Cards of Cthulhu

04.05.25 - Version 1.0.0 – Final

Members:

Luiggie Lucenta (*Quality Assurance*) - 2101103698@student.buksu.edu.ph

Jeter Giambi Melendez (*Project Manager*) - 2201104863@student.buksu.edu.ph

John Winchel Valdez (*Lead Developer*) - 2201108251@student.buksu.edu.ph

II. Table of Contents

III. Document Overview

Scope: This document provides a comprehensive overview of the development process and technical decisions made in creating the Unity-based mobile game "Cards of Cthulhu" by Team T43, BSEMC-DAT 3A, for the course Application Development and Emerging Technologies. It covers all major aspects of the project, including user interface behavior, core gameplay mechanics, API integration, and design rationale. The primary focus is on how the game operates technically and visually within Unity.

The only out-of-scope component is the marketing and promotion strategy, which is not part of the current project requirements.

Audience:

This document is intended for instructors, developers, testers, and academic reviewers involved in evaluating or understanding the project. It also serves as a reference for future development or maintenance work.

IV. Project Overview

- **Executive Summary**: Cards of Cthulhu is a turn-based card game inspired by the horror fiction of H.P. Lovecraft. The game draws gameplay inspiration from roguelike deck-building titles such as Slay the Spire and Inscryption, where players engage in repeatable gameplay loops involving strategy, chance, and card collection. In roguelike games, players typically start from the beginning after each run, with gameplay progression offering new cards, strategies, and challenges over time.

  The target audience includes casual gamers and fans of card-collecting mechanics, particularly those with an interest in dark, Lovecraftian themes. The game is developed for mobile platforms, specifically Android.

- **Objectives**: Technical success in the application is to have a bug-free and stable flow of the gameplay loop. Creatively, the art style, different game functions, strategic gameplay, and simplicity are what we want to achieve. As an entertainment app, we strive to make the application fun, first and foremost. The backend API should be stable and working, not the most professional API system, but a working one. Finally, mobile deployment—our goal is to have the game run smoothly and properly on most of the Android devices we release it to.

- **High-Level Features**: The turn-based combat system successfully implements a dynamic enemy structure with scalable difficulty levels—Easy, Normal, and Hard. Each enemy introduces unique gameplay mechanics. For example, Sauron, a boss-level enemy categorized under Hard difficulty, possesses a health-drain ability, allowing it to heal 15% of the damage it inflicts on the player. Player progress and performance are stored through a simple online profile system. This is managed via an API that communicates with a hosted database, saving essential information such as the player's highest score and the number of consecutive enemies defeated.

- **User Needs**: Players experience progression through beating enemy after enemy, each level raising the difficulty, and if the chances reach the requirements. The player fights a boss entity with its unique abilities and code. Whilst the lore is not explained, the theme and mood of the game (along with the art style) paint an image of what the world of the game is. Lastly, the main purpose of users' needs regarding this game is to entertain, which we try to achieve not only through gameplay and code but also through theme and art.

- **Market Analysis**: Players drawn to Cards of Cthulhu are likely fans of dark and mysterious themes rooted in Lovecraftian horror. They are typically attracted to strategic gameplay, especially in the form of turn-based combat and deck-building mechanics. Whether they are casual gamers looking for an atmospheric mobile experience or dedicated fans of the Cthulhu Mythos and Eldritch horror, this game caters to those who appreciate storytelling, tactical decisions, and the eerie thrill of confronting the unknowable.

V. Functional Specifications

- **Feature List**:
    - **Profile System**: The profile system enables players to create, edit, and delete their respective profiles that store their unique in-game values, such as “enemies defeated”.
    - **Card Drag & Card Drop System:** This system enables the player to hold and drag the cards as game objects and drop them onto the Enemy, which is attached with the Card Drop code. This card drop area detects what specific card has been dropped and runs the following: Attack, Heal, or Magic.
    - **Attack Card**: The attack card is the damage function for the player that chooses a minimum and maximum number, which is used to subtract the health points of the opposing enemy target.
    - **Heal Card**: The heal card is the healing function for the player that chooses a minimum and maximum number, which is used to add health points to the player’s health.
    - **Magic Card**: The magic card is the combo function for the player that applies when, specifically, the same cards are used before this card is played (e.g., Attack -> Attack -> Magic). It then functions with a respective effect called “Critical hit” or “Massive heal”.
    - **Critical Hit**: This function is applied to the magic card when its requirements are met. It will deal a higher minimum and maximum number, which is used to subtract the health points of the opposing enemy target.
    - **Massive Heal**: This function is applied to the magic card when its requirements are met. It chooses a higher minimum and maximum number, which is used to add health points to the player’s health.
    - **Enemy System**: The enemy system is controlled by an enemy “stat” controller, where the values such as maximum HP, min Damage, and maximum Damage are set. These values are different based on the level or difficulty of the enemy.
    - **High-score Count**: This is the value of how many enemies or waves the player has successfully played without their health point bar reaching 0 HP.
    - **Battle Log**: It displays all actions done by both the player and the enemy. (Acts as a log of the game).

- **User Interaction Flow**:

    - **Profile System**: Main Menu -> Play(button) -> Profile Panel -> Create(button) -> Game Panel
    - **Card Drag & Card Drop**: Card (hold) -> Drag -> Enemy (Drop)
    - **Attack Card:** Decide from min(10)–max(20) value -> subtract enemy HP
    - **Heal Card:** Decide from min(10)–max(20) value -> add player HP
    - **Magic Card:** Detects *if else* statement -> (*if yes*) runs "critical hit" or "massive heal" code. /// (*if no*) -> nothing happens and the game proceeds
    - **Critical Hit:** Decide from min(30)–max(60) value -> subtract enemy HP
    - **Massive Heal:** Decide from min(30)–max(60) value -> add player HP
    - **Enemy System:** Choose array [1, 2, 3, 4] (difficulty level & sprite model) -> (*if player wins*) increment enemy damage, hp, etc.
    - **High-score count:** (*if* Player defeats Enemy) -> score: +1
    - **Battle Log:** Action made by Player and Enemy -> Display printed text
- **Use Cases**: Upon launching Cards of Cthulhu, users are greeted with the main menu featuring several navigational options: Play, Credits, Leaderboard, and Quit. Selecting Play directs the user to the Profile Panel, where they can create, edit, or delete user profiles. If no profiles exist, the player must create one. New profiles are added to a scrollable list of selectable user profiles. Once a profile is selected, the game transitions to the core gameplay screen.

  The player begins by selecting a deck, which randomly generates a hand of three cards, consisting of Attack, Heal, or Magic cards. These cards are used strategically in turn-based combat to defeat the enemy that appears. Upon defeating an enemy, the player's high score (enemy win streak) increases, and the next enemy appears with higher difficulty, featuring increased health, damage, or unique traits.

  At higher difficulties, players may encounter bosses with special mechanics, such as HP Drain, which allows the enemy to recover a percentage of the damage it deals. If the player loses, the screen fades to black and transitions to the Leaderboard, where their high score is recorded under their selected profile.

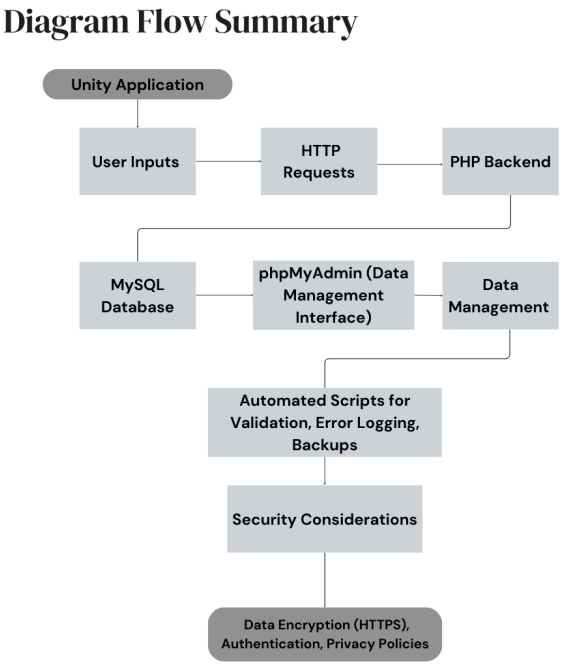
- **User Stories**: "I opened the program and was met with a simple menu screen. I navigated the different buttons before pressing play—when I was fully satisfied with it, I started the game. After pressing play, I

entered the profile creation screen which included three buttons; create, edit, and delete. There were empty profiles there so I pressed create—it prompted me to write down my name and after doing so, I clicked enter."

- **Acceptance Criteria**: "Player can gain XP after each battle."
- **Non-functional Requirements**: Fast load time, 30 FPS minimum, data encrypted, etc.

VI. Technical Specifications

- **Architecture Diagram**:



- **Design Pattern**: Mention if you use a structure like **MVVM** or **Singletons** for managers in Unity.
- **Platform Details**: Unity supports Android/iOS; list supported devices.
- **Data Management**: Explain how data moves: game sends player data → API stores it in the database.
- **Database Schema**: Tables like PlayerProfile (id, name, level, xp), etc.
- **API Endpoints**: List /api/player, /api/player/:id, what methods (GET, POST, etc.), and expected data.

- **Security**: Say something like "Only authenticated requests can edit data. Data is not stored in plaintext."

- **Privacy**: Explain what data is collected and that you won't misuse it.
- **Third-Party Integration**:

Default Android SDK Components and Libraries in Unity

Unity provides specific versions of Android SDK components to ensure compatibility and minimize build-related issues. In Unity version 2023.2, the default SDK components include SDK Tools (v26.1.1), Build Tools (v32.0.0), Command-line Tools (v6), and Platform Tools (v33.0.1). These versions are selected to align with Unity's internal build requirements and are updated periodically.

Additionally, Unity includes built-in libraries for Android development that enhance the engine's capabilities. These libraries—such as `UnityEngine`, `UnityEngine.UI`, `UnityEngine.Android`, `UnityEngine.XR`, and `UnityEngine.InputSystem`—are automatically integrated into all Android builds, providing core functionality, platform-specific features, and support for advanced input and XR development.

Other Applications Used

ChatGPT
ChatGPT, developed by OpenAI, is an advanced AI language model capable of understanding and generating human-like text. Within this project, ChatGPT was utilized to generate code snippets, debug existing code, and provide suggestions for API and database design, thereby enhancing development efficiency and interactivity.

XAMPP
XAMPP is a free, cross-platform software package that provides a local web server environment, bundling Apache (web server), MySQL/MariaDB (database server), PHP, and Perl. In the context of this project, XAMPP was used to host RESTful APIs and manage a MySQL database, enabling backend communication for the Unity application via `UnityWebRequest`.

phpMyAdmin
phpMyAdmin is a browser-based interface for managing MySQL/MariaDB databases, included with XAMPP. It was used to visualize, edit, and manage game-related data such as player profiles, game progress, and leaderboards. While Unity does not directly interface with phpMyAdmin, it facilitated backend data management and administrative task

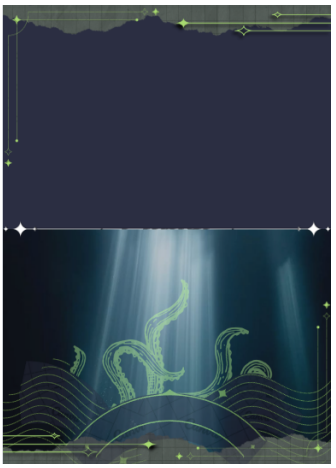
VII. UI/UX Design Specifications

✅ What to do:

- **Wireframes/Mockups**:

Main Menu

Profile



Gameplay

Leaderboard

Credits

Screen List: Main menu, Profile, Gameplay, Leaderboard, and Credits

Navigation Flow: From the Main Menu it has 4 buttons , the play, leaderboard, quit, and credits. The player will go to the profile tab, create a user name then click it and will direct you to the gameplay tab. While the leaderboard will lead you to the leaderboard tab and the same as credits.

Accessibility: The fonts used are sans-serif and easy to read, while a clean background helps users focus on the options. The design uses high-contrast elements white buttons paired with clear, dark text that support easy readability across a variety of screens and lighting conditions.

VIII. Deployment & Maintenance

✅ **What to do:**

- **Release Plan**: Unity build → APK export → shared via Google Drive or Itch.io
- **Update Strategy**: Fix bugs, add content in small patches.

XI. Appendices

✅ **What to do:**

- **Glossary**: Define game terms—e.g., "Combo", "XP", "Cooldown".
- **References**: List sites or docs used—Unity Docs, Express API guides, etc.