a seamless experience for students while providing teachers with tools to monitor progress.

**Modular Decomposition**

**Module 1: User Registration & Profile Management**

* Players create an account with a unique username and password. No real personal data is required.
* Players can view cyberpoints in their profile

**Module 2: Game Missions**

* Players analyze digital case files and drag and drop personal information into “Safe to Share” and “Not Safe to Share” categories.
* Players loot chests and collect password fragments to build stronger, more secure passwords while defending against cyberattacks.
* Players navigate a virtual escape room, identifying phishing emails, suspicious links, and scam messages to uncover security codes.

**Module 3: Leaderboard and Scoring System**

* Displays top players dynamically, updating scores as students progress through missions.
* Scores are based on mission completion time.

**Module 4: Teacher Dashboard**

* Teachers can view individual and class-wide progress, including mission completion and scores.

## User characteristics

CyberKids is designed primarily for elementary and junior high school students, introducing them to cybersecurity concepts in a fun, interactive way. The platform helps students develop safe online habits through gamified learning experiences, while teachers and parents can track their progress and guide them toward better digital awareness

.

1. **Student Users**

The primary users of CyberKids are young learners who are becoming more engaged with technology and the internet. The platform is tailored to their learning style through an intuitive, interactive, and engaging game-based format.

* **Need for Early Cybersecurity Awareness**

Many students lack proper guidance on how to stay safe online, making them vulnerable to phishing, scams, and cyberbullying. CyberKids fills this gap by introducing essential cybersecurity knowledge through engaging missions and challenges that teach digital safety without overwhelming young users with technical jargon.

* **Interactive and Gamified Learning**

Traditional learning materials may not hold the attention of young students. CyberKids ensures engagement by integrating reward systems, level progression, and real-time feedback, making cybersecurity education fun and rewarding. Students can unlock new levels and earn badges as they progress through different cybersecurity challenges.

* **Age-Appropriate Content**

Unlike general cybersecurity platforms designed for adults, CyberKids presents lessons in a way that is simple, visual, and interactive, making it easier for young learners to grasp concepts like password security, phishing awareness, and safe social media usage.

1. **Teachers and Educators**

Teachers play a crucial role in guiding students' learning experience and ensuring they understand the importance of cybersecurity. CyberKids supports teachers by offering a dashboard that tracks student progress and performance.

* **Classroom Integration**

Teachers can integrate CyberKids into their curriculum as an interactive learning tool to supplement traditional cybersecurity education. The platform provides structured challenges that align with key digital literacy learning goals.

* **Student Progress Monitoring**

Educators can view student scores, track progress across different cybersecurity topics, and identify areas where students need improvement. This data-driven approach helps teachers offer targeted guidance to students struggling with certain cybersecurity concepts.

* **Facilitating Cyber Safety Discussions**

Teachers can use CyberKids as a starting point for classroom discussions about online safety, responsible internet use, and digital ethics, ensuring students apply what they learn in real-life scenarios.

1. **Parents and Guardians**

Parents are often concerned about their children’s online activities and exposure to digital threats. CyberKids provides them with a way to ensure their children are learning safe online behaviors through an engaging and structured platform.

* **Parental Supervision and Insights**

Parents can monitor their child’s progress through reports on completed missions, scores, and areas needing improvement. This enables parents to have meaningful conversations about cybersecurity at home.

* **Encouraging Safe Internet Habits at Home**

With CyberKids, parents can actively participate in reinforcing safe online behaviors, helping their children apply cybersecurity lessons beyond the digital classroom.

**User Roles and Privileges**

**1. Students (Players)**

* **Description:** Students in Grades 5-6 at CIT-University who participate in the game to learn about cybersecurity concepts.
* **Roles & Privileges:**
  + Access and play all game missions.
  + View their own scores and rankings on the leaderboard.
  + Customize their profile (avatar and display name only).

**2. Teachers**

* **Description:** Educators who monitor and assess student progress in CyberKids.
* **Roles & Privileges:**
  + Access the Teacher Dashboard to track students' progress, scores, and mission completion rates.
  + Generate performance reports to assess learning outcomes.
  + View leaderboard standings for class rankings.

**3. Admin**

* **Description:** System administrators responsible for managing and maintaining the CyberKids application.
* **Roles & Privileges:**
  + Maintain system functionality and ensure a smooth gaming experience.
  + Manage technical updates, bug fixes, and security measures.
  + Oversee user management, including teacher and student accounts.
  + Ensure leaderboard accuracy and resolve any scoring discrepancies.

## 2.4. Constraints

The development of CyberKids is subject to various of technical, regulatory, and operational constraints that can impact the system design, operation, and deployment. Such constraints need to be critically analyzed to ensure the system is pragmatic, effective, and compliant with the relevant standards.

**Regulatory Policies**

CyberKids must comply with data protection law and education code in a bid to protect user information and enable the consistency of career guidance services. The system should be compliant with:

* The Data Privacy Act of 2012 (RA 10173) intends to protect students' information, grades, and career options by guaranteeing confidentiality and security.

**Hardware Limitations**

The system is designed to run on standard desktop and laptop computers commonly used in CIT-University. However, there are some limitations to consider:

* Support for Low-End Hardware – The system must be optimized to function smoothly on low-end to mid-range hardware, ensuring accessibility for all students, even those with older devices.
* Resource Usage Optimization – To prevent performance issues, the system should minimize excessive CPU and memory usage while still providing a seamless user experience.

**Interfaces to Other Applications**

The system operates independently and does not integrate with external applications, but this comes with certain constraints:

* No Social Media Integration – Unlike other modern educational tools, the system does not connect with social media platforms for sharing progress or achievements.
* Internal Data Storage Only – All user data, including scores, progress, and mission completion information, is stored within an internal database. This limits external access but ensures security and control over student information.

**Parallel Operation**

While the game is designed as a single-player experience, multiple students can play simultaneously. However, there are some operational constraints:

* Independent Sessions – Each student’s game session runs independently, meaning their progress does not interfere with other users.
* Scalability Considerations – Although the system allows multiple concurrent users, excessive simultaneous logins may require server performance optimization.

**Audit Functions**

The system includes mechanisms for monitoring student activity and progress, though there are some limitations:

* Teacher Dashboard Tracking – Teachers can monitor student progress through a dashboard, but they can only view the data and cannot modify student scores or mission completion records.
* Session Logs and Data Retention – The system logs game sessions, scores, and mission completion data, but long-term storage may be limited based on server capacity.

**Control Functions**

Different user roles have varying levels of access and control over the system, but there are restrictions in place:

* Administrative Oversight – Admin users have full control over system maintenance, bug fixes, and security updates. They ensure the system remains functional and up to date.
* Limited Teacher Modifications – While teachers can monitor student progress, they do not have the ability to alter student scores or modify gameplay mechanics.

**Reliability Requirements**

The system aims to be stable and reliable, though certain factors may impact its performance:

* Crash Prevention – The system must function without unexpected crashes during gameplay to ensure a smooth user experience.
* Data Security and Retrieval – Student progress and leaderboard scores must be securely stored and retrievable in case of system failures.

**Criticality of the Application**

Although the system is designed primarily as an educational tool, reliability remains a key priority:

* Not a Mission-Critical System – The game serves as a supplementary educational resource rather than a system that directly impacts grades or official records.
* Importance of Seamless Learning – Despite not being mission-critical, the system must remain stable to maintain student engagement and educational value.

**Safety and Security Considerations**

Ensuring user security and data protection is a fundamental requirement of the system:

* No Collection of Personal Data – The system does not collect real-world personal data, ensuring privacy and compliance with data protection standards.
* Unauthorized Access Prevention – Security measures must be in place to prevent unauthorized access, particularly for student accounts, to protect progress and sensitive game data.

## 2.5. Assumptions and dependencies

The development and functionality of CyberKids rely on the following assumptions and dependencies. Any changes to these factors may impact the system requirements and require modifications to the software.

**Assumptions**

1. **Hardware Availability**
   * The game is assumed to run on standard desktop computers available at CIT-University.
   * Devices used must have basic input peripherals (keyboard and mouse).
2. **Operating System Compatibility**
   * The system assumes that devices will run on a Windows (Google Chrome, Firefox, Edge). If a specific OS is not supported, additional development effort may be required.
3. **Internet Connectivity**
   * The leaderboard and teacher dashboard assume a stable internet connection for real-time updates.
4. **User Digital Literacy**
   * Students (Grades 5-6) are assumed to have basic computer skills, such as navigating a game interface, using a mouse/keyboard, and following on-screen instructions.

**Dependencies**

1. **Database System**
   * The leaderboard, student progress tracking, and user profiles depend on a functional database (e.g., MySQL, Firebase).
2. **Web-Based Frameworks and Technologies**
   * The game relies on web-based technologies (e.g., React, JavaScript, HTML5, CSS) for execution.
3. **Security Policies**
   * The system follows CIT-University’s IT security policies to prevent unauthorized access and protect student data.
   * Changes in security regulations may require updates to authentication and access control mechanisms.
4. **Server Availability**
   * The leaderboard and teacher dashboard depend on a centralized server for data storage and retrieval.
5. **Third-Party Libraries and APIs**
   * The system may use external libraries or APIs for password encryption, data storage, and UI enhancements.
   * If these libraries become deprecated, alternative solutions must be integrated.

# Specific Requirements

## External interface requirements

### 3.1.1. Hardware interfaces

The system is designed to operate on standard computing hardware used at CIT-University, ensuring compatibility with a wide range of devices. The hardware interface requirements include:

* **Client-side Requirements**
  + **Devices:** Desktop and Laptop with a modern web browser.
  + **Browers:** Chrome, Firefox, Edge, and Safari (latest stable versions).
  + **Internet Connection:** Minimum 5 Mbps for smooth interaction.
* **Supported Devices** – The software will run on desktop and laptop computers with Windows operating systems, ensuring accessibility for students. No support for mobile or tablet devices is currently planned.
* **Minimum Hardware Specifications** – The system must be optimized to function on low-end to mid-range computers, requiring at least:
  + Processor: Intel Core i3 (or equivalent) with a 2.0 GHz clock speed
  + RAM: 4 GB minimum, 8 GB recommended for optimal performance
  + Storage: At least 2 GB of available disk space for installation and data storage
  + Graphics: Integrated GPU support (no dedicated graphics card required)
* **Peripheral Device Support** – The system will support standard input and output devices, including:
  + Keyboard and Mouse: Required for user interactions in the system

### 3.1.2. Software interfaces

The system will interact with various software components to ensure smooth functionality, including the operating system, database management system, and web-based services.

* **Operating System Compatibility** – The software will be compatible with the following operating systems:
  + Windows 10 and later – Officially supported, tested for stability and security
* **Database Management System (DBMS)** – The system will store all user progress, scores, and logs in a relational database. Supported databases include:
  + MySQL 8.0 or later – Primary database for storing structured data
  + PostgreSQL 13 or later – Alternative database option for scalability
* **Frameworks and Development Tools** – The system is built using the following technologies:
  + Backend: Java Spring Boot for API handling and business logic processing
  + Frontend: React.js for user interface rendering
  + Data Management: Hibernate ORM for database interactions
* **APIs:**
  + RESTful APIs for communication between frontend and backend.

### 3.1.3. Communications interfaces

The CyberKids requires connectivity to various network services to enable multiplayer interactions, leaderboard updates, and real-time event handling.

* **Internet Connectivity Requirements:**
  + The game requires an active internet connection for live updates, leaderboard rankings, and cloud-based data storage.
* **Protocols Used:**
  + HTTP/HTTPS: Secure communication between client and game server.
  + WebSocket Protocol: For real-time interactions, such as in-game chat, multiplayer elements, or live updates.
  + RESTful API / GraphQL: Handles game progress, leaderboard updates, and authentication requests.

## Functional requirements

### Module 1: User Registration & Authentication

#### **1.1 User Registration**

*Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-001 |
| Use Case Name | User Registration and Log in |
| Actor | Student, Teacher |
| Description | |  | | --- | | This use case describes the process by which a new student registers for an account in the CyberKids application. The user provides a username and password, and upon successful validation, the system stores the credentials and grants access to the login screen. |  |  | | --- | |  | |
| Flow of Events | 1. User opens the CyberKids application.  2. User selects "Register".  3. User chooses an account type (Student).  4. User enters a username.  5. User enters and confirms a password.  6. System validates the input.  7. If input is invalid, an error message is displayed, and the user is prompted to re-enter details.  8. If input is valid, the system securely stores the credentials.  9. System displays a "Registration Successful" message.  10. User is redirected to the login screen. |
| Precondition | |  | | --- | | The user does not have an existing account. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The user successfully creates an account and is redirected to the login screen. |  |  | | --- | |  | |

#### **1.2 User Log in**

1. *Use Case Description*

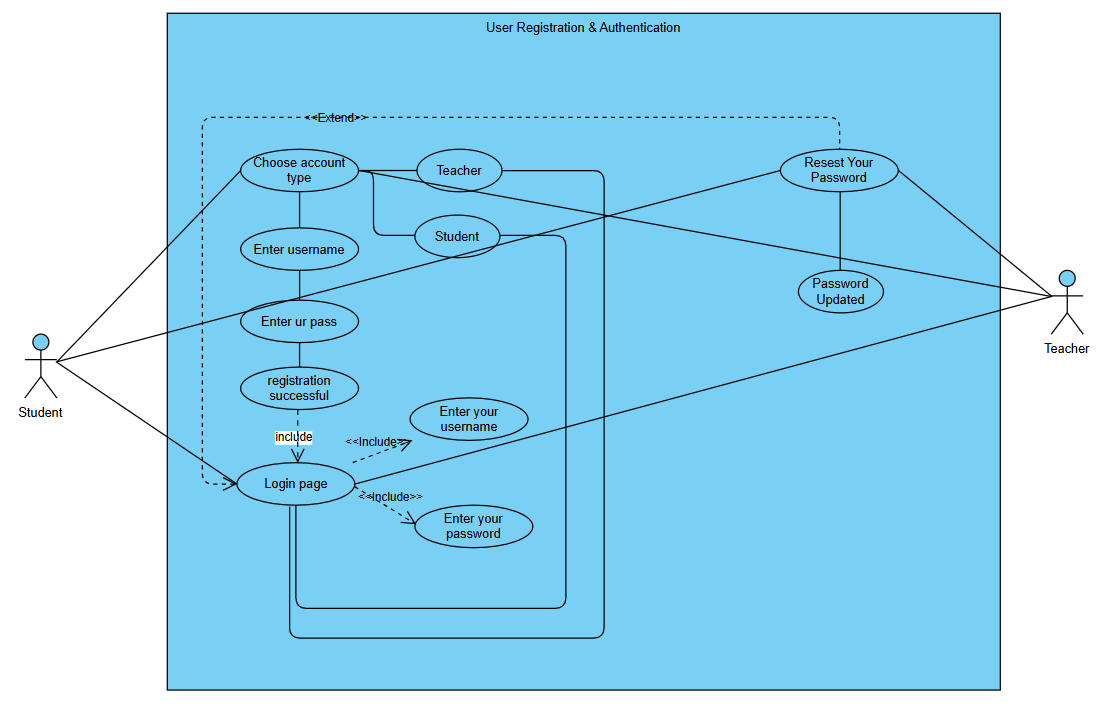
|  |  |
| --- | --- |
| Use Case ID | UC-002 |
| Use Case Name | User Log in |
| Actor | Student, Teacher |
| Description | |  | | --- | | This use case describes the process of logging into the system using a valid username and password. It ensures that only authorized users can access the system, assigning appropriate permissions based on user roles. |  |  | | --- | |  | |
| Flow of Events | 1. The user navigates to the login page. 2. The system prompts the user to enter a **username** and **password**. 3. The user inputs their credentials and submits the login request. 4. The system verifies the provided credentials against the database. 5. If the credentials are valid: 6. The system authenticates the user. 7. The system assigns the appropriate role (Student or Teacher). 8. The system redirects the user to their respective dashboard. 9. The login session is initiated, allowing access to authorized features. |
| Precondition | |  | | --- | | The user must have a registered account in the system.  The system must be online and connected to the database. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | If login is successful, the user gains access to the system with the appropriate role-based permissions. |  |  | | --- | |  | |

#### **1.3 Profile Management**

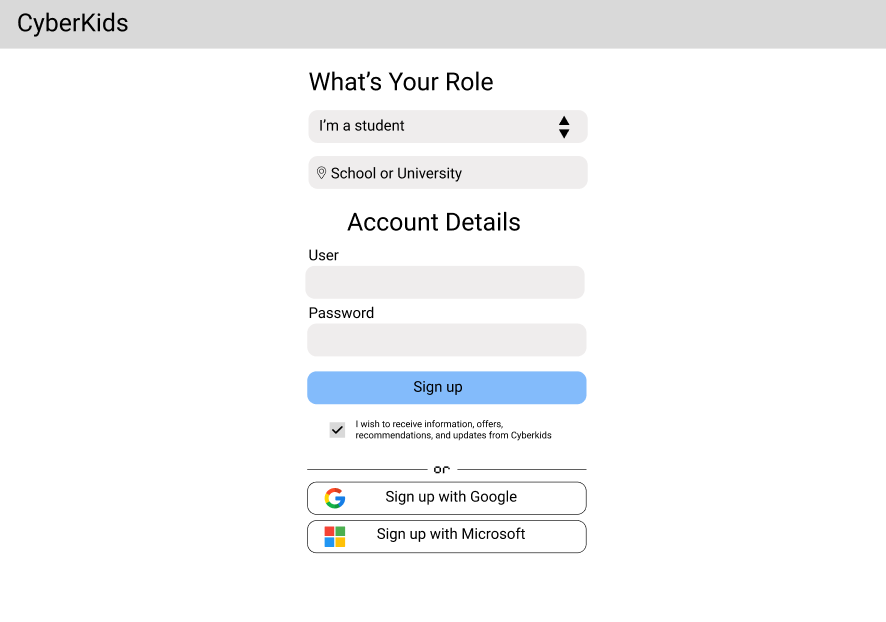
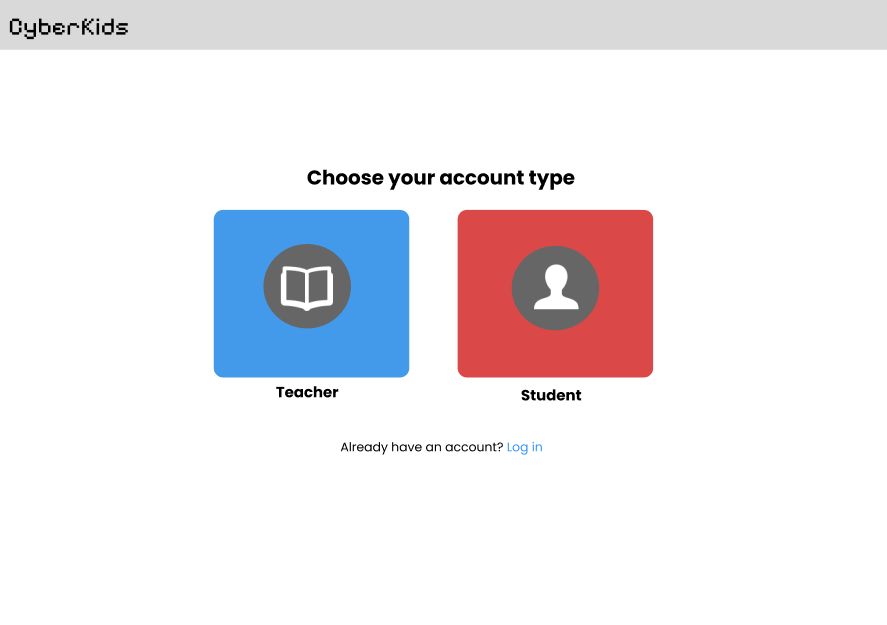
1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-003 |
| Use Case Name | Profile Management |
| Actor | Student, Teacher |
| Description | |  | | --- | |  |   This use case describes how users can view and manage their profile information. Students and teachers have limited profile management options, while administrators have extended control over user profiles. |
| Flow of Events | 1. The user navigates to the **Profile Management** page. 2. The system displays the user's profile information, including:  * Username * Score * Achievements  1. The user selects the **"Edit Profile"** option. 2. The system allows the user to update editable fields:  * Students and teachers can modify password and username.  1. The user submits the changes. 2. The system validates the updated information and saves it to the database. 3.  The system displays a confirmation message: **"Profile updated successfully."** |
| Precondition | |  | | --- | | The user must be logged into the system. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The user’s profile is successfully updated and stored in the database. |  |  | | --- | |  | |

1. *Use Case Diagram*



1. *Activity Diagram*
2. *Wireframe*



### Module 2: Profile Management

#### **2.1 View Profile**

1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-004 |
| Use Case Name | View Profile |
| Actor | Student, Teacher |
| Description | |  | | --- | |  |   This use case describes how a user (Teacher or Student) can view their profile details, including personal information and account-related data. |
| Flow of Events | 1. The user logs into the system. 2. The user navigates to the **Profile** section from the dashboard. 3. The system retrieves and displays the user’s profile details, including:  * Full Name * Email Address * Account Type (Teacher or Student)  1. Other relevant details (e.g., registration date, last login timestamp) 2. The user reviews their profile information. 3.  The user can either stay on the profile page or navigate to another section of the system. |
| Precondition | |  | | --- | | The profile data must already exist in the system. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The user successfully views their profile information. |  |  | | --- | |  | |

#### **2.2 Edit Profile**

1. *Use Case Description*

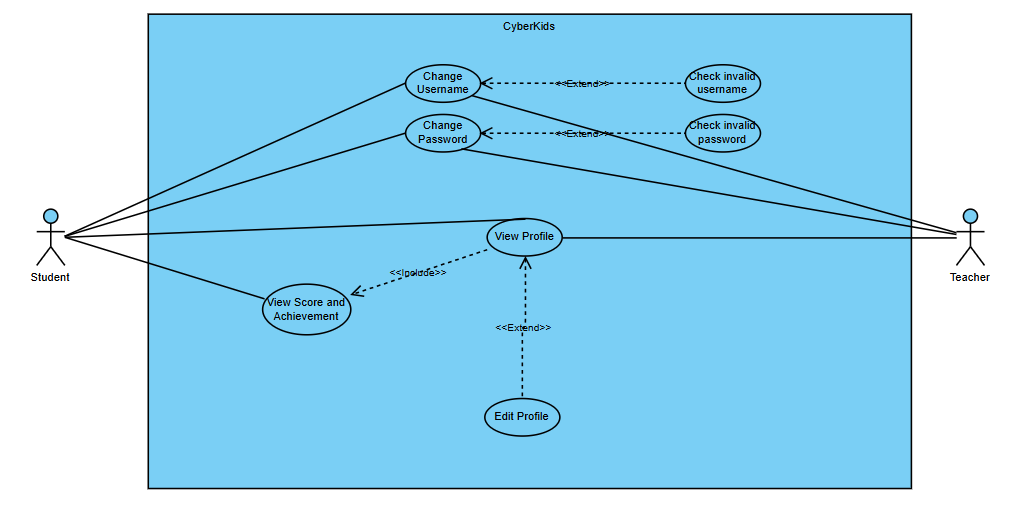
|  |  |
| --- | --- |
| Use Case ID | UC-005 |
| Use Case Name | Edit Profile |
| Actor | Student, Teacher |
| Description | |  | | --- | |  |   This use case describes how a user (Teacher or Student) can update specific profile information, such as contact details and profile picture, while ensuring restricted access to sensitive data. |
| Flow of Events | 1. The user logs into the system. 2. The user navigates to the **Profile** section from the dashboard. 3. The user selects the **Edit Profile** option. 4. The system displays editable fields, including:  * **Username** * **Password**  1. The user modifies the desired fields. 2. The user clicks the **Save Changes** button. 3. If validation passes, the system updates the profile and displays a confirmation message:  * **"Profile updated successfully."**  1.  The user is redirected to the updated profile view. |
| Precondition | |  | | --- | | The user must have a valid account (Teacher or Student). |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The user's profile is successfully updated with the new information. |  |  | | --- | |  | |

#### **2.3 View High Scores and Achievements**

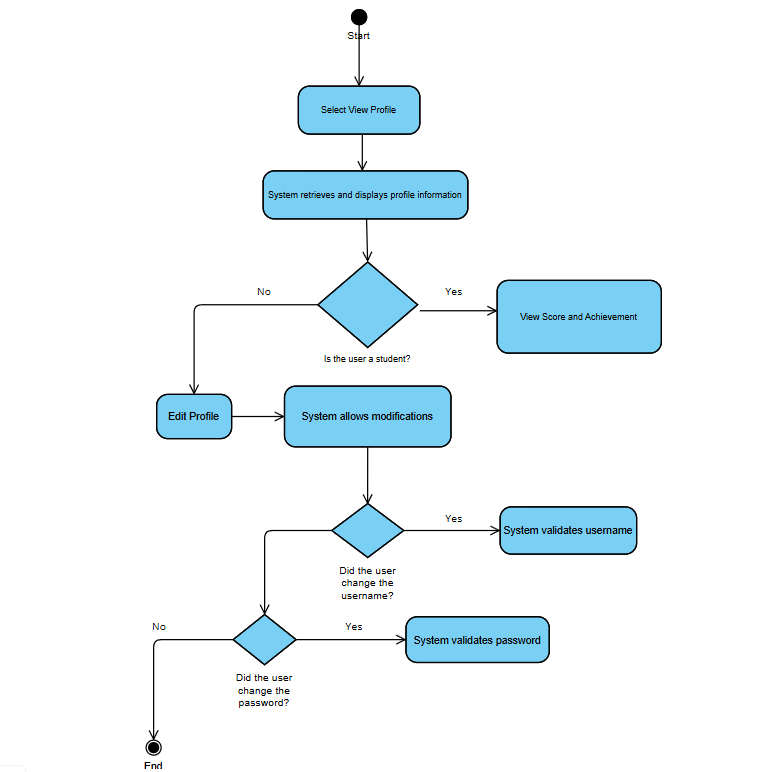
1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-006 |
| Use Case Name | View High Scores and Achievements |
| Actor | Student |
| Description | |  | | --- | |  |   This use case describes how a student can view their accumulated high scores and achievements within the system. The feature provides insights into performance, progress, and milestones achieved in various tasks, missions, or educational challenges. |
| Flow of Events | 1. The student logs into the system. 2. The student navigates to the **Profile** section from the dashboard. 3. The student selects the **High Scores & Achievements** tab. 4. The system retrieves and displays the following:  * **Highest scores achieved in different activities/games.** * **Leaderboard ranking** * **Achievements unlocked (e.g., "Top Performer," "Fast Learner," "Mission Master").** |
| Precondition | |  | | --- | | The student must have participated in at least one game or activity to generate scores. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The student successfully views their progress, including high scores and unlocked achievements. |  |  | | --- | |  | |

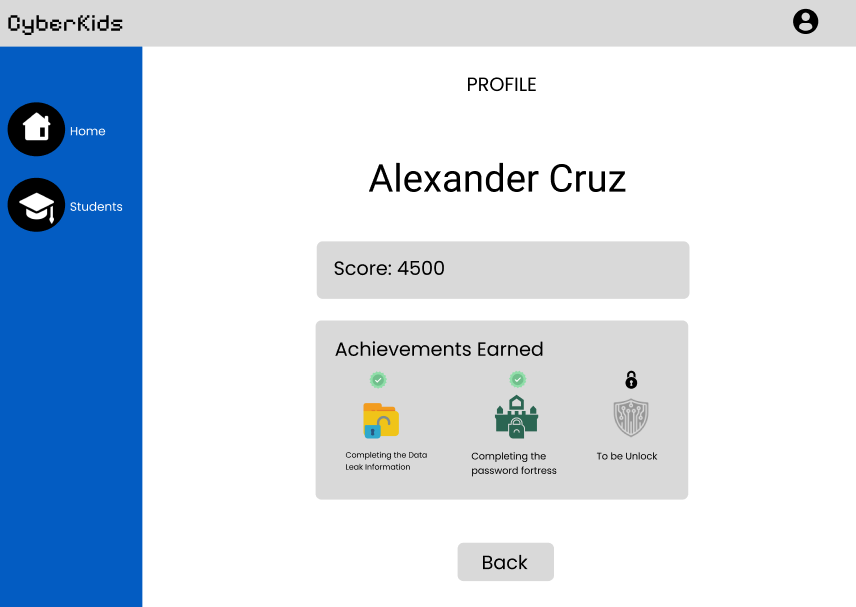
1. *Use Case Diagram*



1. *Activity Diagram*



1. *Wireframe*



### Module 3: Game Mission Module

#### **Start Game**

1. *Use Case Description*

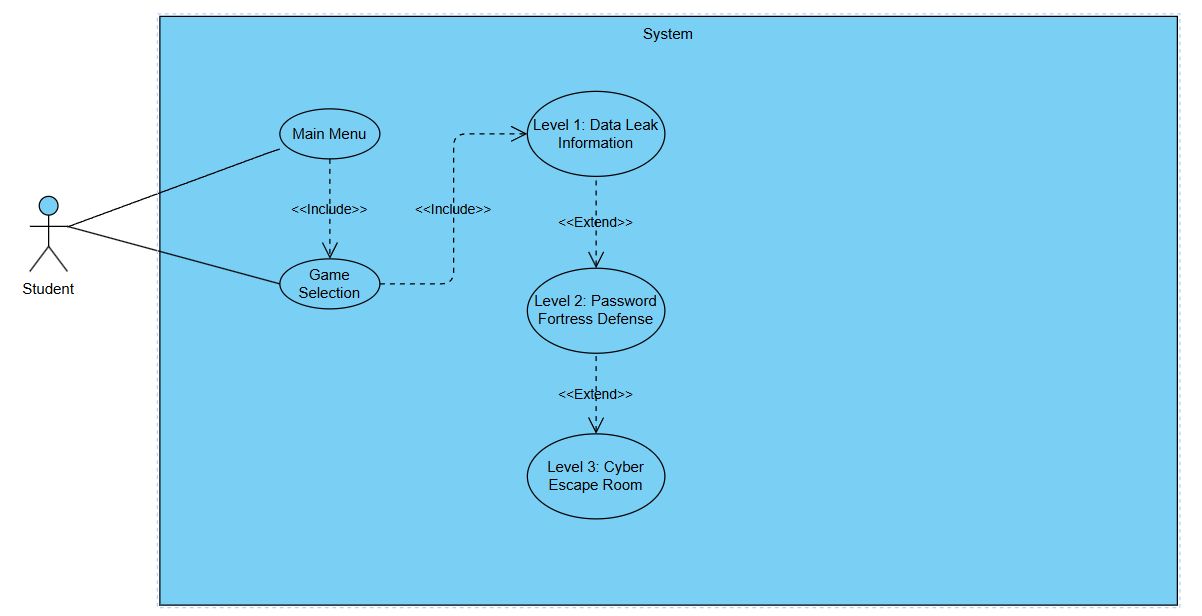
|  |  |
| --- | --- |
| Use Case ID | UC-007 |
| Use Case Name | Start Game |
| Actor | Student |
| Description | |  | | --- | |  |   This use case describes the process of a student starting a game mission within the system. The student selects a game mode or mission, and the system initializes the game environment, loads relevant resources, and begins gameplay. |
| Flow of Events | 1. The student logs into the system. 2. The student navigates to the **Game Mission** section from the dashboard. 3. The student selects a **mission or game mode** from the available options. 4. The system displays the **mission details**, including objectives, difficulty level, and instructions. 5. The student clicks the **"Start Game"** button. 6. The system loads the game environment, assets, and necessary resources. 7. The system initializes the mission and presents the first task or challenge. 8.  The game begins, allowing the student to play and complete tasks. |
| Precondition | |  | | --- | | The student must be logged into the system. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The student successfully enters the game mission and begins gameplay. |  |  | | --- | |  | |

#### **Unlock Levels**

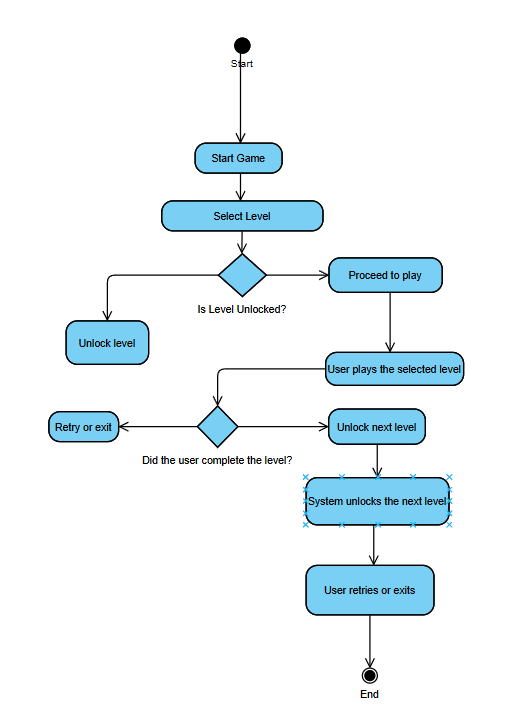
1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-008 |
| Use Case Name | Unlock Levels |
| Actor | Student |
| Description | |  | | --- | |  |   This use case describes how a student can unlock new levels in the game mission module. Levels are unlocked based on the student's performance, achievements, or completion of prerequisite missions. |
| Flow of Events | 1. The student completes a mission or achieves the required score for unlocking the next level. 2. The system evaluates the student's performance against predefined criteria (e.g., mission completion, score threshold, or achievement unlocks). 3. If the criteria are met, the system unlocks the next level. 4. The system notifies the student with a message: **"Congratulations! You have unlocked a new level."** 5.  The unlocked level is now accessible in the **Game Mission** section. |
| Precondition | |  | | --- | | The system must have the next level available for unlocking. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The next level becomes available for selection in the **Game Mission** section. |  |  | | --- | |  | |

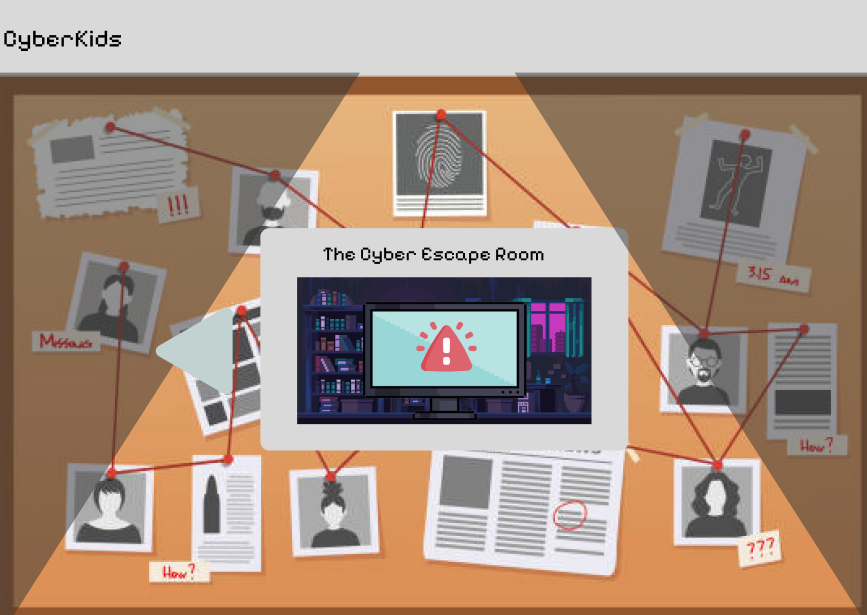
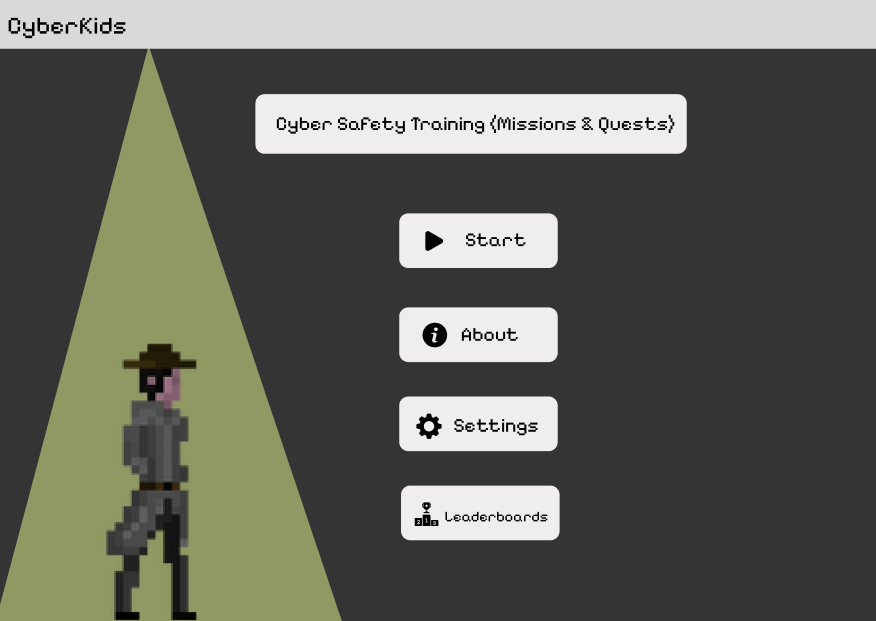
1. *Use Case Diagram*



1. *Activity Diagram*



1. *Wireframe*



### Module 4: Leaderboard & Scoring System

* 1. ***View Leaderboard (Students & Teachers)***

1. *Use Case Description*

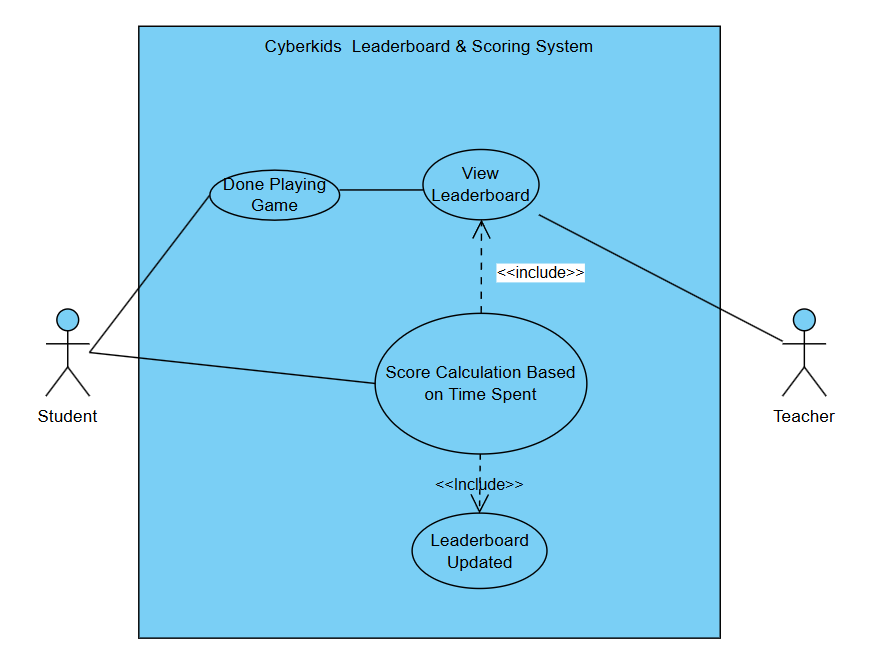
|  |  |
| --- | --- |
| Use Case ID | UC-009 |
| Use Case Name | View Leaderboard |
| Actor | Student, Teacher |
| Description | |  | | --- | |  |   This use case describes how students and teachers can view the leaderboard, which ranks students based on their scores and achievements in the game. The leaderboard provides an overview of top-performing students, encouraging competition and motivation. |
| Flow of Events | 1. The user (student or teacher) navigates to the **Leaderboard** section from the main menu. 2. The system retrieves and displays the leaderboard, ranking students based on scores, achievements, or completed missions. 3. The leaderboard can be filtered by different categories such as:  * **Top Scores** (Overall highest scores) * **Class or Section Rankings** (Filtered view based on student groups)  1. The user can scroll through the rankings to see more students. 2.  The user can exit the leaderboard and return to the main menu. |
| Precondition | |  | | --- | | The system must have recorded scores and achievements of students. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The system updates the leaderboard dynamically as new scores and achievements are recorded. |  |  | | --- | |  | |

#### **Score Calculation Based on Time Spent (Students Only)**

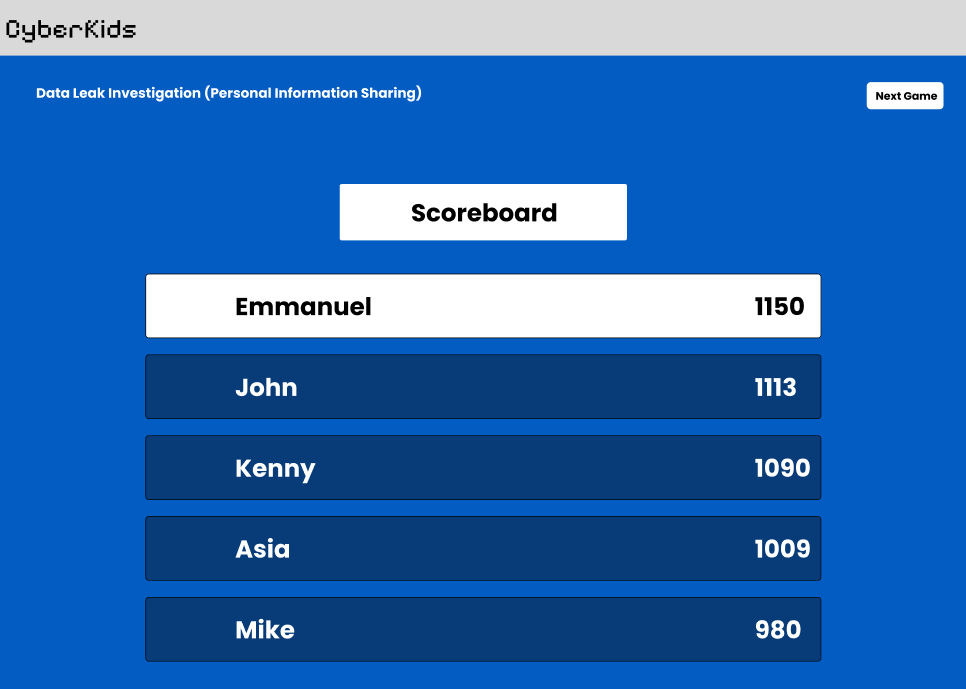
1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-010 |
| Use Case Name | Score Calculation Based on Time Spent |
| Actor | Students |
| Description | |  | | --- | |  |   This use case describes how the system calculates a student's score based on the time spent completing game missions. The scoring mechanism ensures that students are rewarded for both efficiency and persistence, balancing quick problem-solving with sustained effort. |
| Flow of Events | 1. The student starts a game mission. 2. The system tracks the total time the student spends on the mission. 3. Upon mission completion, the system calculates the score using a predefined formula that considers:  * **Time Efficiency Bonus** (Higher scores for faster completion) * **Base Score** (Points assigned for mission completion)  1. The system displays the total earned score to the student. 2. The score is recorded in the database and reflected in the leaderboard. |
| Precondition | |  | | --- | | A game mission must be active for score tracking. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The system successfully calculates and records the student's score. |  |  | | --- | |  | |

1. *Use Case Diagram*



1. *Activity Diagram*
2. *Wireframe*



### Module 5: Teacher Dashboard

***5.1 View Class Progress (Overall Performance Overview)***

1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-011 |
| Use Case Name | View Class Progress |
| Actor | Teacher |
| Description | |  | | --- | |  |   This use case describes how a teacher can view the overall progress and performance of students in their class. The dashboard provides a summarized view of student achievements, scores, and participation in game missions, allowing teachers to track learning outcomes and engagement levels effectively. |
| Flow of Events | 1. The teacher logs into the system and accesses the Teacher Dashboard. 2. The system displays an overview of the class progress, including:  * Completion rates for each assigned mission. * Leaderboard rankings within the class. * Time spent by students on game missions. |
| Precondition | |  | | --- | | The teacher must be logged into the system.  Students must have participated in at least one game mission for data to be available. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The system successfully displays the class progress overview.  The teacher can use the data for student assessment and engagement strategies. |  |  | | --- | |  | |

***5.2 View Top-Performing Students***

1. *Use Case Description*

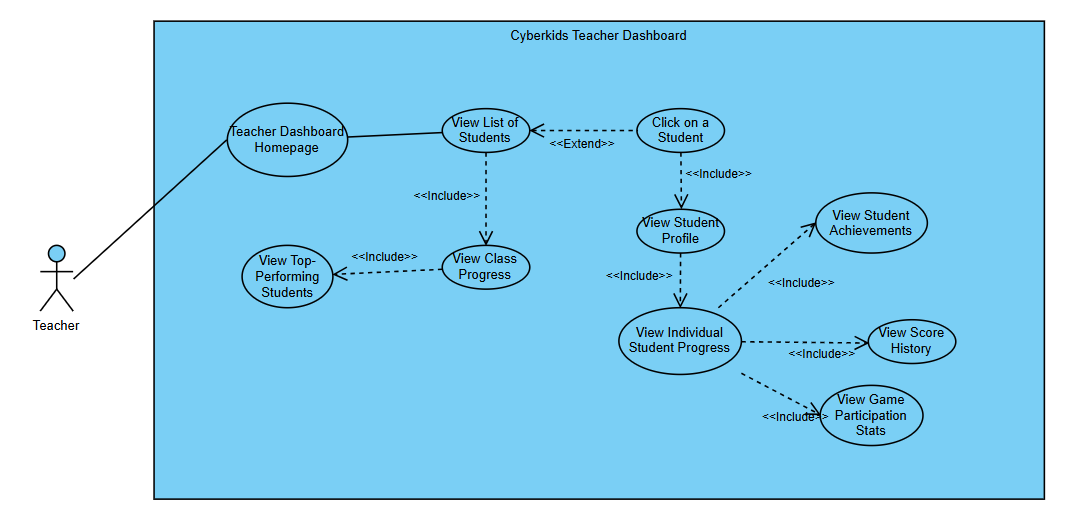
|  |  |
| --- | --- |
| Use Case ID | UC-012 |
| Use Case Name | View Top-Performing Students |
| Actor | Teacher |
| Description | |  | | --- | |  |   This use case describes how a teacher can view a list of the highest-performing students based on their game scores, mission completion rates, and overall participation. The system ranks students according to their performance metrics, allowing teachers to identify and acknowledge outstanding learners. |
| Flow of Events | 1. The teacher logs into the system and accesses the Teacher Dashboard. 2. The system provides an option to view the **Top-Performing Students** section. 3. The system displays a ranked list of students based on:  * Total game scores across missions. * Completion of challenges or difficult levels. * Time efficiency in completing tasks. |
| Precondition | |  | | --- | | Students must have completed at least one game mission for ranking data to be generated. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The system successfully displays the list of top-performing students. |  |  | | --- | |  | |

**5.3 View Individual Student Progress**

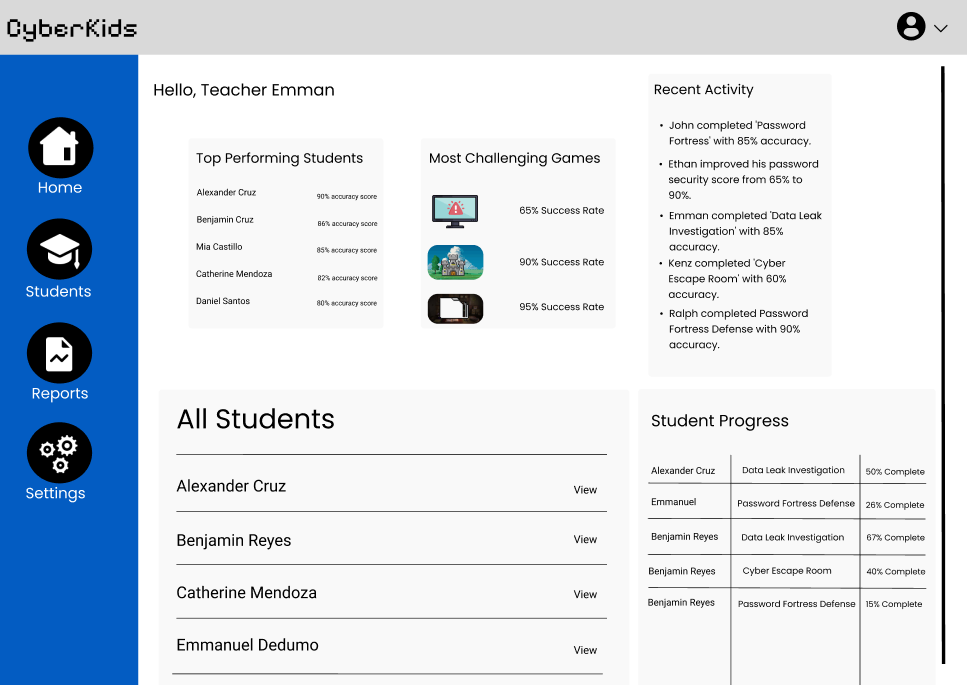
1. *Use Case Description*

|  |  |
| --- | --- |
| Use Case ID | UC-013 |
| Use Case Name | View Individual Student Progress |
| Actor | Teacher |
| Description | |  | | --- | |  |   This use case describes how a teacher can view the detailed progress of a specific student. The system provides insights into the student's performance, including scores, completed missions, time spent on tasks, and areas that need improvement. |
| Flow of Events | 1. The teacher logs into the system and accesses the **Teacher Dashboard**. 2. The teacher selects the **View Individual Student Progress** option. 3. The system displays a list of students in the teacher’s class. 4. The teacher selects a specific student to view detailed progress. 5. The system retrieves and displays:  * Student’s total score and ranking. * Completed missions and levels. * Time spent on each mission. |
| Precondition | |  | | --- | | The student must have played at least one game or completed a mission for progress data to be available. |  |  | | --- | |  | |
| Postcondition | |  | | --- | | The teacher successfully views the student’s detailed progress. |  |  | | --- | |  | |

1. *Use Case Diagram*



1. *Activity Diagram*
2. *Wireframe*



## Non-functional requirements

### Performance

##### **Game Responsiveness**

##### The system must load the main menu within 3 seconds on standard hardware.

##### Each mission should start within 5 seconds after selection.

##### **Leaderboard Updates**

##### The leaderboard should update in real-time (within 1-2 seconds) when a student completes a mission**.**

### Security

##### **User Authentication**

##### All users (students, teachers, and admin) must log in using unique credentials.

##### Teachers and admins should have role-based access control to prevent unauthorized modifications.

##### **Data Protection**

##### No real personal data will be stored—only fictional in-game profiles will be used.

##### **Access Control**

##### Students can only access their own game progress and leaderboard rankings.

##### Teachers can only access data related to their assigned students.

##### The admin has full system control but cannot alter student scores manually.

### Reliability

##### **System Availability**

##### The game should be available **99% of the time**, except for scheduled maintenance.

##### If the server goes down, **offline gameplay should still function**, but leaderboard updates will be delayed.

##### **Error Handling & Recovery**

##### If an error occurs during gameplay, the system should **recover within 5 seconds** without losing progress.

##### If the leaderboard fails to load, users should receive a **retry option** instead of a crash.