CEBU INSTITUTE OF TECHNOLOGY UNIVERSITY

COLLEGE OF COMPUTER STUDIES

Software Requirements Specifications

for

CyberKids

Change History

Version	Date	Amendment	Author
1.0	3/14/2025	Initial	Cultura
1.1	3/18/2025	Updated Modules	Baguio Cultura Dedumo Vequiso
1.2	3/21/2025	Updated Modules (relevant to the problem)	Baguio Cultura Dedumo Vequiso
2.0	5/18/2025	FInalized	Cultura

Table of Contents

1. Introduction

1.1. 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 teachers, system administrators, and developers responsible for maintaining and improving the platform.

1.2. Scope

CyberKids is an educational game designed to teach students about cybersecurity best practices through a series of engaging missions within the Roblox platform. By leveraging Roblox's immersive environment, CyberKids transforms essential cybersecurity conc epts into interactive, game-based challenges that promote active learning and critical thinking.

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. **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 multiplayer interactions.

1.3. 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.

1.4. 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

2. Overall Description

2.1. Product perspective

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

Modular Decomposition

Module 1: Gamified Cybersecurity Game Levels

This module delivers interactive, educational gameplay designed to teach students key cybersecurity concepts through immersive challenges.

- Gamified Online Privacy and Safety
  Players analyze digital case files and sort information into “Safe to Share” vs. “Not Safe to Share” categories, reinforcing critical thinking about data privacy.

- Gamified Password Security
  Players collect password fragments from treasure chests and assemble strong passwords. A simulated hacker bot attempts to crack weak passwords, encouraging players to build robust defenses.

- Gamified Phishing and Scam Awareness
  Players navigate a virtual escape room, identifying phishing emails, fake login pages, and scam messages using social engineering tactics. Successful identification unlocks security codes to escape the room.

Module 2: Student Real-Name Integration

Enables students to input their real names within the Roblox game, linking their in-game activities to identifiable profiles accessible by teachers for clearer monitoring and personalized support.

- Allows students to register and update their real names directly inside the game interface
- Replaces anonymous Roblox usernames with real names in the teacher dashboard for better student identification
- Ensures data consistency between gameplay activity and student records
- Supports privacy and data security by restricting access to real names to authorized teachers only

Module 3: Teacher Web Dashboard

Equips teachers with comprehensive tools to monitor student engagement and performance, facilitating data-driven instruction and support.

- Teachers can lock access to specific game modules
- If a student needs remediation or challenge, teachers can reassign them to replay or skip particular levels.
- View detailed student progress reports and analytics
- Enables teachers to export reports in PDF or CSV format for progress tracking.

Module 4: Notification System

Enhances teacher responsiveness by providing real-time alerts and updates on student activities and system events through a notification interface.

- Automatically generate notifications based on key student actions such as challenge completion.
- Display unread and read notifications for easy tracking of important events

2.2. 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.

2. 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.

3. 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 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 desktop and laptop computers that meet the minimum specifications required by Roblox, the platform hosting the game. To ensure a smooth and enjoyable gaming experience, users should have devices with decent hardware capabilities as outlined below:

- **Minimum System Requirements:**
  The game requires a computer with at least Windows 7, macOS 10.11, or equivalent operating system. A processor with a minimum of 1.6 GHz, 1 GB of RAM, and a dedicated graphics card supporting DirectX 9 or OpenGL 2.1 is necessary to run the game without lag.

- **Recommended System Specifications:**
  For optimal performance, especially with multiple game modules and smooth rendering, devices with a dual-core processor (2.5 GHz or higher), 4 GB of RAM or more, and a modern GPU supporting DirectX 10 or higher are recommended.

- **Resource Usage Optimization:**
  The system is optimized to balance visual quality and performance, aiming to prevent excessive CPU and memory consumption. This ensures that even mid-range hardware can provide a seamless gameplay experience.

- **Justification:**
  These hardware requirements align with Roblox's official recommendations and ensure that students at CIT-University can access and enjoy the cybersecurity game modules without performance bottlenecks or crashes. Ensuring minimum specs helps avoid frustration and supports uninterrupted learning.

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:

- Collection of Minimal Personal Data – Students provide only their real names (no other PII such as birthdates or addresses). These names are used solely for teacher identification and are stored securely in the Azure MySQL database.
- Data Privacy Compliance – Before collecting any real-name information, the student's

faculty adviser and supervising professor will sign a data privacy consent letter, outlining the purpose, storage, and retention policies for the names collected.

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 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.

3. Specific Requirements

3.1. 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, scalability, and maintainability, including operating systems, database management systems, development frameworks, and cloud deployment services.

- **Operating System Compatibility**
  The software will be compatible with the following operating systems:

  - Windows 10 and later – Officially supported and thoroughly tested for stability and security.

- **Database Management System (DBMS)**
  The system will store all user progress, scores, and logs in a relational database with cloud-based hosting for high availability and reliability. Supported databases include:

  - MySQL 8.0 or later – Primary relational database for structured data storage.
  - PostgreSQL 13 or later – Alternative database option offering enhanced scalability and robustness.
  - Microsoft Azure SQL Database – Cloud-hosted database solution providing managed services, automated backups, and scalability.

- **Frameworks and Development Tools**
  The system leverages modern development frameworks and tools for efficient development, scalability, and maintainability:

  - Game Engine: Roblox – The primary game engine hosting the cybersecurity modules, deployed on Roblox servers.

- Backend: Java Spring Boot – Handles API requests, business logic, and integrates with the database layer.
- Frontend: React.js – For building the user interface components of student and teacher dashboards. Additional frontend frameworks include:
  - Shadcn UI – For accessible and customizable UI components.
  - Tailwind CSS – Utility-first CSS framework used to rapidly build and style responsive layouts.
- Data Management: Hibernate ORM – Manages database interactions through object-relational mapping, ensuring data consistency and integrity.

- **APIs**

  Communication between the frontend and backend services is facilitated through RESTful APIs, enabling modularity and separation of concerns.

- **Containerization and Deployment**

  - Docker – Used to containerize the backend and database services, ensuring consistent environments across development, testing, and production.
  - Render – The backend services are deployed on Render, providing scalable and reliable cloud hosting.
  - Vercel – The frontend React application is deployed on Vercel, enabling fast global content delivery and seamless continuous deployment pipelines.

- **Cloud Infrastructure**

  The system utilizes Microsoft Azure for hosting the relational database, leveraging Azure's cloud infrastructure for high availability, security, and automatic scaling.

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

An active internet connection is essential for:

- Live updates of game content and user progress
- Leaderboard synchronization and ranking updates
- Cloud-based data storage and retrieval
- Real-time notifications and event handling within dashboards

Protocols and Technologies Used

- **HTTP/HTTPS**
  Standard protocols for secure communication between clients (web dashboards and game clients) and backend servers, ensuring encrypted data transfer.
- **RESTful APIs**
  The backend exposes RESTful endpoints that handle CRUD operations, user authentication, game progress updates, and leaderboard data management.
- **WebSocket Protocol**
  Enables persistent, full-duplex communication channels between clients and servers, facilitating real-time features such as:
  - Instant leaderboard updates
  - Real-time notifications
  - Live event handling within the teacher and student dashboards

- **Axios**

  A promise-based HTTP client used in the frontend React applications for making asynchronous API requests to the backend, including fetching user data, submitting progress, and managing profile settings.

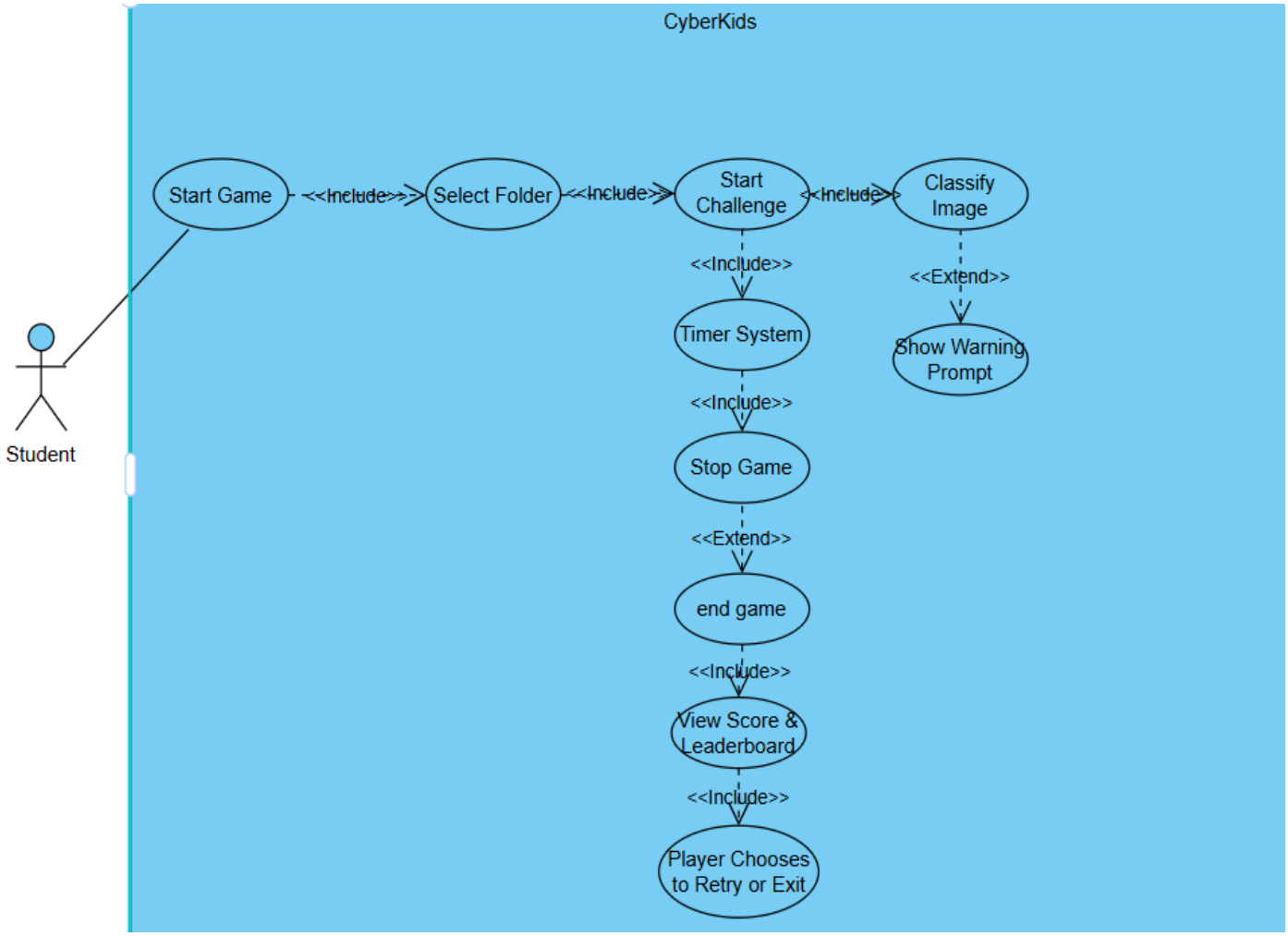
- **Routing and Navigation**
  - React Router handles client-side routing within the web dashboards, enabling smooth navigation between different views such as student progress pages, teacher analytics, and notification panels.

3.2. Functional requirements

Module 1: Gamified Cybersecurity Game Learning Module

Transaction 1.1 Information Classification Sorting Challenge

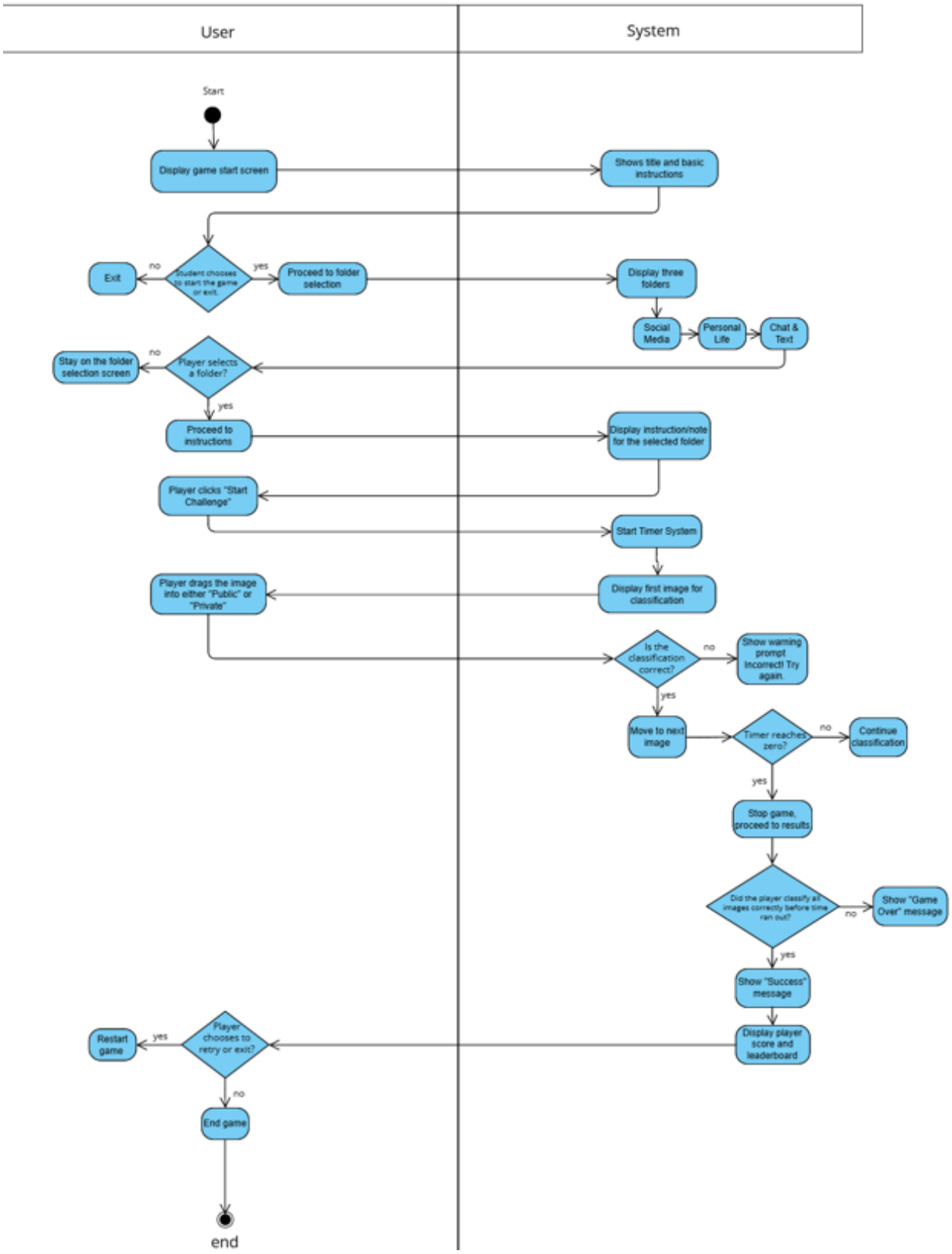
Use Case Diagram *(Information Classification Sorting Challenge)*



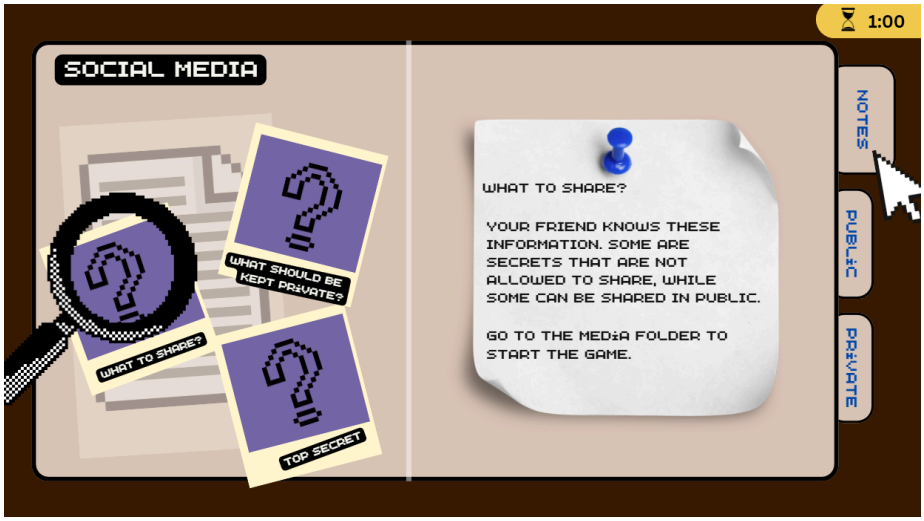
Use Case Description

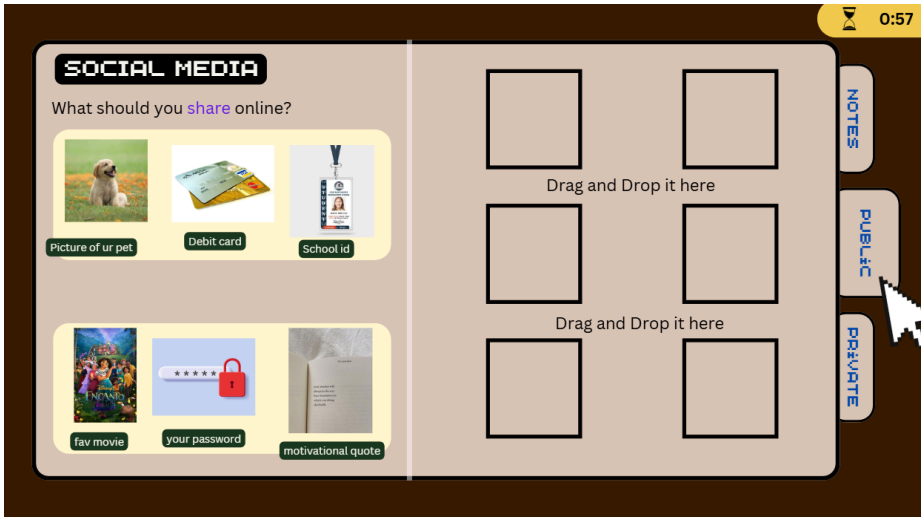
Use Case ID	UC-001
Use Case Name	Information Classification Sorting Challenge
Actor	Student
Description	The Information Classification Sorting Challenge is an interactive game where students categorize pieces of information into Personal (Safe to Share) and Private (Not Safe to Share). This challenge aims to educate students on responsible information sharing and online safety by reinforcing their ability to distinguish between different types of information.
Flow of Events	1. The student starts the challenge, and a set of information items appears on the screen. 2. The student drags and drops each item into either the Personal (Safe to Share) or Private (Not Safe to Share) category. 3. The system validates each selection and provides instant feedback (correct/incorrect). 4. If the answer is incorrect, a brief explanation appears to help the student understand why the information is categorized as such. 5. The challenge continues until all items are sorted or time runs out (if the Timer System is active).
Precondition	The student is logged into the system.
Postcondition	The student's score is recorded and displayed.

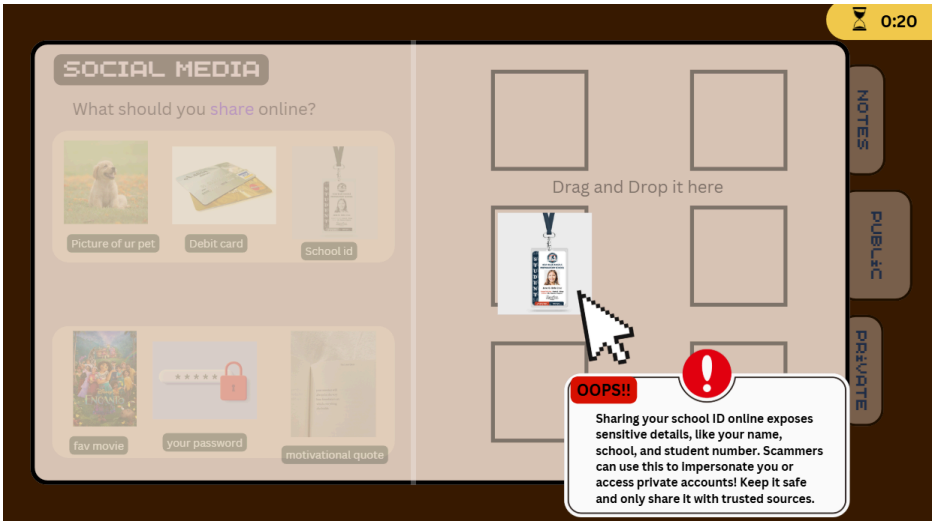
Activity Diagram *(Information Classification Sorting Challenge)*



Wireframe *(Information Classification Sorting Challenge)*

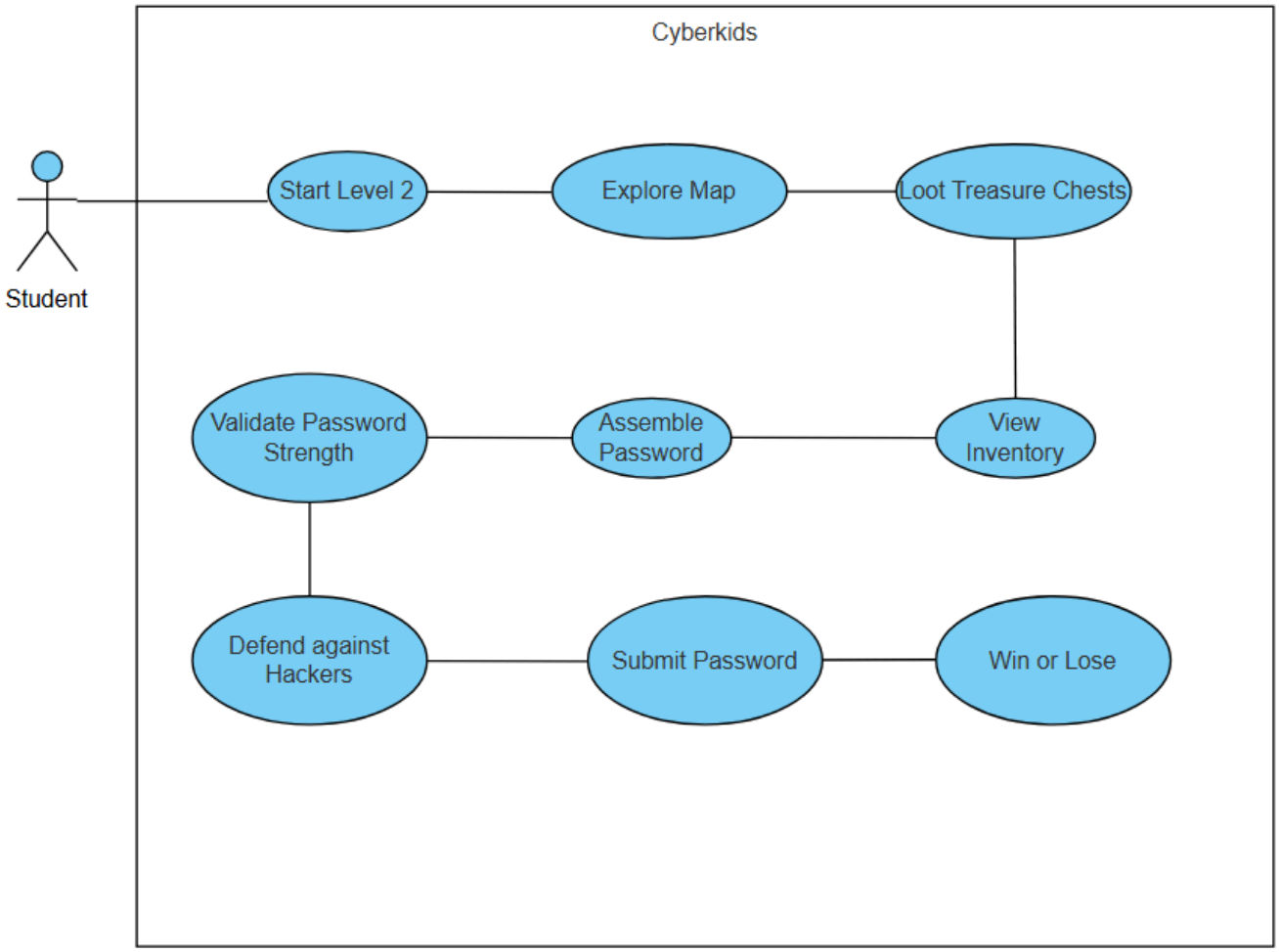






Transaction 1.2 Password Security Challenge

***Use Case Diagram** (Password Security Challenge)*

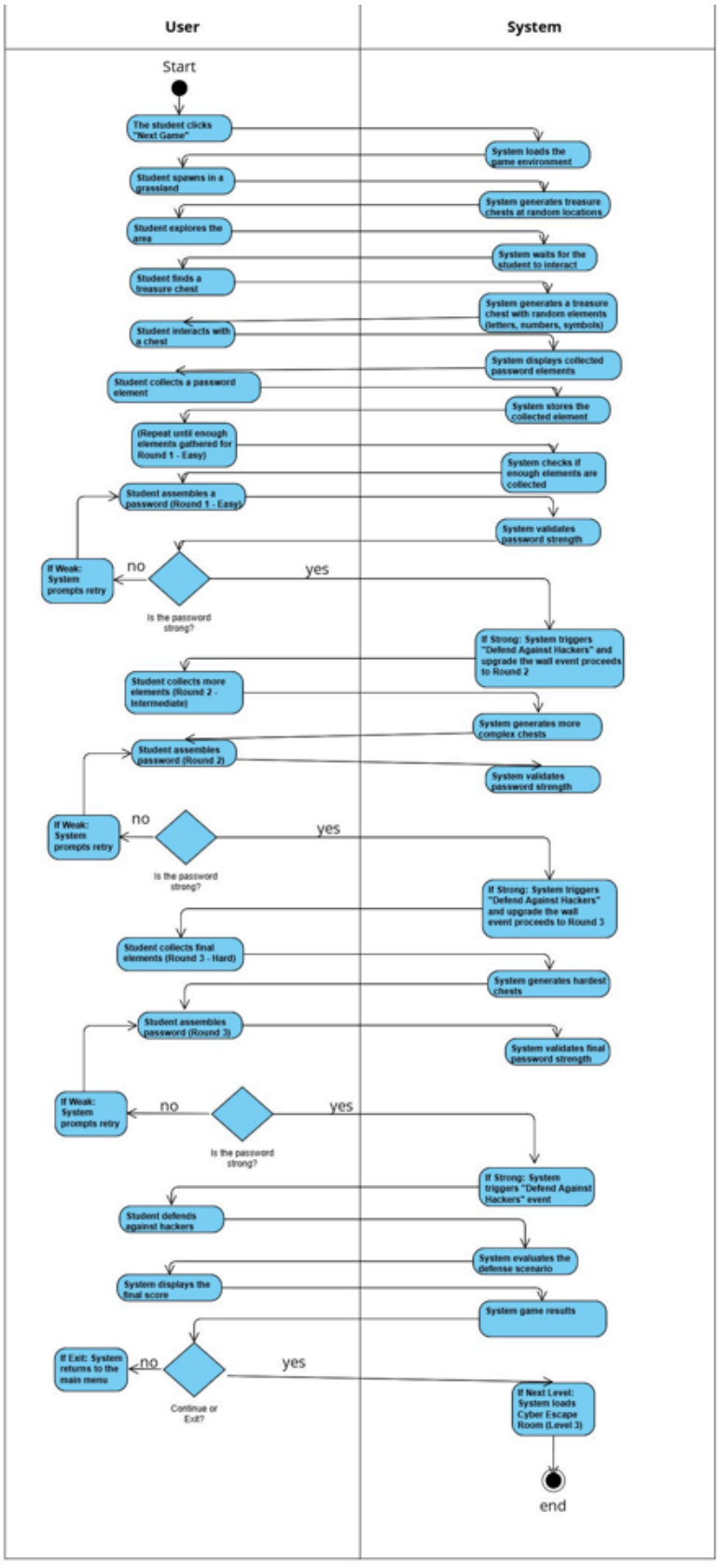


Use Case Description

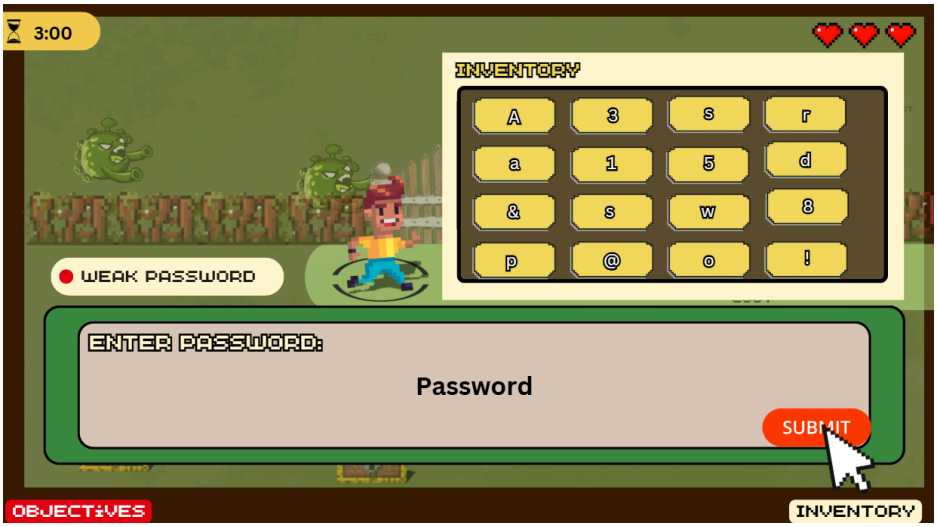
Use Case ID	UC-002
Use Case Name	Password Security Challenge
Actor	Student
Description	The Password Security Game is a gamified learning experience where players must collect password fragments from treasure chests, analyze their strength, and assemble the most secure password before reaching the final gate. If a password is too weak, simulated hacker bots will attack, requiring players to reinforce their defenses before the Cyber HQ is breached.
Flow of Events	1. The player enters the game area and receives an initial weak password. 2. The player explores the environment, searching for treasure chests that contain password fragments (letters, numbers, and symbols). 3. The player collects and reviews the password fragments in their inventory. 4. The player assembles a new password by selecting the strongest fragments. 5. If the password is weak, hacker bots attack, and the player must find stronger fragments. 6. The player proceeds to the security gate and submits the password. 7. If the password is strong, the player successfully secures the gate and wins the challenge. 8. If the password is still weak, the player receives feedback on how to improve it and must try again.

Precondition	The player has logged into the game and started the Password Security mission.
Postcondition	● If the player creates a strong password, the mission is marked as complete. ● If the player fails to create a strong password, they receive feedback and must retry. ● The system records the player's progress and updates their score and leaderboard position.

***Activity Diagram** (Password Security Challenge)*

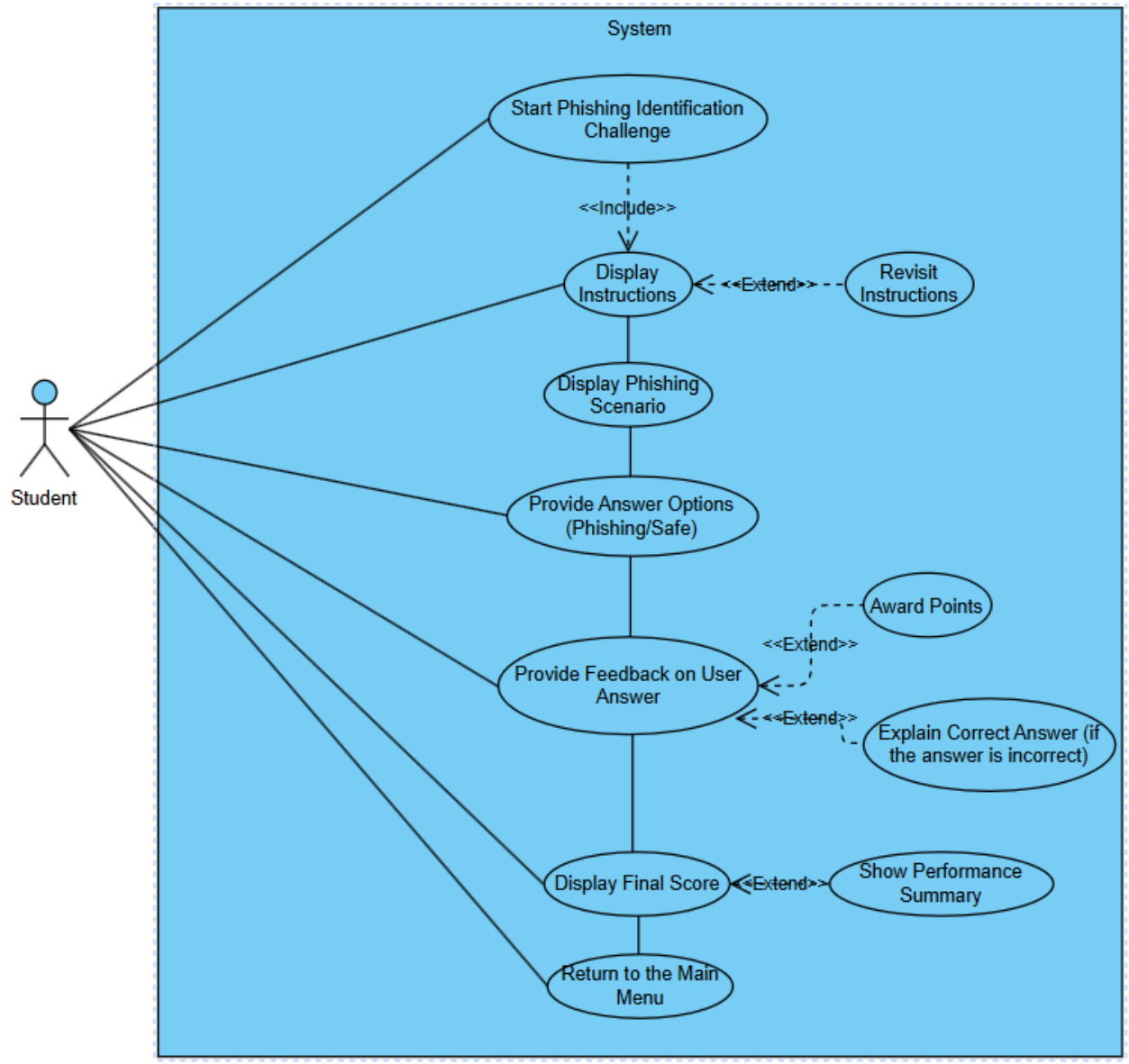


***Wireframe** (Password Security Challenge)*



Transaction 1.3 Phishing Identification Challenge

***Use Case Diagram** (Phishing Identification Challenge)*

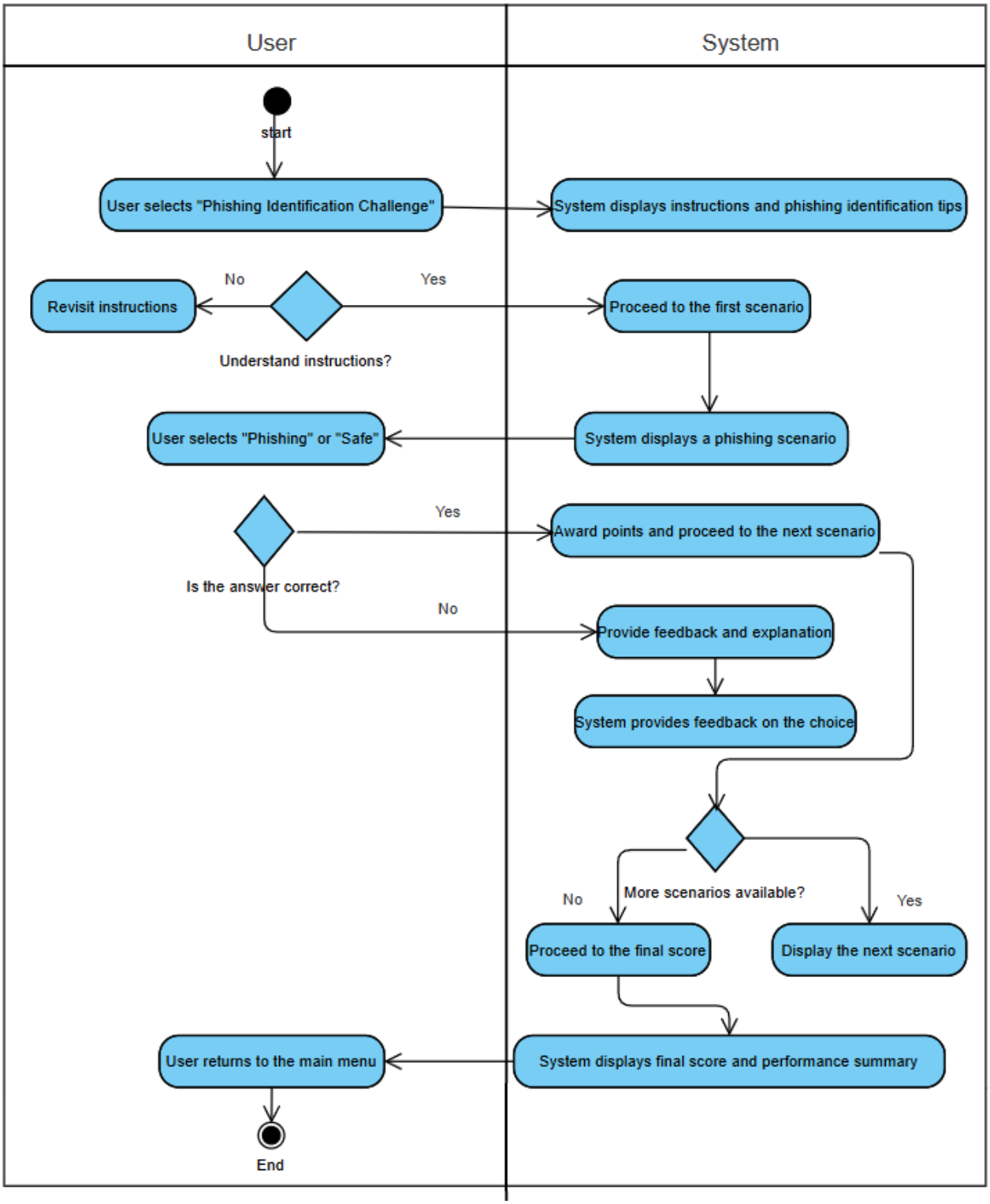


Use Case Description

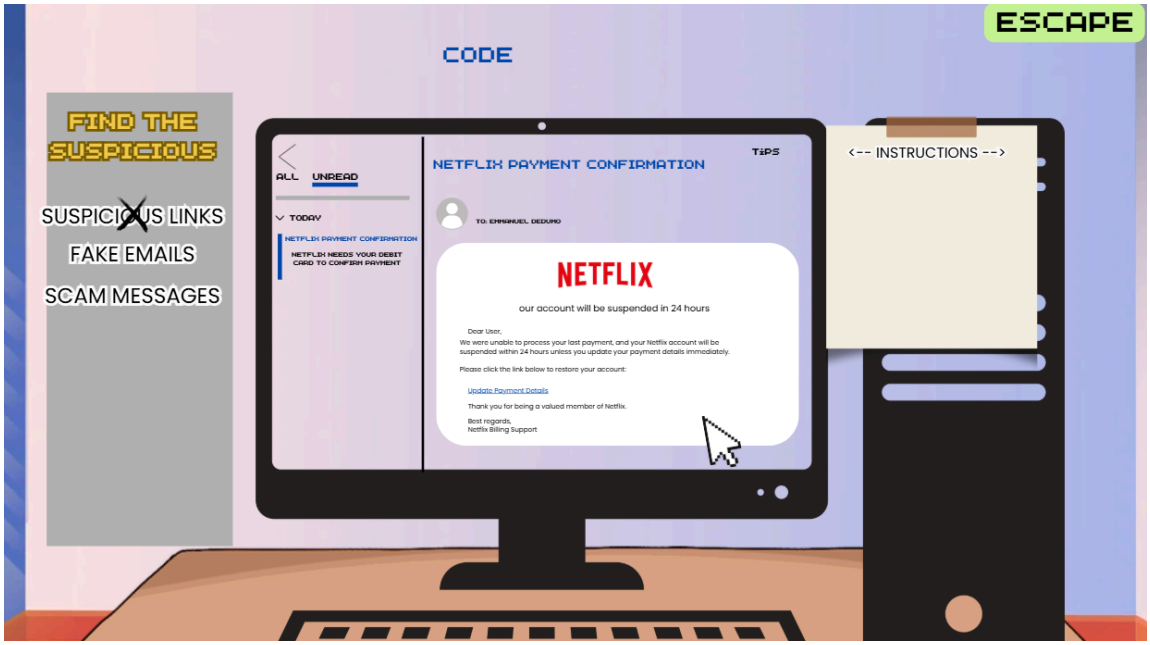
Use Case ID	UC-003
Use Case Name	Phishing Identification Challenge
Actor	Student
Description	The Phishing Identification Challenge is an interactive mystery-solving game where students navigate a simulated PC environment to identify phishing scams, fake links, and malicious messages. Players must analyze emails, websites, and messages to detect deceptive content and successfully unlock their escape from the simulation.
Flow of Events	1. The student starts the challenge and enters the simulated PC environment. 2. The system presents a series of emails, websites, and messages containing both legitimate and phishing elements. 3. The student inspects the content, looking for red flags, such as: ● Suspicious links or URLs. ● Urgent and threatening language. ● Unusual sender addresses. ● Requests for personal or financial information. 4. The student flags phishing attempts by clicking on them or marking them as fraudulent. 5. The system provides immediate feedback on whether the selection was correct or incorrect. 6. The student continues analyzing additional cases until the challenge is completed.

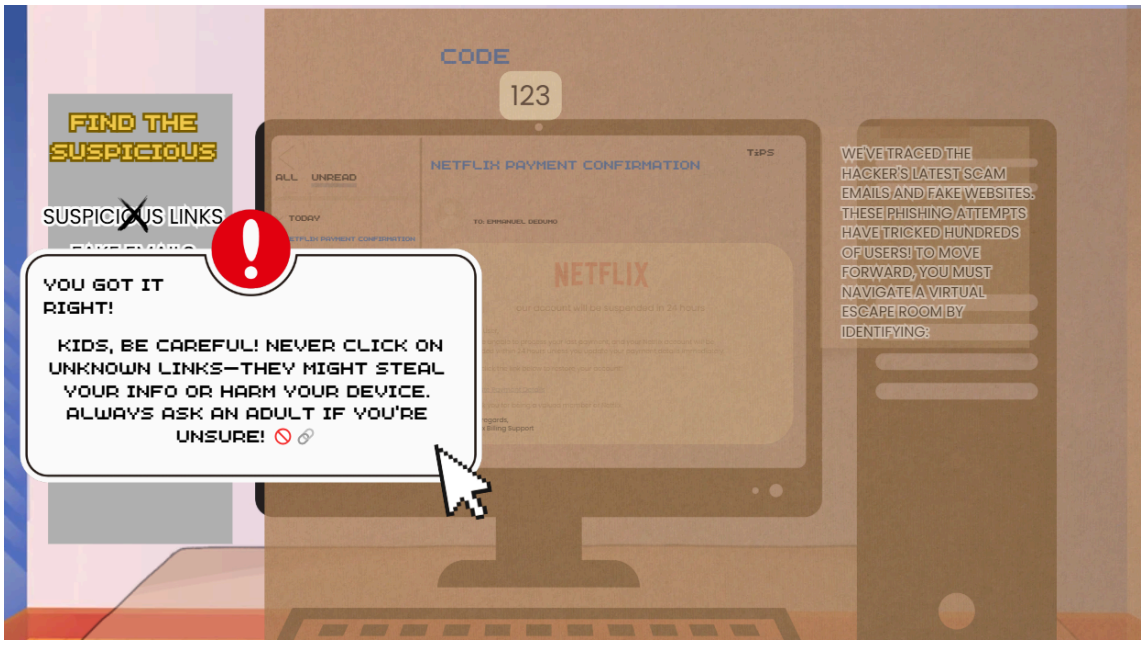
	7. If the student correctly identifies all phishing scams, they successfully "escape" the simulated scenario. 8. The system records the student's performance and updates the leaderboard accordingly.
Precondition	● The student must be logged into the game. ● The Phishing Identification Challenge level must be accessible.
Postcondition	● The student's score and completion status are saved. ● The leaderboard is updated with the student's performance.

***Activity Diagram** (Phishing Identification Challenge)*



Wireframe *(Phishing Identification Challenge)*

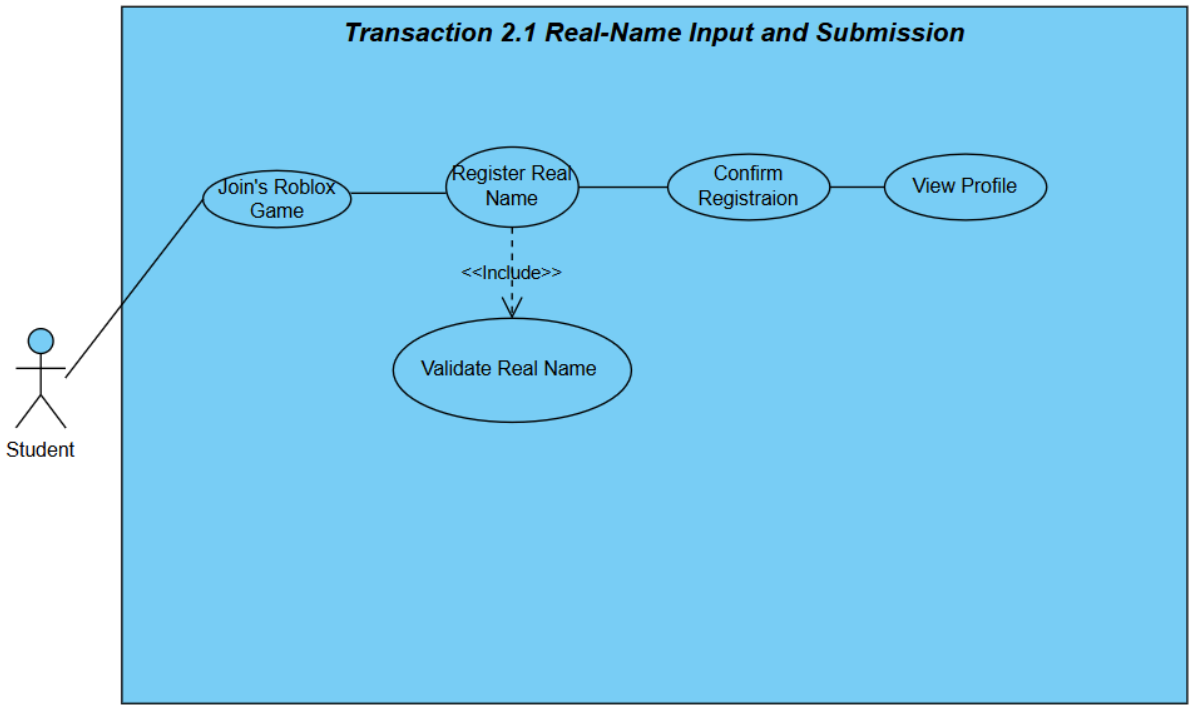




Module 2: Student Real-Name Integration

Transaction 2.1 Real-Name Input and Submission

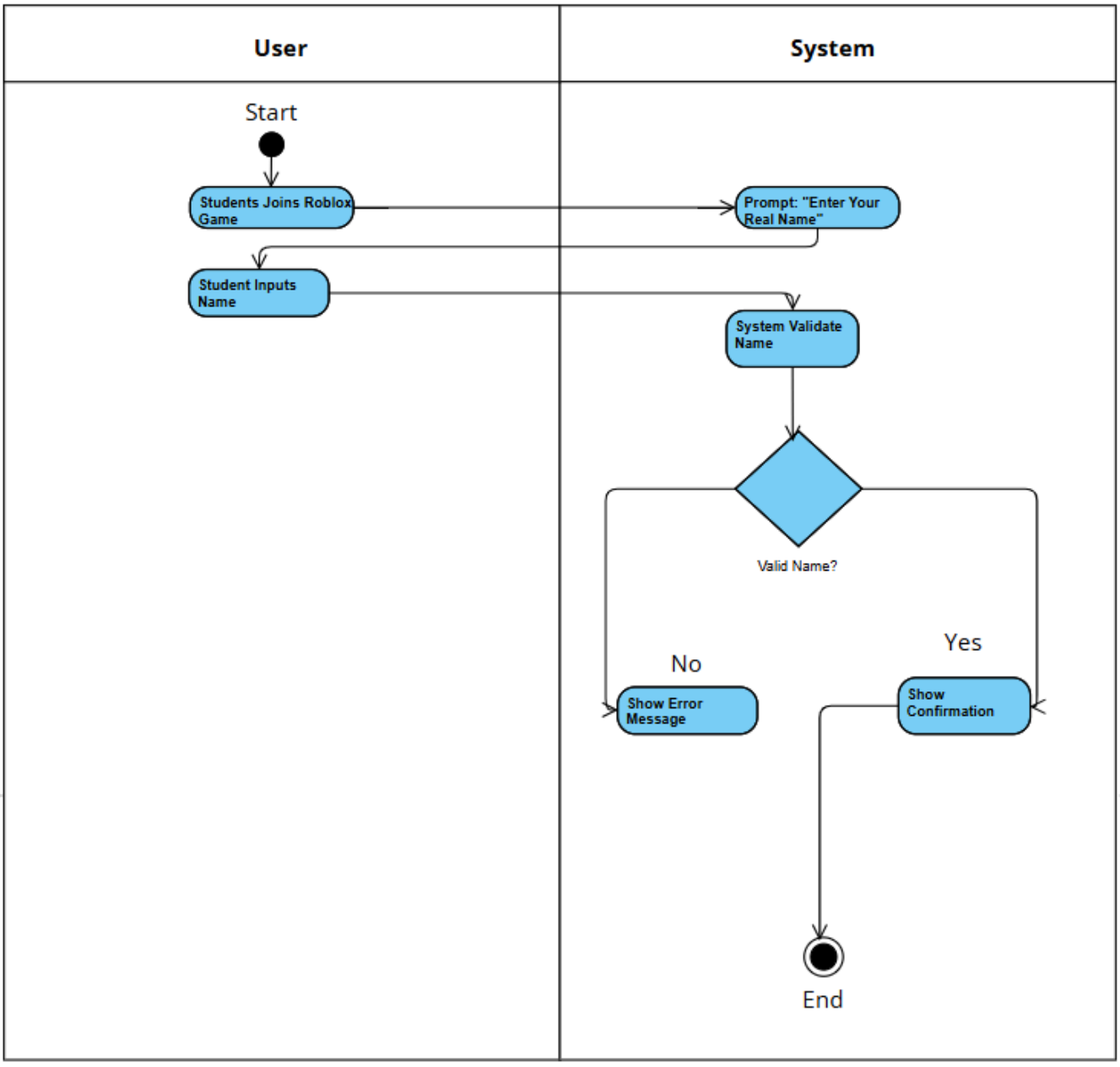
Use Case Diagram



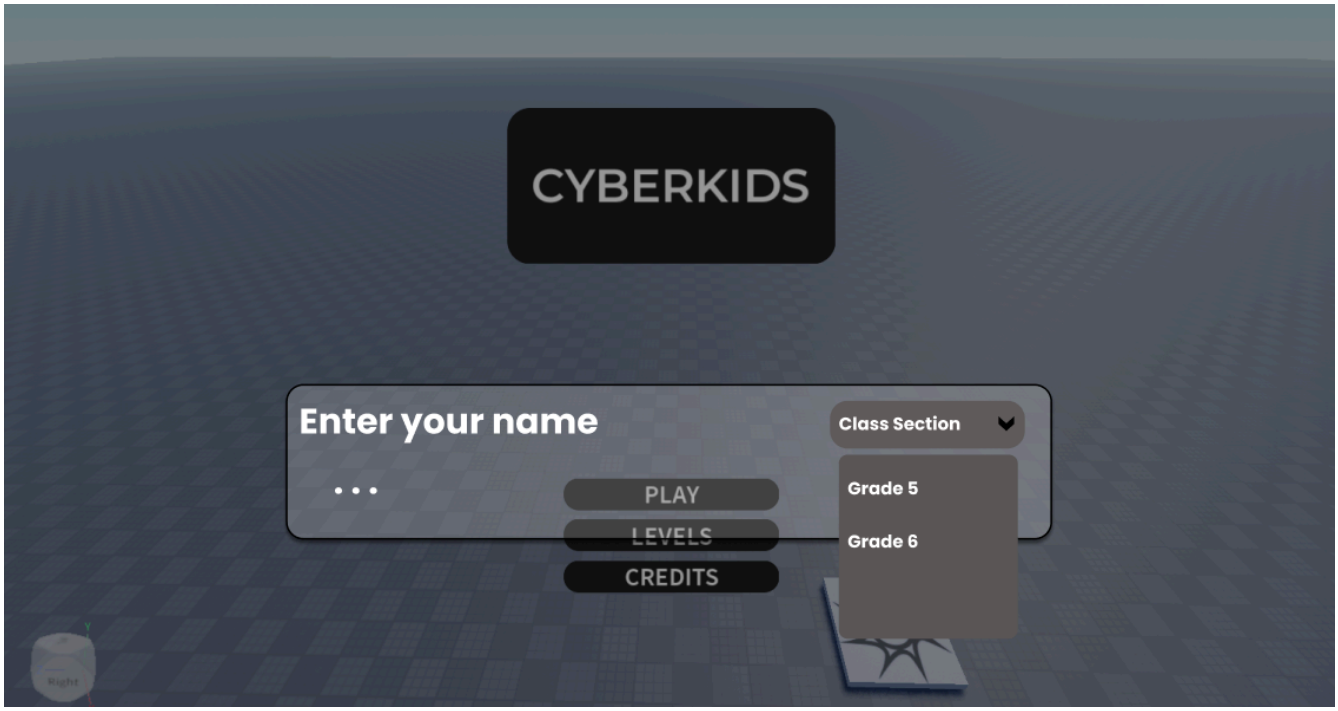
Use Case Description

Use Case ID	UC-001
Use Case Name	Real-Name Input and Submission
Actor	Student
Description	This use case allows students to input and submit their real names upon joining the Roblox game. This real name is used for identifying students in reports, tracking progress, and displaying their achievements.
Flow of Events	1. Student joins the Roblox game. 2. The system prompts the student to enter their real name. 3. Student inputs their real name. 4. System validates the name: ● Name is not empty. ● Only alphabetic characters and spaces are allowed. ● Length does not exceed a defined limit (e.g., 30 characters). 5. If validation passes, system saves the name to the backend database. 6. System displays a confirmation message: “Your name has been registered.” 7. Use case ends.
Precondition	The student has successfully joined the Roblox game.
Postcondition	A confirmation message is shown to the student.

Activity Diagram



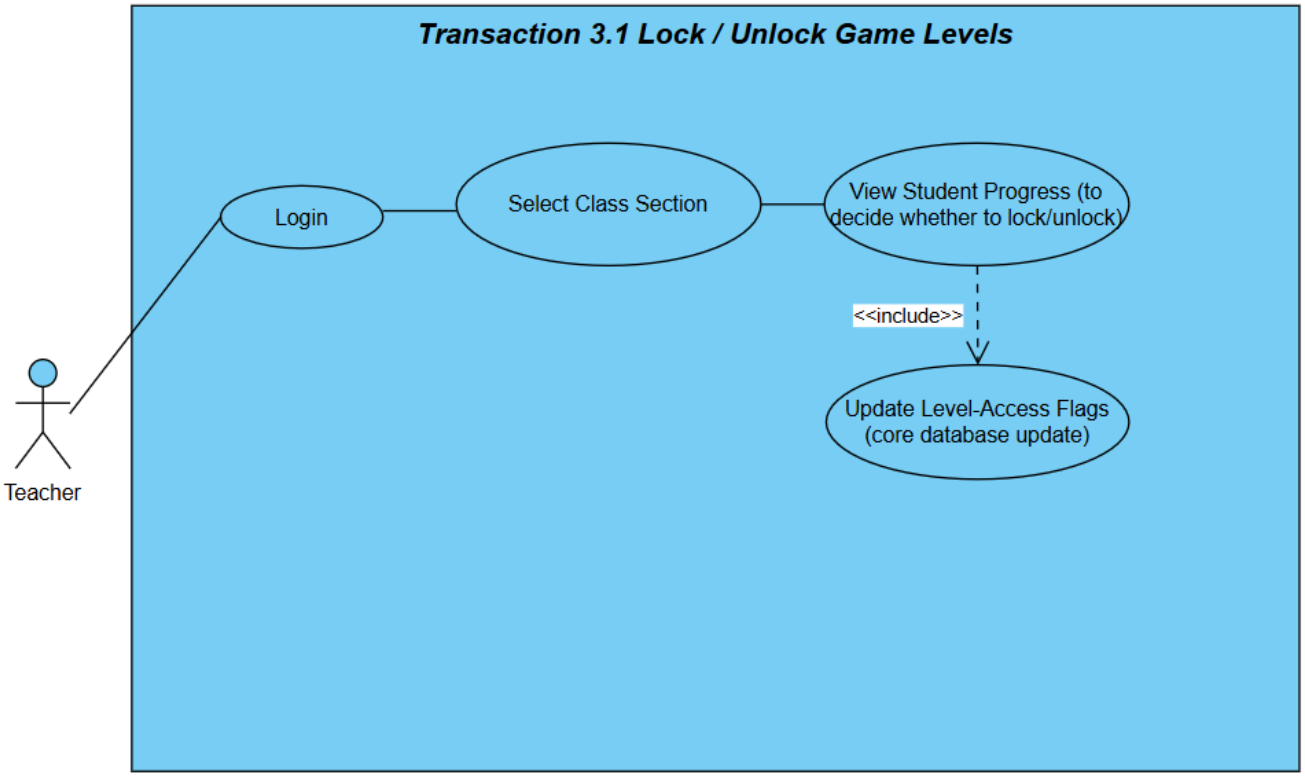
Wireframe


CYBERKIDS
Enter your name
Class Section
Grade 5
Grade 6
PLAY
LEVELS
CREDITS

Module 3: Teachers Dashboard

Transaction 3.1 Lock / Unlock Game Levels

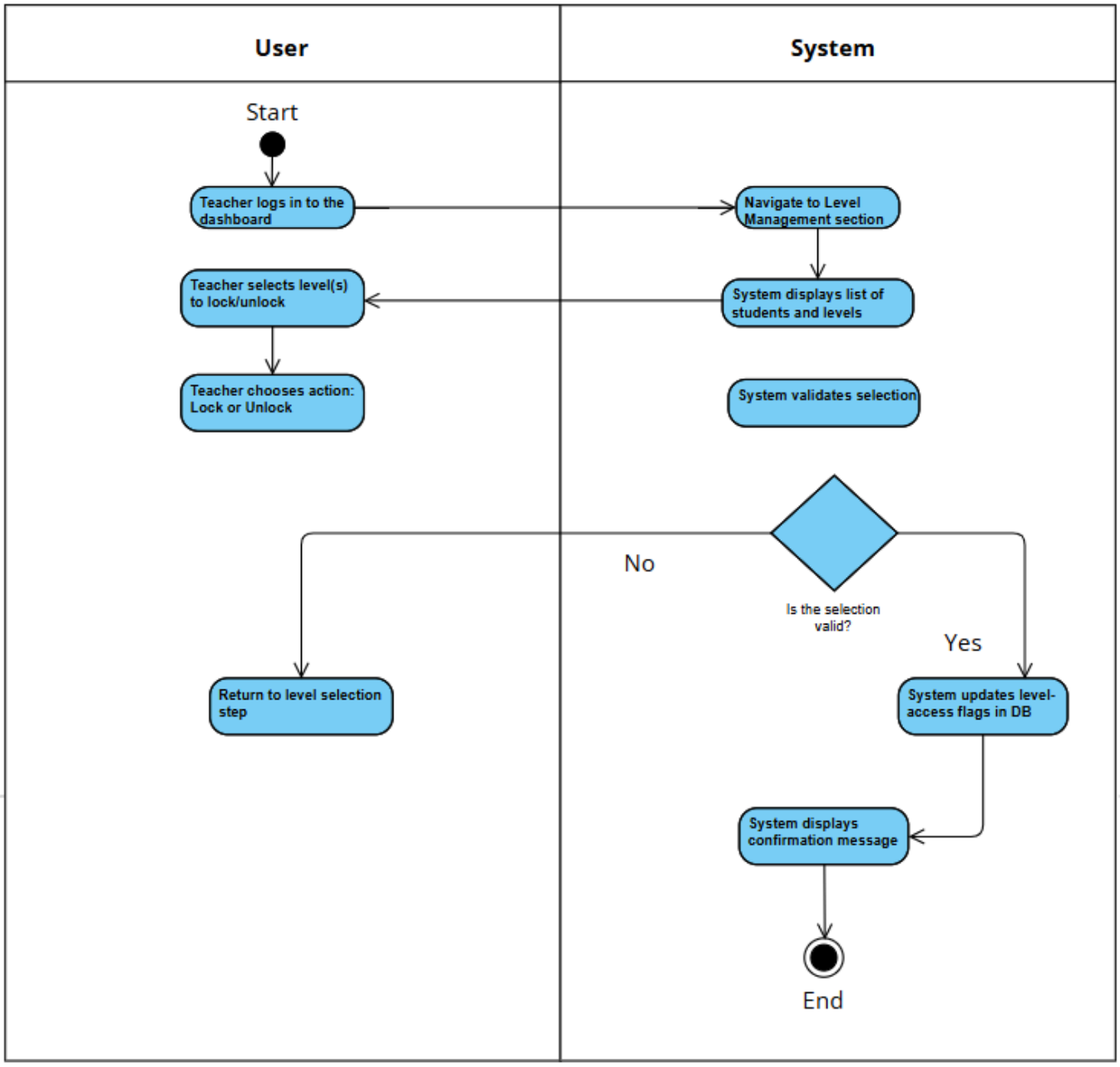
Use Case Diagram



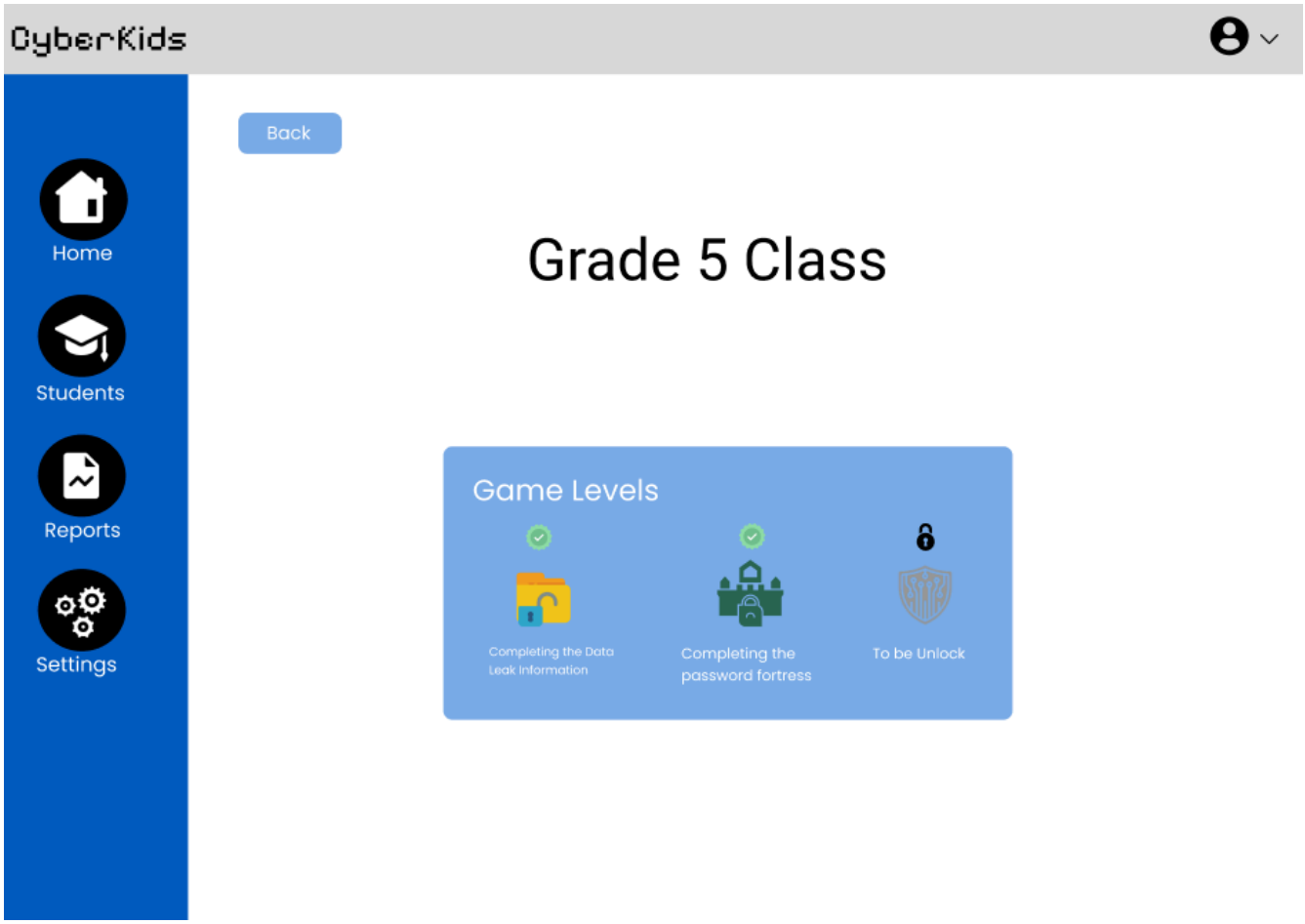
Use Case Description

Use Case ID	UC-001
Use Case Name	Lock / Unlock Game Levels
Actor	Teacher
Description	This use case allows teachers to control access to different game levels via the dashboard. Teachers can lock a level to prevent access until prerequisites are met (e.g., Level 2 must be completed before Level 3) or unlock a level manually for individual or all students (e.g., for testing or support).
Flow of Events	1. Teacher logs in to the dashboard. 2. Teacher navigates to the “Level Management” section. 3. System displays list of students and available game levels. 4. Teacher selects one or more students. 5. Teacher selects the level(s) to lock or unlock. 6. Teacher chooses an action: Lock or Unlock. 7. System updates the level-access flag(s) for each selected student in the backend database. 8. System confirms the changes and displays a success message: ● “Level 3 locked for 10 students.” ● “Level 2 unlocked for John Doe.”
Precondition	● The teacher is authenticated and logged in to the dashboard.
Postcondition	● A confirmation is displayed to the teacher.

Activity Diagram

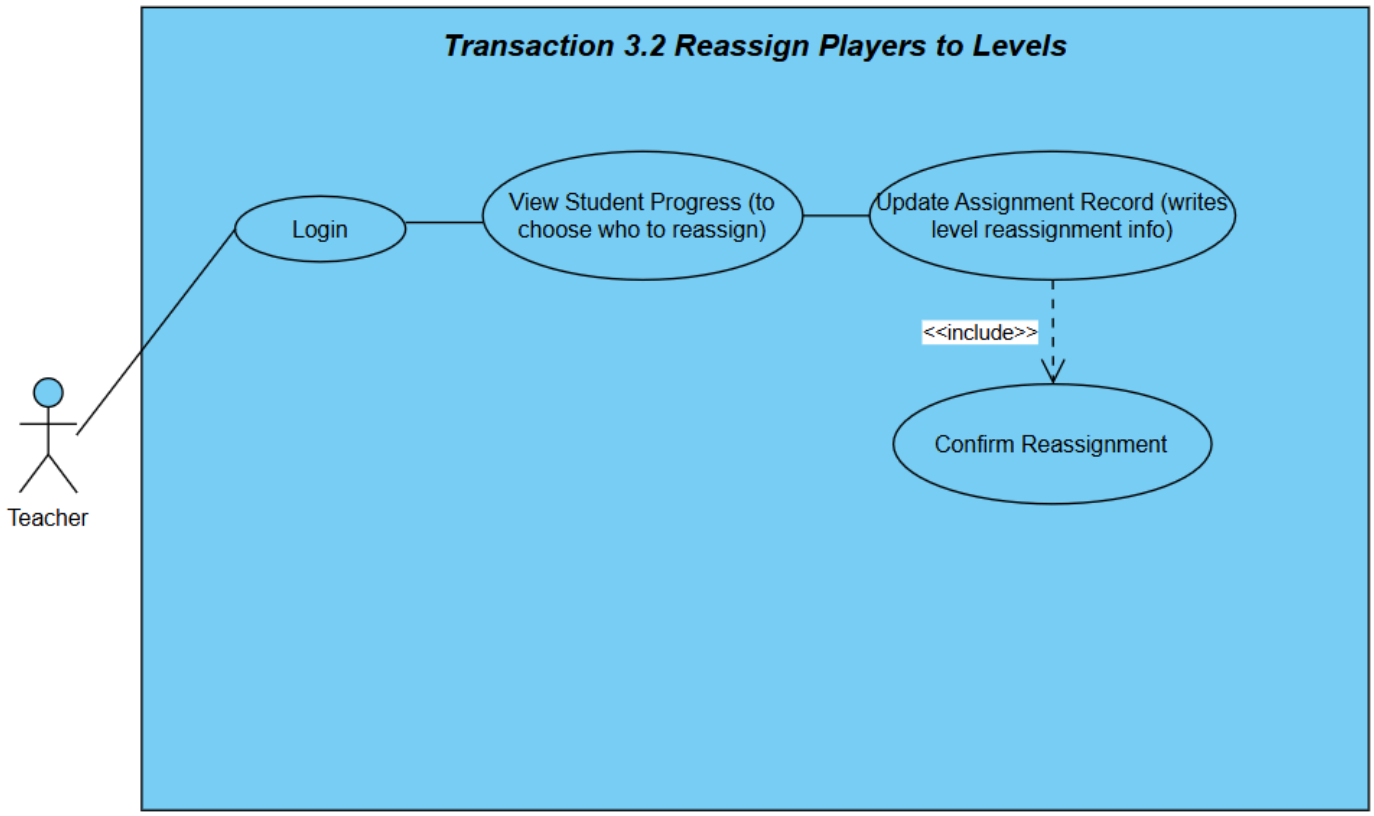


Wireframe


CyberKids
Back
Home
Students
Reports
Settings
Grade 5 Class
Game Levels
Completing the Data Leak Information
Completing the password fortress
To be Unlock

Transaction 3.2 Reassign Players to Levels

Use Case Diagram *(Class Leaderboard)*

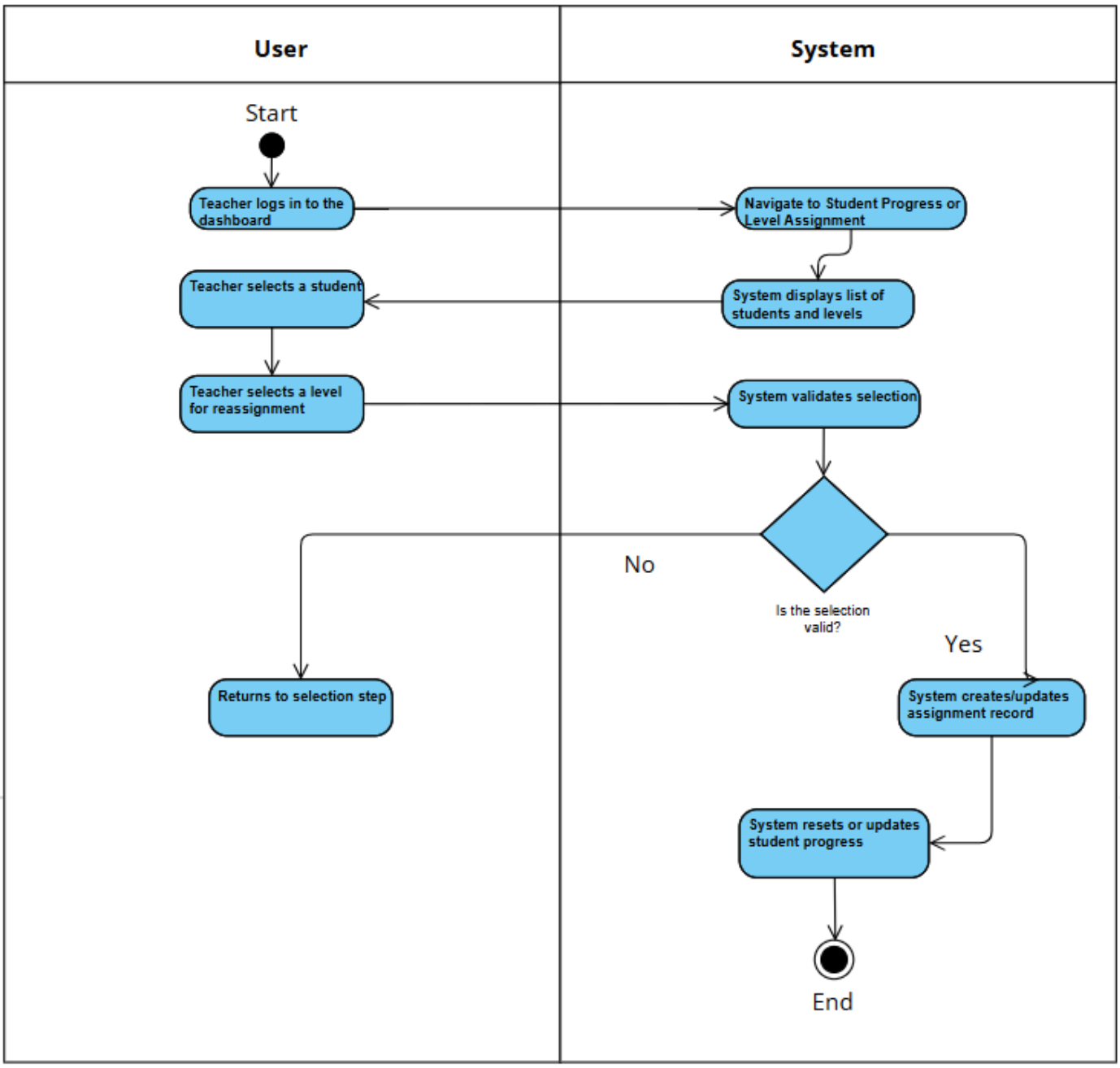


Use Case Description

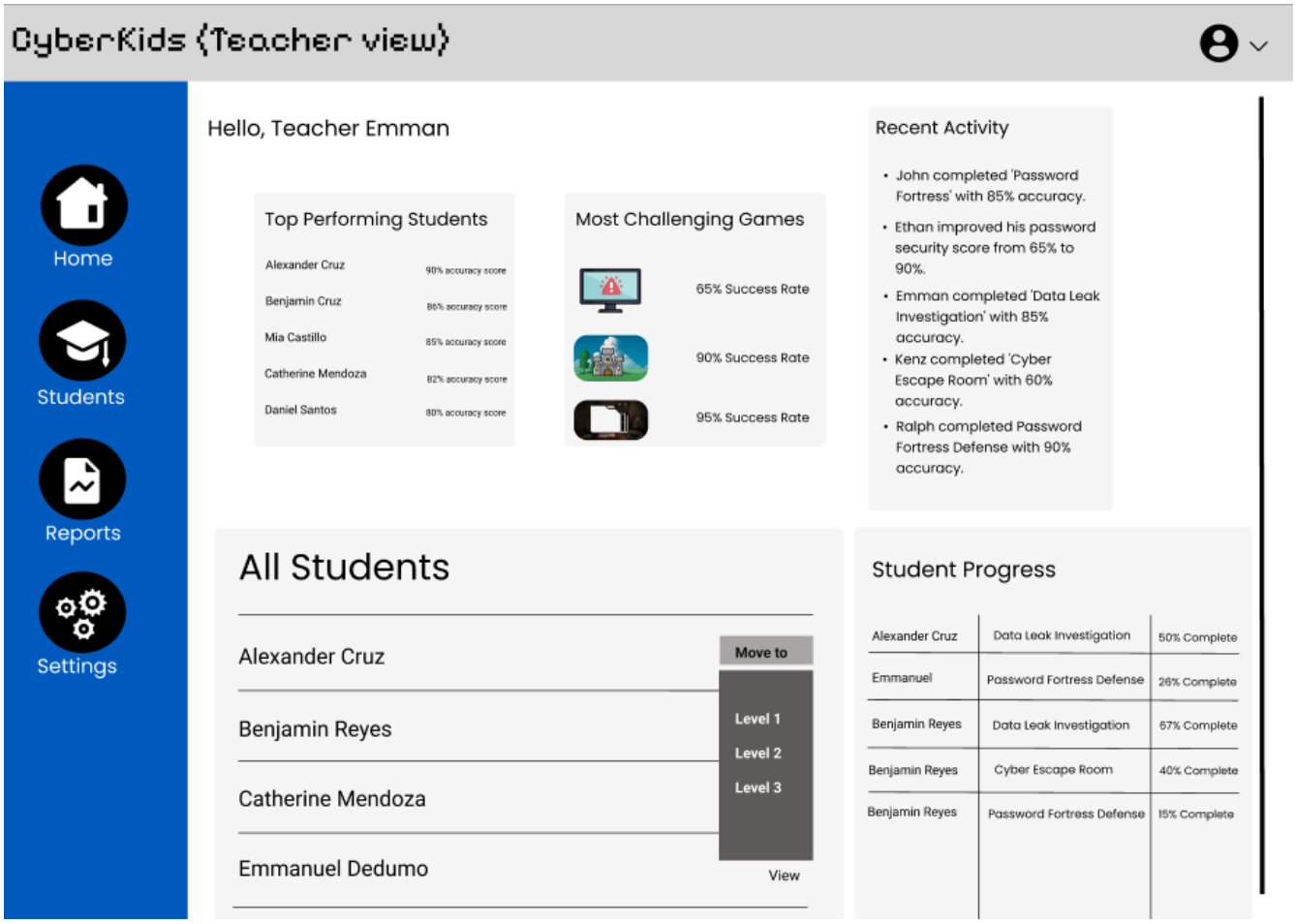
Use Case ID	UC-002
Use Case Name	Reassign Players to Levels
Actor	Teacher
Description	This use case enables teachers to reassign students to specific game levels—either for remediation (replay a level they struggled with) or for advancement (skip a level if they've already mastered it). The system handles this by creating or updating an assignment record that links the student to the desired level and adjusts progress accordingly (e.g., reset or unlock status).
Flow of Events	1. Teacher logs in to the dashboard. 2. Teacher navigates to Student Progress or Level Assignment. 3. System displays students and their current progress. 4. Teacher selects a student who needs reassignment. 5. Teacher selects the target level for reassignment. 6. Teacher chooses an action: ● Replay (Reset Progress) ● Skip to Level 7. System creates or updates an assignment record linking the student to the selected level. 8. System adjusts progress accordingly (reset or unlock).

	9. System displays confirmation message (e.g., “John Doe reassigned to Level 2 for remediation.”)
Precondition	● The student exists and has a valid progress record.
Postcondition	● The student is reassigned to the selected level.

Activity Diagram

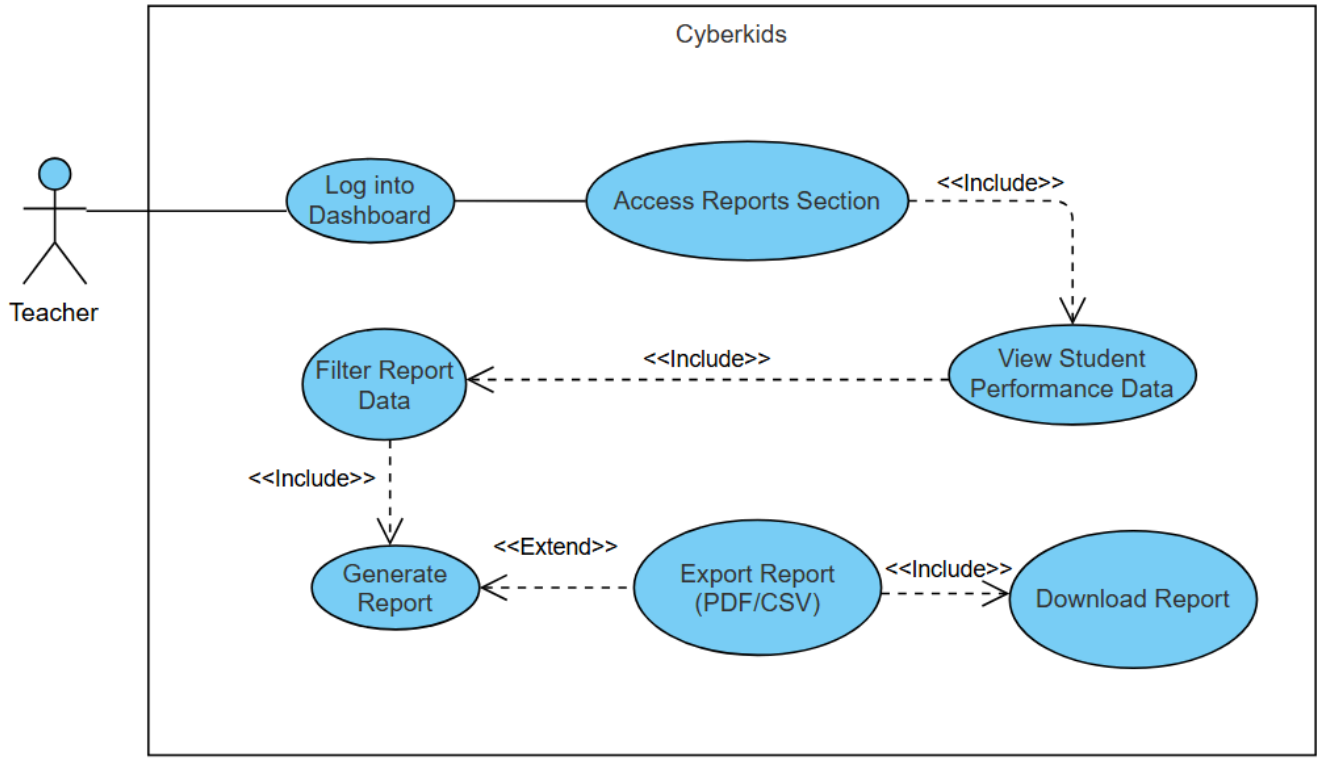


Wireframe


CyberKids (Teacher view)
Home
Students
Reports
Settings
Hello, Teacher Emman
Top Performing Students
Alexander Cruz
90% accuracy score
Benjamin Cruz
86% accuracy score
Mia Castillo
85% accuracy score
Catherine Mendoza
82% accuracy score
Daniel Santos
80% accuracy score
Most Challenging Games
65% Success Rate
90% Success Rate
95% Success Rate
Recent Activity
John completed 'Password Fortress' with 85% accuracy.
Ethan improved his password security score from 65% to 90%.
Emman completed 'Data Leak Investigation' with 85% accuracy.
Kenz completed 'Cyber Escape Room' with 60% accuracy.
Ralph completed Password Fortress Defense with 90% accuracy.
All Students
Alexander Cruz
Move to
Benjamin Reyes
Level 1
Level 2
Level 3
Catherine Mendoza
Emmanuel Dedumo
View
Student Progress
Alexander Cruz
Data Leak Investigation
50% Complete
Emmanuel
Password Fortress Defense
26% Complete
Benjamin Reyes
Data Leak Investigation
67% Complete
Benjamin Reyes
Cyber Escape Room
40% Complete
Benjamin Reyes
Password Fortress Defense
15% Complete

Transaction 3.3 Generate Reports (CSV/PDF)

Use Case Diagram

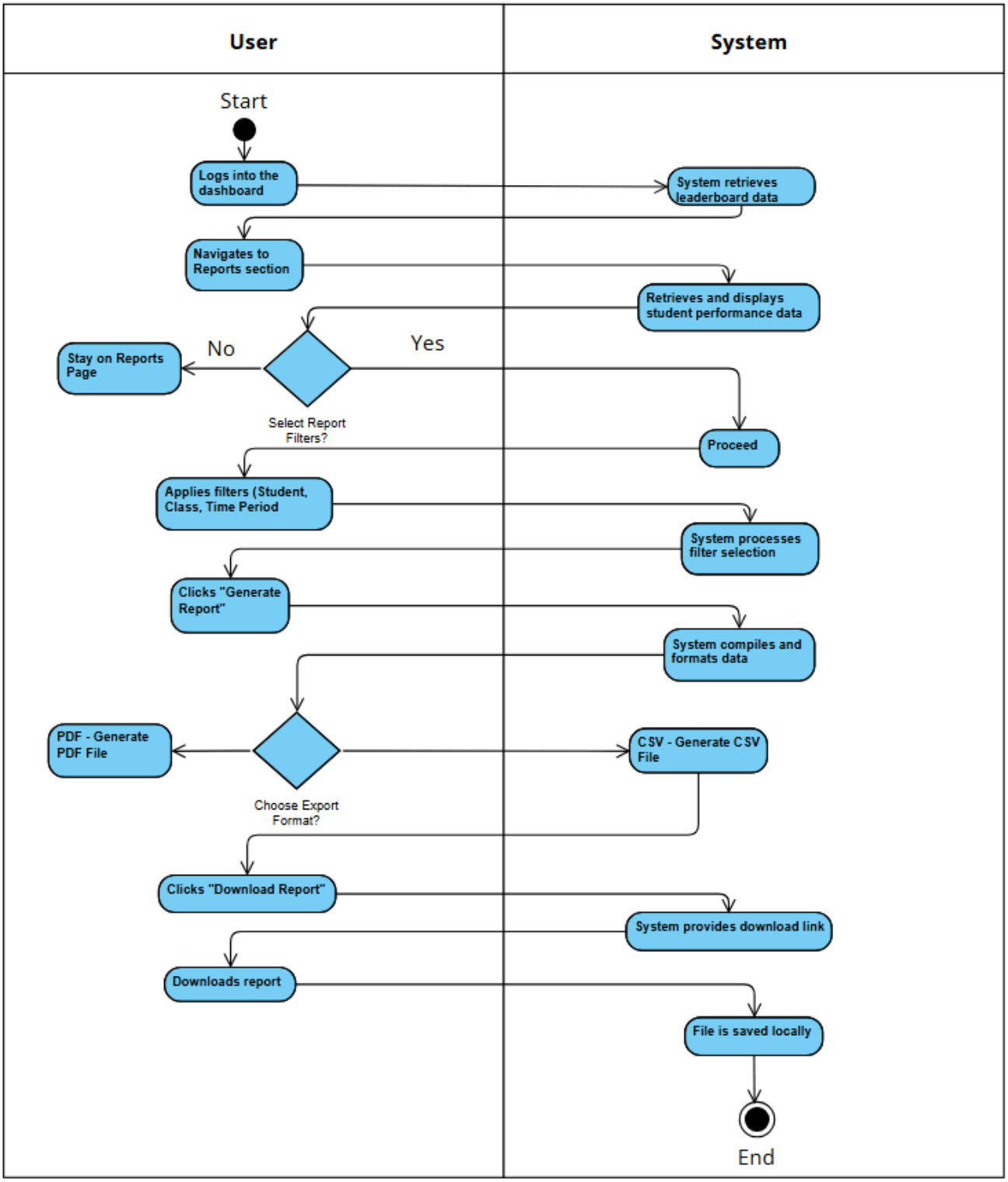


Use Case Description

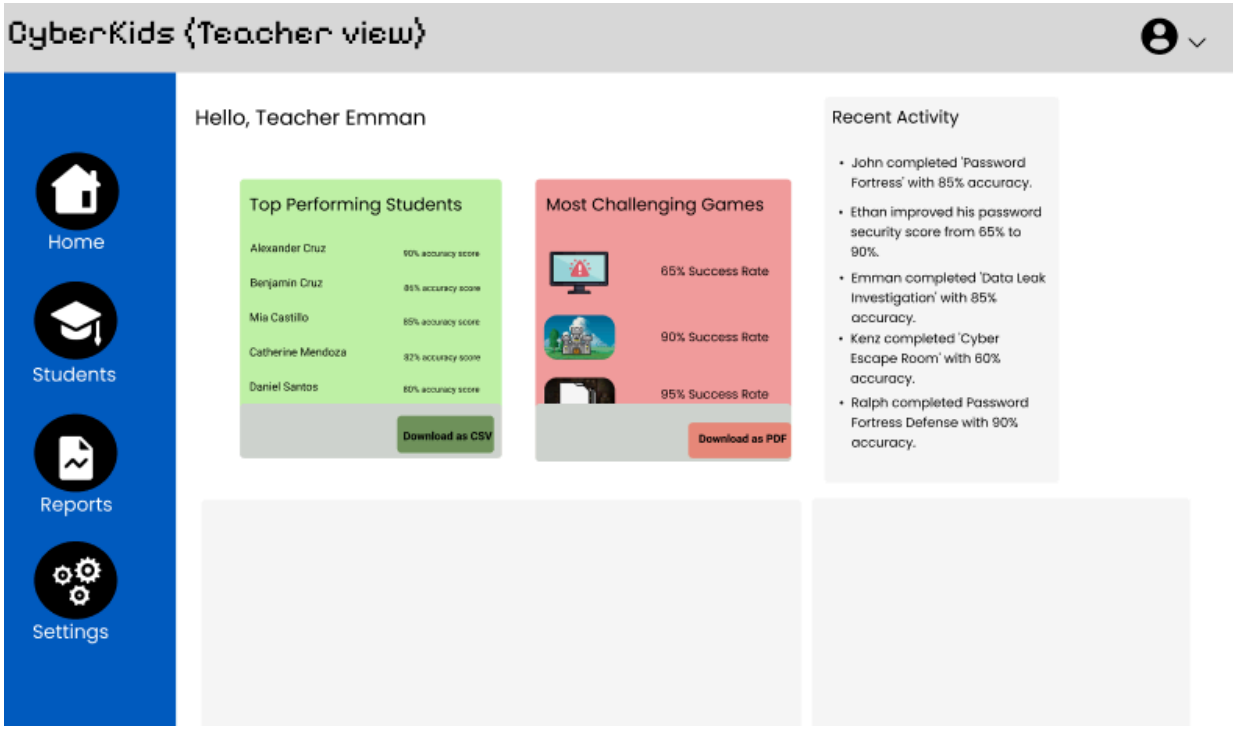
Use Case ID	UC-003
Use Case Name	Generate Reports (CSV/PDF)
Actor	Teacher
Description	The Teacher Dashboard allows teachers to generate and export student performance reports based on game data. Reports can be created for individual students or entire classes, tracking progress, scores, and completion rates for different cybersecurity challenges. Teachers can export reports in PDF or CSV formats for documentation and further analysis.
Flow of Events	1. The teacher logs into the dashboard. 2. The teacher navigates to the Reports section. 3. The system retrieves and displays student performance data, including: ● Scores and rankings for each cybersecurity challenge. ● Completion rates for individual students and the class. 4. The teacher selects report filters, such as: ● Individual student reports. ● Class-wide reports. ● Specific time periods or game levels. 5. The teacher generates the report, and the system compiles the selected data. 6. The teacher chooses an export format (PDF or CSV) and downloads the report.

Precondition	● The teacher has an active account and is logged into the system. ● Students have engaged in cybersecurity learning activities, and data has been recorded.
Postcondition	The teacher receives a detailed student performance report.

***Activity Diagram** (Student Report)*



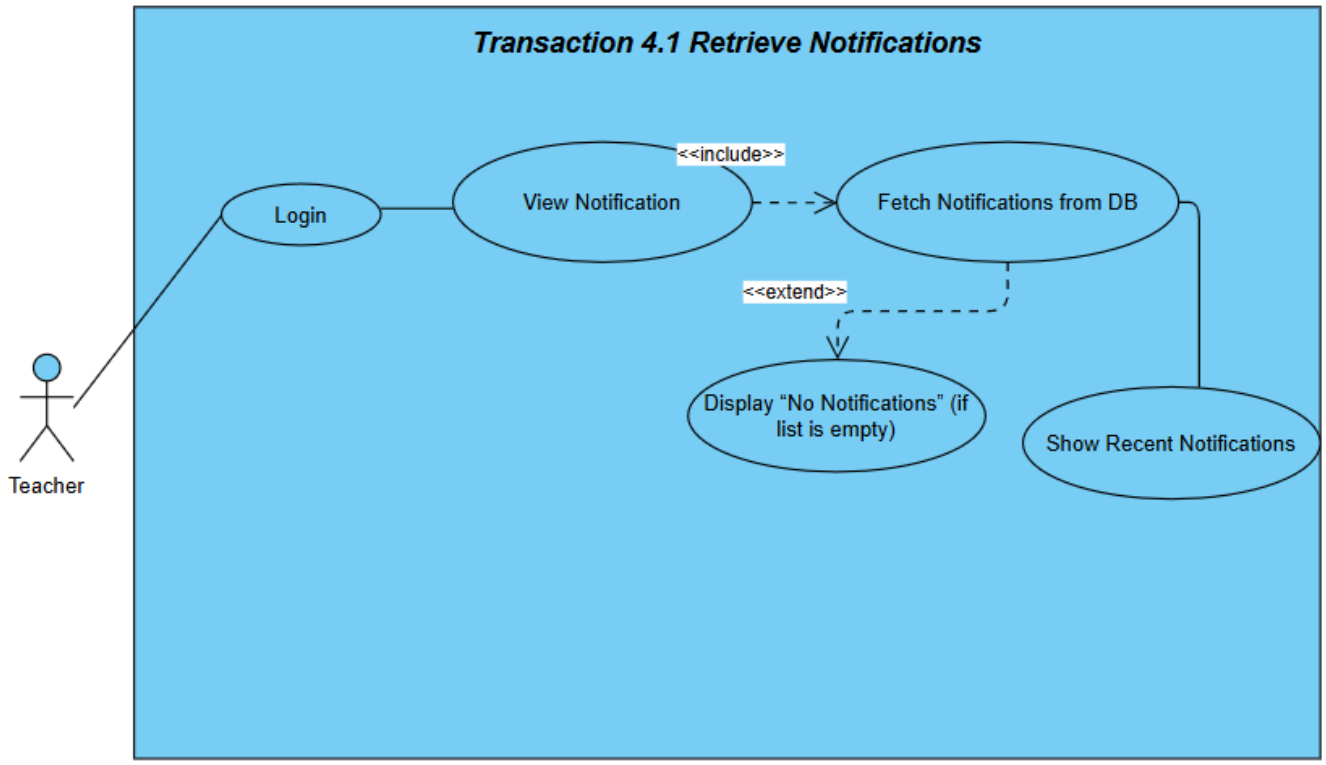
***Wireframe** (Student Report)*



Module 4: Notification System

Transaction 4.1 Retrieve Notifications

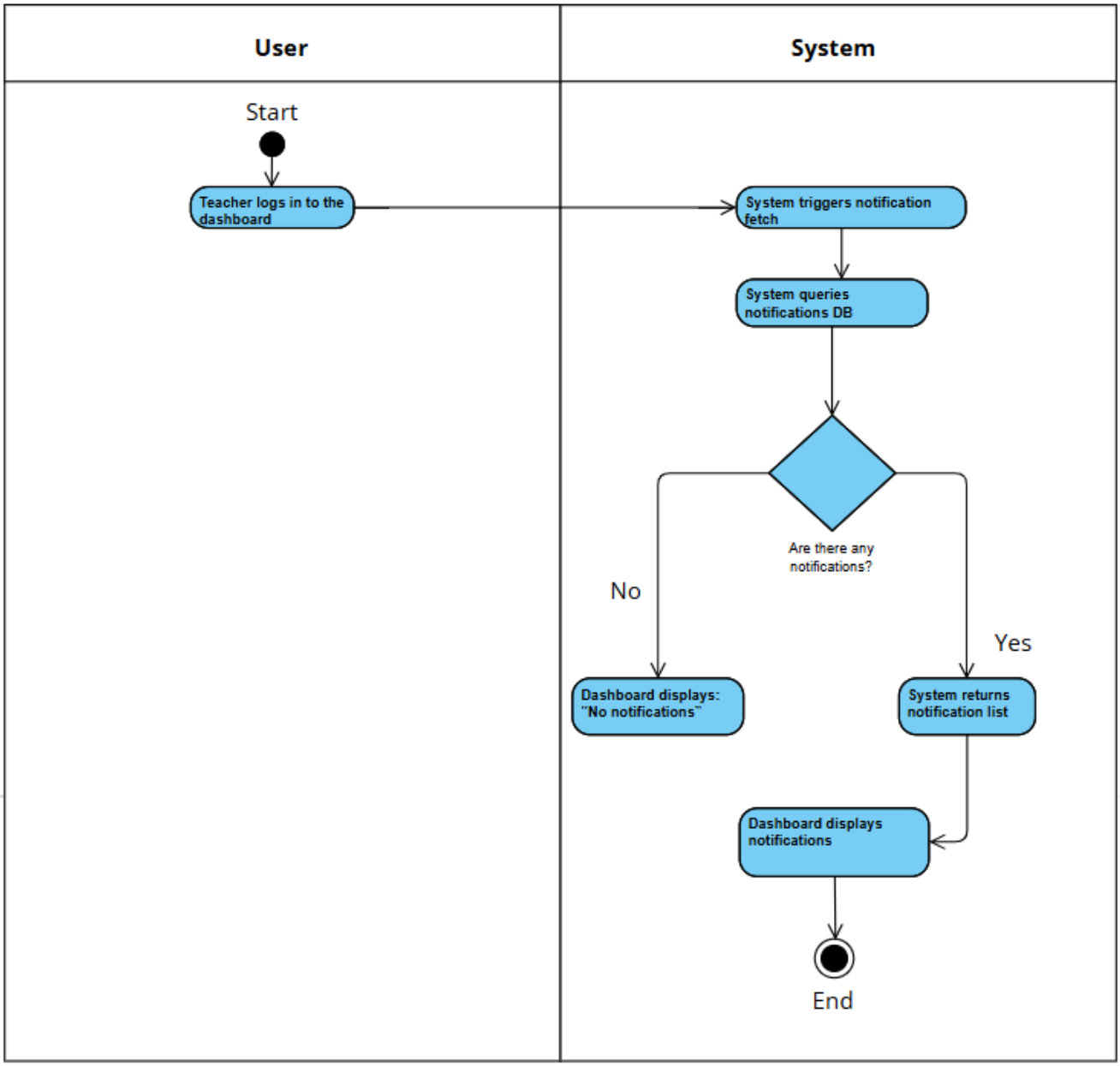
Use Case Diagram



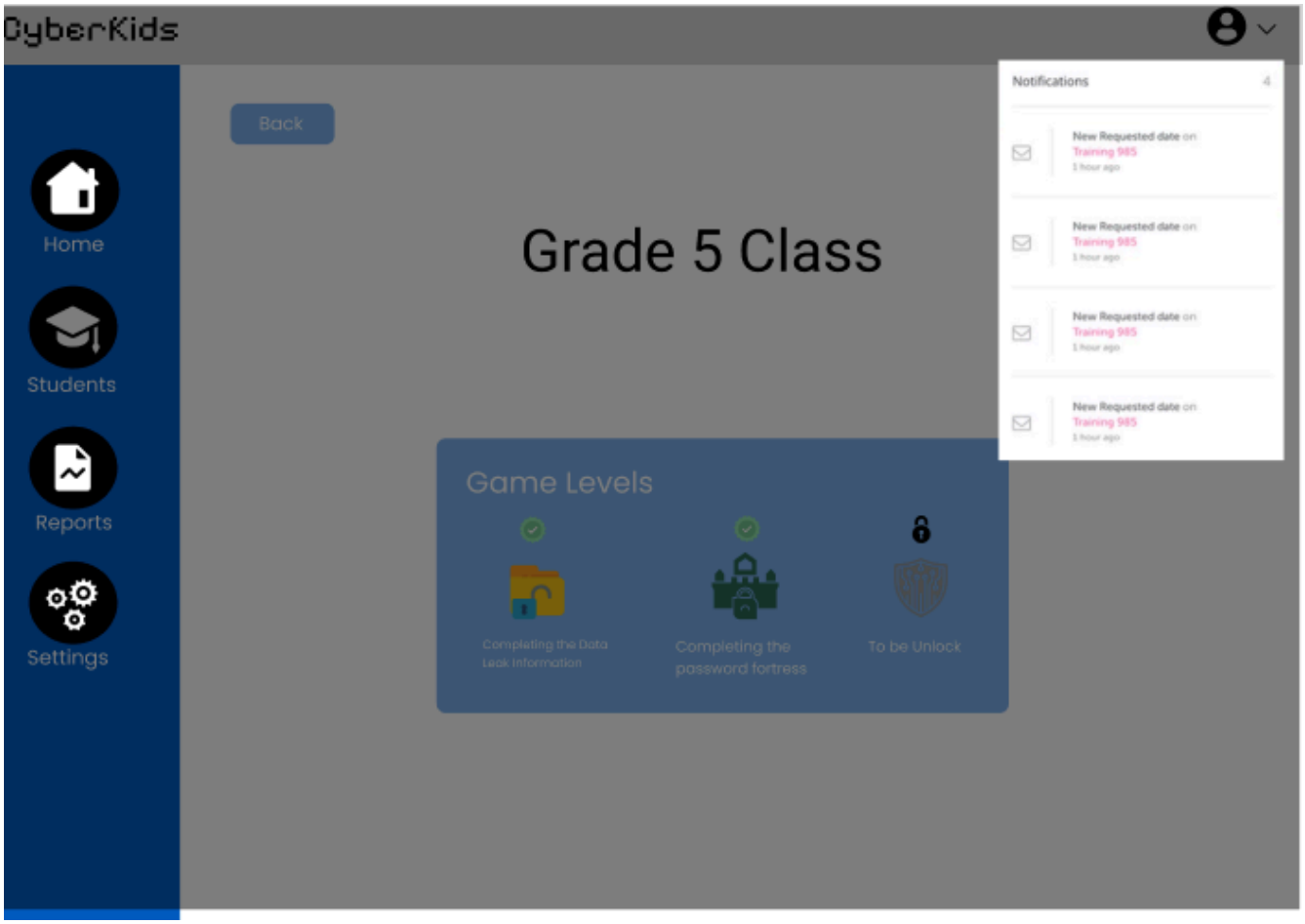
Use Case Description

Use Case ID	UC-001
Use Case Name	Retrieve Notifications
Actor	Teacher
Description	When a teacher logs in to the dashboard, the system retrieves relevant notifications such as: ● Student progress updates ● Reassignment confirmations ● Level completion alerts ● System updates
Flow of Events	1. Teacher logs in to the dashboard. 2. System triggers retrieval of recent notifications. 3. System queries the database for notifications associated with the teacher. 4. System returns the list of notifications (ordered by time). 5. Dashboard displays the notifications in the notification panel.
Precondition	● Teacher is authenticated and logged into the dashboard.
Postcondition	● Teacher sees the most recent notifications.

Activity Diagram

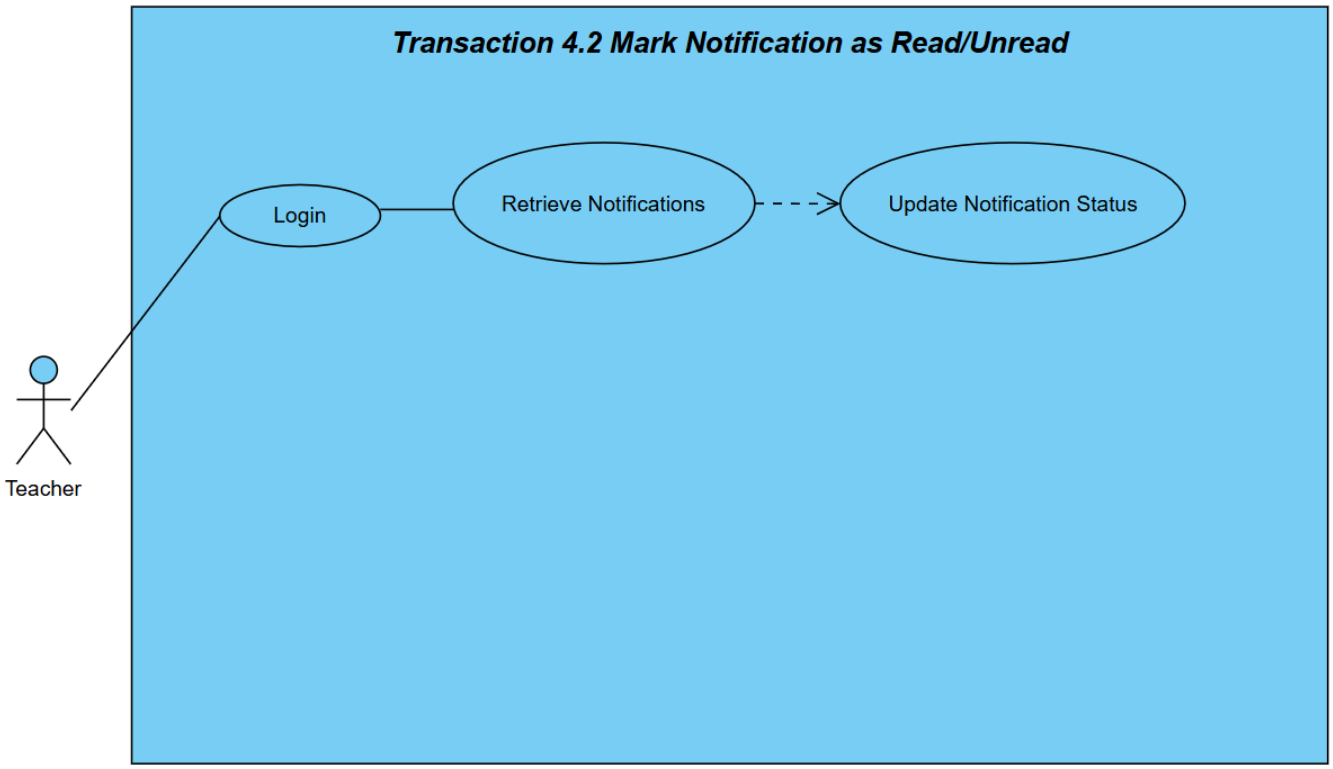


Wireframe


CyberKids
Back
Home
Students
Reports
Settings
Grade 5 Class
Notifications
4
New Requested date on
Training 985
1 hour ago
Game Levels
Completing the Data Leak Information
Completing the password fortress
To be Unlock

Transaction 4.2 Mark Notification as Read/Unread

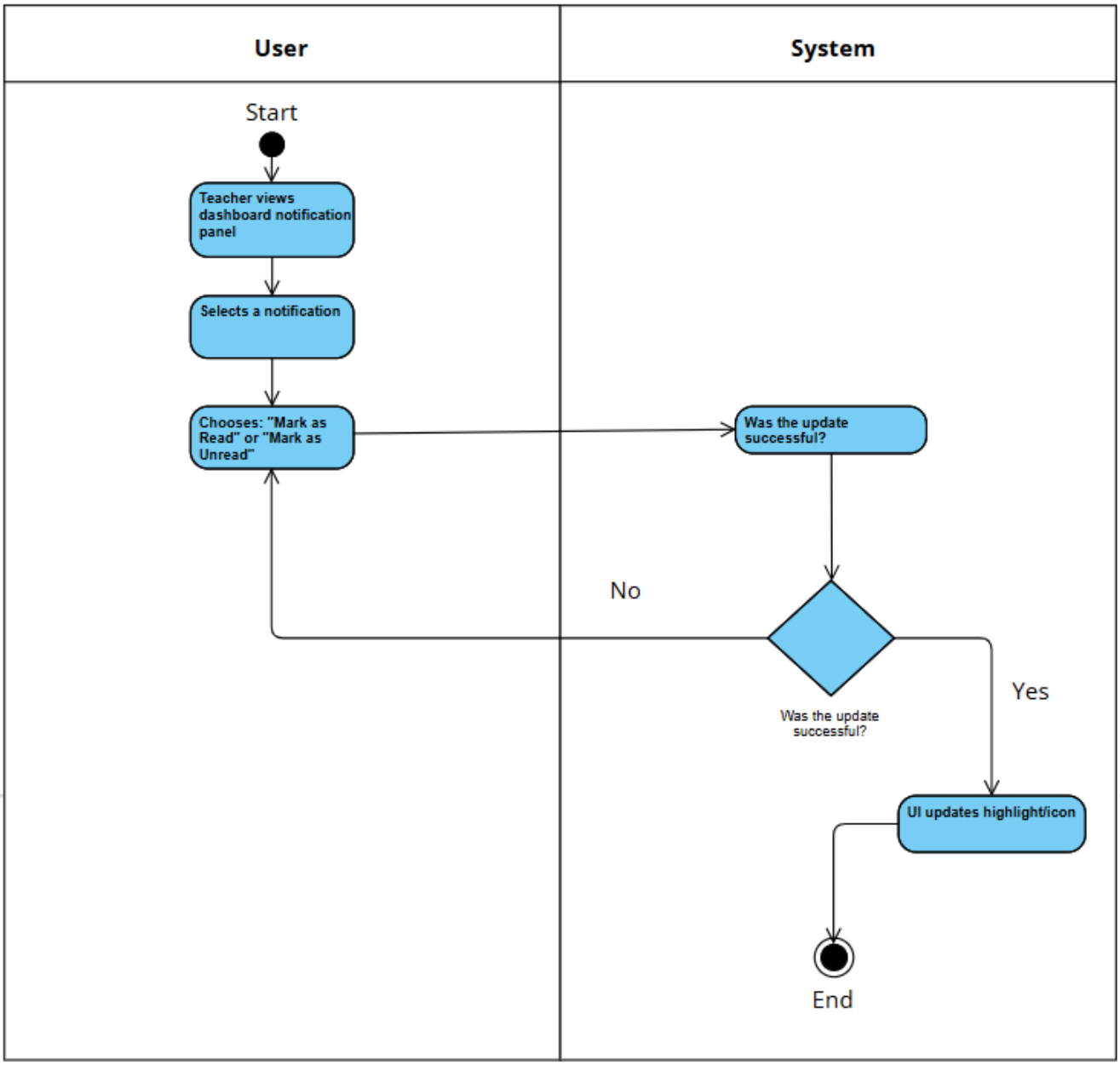
Use Case Diagram



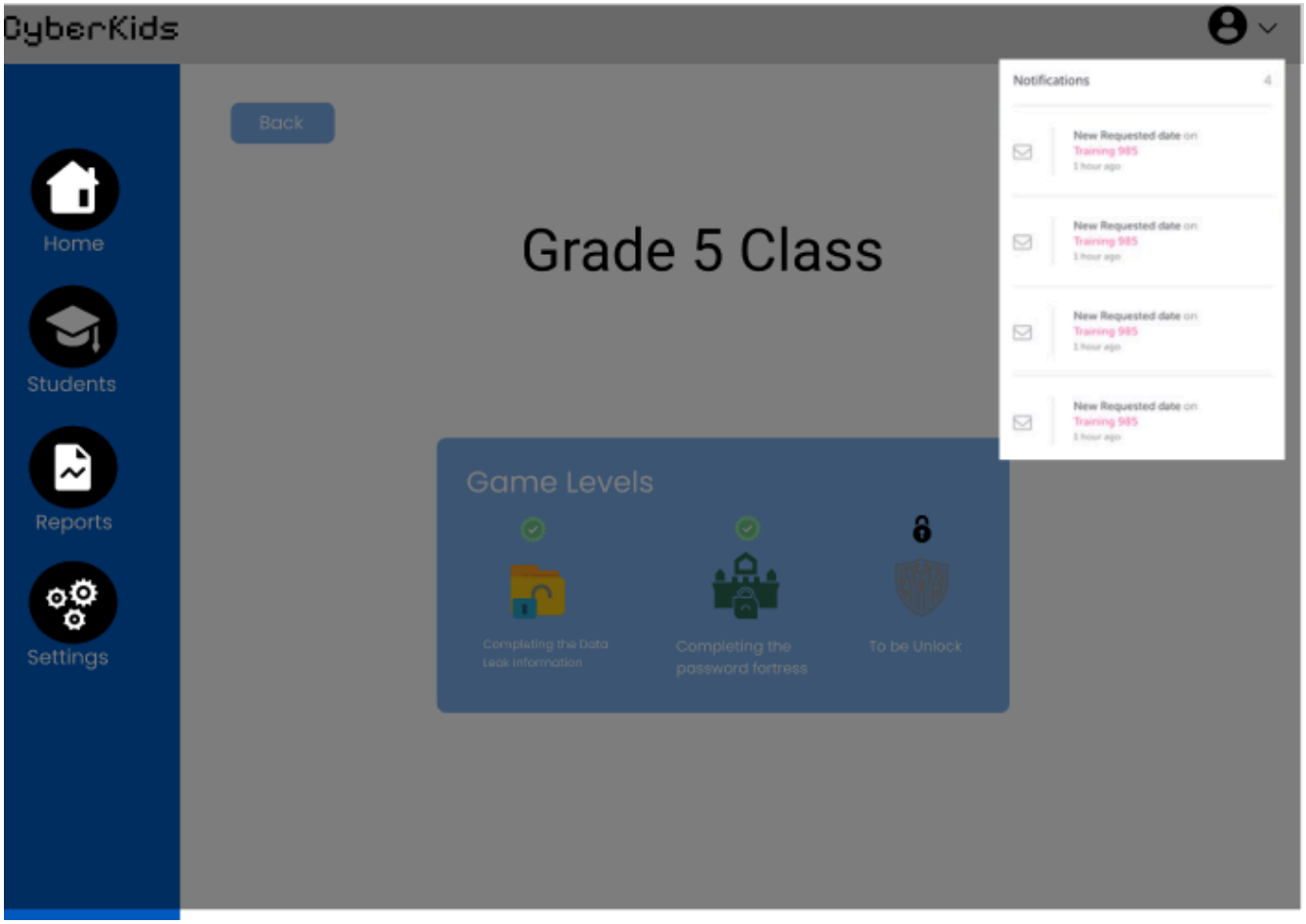
Use Case Description

Use Case ID	UC-002
Use Case Name	Mark Notification as Read/Unread
Actor	Teacher
Description	The teacher can manually mark a notification as either read (e.g., to remove highlight) or unread (e.g., to keep it as a reminder). This updates the read-status field in the database.
Flow of Events	1. Teacher views the notification panel. 2. Teacher selects a notification. 3. Teacher clicks "Mark as Read" or "Mark as Unread". 4. System sends an update request to the database. 5. System updates the `read_status` of the selected notification. 6. UI refreshes to reflect the updated status.
Precondition	● Notifications have already been retrieved and displayed.
Postcondition	● UI reflects the new state (e.g., bold/unbold, icons).

Activity Diagram

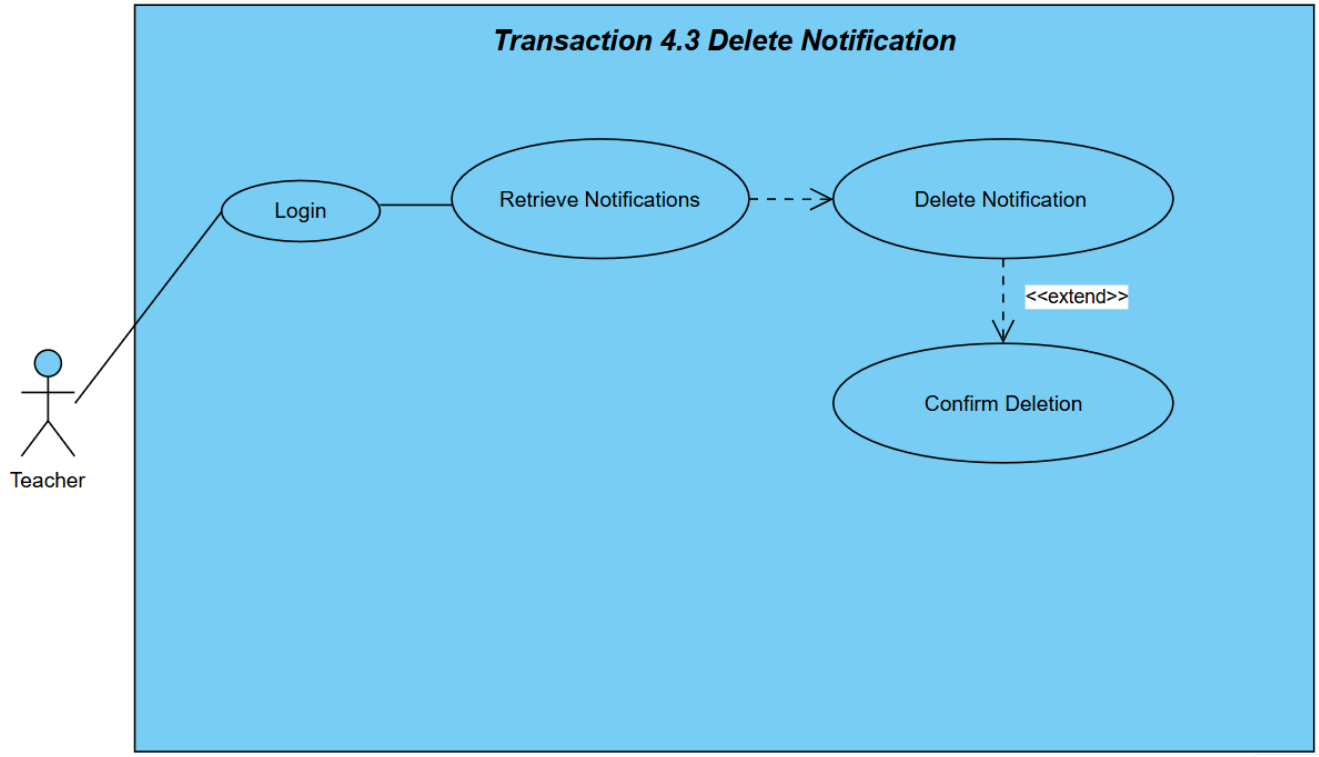


Wireframe


CyberKids
Back
Home
Students
Reports
Settings
Grade 5 Class
Notifications
4
New Requested date on
Training 985
1 hour ago
New Requested date on
Training 985
1 hour ago
New Requested date on
Training 985
1 hour ago
New Requested date on
Training 985
1 hour ago
Game Levels
Completing the Data Leak Information
Completing the password fortress
To be Unlock

Transaction 4.3 Delete Notification

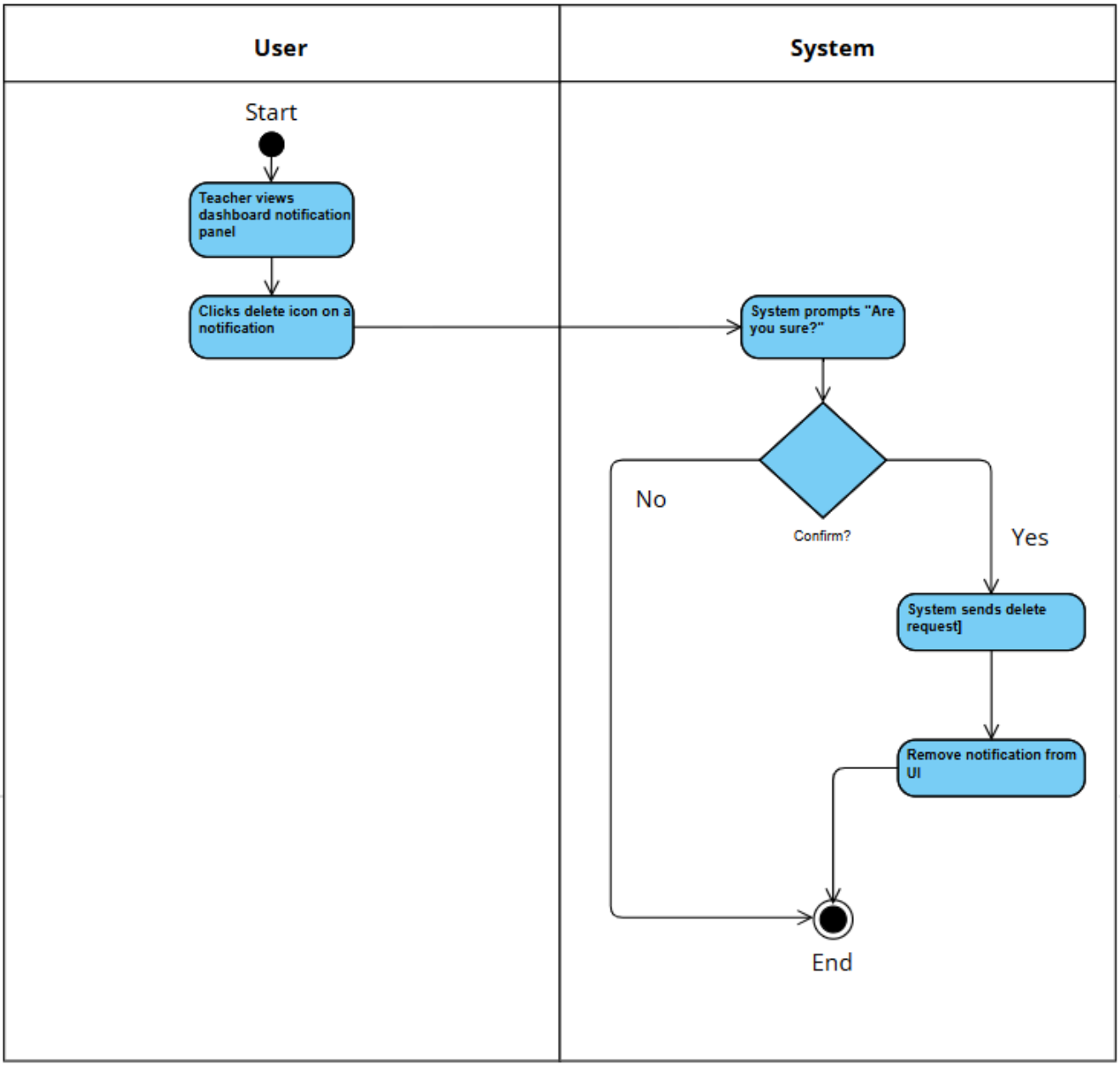
Use Case Diagram



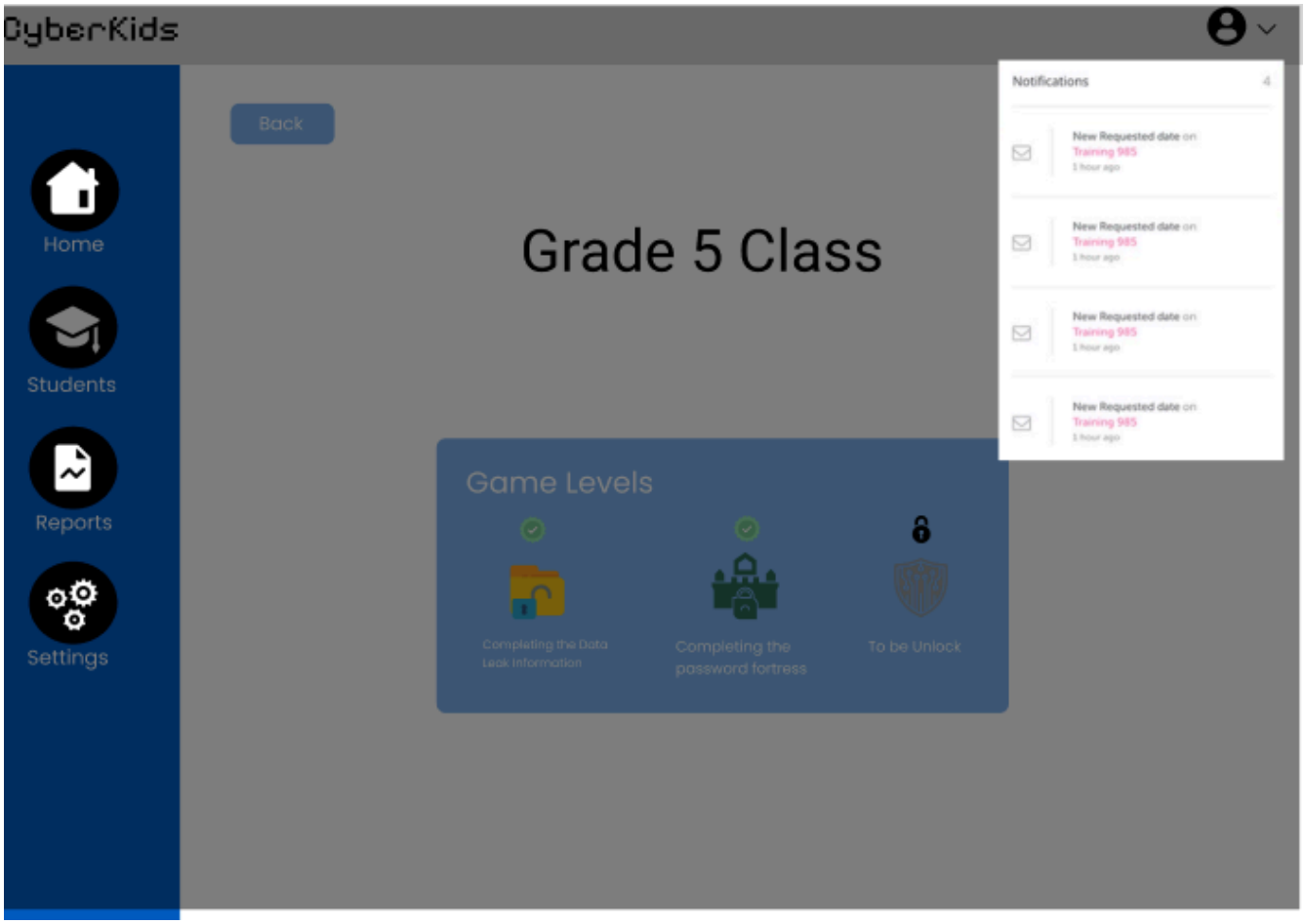
Use Case Description

Use Case ID	UC-003
Use Case Name	Delete Notification
Actor	Teacher
Description	The teacher can permanently delete a notification from their dashboard. The system removes the selected notification record from the database (or flags it as deleted, depending on implementation).
Flow of Events	1. Teacher logs into the dashboard. 2. Teacher views the notifications panel. 3. Teacher clicks the delete icon on a notification. 4. System prompts for confirmation. 5. Teacher confirms deletion. 6. System sends delete request to the backend. 7. System deletes (or flags) the notification in the database. 8. Notification disappears from the UI.
Precondition	● Notification list has been successfully retrieved.
Postcondition	● The selected notification no longer appears in the dashboard.

Activity Diagram



Wireframe


CyberKids
Back
Home
Students
Reports
Settings
Grade 5 Class
Notifications
4
New Requested date on
Training 985
1 hour ago
Game Levels
Completing the Data Leak Information
Completing the password fortress
To be Unlock

3.4 Non-functional requirements

Performance

1. **Game Responsiveness**
   - Initial Load Time: The CyberKids experience should load the main menu on Roblox within 10 seconds on a standard classroom computer (Intel i3, 4 GB RAM, integrated graphics) over a typical school Wi-Fi connection (20 Mbps).
   - Level Transition: Once a student selects a mission, the new level must begin streaming and be playable within 3 seconds. Roblox's builtin asset streaming and the game's optimized terrain/models support this target.
2. **Leaderboard Updates**
   - Polling-Based Sync: The leaderboard refreshes via a standard HTTP GET request to the API. After a student completes a mission and their score is saved, the client will poll the leaderboard endpoint every 5 seconds and update the display. This ensures no more than a 10-second delay in showing new scores.

Security

1. **User Authentication**
   - **Unique Credentials:**
     - All users authenticate via the Roblox platform. On first dashboard access (teacher) or real-name registration (student), the system issues a backend JWT (JSON Web Token) for subsequent API calls.
     - Teachers and admins log in with email/password credentials stored in the Azure MySQL database; passwords are hashed using bcrypt with a salt.
   - **Role-Based Access Control:**
     - **Student Role**: Can only read and write their own game progress, real-name record, and view leaderboard.

- **Teacher Role**: Can read data for all students in their class, generate reports, lock/unlock modules, reassign levels, and manage notifications—but cannot modify student scores.

2. **Data Protection**
   - **Real-Name Storage:** Students' real names are stored alongside their Roblox IDs in the Azure MySQL database. No other personal data (e.g., birth dates, addresses) is collected.
   - **Minimal Data Collection:** Only the real name and Roblox ID are collected; no additional personally identifiable information (PII) is stored.

3. **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

1. **System Availability**
    - CyberKids runs on Roblox's cloud platform, which offers approximately **99.7% uptime** for published experiences.
    - If the Roblox service is unavailable (planned maintenance or outage), students will be unable to play the game until the platform is restored. Leaderboard and progress synchronization will resume automatically once Roblox comes back online.

2. **Error Handling & Recovery**
    - **Backend Resilience:** All critical data (timers, scores, real names, notifications) are stored in the Azure MySQL database. The Spring Boot API handles retries for transient database errors up to three times with exponential back-off. Failed transactions return clear error messages to the client for user feedback.
    - **Script Errors:** Lua runtime errors are caught using protected calls (`pcall`). If an error occurs during a module, the game waits up to **5 seconds** to attempt an automatic reload of that module. If reload fails, the player sees a clear error message with an option to return to the main menu.