Software Engineering Department
Braude College

Capstone Project Phase A – 61998

EscapeCode - Accessible Programming Learning Game Using Unity

Supervisor
Dr. Moshe Sulamy

Team Members

Yinon Lev

E-mail: yinon46levi@gamil.com

Shahaf Israel

E-mail: shahaf564@gmail.com

Github Repository: https://github.com/shahaf5641/EscapeCode

Table Of Content

1 Abstract

EscapeCode is an exciting and forward-thinking project that combines the fun and challenge of escape rooms with the opportunity to learn programming—all while being accessible to people with physical disabilities.

Using innovative technologies like eye-tracking, speech-to-text, and AI assistance, EscapeCode creates an engaging platform where players can solve coding puzzles without needing traditional input devices like keyboards or mice.

2 Introduction

In today's rapidly advancing technological landscape, the availability of diverse and innovative tools opens up countless opportunities to create groundbreaking solutions. From artificial intelligence to augmented reality, these tools not only transform industries but also reshape the way we interact with the world. However, this progress raises an important question: how can we harness these tools to revolutionize traditional fields and make them accessible to underserved populations? This challenge serves as the foundation for EscapeCode, a unique project designed to blend innovation, education, and inclusivity.

EscapeCode is an interactive desktop game that combines the excitement of escape room puzzles with the educational value of programming. Unlike traditional games or learning platforms, EscapeCode is designed to be accessible to individuals with physical disabilities. By integrating cutting-edge technologies such as eye-tracking and voice-to-text, EscapeCode allows players to engage with coding tasks without relying on a keyboard or mouse. This creates a fun, inclusive, and empowering way for users to learn programming skills, fostering both education and accessibility in one seamless experience.

The game leverages standard hardware—basic webcams and microphones—to ensure affordability and wide accessibility. Players solve puzzles by completing code blocks, guided by an AI assistant that provides hints and support throughout their journey. By offering varying levels of difficulty, EscapeCode caters to both beginners and more advanced learners, making programming education accessible to all.

2.1 Bridging the Gap: The Need for Accessible Learning Tools

For individuals with physical disabilities, learning new skills, especially programming, can be a daunting challenge. Many traditional learning platforms and tools are designed with able-bodied users in mind, creating barriers for those who cannot use a keyboard or mouse.

Moreover, specialized accessibility tools, such as advanced eye-tracking devices, often come with prohibitive costs, putting them out of reach for many potential users.

This lack of inclusivity limits opportunities for individuals to gain valuable digital skills, which can improve quality of life, foster independence, and open doors to professional and personal growth. The need for accessible, affordable, and engaging educational tools has never been more pressing. EscapeCode directly addresses this gap, providing an inclusive platform that empowers users to learn programming in an interactive and enjoyable way.

2.2 Inspiration and Vision

EscapeCode draws inspiration from a variety of sources, including escape room games, educational platforms like Scratch and FreeCodeCamp, and accessibility technologies. By combining elements from these domains, the project seeks to create a learning experience that is not only engaging but also transformative.

The vision behind EscapeCode extends beyond the game itself. It aims to demonstrate how technology can be leveraged to break down barriers, making learning accessible to everyone. The project's commitment to inclusivity, innovation, and education underscores its potential to serve as a model for future developments in accessible technology.

By providing an engaging platform for learning programming, EscapeCode aspires to inspire a broader conversation about the importance of accessibility in technology. Whether it's a student discovering coding for the first time or an individual overcoming physical challenges to gain new skills, EscapeCode stands as a testament to the power of technology to transform lives and create opportunities for all.

2.3 The Problem

Unfortunately, individuals with physical disabilities often face significant challenges when it comes to learning new skills, particularly in programming and digital literacy. For many, traditional input methods

such as keyboards and mice are not viable options, which can deter them from pursuing opportunities in these fields. This exclusion not only limits their personal growth but also perpetuates a lack of diversity in the tech industry.

Beyond physical barriers, financial obstacles further exacerbate the issue. Existing accessibility tools, such as high-end eye-tracking systems or advanced speech-to-text solutions, often require specialized hardware that is prohibitively expensive for most individuals. As a result, many people with disabilities are unable to access the resources needed to develop critical skills, leaving them at a disadvantage in an increasingly digital world.

Furthermore, traditional learning platforms, while valuable, are not tailored to meet the unique needs of individuals with disabilities. For example, platforms like Scratch or FreeCodeCamp offer excellent programming tutorials but rely heavily on conventional input methods. This lack of adaptation makes it difficult for individuals with physical disabilities to engage with the content effectively, leading to frustration and exclusion.

The consequences of these challenges are far-reaching. Individuals who are unable to learn programming due to physical or financial constraints miss out on opportunities to improve their quality of life, gain independence, and unlock new career paths. Moreover, society as a whole loses out on the contributions of talented individuals who could bring unique perspectives and innovations to the tech industry.

EscapeCode aims to address these challenges by providing an accessible, engaging, and affordable platform for learning programming. By utilizing readily available hardware, such as standard webcams and microphones, and integrating innovative technologies like eye-tracking and speech-to-text, EscapeCode eliminates the need for expensive specialized equipment. The game's design prioritizes inclusivity, ensuring that individuals with physical disabilities can participate fully and enjoyably in the learning process.

By breaking down these barriers, EscapeCode empowers individuals with disabilities to develop valuable digital skills, fostering a more

inclusive and diverse tech community. The project represents a significant step toward a future where technology is accessible to all, regardless of physical or financial limitations.

2.4 Existing Solutions

There are currently technologies like eye-tracking devices or advanced speech-to-text solutions that enable users to type or perform actions without a keyboard. These technologies represent a significant step forward in accessibility, allowing users to interact with digital systems in ways that were previously impossible. However, most of these tools require specialized hardware, such as high-end cameras or sensitive microphones, which are costly and out of reach for many users. This financial barrier significantly limits their adoption and excludes individuals who could greatly benefit from these innovations.

Learning platforms like Scratch or FreeCodeCamp provide tools for learning programming, but they are not tailored for users with physical disabilities that make keyboard or mouse usage difficult. While these platforms offer engaging and effective methods for acquiring programming skills, their reliance on traditional input methods creates a significant barrier for those with physical limitations. Without alternative input mechanisms, users with disabilities may find these platforms inaccessible and frustrating to use.

Escape by CodinGame is an online platform that combines coding challenges with a fun escape room experience. Players solve puzzles by writing code to progress through levels, using multiple programming languages. While the platform is both engaging and educational, it relies exclusively on conventional input methods, such as keyboards, making it less accessible for individuals with physical disabilities. This reliance highlights the broader issue of inclusivity in digital learning tools, as many existing solutions fail to consider the diverse needs of all potential users.

Although these solutions have made strides in promoting programming education, their limitations in accessibility underscore the need for more inclusive alternatives. Tools and platforms must evolve to accommodate users with varying physical abilities, ensuring that no one is left behind in

the digital age. EscapeCode seeks to address this gap by providing an innovative, inclusive platform that removes barriers and empowers individuals with disabilities to learn programming in an engaging and accessible way.

2.5 **Project Objectives**

We plan to develop a game called EscapeCode, which integrates eye-tracking and speech-to-text technologies, designed to work with basic webcams and microphones. The game focuses on learning programming through engaging, interactive escape room challenges. Players will be able to select lines of code, move and interact with elements in the game using their eyes, and write code using voice commands. By combining these technologies, we aim to make programming education accessible to individuals with physical disabilities.

EscapeCode will go beyond merely offering accessibility; it will provide a dynamic and rewarding experience that encourages exploration and learning. The gameplay is designed to adapt to individual skill levels, ensuring that users are neither overwhelmed nor under-challenged. Players will benefit from real-time feedback and guidance from an AI assistant, which will help them progress through increasingly complex puzzles while maintaining engagement and enjoyment.

The game's design prioritizes inclusivity and affordability. By relying solely on open-source tools and readily available resources, EscapeCode ensures that the platform is accessible to a wide audience. This commitment to free and open-source technologies underscores the project's mission to eliminate financial barriers, making programming education universally attainable.

In addition to its educational value, EscapeCode aims to inspire confidence and creativity among its players. By solving puzzles and learning to code in an interactive environment, users will not only gain valuable technical skills but also a sense of achievement and empowerment. EscapeCode's ultimate objective is to demonstrate that programming is a skill that can be mastered by anyone, regardless of physical or financial limitations.

2.6 Target Audience and Stakeholders

People with Physical Disabilities:
EscapeCode will enable them to learn programming and develop digital skills comfortably and interactively, without the need for expensive or advanced equipment.

Teachers and Educational Institutions:
The solution can serve as a teaching tool that promotes equal opportunities and introduces innovative learning methods.

A General Audience Interested in Learning Programming:
Even individuals without disabilities can enjoy the game, which provides a fun and innovative approach to learning programming.

The Developer Community:
By using open-source and free tools, the project encourages further development of accessible solutions.

3 Literature Review

To build an accessible and interactive learning experience, EscapeCode leverages a combination of eye-tracking, speech-to-text, and AI technologies, all designed to work with basic hardware such as standard webcams and microphones. This section will present a comprehensive and detailed literature review of the technologies chosen for the project and how they align with the goal of creating an inclusive and engaging game for users with physical disabilities. Additionally, we will provide a comparison between the technologies currently available in the market and the specific tools we are using, highlighting their advantages, limitations, and suitability for our project's goals.

3.1 Eye-Tracking

Eye-tracking is a key feature of EscapeCode, allowing players to select parts of the code by looking at the screen

Eye-Tracking with Specialized Hardware

Currently, the market offers high-quality, advanced eye-tracking solutions that rely on specialized hardware. These tools are widely utilized in various fields such as research, gaming, and accessibility, where high precision and reliability are essential.

Tobii

Tobii is one of the leading companies in eye-tracking technology, offering a range of devices designed for both research and consumer applications. Their hardware provides highly accurate gaze detection and allows for sophisticated eye-tracking features, such as tracking multiple points of gaze and integrating with immersive virtual reality experiences. While Tobii devices are more expensive than webcam-based solutions, they offer exceptional precision and reliability, making them ideal for professional applications.

Pupil Labs

Pupil Labs provides open-source eye-tracking solutions that require specialized hardware, such as a pair of glasses or a camera system. These systems are highly customizable and are often used in research settings where accuracy and flexibility are crucial. The hardware can provide detailed eye movement tracking, which is beneficial for applications requiring high performance, such as detailed user behavior studies or real-time interactive feedback.

These specialized tools are ideal for scenarios where precision and advanced tracking features are essential but come with higher costs and complexity, making them less accessible for broader audiences. While they provide a high level of detail and accuracy, they are not necessary for EscapeCode's purpose, as we prioritize accessibility and affordability.

Eye-Tracking Without Special Hardware

To make the game accessible to a broader audience, we sought solutions that don't require any specialized hardware and instead rely on standard webcams.

OpenCV

OpenCV is an open-source computer vision library that is frequently used for eye-tracking applications. It is a free tool that can handle eye tracking using only a basic webcam. By detecting facial landmarks, OpenCV can estimate where the player is looking on the screen. It can also be integrated with Unity through additional libraries like Emgu CV, making it a flexible choice for real-time applications.

GazePointer

GazePointer is a simple, free, open-source solution for eye-tracking interactions, converting a basic webcam into a gaze-tracking device. It is straightforward to implement in Unity-based applications and offers basic eye-tracking features without the complexity of more advanced solutions.

Comparison:

While OpenCV offers a broader range of customization and more advanced capabilities, GazePointer is easier to implement and more focused on basic functionality. Depending on the specific requirements of the game, either tool can be chosen to deliver effective eye-tracking interactions. Both solutions make eye-tracking accessible without the need for specialized hardware, keeping the game affordable and user-friendly.

3.2 Turning Speech into Text

Once the player selects a line of code using their eyes, they'll complete it by speaking. This requires speech-to-text technology that's accurate, fast, and works well with standard microphones.

OpenAI Whisper

Whisper is an open-source tool that can convert speech into text with impressive accuracy. It doesn't need an internet connection, which is a big plus since it keeps everything private and works offline. Whisper is also free, making it a great fit for our project.

Vosk

Another excellent option is Vosk, a lightweight, open-source speech recognition toolkit. It's quick, works offline, and integrates easily with Unity. Both Whisper and Vosk are designed to handle speech from a variety of accents and environments, which means players won't have to worry about perfect pronunciation or noise.

We're leaning toward using Whisper because it's been tested widely and has excellent accuracy, but Vosk remains a strong alternative if we run into performance issues.

3.3 AI Bot Assistance

In EscapeCode, players aren't left to figure things out on their own. An AI bot will guide them through challenges, suggest solutions, and respond to commands like "Help" or "Write." This makes the game interactive and engaging.

Rasa

Rasa is an open-source tool for building conversational AI. What we like most about Rasa is that it runs entirely on the player's computer, so there's no need for an internet connection. It's also fully customizable, which means we can tailor it to understand specific commands related to gameplay.

ChatGPT API

Another option is the free version of ChatGPT. It's more advanced in terms of natural conversation and can provide dynamic suggestions. However, the free tier has limits, and we'll need to manage how much we rely on it to keep the experience smooth.

Rasa seems like the best fit because it's local, free, and reliable, but we might use ChatGPT for specific features requiring more depth.

3.4 Building the Game with Unity

Unity is the backbone of our project. It's a game engine that allows us to create desktop applications with ease. Unity's free personal license includes all the features we need, from 3D modeling to advanced scripting.

Unity comes with built-in features like Visual Scripting, which lets us create game mechanics without writing traditional code. This is great for designing complex interactions like gaze locking and speech-based code input. The Unity Asset Store also offers free plugins for eye tracking and speech recognition, which we'll explore to speed up development.

We chose Unity because it's user-friendly, powerful, and widely supported by free resources and tutorials.

3.5 Coding Challenges and Player Experience

At its core, EscapeCode is about solving coding puzzles in a fun and interactive way. For this, we looked at educational platforms that gamify coding to make it more engaging.

Scratch

Scratch is a free platform that teaches coding through block-based puzzles. It's simple, interactive approach is a big inspiration for how we'll design the coding challenges in EscapeCode. Players won't just write code; they'll solve problems and see immediate results, much like in Scratch.

FreeCodeCamp

FreeCodeCamp is another open-source platform with structured coding lessons. Its progressive difficulty model inspires how we'll design the game's levels, starting with easy puzzles in the tutorial and moving toward more complex challenges as the game progresses.

Escape By CodinGame

is an online platform that combines coding challenges with a fun escape room experience. Players solve puzzles by writing code to progress through levels, using multiple programming languages. While engaging and educational, it relies on traditional input methods like keyboards, making it less accessible for individuals with physical disabilities.

3.6 Why These Tools Work

Every tool we've chosen fits within our constraints: free to use, accessible on standard hardware, and compatible with Unity for desktop applications. OpenCV ensures that eye tracking works smoothly with just a basic webcam. Whisper and Vosk handle speech-to-text accurately without the need for internet. Rasa and Unity tie everything together to create an interactive experience, while platforms like Scratch inspire the game's design.

By combining these tools, we're confident we can create an engaging, accessible game that players will enjoy. Each piece of this puzzle

supports the overall vision: a fun, coding-focused escape room game that anyone can play, no matter what equipment they have.

4 Expected Achievements

The EscapeCode project aims to achieve several tangible and measurable outcomes, ensuring that the game is both functional and impactful for its intended audience. Below are the key deliverables and success criteria:

4.1 Expected Deliverables

A Fully Functional EscapeCode Game

Desktop game developed in Unity, accessible for users with physical disabilities.

Eye-tracking integration using standard webcams for interaction.

Speech-to-text functionality for coding input via voice.

AI bot assistance for guidance and in-game support.

Interactive escape room challenges with progressive coding tasks.

Tutorial and multi-level difficulty to accommodate various skill levels.

Eye-Tracking System

A software-based eye-tracking system using OpenCV or GazePointer, enabling players to select code lines using a standard webcam.

The system will be calibrated for accuracy and responsiveness, ensuring a smooth user experience.

Speech-to-Text Integration

Implementation of an offline speech-to-text system (e.g., Whisper or Vosk) to convert spoken commands into text, allowing players to write code without a keyboard.

The feature will handle diverse accents and background noise effectively.

AI Assistance Bot
An AI bot (using Rasa or a similar tool) that helps players with hints, guidance, and commands such as “Help” or “Write.”

The bot will be context-aware and able to provide tailored support based on the player’s progress.

Tutorial and Difficulty Levels
A detailed tutorial to teach players how to use the eye-tracking and speech-to-text systems.

Multiple levels of difficulty to accommodate both beginners and advanced learners.

Accessibility Features
Design elements specifically tailored for users with physical disabilities, ensuring the game is inclusive and easy to use.

4.2 Success Criteria

Functionality
The game must operate smoothly on standard PCs with basic webcams and microphones, without requiring advanced hardware.

Usability
Players with no prior programming experience or technical knowledge should be able to navigate and complete tasks within the game.

Accessibility
The game should provide a seamless experience for users with physical disabilities, particularly those who cannot use a keyboard or mouse.

Engagement
Players should find the game enjoyable and educational.

Performance
The eye-tracking and speech-to-text systems should achieve at least 90% accuracy

By delivering these features and meeting these criteria, EscapeCode will demonstrate how technology can break down barriers and provide equal opportunities for learning and growth.

5 Engineering Process

The engineering process for EscapeCode is a structured and iterative approach designed to develop a highly accessible and engaging educational platform. By leveraging cutting-edge technologies such as eye-tracking and speech-to-text, the project focuses on creating a seamless experience for users with physical disabilities. This section outlines the key steps taken in the development process, along with detailed descriptions of the tools, algorithms, and testing strategies employed to ensure a robust final product.

5.1 Process

The process of developing EscapeCode is driven by user-centered design and agile methodologies. This section provides an overview of the steps involved, including requirement analysis, design, implementation, and testing. Each phase has been carefully crafted to address the unique challenges of creating an accessible game while integrating innovative technologies efficiently.

5.2 Development Approach

Requirement Analysis:

- Identifying the specific needs of users with physical disabilities.
- Researching existing technologies and methodologies for eye-tracking and speech-to-text integration.
- Defining system requirements, user interface (UI) design, and gameplay mechanics.

Design Phase:

- Prototyping core features such as eye-tracking and speech-to-text functionality.
- Developing wireframes for the UI and overall gameplay flow.
- Planning the architecture for Unity integration and AI bot implementation.

Implementation:

- Creating modular components for seamless integration of various features.

- Iteratively coding and testing, beginning with eye-tracking and advancing to speech-to-text and AI bot interactions.

Testing and Evaluation:

- Performing unit tests for individual modules, such as OpenCV-based eye tracking.
- Conducting usability testing with users to gather actionable feedback.
- Refining features and mechanics based on test outcomes.

5.3 Constraints and Challenges

- **Hardware Accessibility:** Ensuring compatibility with standard webcams and microphones.
- **Performance Optimization:** Delivering smooth gameplay while supporting real-time interactions.
- **Budgetary Restrictions:** Relying on open-source tools and free resources to adhere to cost constraints.
- **Inclusive Design:** Catering to a diverse range of players with varying physical abilities.

5.4 Motivation for the Chosen Approach

The iterative and agile methodology was selected for its emphasis on user-centered development and adaptability. By segmenting the project into manageable phases, we can incorporate user feedback efficiently and refine features incrementally. This approach prioritizes accessibility and inclusivity, aligning with EscapeCode's mission to deliver a universally engaging experience. Additionally, agile methods facilitate the integration of innovative technologies like eye-tracking and speech-to-text without requiring extensive upfront development.

5.5 Product

The product component of EscapeCode focuses on the technical and user-experience-driven elements that bring the educational platform to life. This section describes the algorithms, models, data structures, and user interface design that underpin the game, highlighting how each component contributes to the project's accessibility and engagement goals.

5.6 Algorithms and Models

Eye-Tracking:

OpenCV

GazePointer

DeepGaze: An advanced library that uses deep learning to enhance eye-tracking accuracy. While more resource-intensive, it can improve the precision of gaze-based interactions.

Speech-to-Text:

OpenAI Whisper

Vosk

CMU Sphinx: A classic open-source speech recognition library that is efficient for basic transcription needs, though less advanced than Whisper or Vosk.

AI Bot:

Rasa

ChatGPT API

Hugging Face Transformers: Explored for generating natural language assistance to enhance the conversational experience.

6 Data Structures

6.1 Game State Management:

1. **JSON:** Used for storing game progress and puzzle states, ensuring portability and ease of integration.
2. **SQLite:** A lightweight database solution for managing player data and real-time updates efficiently.

6.2 User Input Handling:

1. **Event-Driven Pipelines:** Designed to process eye-tracking data and audio transcription in real time, ensuring seamless interaction.
2. **Queue-Based Systems:** Utilized for managing concurrent inputs from gaze and voice to maintain smooth gameplay.

6.3 User Interface

1. **Visually Intuitive Layout:** Large, clearly marked interactive areas tailored for ease of use.
2. **Tutorial Mode:** A comprehensive guide introducing players to eye-tracking and speech-to-text mechanics, ensuring a smooth onboarding experience.
3. **AI Bot Interface:** Context-sensitive suggestions and hints displayed in a user-friendly manner, fostering engagement and reducing frustration.
4. **Accessibility Options:** Customizable settings for font size, contrast, and input sensitivity to accommodate diverse user needs.

By combining these elements, the product design ensures that EscapeCode delivers a highly accessible, engaging, and educational experience for all players, regardless of physical limitations.

7 Testing Plan

The testing plan for EscapeCode is a comprehensive framework designed to ensure the reliability, accessibility, and overall quality of the game. This section details the strategies, constraints, and environment considerations for thoroughly validating all aspects of the platform.

7.1 Testing Strategy

Unit Testing:

Each individual component, such as gaze-locking mechanisms, speech-to-text conversion, and AI bot responses, will undergo rigorous testing to validate their standalone functionality. This includes verifying the accuracy of eye-tracking calibration, speech recognition under different acoustic conditions, and AI responses to a variety of user commands.

Integration Testing:

Integration testing ensures seamless interactions between core components, such as the Unity game engine, the AI bot, and the input mechanisms. For example, real-world scenarios will be simulated where players select code lines using their eyes and then complete those lines via voice commands, ensuring smooth transitions and compatibility between systems.

Usability Testing:

This critical phase involves engaging participants from the target audience, including individuals with physical disabilities, to assess the game's accessibility, ease of navigation, and overall user experience. Feedback will be gathered to refine the interface and optimize features for usability.

Performance Testing:

Metrics such as frame rates, input latency, memory usage, and overall system resource consumption will be measured to ensure the game operates smoothly on mid-range PCs. Stress testing will also be conducted to evaluate how the system performs under heavy

computational loads, such as simultaneous eye-tracking and speech-to-text processing.

7.2 Constraints and Assumptions

- Testing will focus on mid-range PCs equipped with standard webcams and microphones, as these represent the target hardware for the game.
- Open-source libraries, such as OpenCV and Whisper, will be assumed to function as documented, with community support available for troubleshooting and optimization.
- Accessibility standards will guide the testing criteria to ensure the game meets the needs of users with a variety of physical disabilities.

7.3 Testing Environment

- Development and testing will be conducted on Unity-compatible systems to ensure all features function as intended within the chosen game engine.
- Simulated environments will replicate real-world scenarios, such as varying lighting conditions for webcam-based eye tracking and different acoustic conditions for speech-to-text inputs.
- Controlled testing of tools like OpenCV, Whisper, and Rasa will verify their compatibility with Unity and the specific requirements of EscapeCode.
- Cross-platform testing will be carried out to confirm the game operates consistently across different operating systems and hardware configurations.

7.4 Additional Testing Considerations

- **Localization Testing**: Ensuring that the speech-to-text system supports multiple languages and accents, allowing for a global user base.
- **Error Recovery Testing**: Simulating errors such as incorrect speech recognition or unresponsive eye-tracking to evaluate the robustness of fallback mechanisms.

- **Regression Testing**: Re-running previous tests after each update to confirm that new changes do not introduce bugs or break existing functionality.
- **Accessibility Compliance Testing**: Comparing the game's features against established standards such as WCAG (Web Content Accessibility Guidelines) to ensure an inclusive experience for all users.
- By employing this detailed testing plan, EscapeCode aims to deliver a polished, reliable, and inclusive gaming experience that exceeds user expectations while adhering to high standards of quality and accessibility.

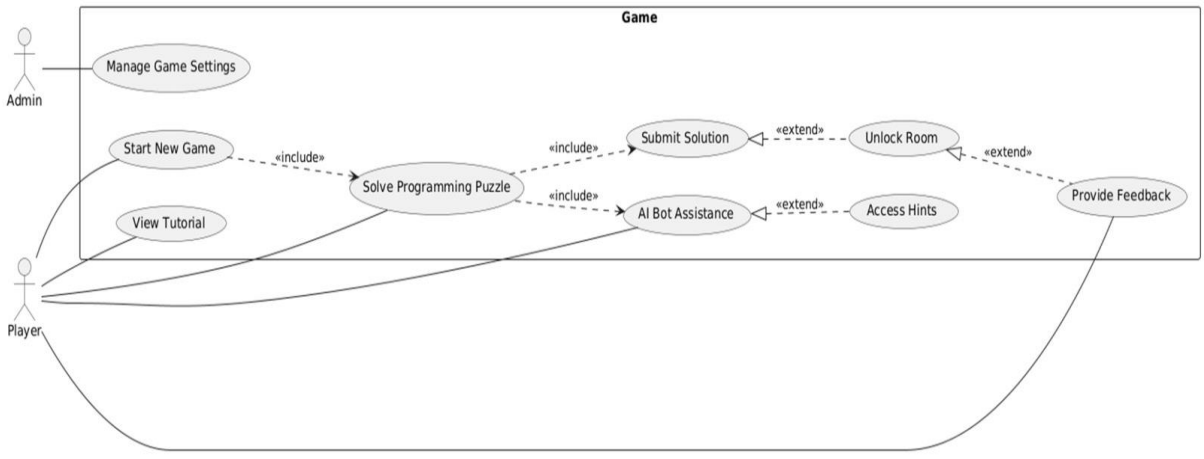
8 Project Design

The design of this project focuses on creating an interactive puzzle-solving game that incorporates elements of programming challenges and room unlocking mechanics. The system is structured to include various roles, such as the Player and Admin, and aims to provide an engaging learning experience. The architecture leverages multiple design patterns to ensure that the game is both flexible and scalable. Various UML diagrams are employed to visually represent the system, detailing interactions, object relationships, and state transitions throughout the game.

8.1 Use Case Diagram

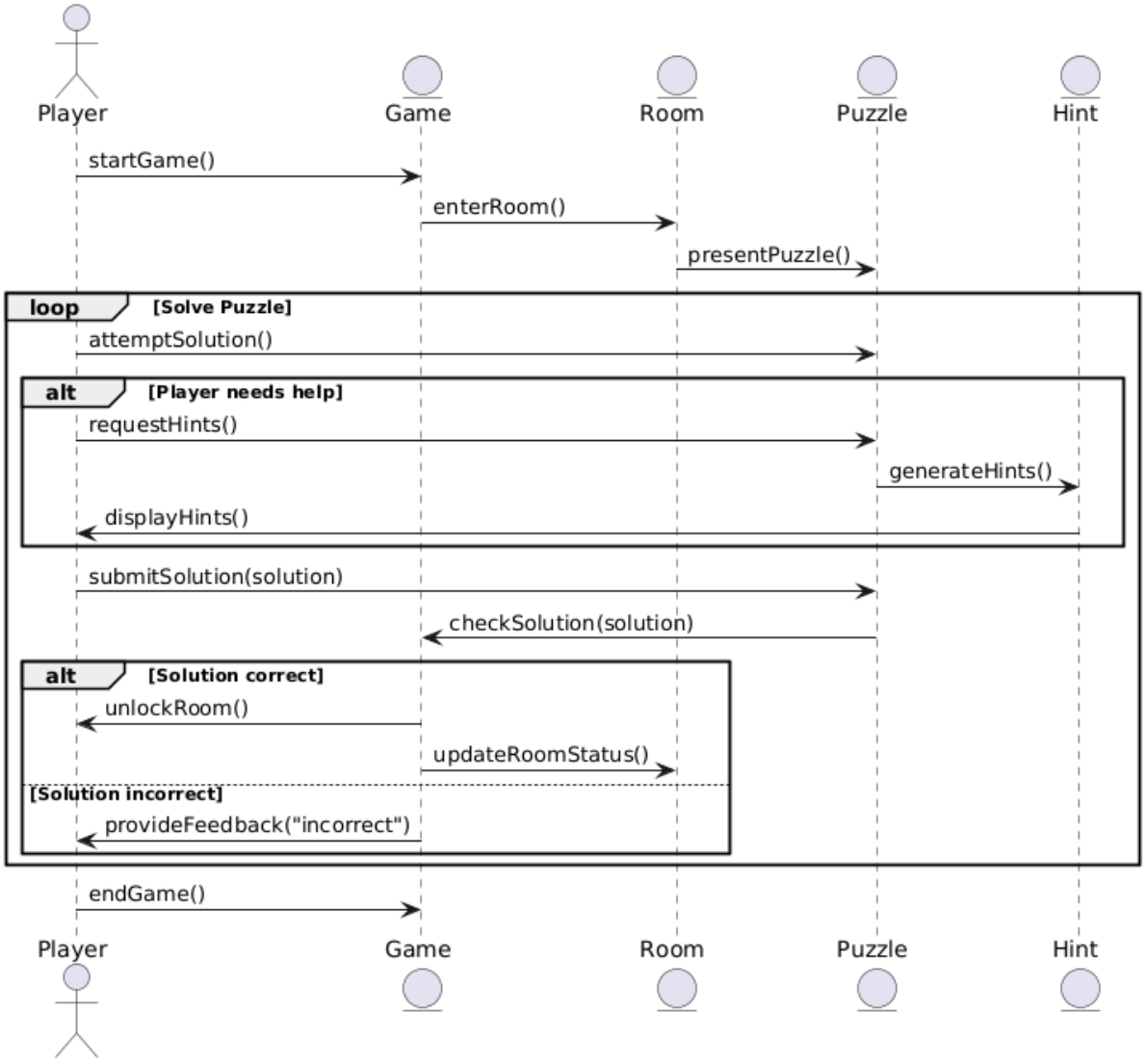
The Use Case Diagram illustrates the high-level functionalities of the system and the interactions between different actors, namely the Player and Admin. The Player is able to engage in activities such as solving puzzles, requesting hints, submitting solutions, unlocking rooms, and providing feedback.

The Admin has control over managing game settings and starting new games. Several use cases, such as "Solve Programming Puzzle" and "Unlock Room," demonstrate core actions, with some extending or including other use cases to reflect more complex behaviors.



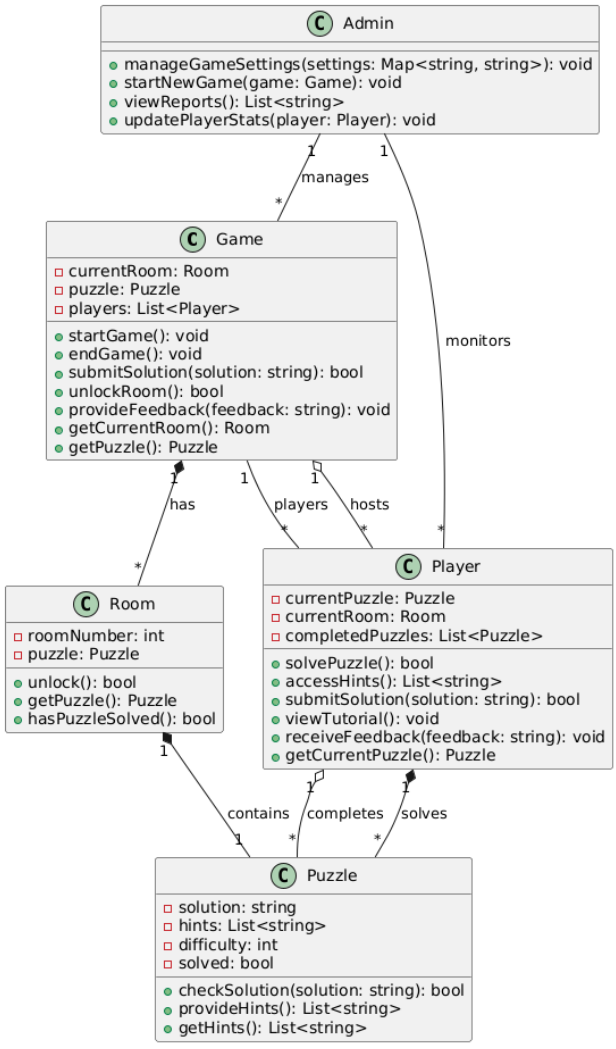
8.2 Sequence Diagram

The Sequence Diagram outlines the flow of interactions between the Player, Game, Room, Puzzle, and Hint entities. Starting from the Player initiating the game, it shows the process of entering rooms, solving puzzles, and interacting with hints. The diagram also includes feedback loops for incorrect solutions and progresses the game state accordingly. This detailed sequence helps visualize the specific order of operations required to solve puzzles and advance in the game.



8.3 Class Diagram

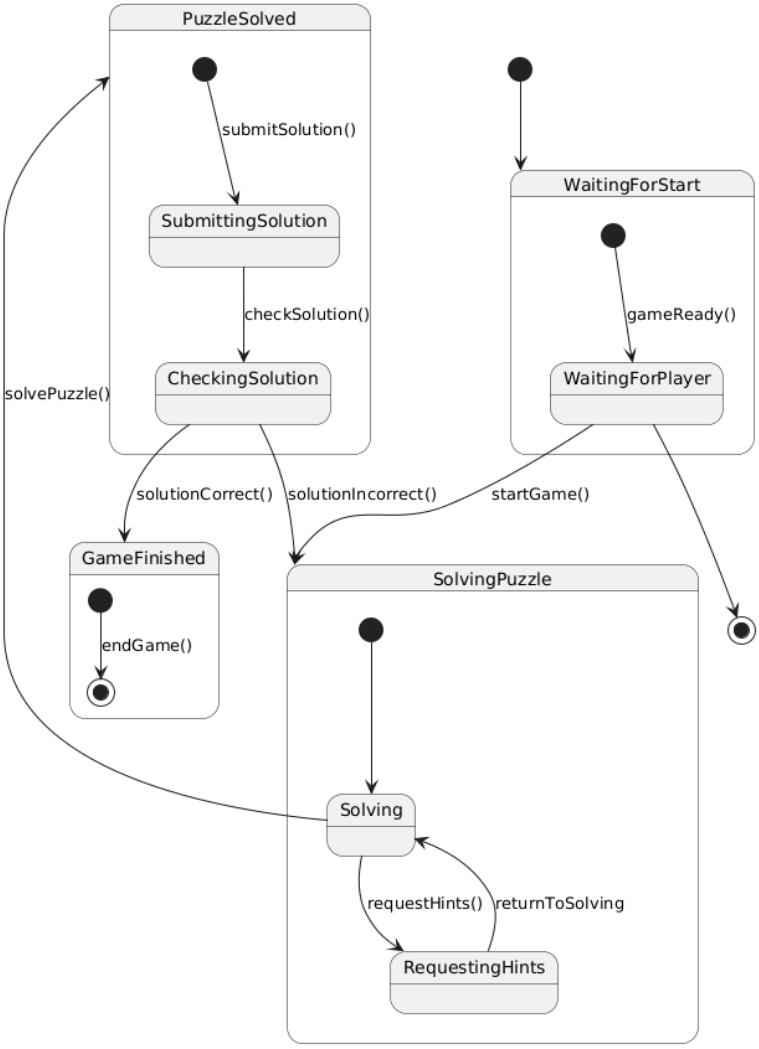
The Class Diagram depicts the underlying object structure of the game, detailing the classes and their relationships. The Game class manages game flow, hosts players, and contains rooms and puzzles. The Player class interacts with puzzles, receives feedback, and completes challenges. The Room class manages puzzle availability, while the Puzzle class checks solutions and provides hints. The Admin class is responsible for managing settings, viewing reports, and updating player statistics. The diagram also highlights key attributes and methods, along with the relationships between these entities.



8.4 State Diagram

The State Diagram represents the different stages of a player's progression through the game. Initially, the game waits for the player to start. Once the player begins, the state transitions to "SolvingPuzzle," where the player can attempt to solve puzzles or request hints. If the puzzle is solved, the state changes to "PuzzleSolved." If the solution is incorrect, the player is prompted to continue solving.

After a correct solution, the game enters the "GameFinished" state. This diagram highlights the key states and transitions that govern the player's interaction with the game.



9 Conclusion and Benefits

EscapeCode represents a transformative approach to learning programming, addressing the critical need for accessible and inclusive tools. The project offers the following key benefits:

9.1 Enhanced Accessibility:

- Empowers individuals with physical disabilities to learn programming using eye-tracking and speech-to-text technologies.
- Removes the reliance on traditional input methods like keyboards and mice.

9.2 Inclusive Design:

- Provides a platform that is easy to use for everyone, regardless of physical abilities.

9.3 Fun and Engaging Learning Experience:

- Combines escape room challenges with coding tasks, making programming both enjoyable and educational.

9.4 Free and Open-Source Tools:

- Ensures affordability and accessibility by using only free resources and open-source technologies.

9.5 Broader Educational Impact:

- Serves as a valuable tool for schools and educational institutions to promote equal learning opportunities.

9.6 Innovation and Inspiration:

- Demonstrates how technology can break down barriers, inspiring further development of inclusive solutions in the tech community.

In meeting these goals, EscapeCode aligns with the broader vision of using technology to empower individuals and create opportunities for all. Whether it's a student discovering programming for the first time or someone overcoming physical challenges to learn a new skill, this project stands as a testament to the potential of inclusive design.

10 Game Interface Mockups