Cool Cyber Games – Interactive Platform for Teaching Cybersecurity

Design Document

Matthew Goembel, Anthony Clayton, Ludendorf Brice, Ben Allerton
Team Members

Sneha Sudhakaran
Faculty Advisor

College of Science and Engineering
Florida Institute of Technology
Melbourne, United States

February 2025

Index

1. Introduction

1.1 Project Overview

Project Name: Cool Cyber Games – Interactive Platform for Teaching Cybersecurity
Team Members: Matthew Goembel, Anthony Clayton, Ludendorf Brice, Ben Allerton
Faculty Advisor: Sneha Sudhakaran
Client: Sneha Sudhakaran, College of Engineering and Science

1.2 Purpose

Cool Cyber Games is designed to provide an engaging, interactive, and accessible platform to educate users about cybersecurity. The system aims to improve cybersecurity awareness, teach practical skills through hands-on experience, and make learning accessible to global audiences through multilingual support.

1.3 Scope

The platform will include:

- **Interactive tutorials** and **quizzes** to teach cybersecurity concepts.
- **Real-world cybersecurity challenges** and **gamification elements** to enhance engagement.
- **Progress tracking** and **certification features** to motivate users.
- A **web-based application** with cross-platform compatibility for desktop and mobile devices.

---

2. System Architecture

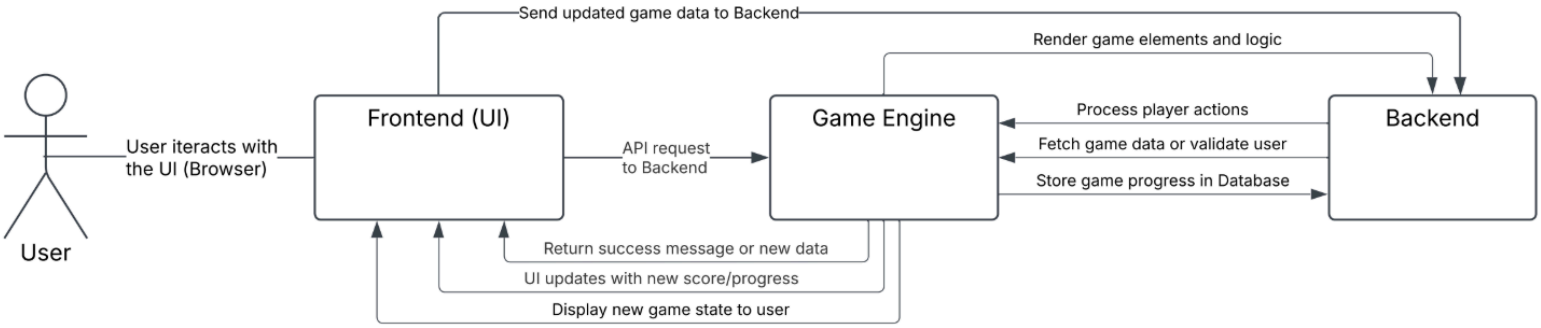
2.1 Overview

The system follows a **modular, service-oriented architecture** consisting of the following components:

1. **Frontend Module:** Implements user interfaces using **HTML** and **Godot** for game elements.
2. **Backend Module:** Handles game logic, user authentication, progress tracking, and real-time interactions using **Node.js, Render,** and **Express.js**.
3. **Database Module:** Stores user progress, scores, and game data using **MongoDB** (separate databases for **User DB** and **Game DB**).

4. **Game Engine:** Implements the core game logic, cybersecurity challenges, and scoring system using **Godot**.
5. **Authentication Service:** Manages user registration, login, and session management with **OAuth 2.0** (Google integration).
6. **Progress Tracking Service:** Logs user achievements, quiz results, and certificates.

2.2 System Architecture Diagram



---

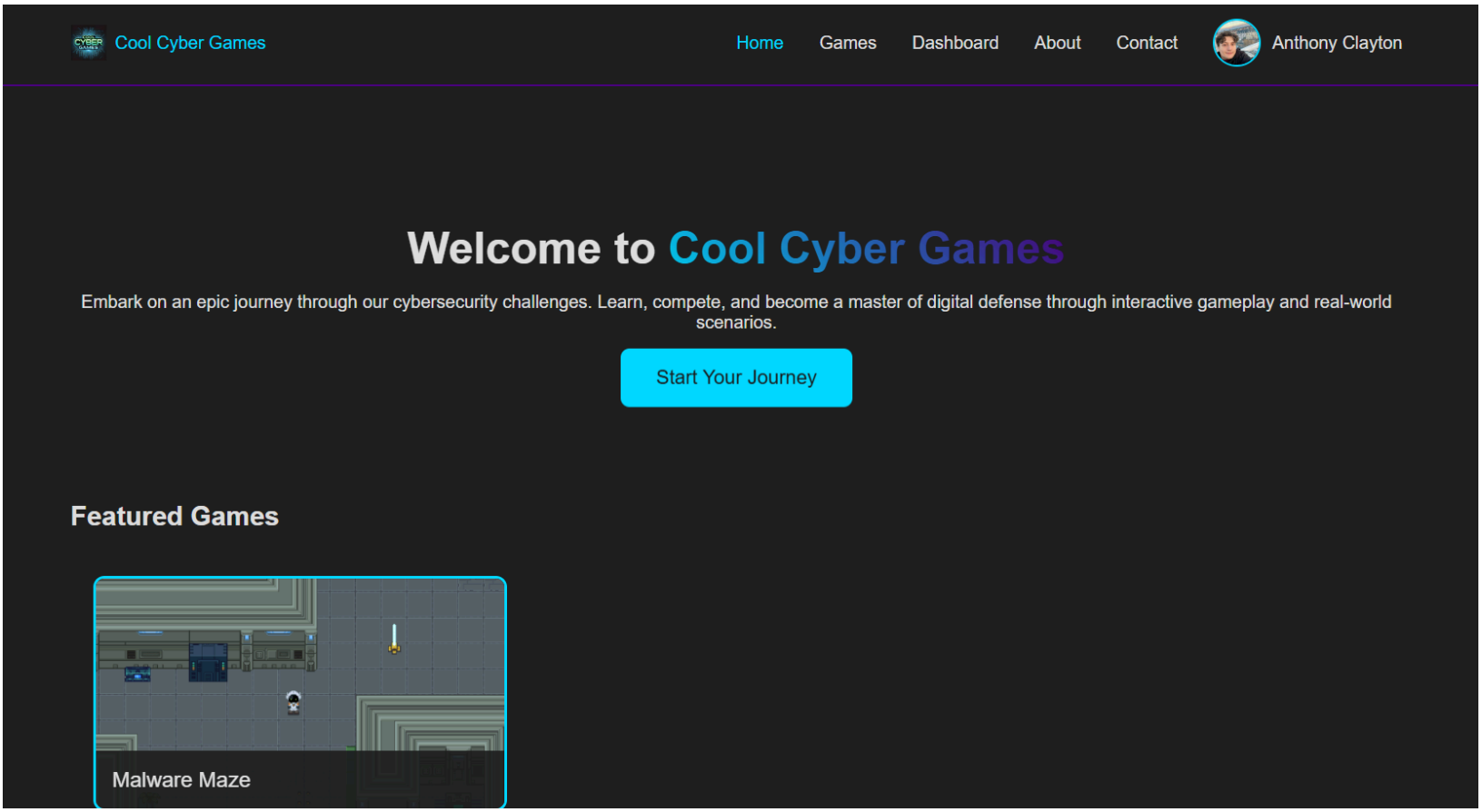
3. Module Design

3.1 User Interface (UI)

The UI will include the following screens:

1. **Homepage:** Features an overview of the platform, login/register options, and access to tutorials and challenges.
2. **Dashboard:** Displays user progress, achievements, and recommended learning paths.
3. **Game Interface:** Hosts interactive cybersecurity challenges with real-time feedback.
4. **Evaluations and Challenges:** Scenario-based quizzes and interactive challenges within the gamified module.
5. **Leaderboard:** Displays user rankings based on challenge completion and scores.
6. **Profile Page:** Shows user details, settings, and logout option.

3.2 User Interface (UI) Image



3.3 Backend API

The backend will expose **RESTful API endpoints** to:

- Handle user authentication and sessions.
- Fetch and update game progress.
- Retrieve leaderboard data.
- Store and validate quiz responses.

3.4 Game Engine

The game engine will:

- Render 2D and 2.5D interactive game elements using **Godot**.
- Simulate real-world cybersecurity challenges.
- Manage game state, scoring, and level progression.

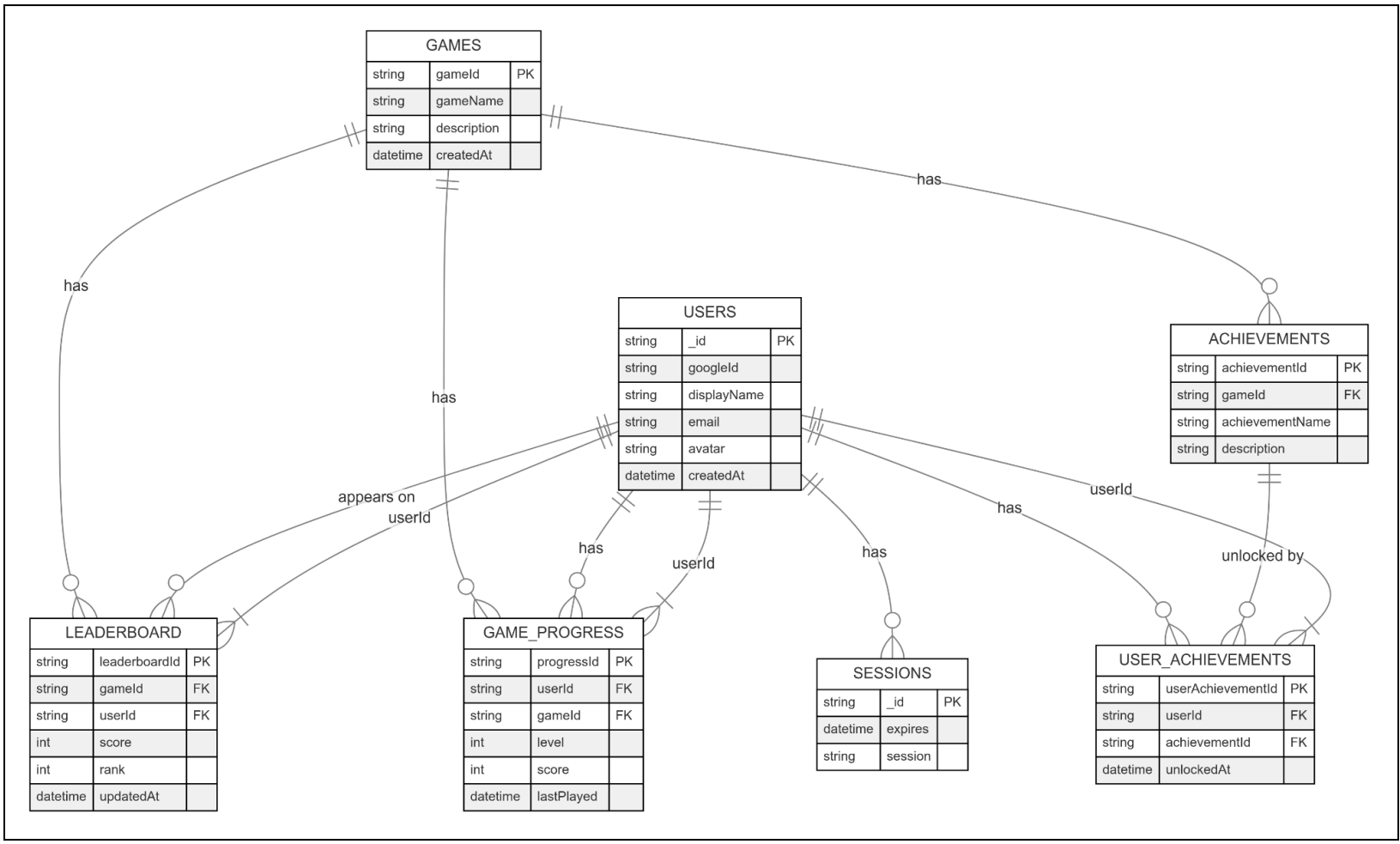
3.5 Database Schema

A Part of User DB:

1. **Users Table:** Stores user information (e.g., name, email, avatar) and authentication details.
2. **Sessions Table:** Stores information about session details.
3. **User Achievements Table:** Stores information about achievements the user has earned.
4. **Game Progress table:** Stores information about users' game progress.

A Part of the Game DB:

5. **Games Table:** Tracks user engagement and completion.
6. **Leaderboard Table:** Stores information about who's on the leaderboard.
7. **Achievements Table:** Stores information about all achievements.



3.6 Network Communication

The system follows a client-server model with secure API communication:

- **Authentication**: The system shall use **Google OAuth 2.0** for user authentication. Upon successful login, the system will receive an OAuth access token from Google, which will be used to authenticate API requests.
- **Secure Communication**: All communication between the client and server shall use **HTTPS** to ensure data encryption and prevent interception.
- **Real-Time Interactions**: **WebSockets** may be implemented for real-time interactions in cybersecurity simulations, such as multiplayer challenges or live leaderboard updates.

---

4. Algorithm Design

4.1 Example Pseudocode for API Test functionality

```
document.getElementById('testApiButton').addEventListener('click', async () => {
          try {
            const response = await fetch('/api/test', { credentials: 'include' });

            // Log the response for debugging
            const text = await response.text();
            console.log('API Response:', text);

            // Try to parse the response as JSON
            const data = JSON.parse(text);
            document.getElementById('api-result').textContent = JSON.stringify(data, null, 2);
          } catch (error) {
            console.error('API Test Error:', error);
            document.getElementById('api-result').textContent = 'Error: ' + error.message;
          }
        });
```

---

5. Additional Design Considerations

5.1 Security Measures

- Ensure secure API communication using **HTTPS**.

- Follow **OWASP security guidelines** to prevent common vulnerabilities (e.g., SQL injection, XSS).

5.2 Deployment Strategy

- Host the application on **Render** for backend services and **Godot** for game hosting.
- Use **MongoDB Atlas** for cloud-based database management.

---

6. Conclusion

Cool Cyber Games is designed to provide an innovative and engaging approach to cybersecurity education. With its modular architecture, gamified challenges, real-world simulations, and multilingual support, the platform ensures accessibility, scalability, and security for a broad audience.

---

7. Appendix

This document is subject to change as development progresses. Future iterations will include additional details on implementation, testing, and deployment strategies.