



Software Engineering Department
ORT Braude College

Capstone Project Phase A – 61998

House of Riddles using ML

24-1-D-42

Supervisor:

PhD. Moshe Sulamy

MosheSu@braude.ac.il

Students:

Seren Hanany- seren.hanany@e.braude.ac.il

Maisalon Safory- maisalon.safory@e.braude.ac.il

GIT:

<https://github.com/Maisalon/Final-Project>

Abstract

The proposed game "House of Riddles using ML" offers players a multi-faceted experience designed to exercise their brains and foster cognitive stimulation.

The game features a sophisticated system comprising multiple puzzles secured within a locked frame. Players engage in a race to unlock all cells and solve all puzzles ahead of their friends. Moreover, there's a solo mode available, where players can tackle each puzzle within a designated time limit. Upon entering the game, each player selects their preferred mode solo or with friends and the game commences with each player's house equipped with locks. Players must decipher the puzzles to unlock the locks, and the first player to solve all their puzzles emerges victorious.

Artificial intelligence enhances the gaming experience by providing personalized recommendations, subtle hints and dynamic difficulty adjustment to keep players interested and engaged. The integration of ML within the game not only helps improve the player's aim but also provides an engaging method for learning.

The goal is to offer a gaming experience that adds value by stimulating cognitive activity and providing a more engaging alternative to traditional idle games.

Table of Contents

1. Introduction	4
2. Literature review	5
3. Expected Achievements	7
3.1 Outcomes	7
3.2 Unique Features	7
3.2.1 ML - Player behavior analysis	7
3.2.2 Unity Engine – Modeling a 3D environment	8
3.2.3 Personalized Player Experience	8
3.3 Criteria for Success	9
4. Engineering Process	9
4.1 Research- Unity and game development	9
4.1.1 Constraints and Challenges	10
4.1.2 Conclusions from Research	10
4.2 Research- Machine Learning	10
4.2.1 Constraints and Challenges - Machine Learning	11
4.3 Methodology and Development Process	12
5. Product	13
5.1 Requirements	13
5.2 Architecture overview	14
5.2.1 Machine learning Application	15
5.2.2 Scenes	16
Player 2:	20
Player 3:	21
5.3 Diagrams	22
5.3.1 Use Case	22
5.3.2 Activity Diagram	23
5.3.3 Class Diagram	23
6. Verification and Evaluation	24
6.1 Evaluation	24
6.2 Verification	25
7. References	27

1. Introduction

Our project introduces a puzzle game developed in Unity employing machine learning techniques, designed to engage players' cognitive faculties and adapt challenges to their behaviors and preferences. It features an advanced framework with several puzzles that are not accessible until they are unlocked, where players aim to solve them faster than their friends. Using machine learning, the game offers personalized recommendations, subtle hints and dynamic difficulty adjustments to keep players interested and engaged. Our goal is to enrich the gaming experience while addressing common challenges encountered in puzzle solving games, including fixed difficulty levels, the need for personalized experiences, overcoming difficult puzzles and ensuring player engagement and retention, thus providing a more satisfying gaming experience overall.

Machine learning (ML) algorithms analyze player behavior and preferences, adjusting the challenges dynamically to ensure an appropriate level of difficulty. This solves the problem of fixed difficulty levels by providing a personalized and engaging experience. ML also enables personalized recommendations and hints based on player profiles, addressing the need for a tailored experience and enhancing player connection to the game. Additionally, ML algorithms can detect patterns indicating player frustration or difficulty, offering subtle hints and assistance to help players overcome obstacles and progress smoothly. By continuously analyzing engagement metrics, ML optimizes gameplay elements to maximize player engagement and retention, ultimately leading to increased satisfaction and long-term player commitment. ML's ability to adapt, personalize, and optimize the gaming experience makes it an effective solution to the challenges faced in puzzle-solving games.

Furthermore, there are different types of players that are the main ones in the game, each with unique characteristics, preferences and play styles. Some players may be casual gamers who enjoy solving puzzles in their free time without seeking intense challenges. These players prefer relaxed gameplay with simple puzzles and may appreciate gentle hints and guidance to help them progress smoothly. On the other hand, there are hardcore gamers who thrive on difficult challenges and complex puzzles that test their skills and problem-solving abilities. These players may prefer minimal assistance and enjoy the satisfaction of overcoming tough obstacles on their own. Additionally, there are completionists who aim to fully explore and master every aspect of the game, strictly solving every puzzle and uncovering every secret. They may appreciate additional content, such as bonus levels or hidden challenges, to extend their gameplay experience. By recognizing and accommodating these different player types, your game can offer a more

inclusive and enjoyable experience for all players, regardless of their preferences and skill levels.

To address these challenges, our project uses Unity and machine learning to create a dynamic puzzle game that tests players' cognitive abilities while adapting to their preferences and behaviors. With a variety of puzzles locked in a secure frame, players compete to unlock and solve them faster than the other players. Machine learning algorithms offer tailored hints and recommendations, evolving the game and keeping players interested. By addressing common obstacles such as fixed difficulty levels and the need for personalized experiences, our game ensures a satisfying gaming experience and stimulates cognitive activity and offers a fascinating alternative to traditional idle games.

2. Literature review

Puzzle games like Lumosity and Duolingo combine entertainment with education, helping users develop cognitive and language skills. Lumosity users report a 97% improvement in mental sharpness, while Duolingo can equate 34¹ hours of use to a semester of university-level language learning. These games demonstrate the effective blend of fun and learning on digital platforms.

A new generation of puzzle games is leveraging machine learning (ML) to significantly enhance the gaming experience and educational benefits². These ML-driven games adapt dynamically to the player's skill level, providing personalized challenges that evolve based on the player's progress. One such example is "SkillCraft," which uses advanced algorithms to analyze player performance and adjust the difficulty in real-time, ensuring an optimal balance of challenge and learning. By continuously adapting to the user's abilities, these games not only keep players engaged but also promote more effective learning and skill development.

Early studies indicate that puzzle games enhanced with machine learning (ML) show promising results in improving players' learning outcomes and skill retention³. ML techniques enable these games to dynamically adapt their difficulty based on the player's performance, providing personalized challenges that enhance engagement and learning efficiency. This adaptive approach contrasts with traditional static puzzle games, which often fail to sustain player interest and learning progression at the same rate. Research highlights that ML-enhanced games not only help players improve faster but also retain their acquired skills longer, demonstrating a significant advancement in the field of educational gaming.

Now you will discuss how ML contributed to the development of games. Reinforcement Learning (RL) has been extensively used to train Non-Player Characters (NPCs) for intelligent behavior. Examples include teaching agents

¹ https://www.languagezen.com/pt/about/english/Duolingo_Efficacy_Study.pdf

² <https://www.mdpi.com/2078-2489/12/4/161>

³ <https://www.mdpi.com/2078-2489/12/4/161>

to navigate mazes or solve puzzles autonomously. A notable example is the use of RL in Unity's ML-Agents Toolkit, which provides a platform for training intelligent agents within the Unity environment⁴.

Unity is a leading game development engine known for its user-friendly interface, cross-platform compatibility, extensive asset store, and strong community support. Developers prefer Unity for its rapid prototyping capabilities, seamless deployment across multiple devices, and access to a wealth of ready-made assets and tools, making it a top choice for creating games of all sizes and genres.

Integrating machine learning into Unity for puzzle-solving games opens up a wealth of possibilities for creating intelligent, adaptive, and engaging experiences. By leveraging various ML techniques such as reinforcement learning, supervised and unsupervised learning, and evolutionary algorithms, developers can craft games that not only challenge players but also adapt to their unique play styles and preferences. The Unity ML-Agents Toolkit and other ML frameworks provide the necessary tools to bring these advanced game mechanics to life.

⁴ <https://arxiv.org/abs/1809.02627>

3. Expected Achievements

3.1 Outcomes

Our project aims to advance exposure to puzzle-solving and adapt the learning process to suit diverse learners, encompassing individuals from various fields and ages. This entails crafting a wide array of puzzles that challenge problem-solving skills in diverse ways, catering to different learning preferences and cognitive styles. To achieve this, we'll integrate features like adjustable difficulty levels, hint systems, and multiple progression paths, accommodating players with varying skill levels and learning needs⁵. Our design will prioritize inclusivity, ensuring clarity in game mechanics, instructions, and feedback for players of diverse backgrounds, including those with disabilities or language barriers. Additionally, we'll seamlessly embed educational content into the game environment, offering opportunities for players to learn and apply knowledge while solving puzzles. By focusing on these achievements, our game will not only deliver an engaging gaming experience but also serve as a valuable educational resource accessible to learners of all ages and backgrounds.

3.2 Unique Features

3.2.1 ML - Player behavior analysis

Machine learning enhances player behavior analysis in the game by providing personalized challenges, dynamically adjusting difficulty levels, offering adaptive hints, and enabling predictive analytics⁶. Through continuous monitoring of player performance and preferences, machine learning algorithms tailor challenges to each player's skill level and gameplay style, ensuring an engaging experience. The system adapts the difficulty of puzzles based on player progress, offers subtle

⁵https://www.researchgate.net/profile/Jeffrey-Craighead/publication/265284198_Using_the_Unity_Game_Engine_to_Develop_SARGE_A_Case_Study/links/55d33fdf08aec1b0429f32f4/Using-the-Unity-Game-Engine-to-Develop-SARGE-A-Case-Study.pdf

⁶https://www.theseus.fi/bitstream/handle/10024/496604/Breugelmans_Arno.pdf?sequence=2&isAllowed=y

hints when needed, and predicts future behavior to anticipate player needs, ultimately enhancing the overall gaming experience.

3.2.2 Unity Engine – Modeling a 3D environment

Creating a captivating 3D environment is pivotal to our game's success. We must carefully choose the setting that best compliments our post-apocalyptic narrative and meticulously craft it using Unity, paying close attention to lighting, object placement, and scale to enhance immersion⁷. Moreover, optimizing our environment to run smoothly within hardware constraints are crucial. Balancing the number of rendered objects to maintain a stable framerate is essential to prevent user discomfort like nausea or motion sickness, ensuring an enjoyable gaming experience.

3.2.3 Personalized Player Experience

The integration of machine learning algorithms to analyze player behavior and preferences is another key achievement. This system should tailor challenges, provide hints, and adjust gameplay elements to suit individual players, enhancing immersion and satisfaction.

3.2.4 Improve mental skills

Creating a dynamic puzzle game that not only activates the mind but also adjusts its challenges based on player behavior and preferences. The game features a complex system with multiple puzzles locked behind a secured frame, and players compete to open all cells and solve all puzzles before other players.

⁷<https://isprs-annals.copernicus.org/articles/IV-4-W4/161/2017/isprs-annals-IV-4-W4-161-2017.pdf>

3.3 Criteria for Success

- Clear Guidance for Flexible Gameplay: Helping Players Understand and Adapt to the Game.
- A distinct game that can be used for learning while being entertained.
- Creating an intuitive and reliable 3D/2D scene.
- Continue playing incentive: earn houses for each win, accumulating points to help progress in future puzzles.
- We aim to leverage machine learning to ensure optimal functionality, while providing timely and engaging prompts that sustain players' enthusiasm throughout the game experience.

4.Engineering Process

4.1 Research- Unity and game development

Regarding expanding our understanding of the game, we focused on answering the following questions:

- How does the game affect the expansion of the knowledge of the players?
- Who are the interested parties?
- Which age group is most eager to play?
- What bottlenecks exist today in providing rapid screening and assessment for our game?

To tackle these questions and expand our knowledge, we used various resources, from scientific papers and articles to videos. After going through the available resources, we met to discuss our findings and share the main points we should focus on when developing the app. Some of the conclusions we reached while reading about puzzle games when it comes to software development, is that the puzzle needs to be adjusted to the age of the player himself so that he gets excited about the game and continues to play in the future. Therefore, the suitability of puzzle games for a certain age group depends on the individual's cognitive development, interests and preferences. Some people may enjoy puzzles from a very young age, while others may not find them engaging until later in life. The key is to find puzzles that match the skill level of the player and provide an appropriate level of challenge and enjoyment.

4.1.1 Constraints and Challenges

Learning how to use Unity presents several constraints and challenges, primarily due to the platform's complexity and expansive feature set. Novice developers may find themselves overwhelmed by the multitude of tools, functions, and scripting languages on Unity.

Navigating this learning curve requires dedication, patience, and a significant investment of time to understand the fundamentals of game development and Unity's specific workflows. Additionally, keeping pace with Unity's frequent updates and evolving technologies can pose a challenge, as developers must continuously adapt their skills and workflows to leverage new features effectively. Moreover, problem solving issues and debugging errors within Unity projects can be arduous, particularly for those without prior programming experience. Despite these challenges, overcoming the learning curve in Unity can lead to rewarding opportunities for creative expression and innovation in game development.

4.1.2 Conclusions from Research

The conclusions are that the suitability of puzzle games for a certain age group depends on the individual's cognitive development, interests and preferences. Some people may enjoy puzzles from a very young age, while others may not find them engaging until later in life. The key is to find puzzles that match the skill level of the player and provide an appropriate level of challenge and enjoyment.

4.2 Research- Machine Learning

As we delved into the world of machine learning algorithms, we realized that there are many options available, each tailored to different tasks and problem areas. However, as our journey progressed, we found ourselves drawn into the realm of reinforcement learning. This decision was partly influenced by the Unity Machine Learning Agents (ML-Agents) toolset, which prioritizes reinforcement learning algorithms. With its seamless integration into Unity projects and support for various reinforcement learning approaches. So the agents are modeled in the environment and receive rewards based on their actions⁸.

8

https://books.google.co.il/books?hl=ar&lr=&id=OMNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Reinforcement+Learning+Algorithms+with+unity+game&ots=CkBONy-A92&sig=OHdCTANtyNth_N5-Auc_sduxZq8&redir_esc=y#v=onepage&q=Reinforcement%20Learning%20Algorithms%20with%20unity%20game&f=false

4.2.1 Constraints and Challenges - Machine Learning

- **Integrating Unity:** Understanding how to integrate machine learning algorithms within Unity. This integration can be a challenge for new users. This includes installing and configuring the appropriate plugins, understanding how to interface with Unity's game objects and environments, and managing the interaction between machine learning agents and the Unity environment. For example, it was difficult for us to test the ML infrastructure within Unity and understand how it works with the puzzles.
- **Unity ML-Agents:** While Unity ML-Agents simplifies the process of integrating machine learning into Unity projects, learning how to use the ML-Agents toolkit effectively still requires familiarity with reinforcement learning concepts and algorithms. Understanding how to configure and train agents, design reward functions, and interpret training results are important skills to develop.
- **Performance Optimization:** Ensuring that machine learning algorithms run efficiently within the Unity runtime environment is essential, especially for real-time applications such as games. Optimizing compute performance, memory usage, and runtime efficiency while maintaining smooth gameplay can be a challenging task.
- **Unity-specific challenges:** The Unity development environment may pose its own unique challenges when working with machine learning algorithms, such as compatibility issues with specific versions of Unity.

4.3 Methodology and Development Process

For our development approach, we opted for the Agile methodology, which we believe aligns well with our specific needs. By adopting an iterative approach, we can break down our feature delivery into manageable components, allowing for maximum adaptability to changes. Our development process will be segmented into:

- Building the scene using the Unity 2D/3D engine.
- Adding tracking and movement in the scene.
- Added option to store user data.
- Checking the correlations of our data with questionnaires.
- Applying changes to any problem that arises as a result of the results.
- Creating a web-based interface that will be presented to the screening initiators.

After each iteration, we will conduct an evaluation to assess our progress, and make any necessary adjustments before starting the next cycle. Our priority is to provide functional software that incorporates input from users who will participate in testing our product.

5.Product

5.1 Requirements

Functional

1	The system should choose puzzles dynamically, adjusting the complexity based on player behavior and preferences.
2	The system should track players' progress, including completed puzzles and overall game achievements.
3	The system should analyze player behavior to provide personalized recommendations and hints.
4	The system stores data in a database
5	The system should have a secure mechanism for locking and unlocking puzzles to ensure fair gameplay.
6	The system should support multiplayer functionality, allowing players to compete against each other in real-time or asynchronously.
7	The system should collect and analyze data on player interactions, puzzle completion rates, and overall engagement.

Non-functional

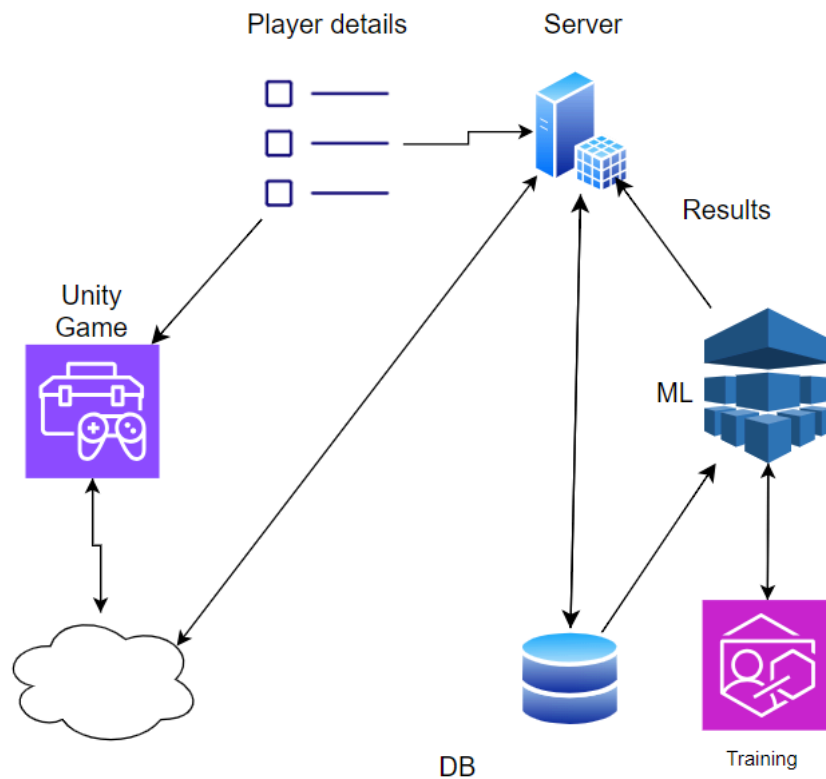
1	Easy to use graphical interface for the user.
2	Players should be able to navigate between different puzzles, levels and game features with ease.
3	Different types of puzzles should be available, offering variety and challenging gameplay.
4	The difficulty of puzzles should be dynamically adjusted based on player performance to maintain engagement.
5	Access to new puzzles should be gated behind previous achievements or challenges.
6	Reports and analytics should be available to help developers understand player behavior and improve the game experience over time.

5.2 Architecture overview

Our architecture consists of several key components:

- **Input:** Receives data from the environment.
- **Preprocessing:** Cleans and formats data for learning.
- **Neural Network:** Core component for pattern recognition.
- **Activation Functions:** Introduces non-linearity for complex learning.
- **Output:** Produces actions or predictions.
- **Loss Function:** Measures performance during training.
- **Optimizer:** Adjusts model parameters to minimize loss.
- **Training Pipeline:** Process for data collection, training, and evaluation.
- **Memory and State:** Retains information for decision-making.
- **Deployment:** Putting the trained model into action for real-time use.

5.2.1 Machine learning Application



Our application will be using the MVC architecture in the following method:

Model

The database will receive data like distance calculations and user details from the Controller. Its main role is to store this information securely.

Controller

The controller layer will dynamically handle scene creation and management based on user interactions, utilizing Unity3D/2D Engine for tasks like texture rendering, lighting computations, and object

manipulation. It will also interact with the Model layer to store any necessary data.

View

The View layer displays the user interface, including navigation buttons, players, and objects. Users interact with this interface, and the Controller manages the input and logic for the scene accordingly.

5.2.2 Scenes

We create a variety of social situations for users to experience, from pleasant and familiar settings such as peaceful homes to more challenging environments such as streets. Our goal is to offer a wide variety of scenarios that help users build confidence and adaptability in different social contexts.

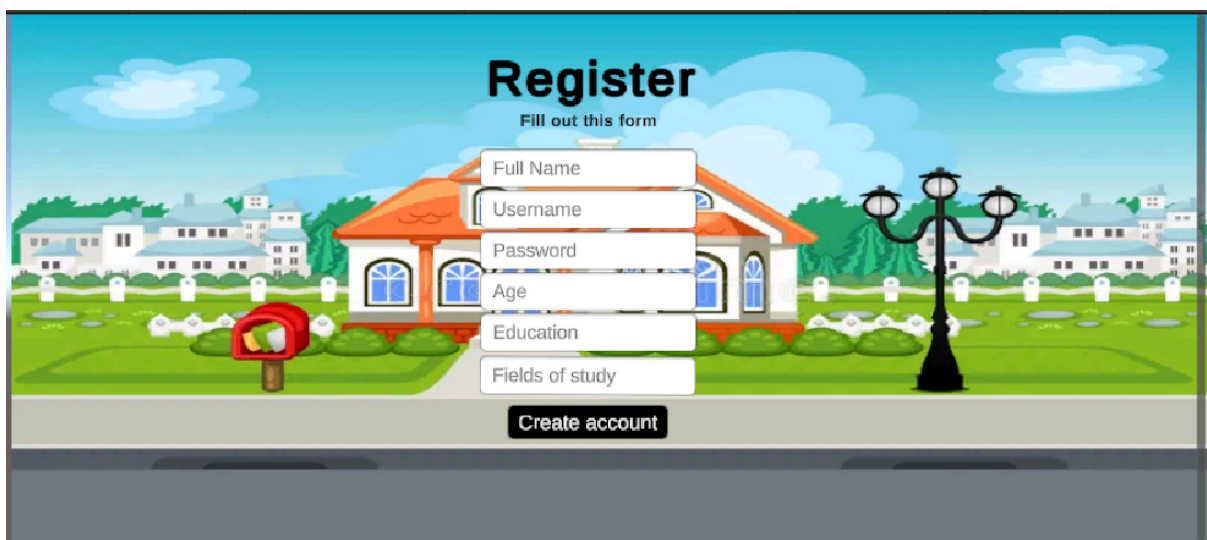
Login screen:

On installing the app, users are faced with the initial screen. Returning users can log in by entering their username and password. If they forget their password, they can reset it using the "Forgot Password" option. New users simply click "Register Now" and provide basic information to complete the registration process. With the login feature, any player can conveniently access the game using their saved information.



Registration screen:

When a player embarks on their first gaming session, they'll register by inputting personal information like their full name, username and password to establish an account. Furthermore, they'll provide details about their age, education, and specific areas of interest. This ensures that they are matched with puzzles that align with their preferences and engage their curiosity.



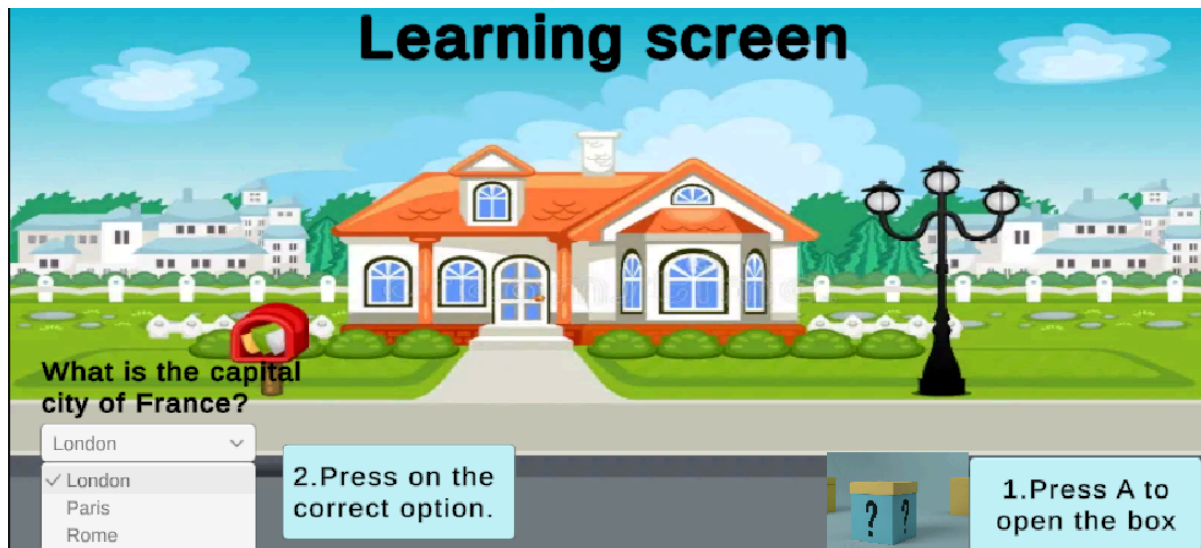
practice prompt:

Once the user inputs their name and age, they'll encounter a screen prompting them to choose whether they'd like to participate in a practice session. This option is particularly encouraged for novice users utilizing Unity.



character walking towards the user:

During the practice session, users' input won't be stored in their accounts, distinguishing it from the simulation where all user actions are saved in our database. Users will encounter a figure that approaches and then retreats. They'll be prompted to halt the figure at a comfortable personal space distance. Additionally, they'll find boxes with instructions to open one and solve the puzzle inside.

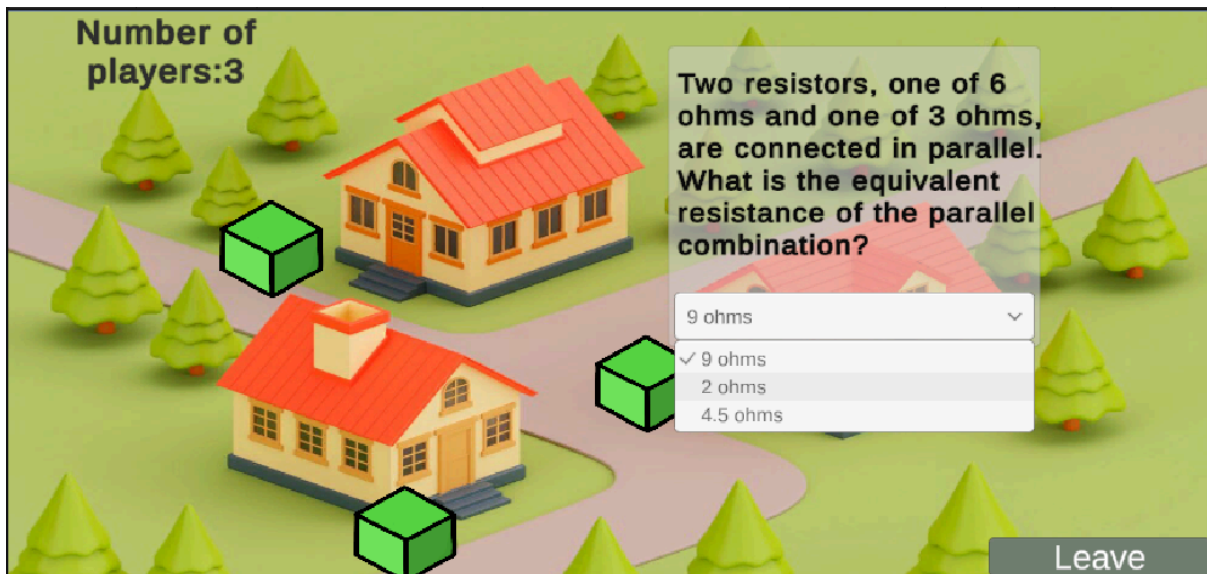


Start Play Screen:

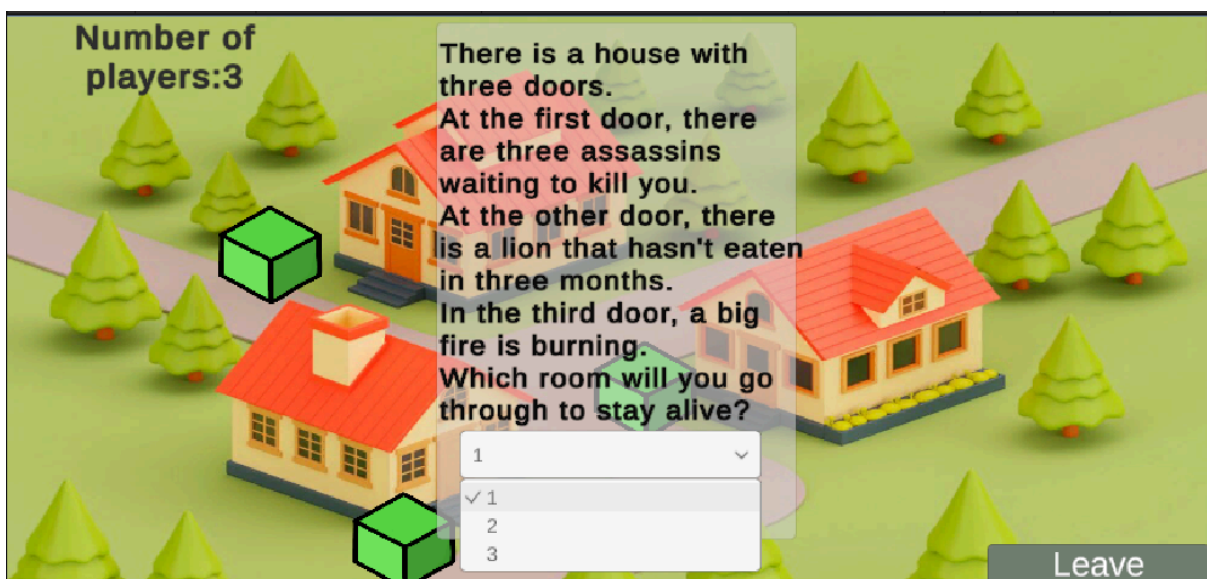
As players commence their gameplay, their progress and data are securely stored in the database. They find themselves in a neighborhood where each house corresponds to a player and features a unique lock and prize. Players endeavor to unlock each house and solve the puzzles within. The first player to finish all puzzles wins the respective house. Each player is assigned puzzle difficulty levels based on the number of houses they've won, ensuring a balanced challenge for all participants.



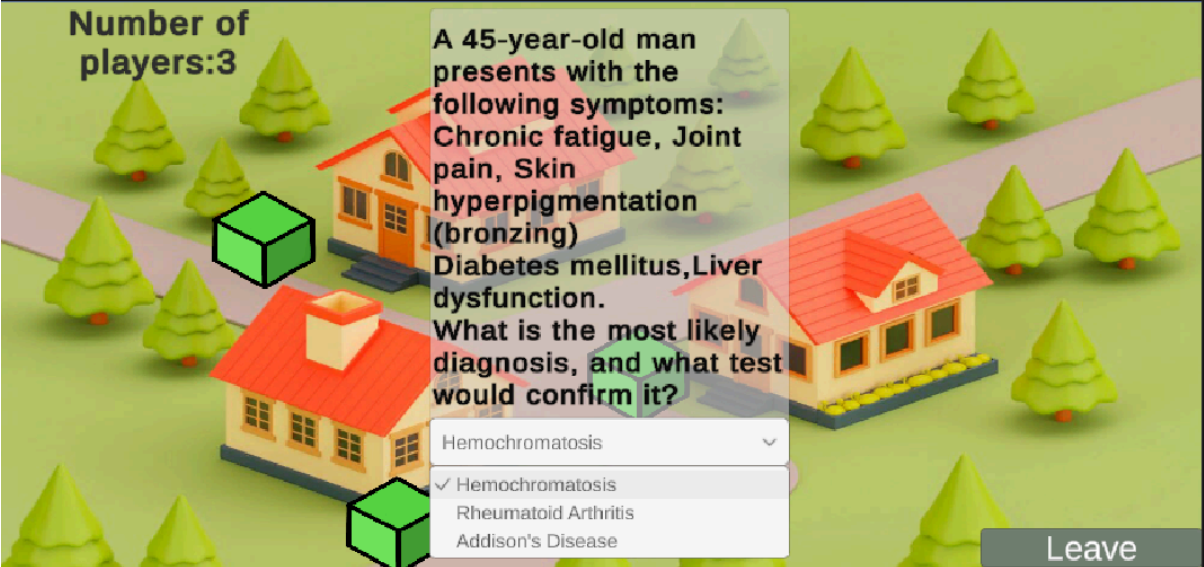
Player 1:



Player 2:



Player 3:



Number of players:3

A 45-year-old man presents with the following symptoms:
Chronic fatigue, Joint pain, Skin hyperpigmentation (bronzing)
Diabetes mellitus, Liver dysfunction.
What is the most likely diagnosis, and what test would confirm it?

Hemochromatosis

✓ Hemochromatosis

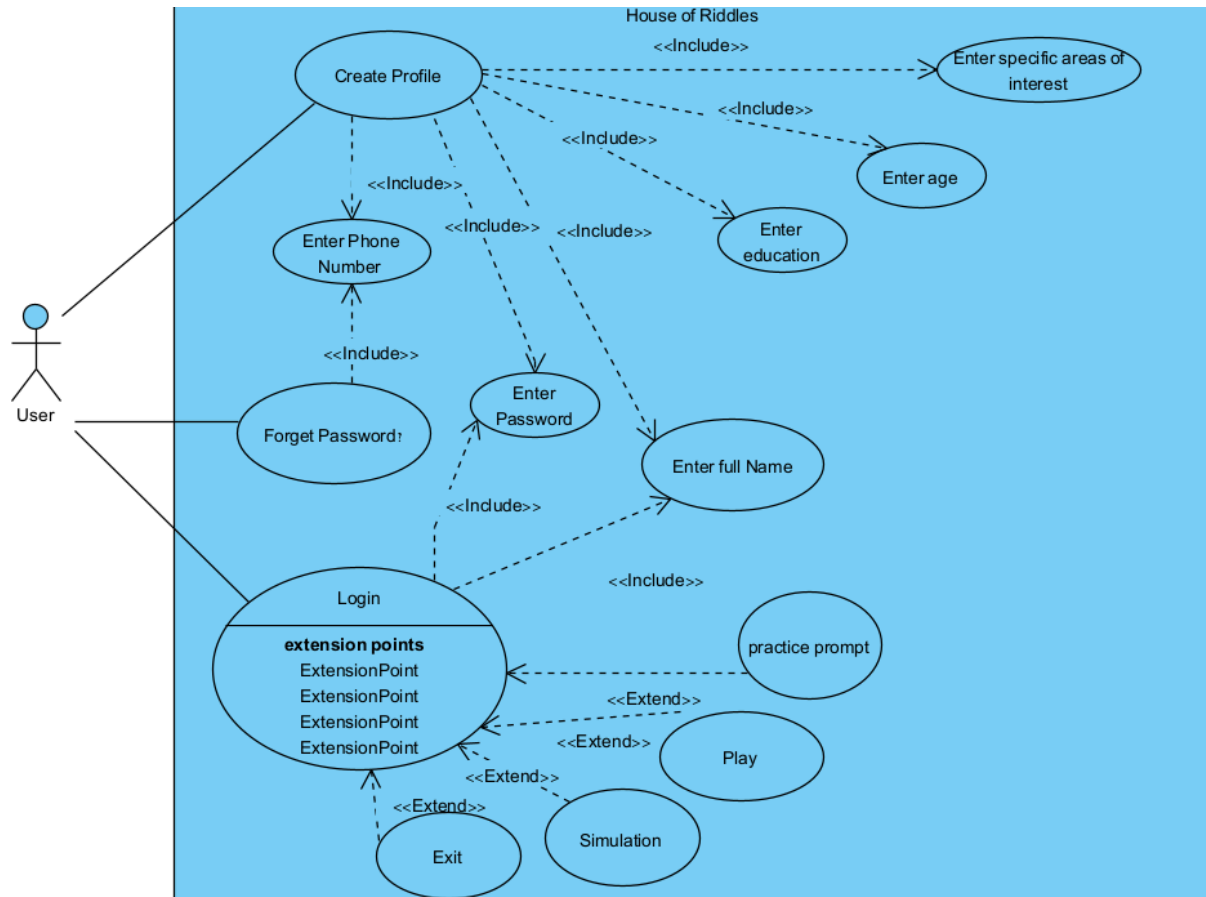
Rheumatoid Arthritis

Addison's Disease

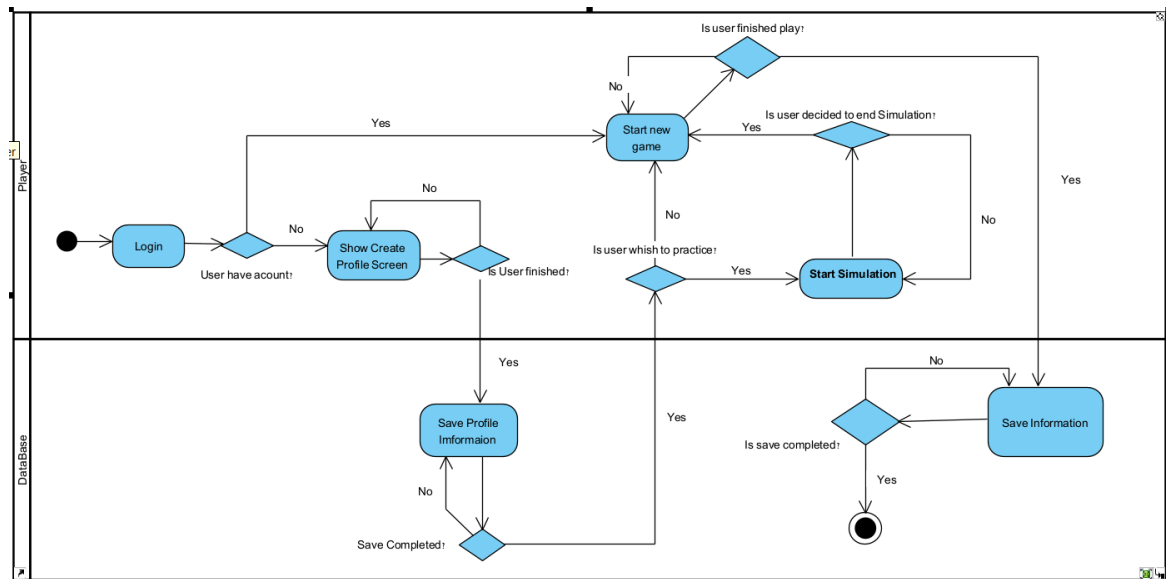
Leave

5.3 Diagrams

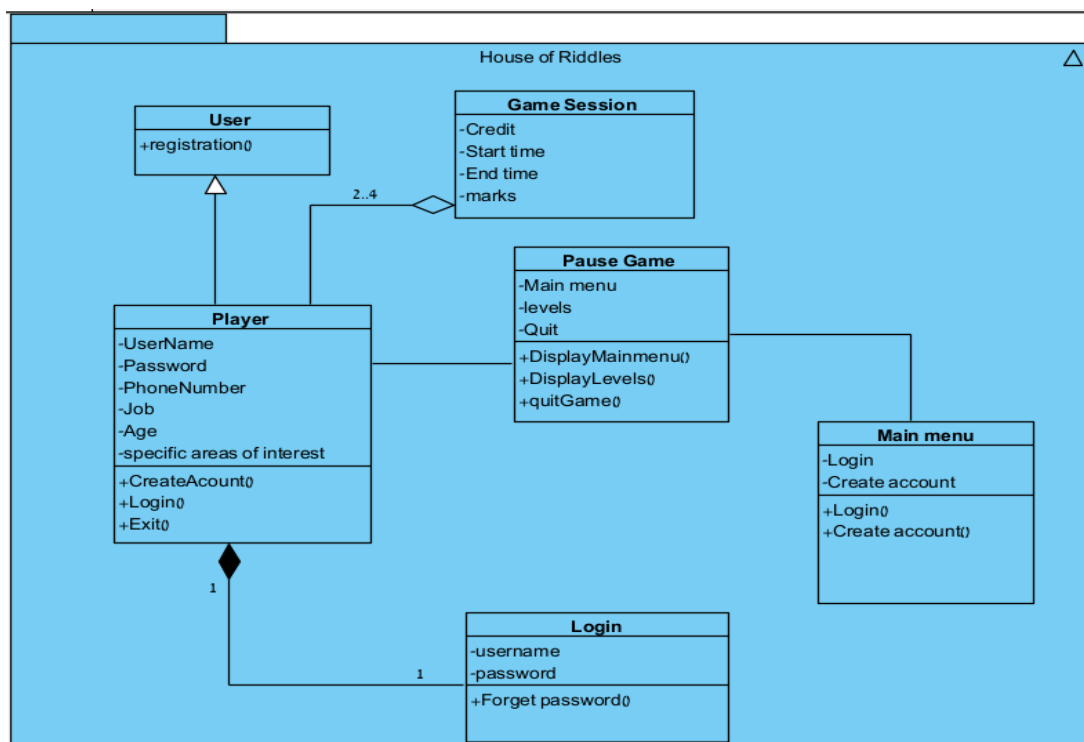
5.3.1 Use Case



5.3.2 Activity Diagram



5.3.3 Class Diagram



6. Verification and Evaluation

Verification and evaluation in game development ensure that the game functions as intended and meets quality standards. Verification involves testing the game's features, mechanics, and functionalities to identify and fix any bugs or errors. The evaluation assesses the overall player experience, including gameplay, graphics, audio, and user interface, to ensure the game meets design objectives and player expectations.

6.1 Evaluation

Evaluation in game development evaluates various aspects of the game to ensure it meets the design goals and player expectations. Through testing and gathering feedback from players, evaluation helps developers identify strengths and weaknesses, iterate on design choices, and ultimately create a more enjoyable and engaging game for the target audience.

- **Evaluating Core Gameplay Mechanics:** Test the core gameplay mechanics of the game. This includes the actions that players can perform, such as player action and puzzle solving. How players interact with the game world, and how the game systems interact with each other.
- **Evaluating Game Design Elements:** Evaluating the design of games and the elements of the game, such as gameplay, appropriate graphics, pleasant sound design. These should be immersive, engaging and tie in with the overall theme and narrative of the game.
- **Feedback Effectiveness Evaluation:** Collect player feedback on the clarity and helpfulness of feedback provided after each question. Use surveys to understand how players perceive the feedback and whether it helps them improve.

6.2 Verification

Verification in game development focuses on ensuring that the game functions correctly and meets specified requirements. Here are three examples:

- **Performance Testing:** Developers conduct performance testing to assess how the game performs under various conditions, such as different hardware configurations, network environments, and player loads. This helps optimize the game's performance, including frame rates, loading times, and server responsiveness, to provide a smooth and enjoyable experience for players.
- **Unit Testing:** Verification involves conducting unit tests on individual components or units of the game's code to ensure that each unit functions correctly in isolation. Developers write test cases to validate the behavior of functions, classes, and modules, helping to identify and fix errors early in the development process and ensure the stability and reliability of the codebase.
- **Play Mode Tests:** Play mode tests are run in the Unity editor and simulate actual gameplay to ensure that the game functions correctly in a real-world scenario.

- Here are a few unit tests we're going to use:

Test	Component	Test case description	Expected Result
1	Login	Correct username and password, press login	Go to the next screen
2	Login	Incorrect username and/or password, press login	Display error message
3	Login	Empty username or password field, press login	Display error message
4	Register	Enter valid details (Full Name, Username, Password, etc.) and press 'Create Account'	Account created
5	Register	Enter existing username and press 'Create Account'	Display error message
6	Rigister	Leave required fields empty (Username, Password, etc.), press 'Create Account'	Display error message
7	Home Page	Press 'Play' without logging in	Display error message
8	Home Page	Press 'Create Account' from the home screen	Navigate to the register page
9	home Page	Press 'Help' from the home screen	Display the help panel with the correct information
10	Play Scene	Player selects a house and answers a question correctly	House unlocks, and score updates
11	Play Scene	Player selects a house and answers a question incorrectly	Display an error message
12	Play Scene.	Player presses 'Hint' button	Display a personalized hint. progress
13	Logout	Player presses 'Logout' button	Session ends, navigate to home

			page
14	Practice Scene	Player selects 'Yes' for practice mode	Navigate to the practice game with sample questions
15	Practice Scene	Player selects 'No' for practice mode.	Navigate back to the previous screen
16	Help Panel	Press the 'Help' button	Display the help panel with clear instructions

7. References

1. https://www.researchgate.net/profile/Jeffrey-Craighead/publication/265284198_Using_the_Unity_Game_Engine_to_Develop_SARGE_A_Case_Study/links/55d33fdf08aec1b0429f32f4/Using-the-Unity-Game-Engine-to-Develop-SARGE-A-Case-Study.pdf
2. https://www.theseus.fi/bitstream/handle/10024/496604/Breugelmans_Arno.pdf?sequence=2&isAllowed=y
3. <https://isprs-annals.copernicus.org/articles/IV-4-W4/161/2017/isprs-annals-IV-4-W4-161-2017.pdf>
4. https://books.google.co.il/books?hl=ar&lr=&id=OMNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Reinforcement+Learning+Algorithms+with+unity+game&ots=CkBO Ny-A92&sig=OHdCTANtyNth N5-Auc sduxZq8&redir_esc=y#v=onepage&q=Reinforcement%20Learning%20Algorithms%20with%20unity%20game&f=false
5. https://www.languagezen.com/pt/about/english/Duolingo_Efficacy_Study.pdf
6. <https://www.mdpi.com/2078-2489/12/4/161>
7. <https://arxiv.org/abs/1809.02627>