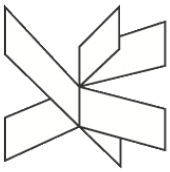

VIA University College

IT-SEP4C-S18 – Serious Game

Project report

The Frangovers

Karolina Beliharova (253810)

Marek Löwy (253652)

Supervisors

Kasper Knop Rasmussen

Jakob Knop Rasmussen

TABLE OF CONTENT

LIST OF FIGURES AND TABLES

ABSTRACT

The customer is the Ensight Games, a Danish company that focuses on developing fun and educational games for use in companies as well as schools and universities. The company required a serious game for memorization training that will be developed for both mobile and pc platforms. The system is simple and can be used by anyone who wants to play the simple lightweight game. A user can choose between playing a random level or the specific one. After the user starts the game, he runs around the maze and finds statues and gets access to different minigames for memory training. After the user finishes the minigame he will get a fire point and can continue in the maze until he collects 5 point and can finish the level.

INTRODUCTION

A game which can be used for educational purposes or spare time enrichment, has been known since around 2600 BC. To achieve the meaningful game experience, games needed to be defined by rules, to be understood clearly by the players.

French sociologist Roger Caillois (Caillois, 1953) defined game as an activity that needs to have the following characteristics: fun, separate, uncertainty, non-productive, governed and fictious.

The first video games can be dated to the early 50s, when the technology became advanced enough for scientist to design simple games and simulations using electronic circuits. The discovery of CRT lead to tremendous rise of game development.

Video games have great educational potential in addition to their entertainment value. Games designed for specific problem, or to teach a specific skill have been very successful, since they are motivating, engaging, and provide rewards and chance to improve.

Learning by playing games encourage students to learn outside of class. There are no consequences, it is only a game that means if players lose, they can simply start the game over, try it again and learn from previous mistakes. This is not possible with grades at school, so it is not possible to correct mistakes. Games make learning more fun, and student will be more motivated to study and learn something.

The project presented is based on a requirement from the Ensight Games company which came with a demand for a serious game for memorization thinking. They focus and develop fun and educational games for use in companies as well as schools and universities. The purpose is to create a user-friendly game that would help people with training their short-term memory. It should be in a form that is both entertaining and appealing to young people.

Main features of the game must include simple controls and clear rules. The game must be able to run on both PC and mobile platform.

REQUIREMENTS

1. A user should be able to use application on both PC and mobile platform.
2. The system should contain procedurally generated mazes.
3. The system should have different types of memory games for brain training.
4. The system should contain statues.
5. A user should be able to move by using game controllers.
6. The user should be able to access minigame by statues.
7. The system should have pexeso minigame.
8. The system should have Simon says minigame.
9. The system should have hidden object in room minigame.
10. The system should have Sailor puzzle minigame.
11. A user should be able to launch the application through the menu.
12. The user should be able to exit the labyrinth through the temple.
13. The system should have smooth animations to change scenes.
14. The system should have a least one character.
15. The user should have view from third person.
16. The system should be optimized.
17. A user should be able to easily quit from the game.
18. The system should have nice skybox.
19. The system should have more levels.
20. The system should give instructions to the player.
21. The system should have sound effects.

ANALYSIS

USE CASE DIAGRAM

The following figure represents the Use Case Diagram for the system based on the above mentioned important requirements.

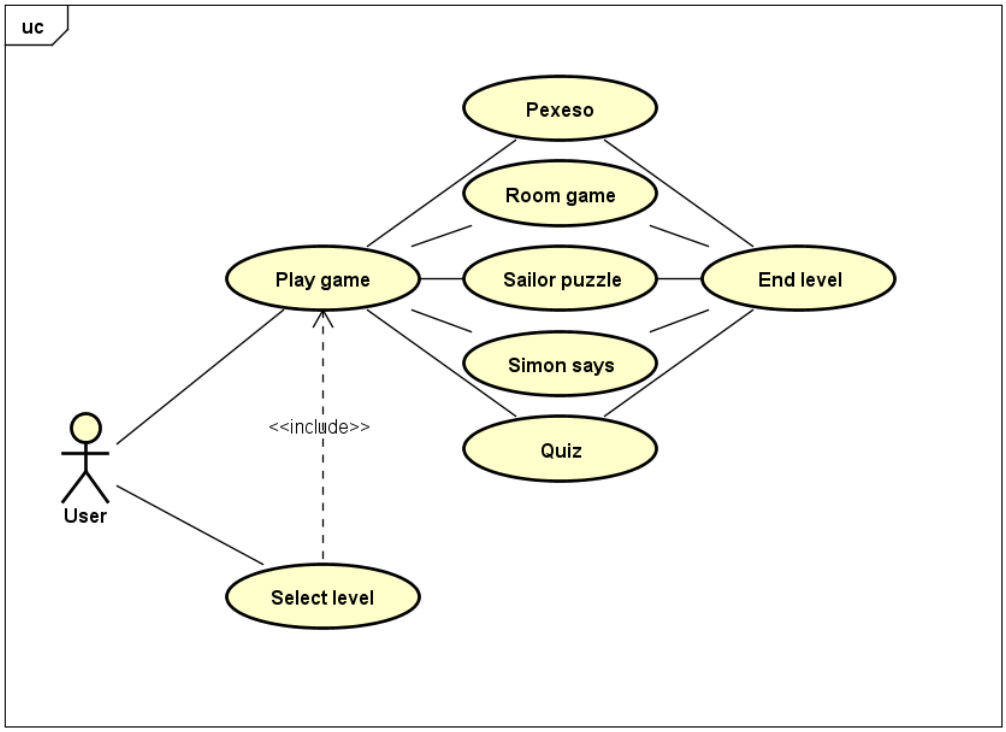

FIGURE 1: USE CASE DIAGRAM

When a user decides to play the game, the first level is loaded. In the game, the player navigates the labyrinth. While searching through the maze, the player encounters various statues, which lead to one of five minigames. One fire point is gained for completing each of the minigames. The player cannot end level until 5 fire points were collected.

USE CASE DESCRIPTION

From the use case description for Play Game, we can see that the user must first choose to play New Game in the main menu. When the New Game is selected, the first level is loaded. Afterward, the user plays the game, navigates through the maze and finds statues to launch the minigames. When the user collects five fire point, he can access the next level via temple.

ITEM	VALUE
UseCase	Play game
Summary	User plays the game.
Actor	User
Precondition	User selected to play new game.
Postcondition	User played game.
Base Sequence	1. The first level is loaded. 2. The user plays game. 3. The user navigates through the maze. 4. The user locates the minigames statues. 5. The user locates the next level temple.
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

FIGURE 2: USE CASE DESCRIPTION

DOMAIN MODEL

The game will contain player, which will be controlled by the user and will contain the camera. The player can interact with statues, through which access to minigame is granted. Labyrinths will be generated using Maze Generator. The level contains Player, Maze, and the audio controls.

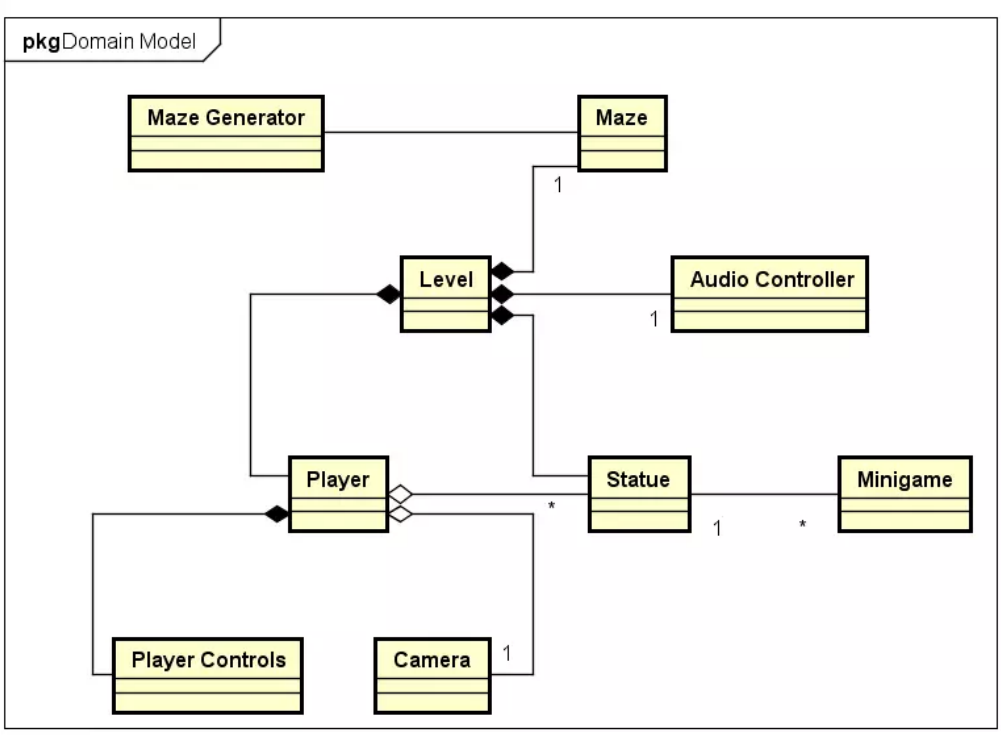

FIGURE 3: DOMAIN MODEL

DESIGN

We have decided to make a game, which will be appealing to young people and will convince them to train their memory. We will use Unity 5 to implement it since it is game engine we are most familiar with and will suffice our needs. We will make a 3D game with third-person controls. We have decided to do so because this type of games is quite popular in young generations and will help us in engaging them to play.

CLASS DIAGRAMS

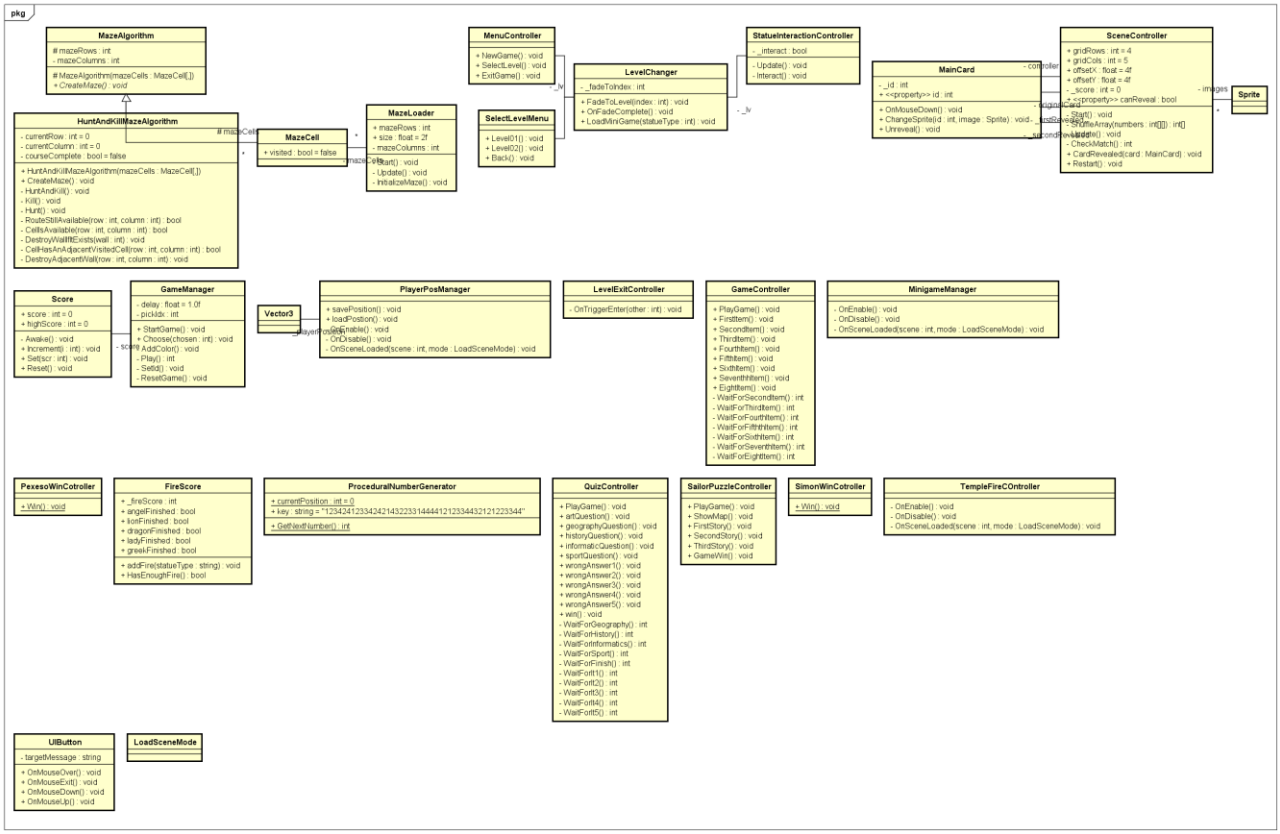

FIGURE 4: CLASS DIAGRAM

ACTIVITY DIAGRAM

From the activity diagram for Play Game use case we can see that after navigating labyrinth and finding the statue, a user can interact with the statue which will lead to playing the mini-game. When mini-game is finished, fire point is received, and the user is back in the labyrinth. If the user has collected 5 fire points, the level can be finished and next one is loaded.

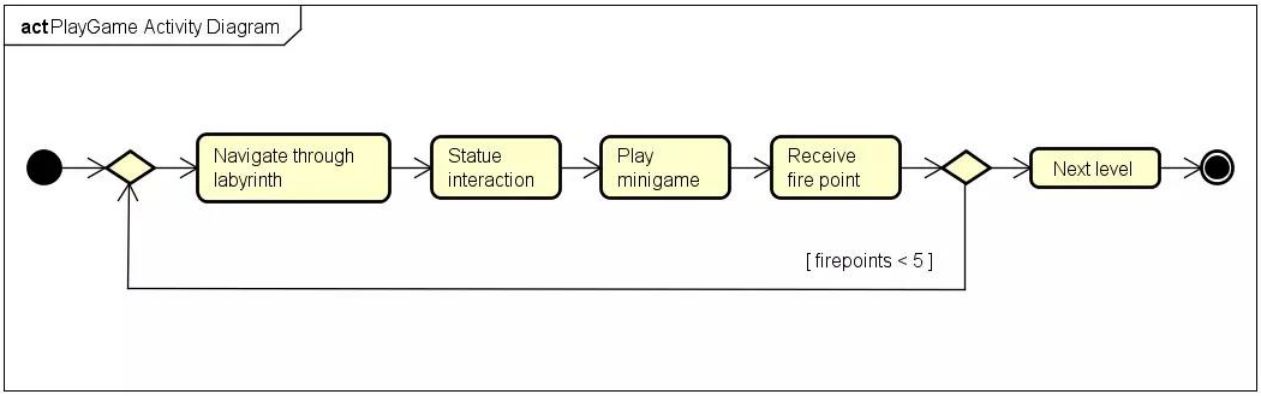

FIGURE 5: ACTIVITY DIAGRAM

SEQUENCE DIAGRAM

From the sequence diagram for playing mini-game we can see that when game registers user input for interacting with the statue, a method to load mini-game is called on the level changer. The method takes a string with statues name as a parameter, using which it determines which mini-game to load. Before the new scene is loaded, player position is saved. When the game is finished, method Win() from win manager for a particular mini-game, which gives player one fire point and loads back the level. When the level scene is loaded, a previously saved player position is loaded.

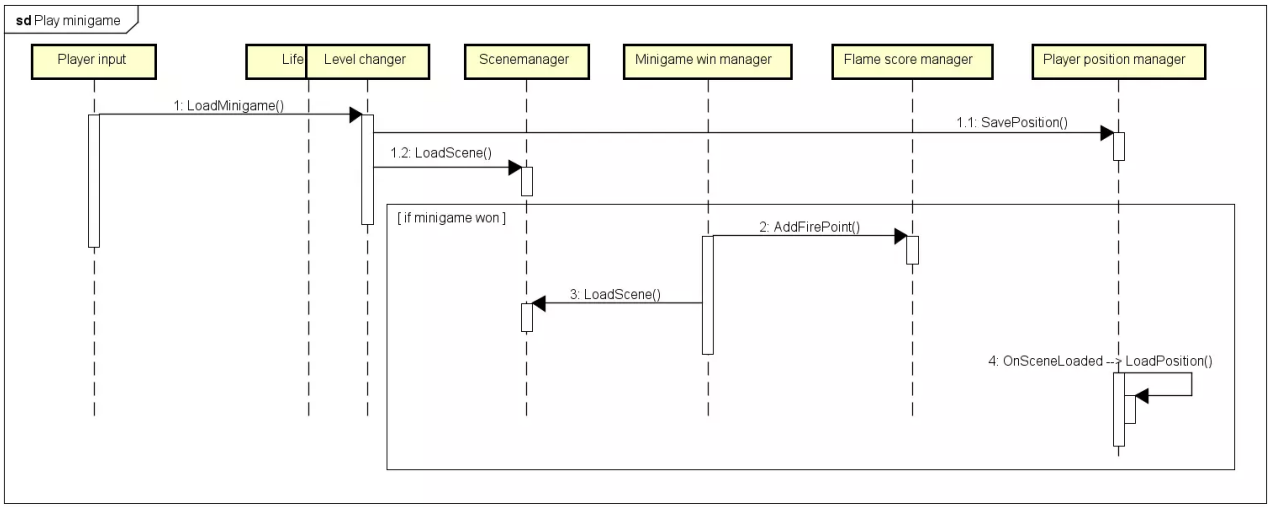

FIGURE 6: SEQUENCE DIAGRAM

UI DESIGN CHOICES

The design of *BrainPath* is very simple and easy to use for everyone. The first thing the user will come across is the Game Menu where the user can choose between several options. New Game option will load the first level by default.

FIGURE 7: GAME MENU (UI)

When the select level is clicked, the user can choose which level will be loaded. The levels are not ordered by difficulty but different minigames are included. The user has the option to go back to the main menu either.

FIGURE 8: SELECT LEVEL (UI)

The character was imported from unity Asset Store.

FIGURE 9: CHARACTER (UI)

The user can see the game from third person view to achieve better outlook of the maze and character itself. The maze contains stone roads, grassy walls, and nice skybox.

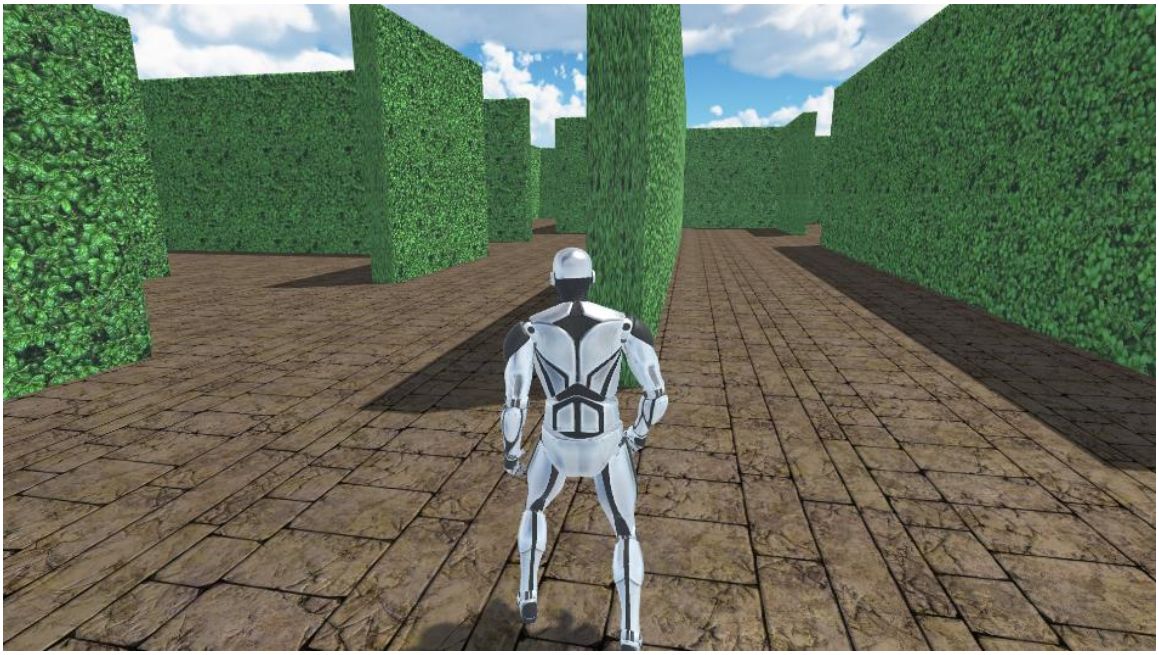

FIGURE 10: LEVEL 1 SCREEN (UI)

While the user is navigating the labyrinth, he finds five different statues which he can interact with and it leads to load the minigames.

FIGURE 11: THE ANGEL STATUE (UI)

FIGURE 13: THE DRAGON STATUE (UI)

FIGURE 12: THE SCREAMING LADY STATUE (UI)

FIGURE 15: THE LION STATUE (UI)

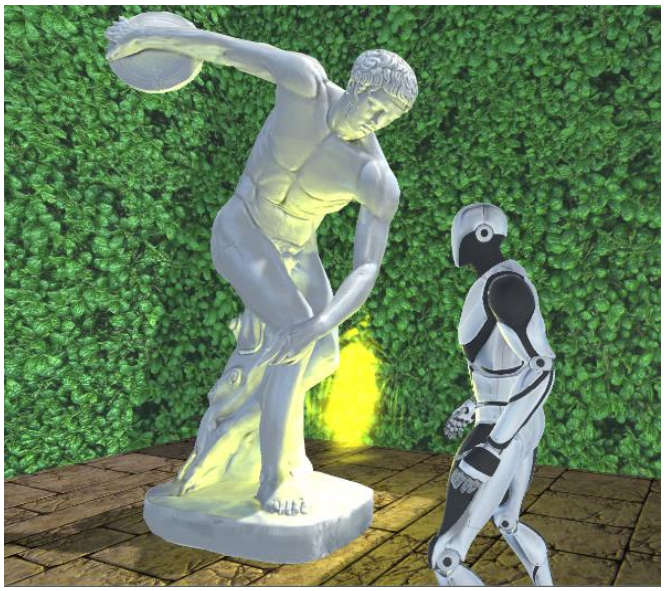

FIGURE 14: THE GREEK MAN STATUE (UI)

If the user has collected 5 fire points, the temple will activate and light up which allows to the user to end the level.

FIGURE 16: THE TEMPLE (UI)

IMPLEMENTATION

The game was implemented using Unity 5 and C# scripts. Most of scripts derive from MonoBehavior, which allows them to be parts of GameObjects.

Example 01: *StatueInteractionController*

- This component is responsible for interacting with statue when player input is given
- Method Interact first gets all objects with tag *Statue*
- If a statue is in range, method from LevelChanger is called to load the minigame

```
public class StatueInteractionController : MonoBehaviour {

    [SerializeField] private LevelChanger _lv;
    private Transform _player;

    private bool _interact;

    void Update() {
        if (Input.GetKeyDown(KeyCode.F))
            Interact();
    }

    private void Interact() {
        GameObject[] objs = GameObject.FindGameObjectsWithTag("Statue");
        _player = gameObject.transform;
        foreach (var obj in objs) {
            if (Vector3.Distance(_player.position, obj.transform.position) < 4) {
                _lv.LoadMiniGame(obj.name);
            }
        }
    }
```

Example 02: *LevelChanger*

- This class is responsible for loading levels and minigames
- Method FadeToLevel takes build index of scene to load as argument
- Before loading next scene, animation to fadeout is triggered
- Minigame to load is determined by the statue name

```
public class LevelChanger : MonoBehaviour {

    [SerializeField] private Animator _animator;
    private int _fadeToIndex;

    public void FadeToLevel(int index) {
        _fadeToIndex = index;
        _animator.SetTrigger("FadeOut");
        GameObject go = GameObject.FindGameObjectWithTag("PosMan");
        go.GetComponent<PlayerPosManager>().savePosition();
    }

    public void OnFadeComplete() {
        SceneManager.LoadScene(_fadeToIndex);
    }

    public void LoadMiniGame(String statueType) {
        switch (statueType) {
            case "Angel": {
                FadeToLevel(SceneManager.GetActiveScene().buildIndex + 1);
                break;
            }
            case "Lion": {
                FadeToLevel(SceneManager.GetActiveScene().buildIndex + 2);
                break;
            }
            case "Greek Man": {
                FadeToLevel(SceneManager.GetActiveScene().buildIndex + 3);
                break;
            }
            case "Screaming Lady": {
                FadeToLevel(SceneManager.GetActiveScene().buildIndex + 4);
                break;
            }
            case "Dragon": {
                FadeToLevel(SceneManager.GetActiveScene().buildIndex + 5);
                break;
            }
            default:
                Debug.Log("Error");
                break;
        }
    }
}
```

Example 03: *PlayerPositionManager*

- This class is responsible for saving and loading players position when minigames are loaded
- Position is stored as Vector3 and saved from LevelChanger when next scene is loaded
- When Level scene is loaded, method to load position is triggered

```
public class PlayerPosManager : MonoBehaviour {
    private GameObject _player;

    private Vector3 _playerPosition = new Vector3(0,0,0);
    private Quaternion _playerRotation;

    public void savePosition() {
        _player = GameObject.FindGameObjectWithTag("Player");
        if (_player != null) {
            _playerPosition = _player.transform.position;
            _playerRotation = _player.transform.rotation;
            _player = null;
        }
    }
    public void loadPostion() {
        _player = GameObject.FindGameObjectWithTag("Player");
        if (_player != null) {
            if (_playerPosition.Equals(new Vector3(0,0,0))) {
                _playerPosition =transform.position;
                _playerRotation = transform.rotation;
            }
            _player.transform.position = _playerPosition;
            _player.transform.rotation = _playerRotation;
            _player = null;
        }
    }
    void OnEnable() {
        SceneManager.sceneLoaded += OnSceneLoaded;
    }
    void OnDisable() {
        SceneManager.sceneLoaded -= OnSceneLoaded;
    }
    private void OnSceneLoaded(Scene scene, LoadSceneMode mode) {
        loadPostion();
    }
}
```

Example 04: *MinigameWinController*

- Each minigame has either WinController, or contains Win method somewhere in its main script
- Method Win grants a fire point to the player and then loads level from which minigame has been loaded

```
public class SimonWinCotroller : MonoBehaviour {

        public static void Win() {
                GameObject go = GameObject.FindGameObjectWithTag("FireScore");
                go.GetComponent<FireScore>().addFire("Lion");
                SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex - 2);
        }
}
```

Example 05: *TempleFireController*

- This class is responsible for checking whether the player has collected 5, or more Fire points
- When sufficient amount of point is collected, the lights and a flame in next level temple are activated, which allows player to enter and finish the level

```
public class TempleFireCOntroller : MonoBehaviour {
        void OnEnable() {
                SceneManager.sceneLoaded += OnSceneLoaded;
        }
        void OnDisable() {
                SceneManager.sceneLoaded -= OnSceneLoaded;
        }
        private void OnSceneLoaded(Scene scene, LoadSceneMode mode) {
                GameObject go = GameObject.FindGameObjectWithTag("FireScore");
                FireScore fs = go.GetComponent<FireScore>();

                if (fs._fireScore >= 5) {
                        Transform[] trans =
GameObject.Find("Temple").GetComponentsInChildren<Transform>(true);
                        foreach (Transform t in trans) {
                                if (t.gameObject.name == "Lights") {
                                        t.gameObject.SetActive(true);
                                }
                        }
                }
        }
}
```

TEST

The final version of the game was tested by checking if all the requirements were accomplished.

Requirement	Status
A user should be able to use application on both PC and mobile platform.	✗
The system should contain procedurally generated mazes.	✓
The system should have different types of memory games for brain training.	✓
The system should contain statues.	✓
A user should be able to move by using game controllers.	✓
The user should be able to access minigame by statues.	✓
The system should have pexeso minigame.	✓
The system should have Simon says minigame.	✓
The system should have hidden object in room minigame.	✓
The system should have Sailor puzzle minigame.	✓
A user should be able to launch the application through the menu.	✓
The user should be able to exit the labyrinth through the temple.	✓
The system should have smooth animations to change scenes.	✓
The system should have a least one character.	✓
The user should have view from third person.	✓
The system should be optimized.	✓
A user should be able to easily quit from the game.	✓
The system should have nice skybox.	✓
The system should have more levels.	✓
The system should give instructions to the player.	✓
The system should have sound effects.	✓

RESULTS AND DISCUSSION

BrainPath is a serious game aimed at young people to help them train memory. Users must orient in labyrinths and remember their way. While exploring they can encounter several statues leading to one of five memory training mini-games. Pexeso is game where players must match two cards with the same number and picture. Quiz, where players must recall and answer several questions from general knowledge. Sailors puzzle in which players read through a story from old sailor's journal and must decipher and remember trace the sailor made. They mark this route on the map later. Room game is mini-game where the player is shown a picture of a house plan for few seconds. After the time is up, 4 objects in the plan are covered and the player is asked to mark a certain object. The last mini-game is Simon Says. A sequence of color and sound is played, and the player must remember it and then repeat. The sequence starts at length of one and prolongs to ten. Due to time shortage, the game not available for mobile devices. This is something which needs to be improved later in future.

CONCLUSIONS

During this project, a serious game for memory training has been developed. This game is a fun solution for young people to engage in short-term memory training. It is a PC game, which is entertaining for younger generations. It contains effective memory training exercises concealed enough to seem appealing and fun to young.

PROJECT FUTURE

BrainPath has many opportunities for future development. First of all, mobile version needs to be finished, after which the aesthetics can be addressed. At this point, when mini-game has finished the statue, which led to it, disappears. Putting out of only the flame on the statue would, however, look much nicer. Some additional animations could be also added, which would give the game more developed look. Implementation of several more mini-games would give the game more diverse feel. Currently, the game has only two levels, which makes the gameplay quite short. In the future, it would be preferable to add more levels. The game has also no background music and lacks more sound effects. Even though the main idea for the game was the training of the memory, telling some sort of story could be potentially more engaging for users. Having a story to tell would also create an opportunity to prolong gameplay by making a spinoff game, or even making a standalone game for one of the minigames.

SOURCE OF INFORMATION

Why Use Games to Teach. *SERC* [ONLINE].
Available at: https://serc.carleton.edu/introgeo/games/whygames.html

Caillois, R. (1953). *Les jeux et les hommes.* Paris: Gallimard. Retrieved from Wikipedia
Available at: https://en.wikipedia.org/wiki/Game#Roger_Caillois

Playing video games is good for your brain – here's how. *The Conversation* [ONLINE].
Available at: http://theconversation.com/playing-video-games-is-good-for-your-brain-heres-how-34034

Wikipedia. *History of video games*. [ONLINE]
Available at: https://en.wikipedia.org/wiki/History_of_video_games

Wikipedia. *Game*. [ONLINE]
Available at: https://en.wikipedia.org/wiki/Game

69 Trick Questions and Answers: How Many Can You Answer Correctly?. *Prisoner Of Class* [ONLINE].
Available at: https://www.prisonerofclass.com/trick-questions-and-answers/

Woody Woodpecker. In: *Wikipedia* [ONLINE].
Available at:
https://www.google.dk/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjP34HpyLzbAhXHfiwKHf1MDwAQjRx6BAgBEAU&url=https%3A%2F%2Fg.sina.re%2Findex.php%3Fq%3DoKipp7eAc2SWr3HmtNTZ4JzJqsRgwuHce-fP2bV-w-Cmp9G5jdja07esqdibpA&psig=AOvVaw0tj8-6LIJP8-Qk2ckf18lb&ust=1528289655515766

Ensight Games [ONLINE].
Available at: http://www.ensightgames.com

Busina. 2015. Angel Statue. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/environments/fantasy/angel-statue-27594

NVizzutti. 2017. Simon Says Minigame. [ONLINE]
Available at: https://github.com/NVizzutti/SimonSays

ChamferBox Studio. 2018. Greek Man Statue. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/props/discobolus-statue-107544

VSIFY. 2016. Screaming Lady Statue. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/environments/dungeons/screaming-statue-60499

Viacheslav Titenko. 2016. Dragon Statue. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/dragon-statue-59053

organicpolygons. 2015. Lion Statue. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/props/exterior/lion-statue-34247

Nobiax / Yughues. 2015. Pavement Material. [ONLINE]
Available at: https://assetstore.unity.com/packages/2d/textures-materials/roads/yughues-free-pavement-materials-12952

Thunderent's Assets. 2014. Temple Prefab. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/props/exterior/temple-props-19555

OV. 2017. Skull Platform. [ONLINE]
Available at: https://assetstore.unity.com/packages/3d/props/skull-platform-105664

ProAssets. 2016. Skybox. [ONLINE]
Available at: https://assetstore.unity.com/packages/2d/textures-materials/sky/free-hdr-sky-61217

Invector. 2017. Third Person Controller. [ONLINE]
Available at: https://assetstore.unity.com/packages/templates/systems/third-person-controller-basic-locomotion-free-82048

Gameshard. 2017. Quiz Birdz. [ONLINE]
Available at: https://assetstore.unity.com/packages/2d/characters/cute-birds-89649

Unity Technologies. 2018. TextMesh Pro. [ONLINE]
Available at: https://assetstore.unity.com/packages/essentials/beta-projects/textmesh-pro-84126

Google Maps. 2018. Sailors Puzzle Map. [ONLINE]
Available at: https://www.google.com/maps/@35.9470164,-5.5265056,145601m/data=!3m1!1e3

Quiz Time. In: *Removing the Mask* [ONLINE]
Available at: https://www.fromtpm.com/blog/2016/12/08/quiz-time-open-book-examination/

Start. In: *Sell simple* [ONLINE]
Available at: http://www.sellsimpleestateagency.com/wp-content/uploads/2016/09/start.png

Wrong answer. In: *Clker.com* [ONLINE]
Available at:
http://www.clker.com/cliparts/7/e/1/6/1206557186603844069mcol_cross.svg.hi.png
Colorful cartoon nature. In: *Storyblocks* [ONLINE]
Available at: https://d2v9y0dukr6mq2.cloudfront.net/video/thumbnail/S15GBCm/colorful-cartoon-nature-background-with-space-for-your-text-or-logo-nice-sunny-day-with-some-clouds_ehseqyaig__F0000.png

Richard Hawkes. 2016. Maze Generating Algorithm. [ONLINE]
Available at: https://www.youtube.com/watch?v=IrO4mswO2o4&t=3s

ICT Engineering, 2017 Project Report – VIA Engineering Guidelines

APPENDENCIES

A – USE CASE DESCRIPTIONS

ITEM	VALUE
UseCase	Select level
Summary	User selects level.
Actor	User
Precondition	User chose option to select level.
Postcondition	User played game.
Base Sequence	1. The chosen level is loaded. 2. The user plays the game.
Branch Sequence	
Exception Sequence	
Sub UseCase	Play game
Note	

FIGURE 17: SELECT LEVEL DESCRIPTION

ITEM	VALUE
UseCase	Pexeso
Summary	User looks for matching pairs.
Actor	
Precondition	User located and activated Angel statue.
Postcondition	User got one fire point.
Base Sequence	1. The system displays twenty cards in random order. 2. The user press start button to start or restart the game. 3. The user selects two cards to find out what is on the other side. 4. The user looks for two same cards. 5. The user finishes the minigame by finding ten matching pairs.
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

FIGURE 18: PEXESO MINIGAME DESCRIPTION

ITEM	VALUE
UseCase	Room game
Summary	User looks for hidden objects.
Actor	
Precondition	User located and activated Screaming Lady statue.
Postcondition	User got one fire point.
Base Sequence	1. The system displays open house with objects. 2. The user press the play game button to start the minigame. 3. The system displays four red buttons and hides different objects. 4. The system displays information which object user has to find. 5. The user finishes the minigame by finding all required objects.
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

FIGURE 19: ROOM GAME DESCRIPTION

ITEM	VALUE
UseCase	Sailor puzzle
Summary	User solves sailing puzzles.
Actor	
Precondition	User located and activated Greek man statue.
Postcondition	User got one fire point.
Base Sequence	1. The system displays instructions for the user. 2. The user press the play button to start the minigame. 3. The system displays instructions of story part 1 for the user. 4. The user marks the solution on the map. 5. The system displays instructions of story part 2 for the user. 6. The user marks the solution on the map. 7. The system displays instructions of story part 2 for the user. 8. The user marks the solution on the map. 9. The user finishes the minigame by finding the right solution of the story.
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

FIGURE 20: SAILOR PUZZLE DESCRIPTION

ITEM	VALUE
UseCase	Simon says
Summary	User repeats sequences after Simon.
Actor	
Precondition	User located and activated Lion statue.
Postcondition	User got one fire point.
Base Sequence	1. The user starts the minigame by pressing begin button. 2. The system shows random sequences of pressing the different buttons. 3. The user repeats sequence after the system. 4. The user finishes the game by getting 10 points.
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

FIGURE 21: SIMON SAYS MINIGAME DESCRIPTION

ITEM	VALUE
UseCase	End level
Summary	Next level is loaded.
Actor	
Precondition	User played all minigames and got 5 fire points.
Postcondition	
Base Sequence	1. The user enters the next level temple. A) Next level is loaded, if the user finished the first level. B) The system displays menu, if the user finished the second level.
Branch Sequence	
Exception Sequence	
Sub UseCase	
Note	

FIGURE 22: END LEVEL DESCRIPTION

B – PROJECT DESCRIPTION

BACKGROUND DESCRIPTION

A game, as a form of recreation, which can be used for educational purposes, or spare time enrichment, has been known since around 2600 BC (Royal Game of Ur, Iraq). The games generally spread among people has developed to be more complicated from that time on.

Rules, goals challenges and player interactions are some of the examples that became parts of games. Playing against the opponent was more entertaining because it was challenging, required higher skills, and the joy from winning was more satisfying.

To achieve the meaningful game experience, games needed to be defined by rules, to be understood clearly by the players. Wining conditions were needed to determine winner from looser, or to determine draw.

Some games were pushed so far, that certain level of skill and strategy was required to play for each participant. Fulfilling those rules ensures best game experience.

French sociologist Roger Caillois (Caillois, 1953) defined game as an activity that needs to have the following characteristics: fun, separate, uncertainty, non-productive, governed and fictitious.

The first video games can be dated to the early 50s, when the technology became advanced enough for scientists to design simple games and simulations using electronic circuits. Until that point, the computers were mainly used to solve mathematical problems. The discovery of CRT lead to tremendous rise of game development; however, it was not until the 70s and 80s that the games reached the mainstream popularity with arrival of arcade games and gaming consoles.

According to *The Conversation*, video games have great educational potential in addition to their entertainment value. Games designed for specific problem, or to teach a specific skill have been very successful, since they are motivating, engaging, and provide rewards and chance to improve.

Gameplay involves repeated actions that strengthen the brain cell connections underlying memory and learning. Games as Tetris or Othello activate brain areas which control decision making. Some games require real-time action and activate areas, which control sensory movement.

Definition of Purpose

The purpose of this project is to develop an application that will help people with training their short-term memory. It should do this in a form that is both entertaining and appealing to young people.

Problem Statement

We want young people to have fun in their free time while still getting some benefits in terms of short-term memory improvement.
Today, young people see memory training exercises as boring and unattractive. We need to find a way to change this. We need to find something that will help them train their short-term memory in a fun way. We need to make this solution available for them at almost any time, whether they are home, or bored outside. We need to make this solution appealing to them, while keeping it effective.

Delimitation

This solution can be used by only one person at the time. The solution will be made only on computer and mobile devices. The solution won't necessarily improve long-term memory.

Choice of Models and Methods

What Partial problem	**Why** Why study this problem?	**Which** Which models/theories are expected to be used to solve the problem?
How to make memory training fun for young	Because making training fun will convince them to train more	We will make a serious game in using Unity 5
How to make something to train short-term memory	Short term-memory can be very useful to young people	We will include several types of puzzles in the game
How can we make the solution available at almost any time	This will give them opportunity to train whenever they feel like it, or if they are bored.	We will make the game available for both computer and mobile devices

TIME SCHEDULE

	Task Name	Duration	Start	ETA
1	Complete project execution	22 days	16.05.18	06.06.18
2	Coding	18 days	19.05.18	05.06.18
3	Scrum setup and game idea planning	1 day	18.05.18	18.05.18
4	First Sprint	3 days	19.05.18	21.05.18
5	Second Sprint	3 days	22.05.18	24.05.18
6	Third Sprint	3 days	25.05.18	27.05.18
7	Fourth Sprint	3 days	28.05.18	30.05.18
8	Fifth Sprint	3 days	31.05.18	02.06.18
9	Sixth Sprint	16 days	03.06.18	05.06.18
10	Documentation work	10 days	16.05.18	06.06.18
11	Hand In	1 day	06.06.18	06.06.18

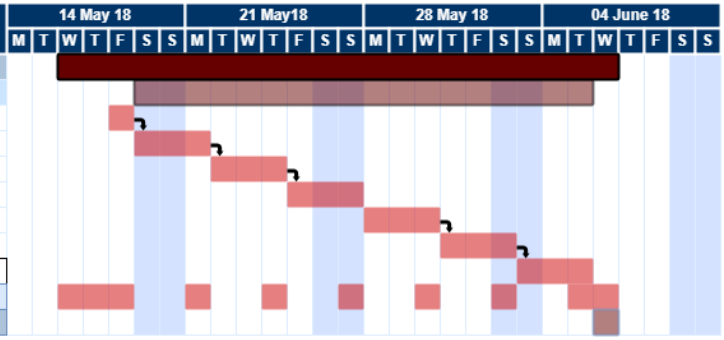

FIGURE 23: TIME SCHEDULE (PROJECT DESCRIPTION)

Risk Assessment

RISKS	DESCRIPTION	L	S	PERSON IN DANGER	CONTROL RECOMMENDATIONS
MOBILE PROBLEMS	SOMETHING CAN'T BE IMPLEMENTED ON MOBILE PLATFORM	3	3	ALL GROUP MEMBERS	ENSURE THAT FUNCTIONALITY THAT IS TO BE IMPLEMENTED IS COMPATIBLE BEFORE IMPLEMENTING
NOT ENOUGH TIME	GAME COULDN'T BE FINISHED ON TIME	2	3	ALL GROUP MEMBERS	DON'T OVERESTIMATE SPEED OF OUR IMPLEMENTATION
NOT OPTIMIZED	GAME IS TOO RESOURCE HEAVY	2	2	ALL GROUP MEMBERS	TRY NOT TO IMPORT OVER COMPLICATED ASSETS
ASSETS	SOME ASSETS NEED COULDN'T BE FOUND	3	2	ALL GROUP MEMBERS	DO INITIAL ASSET RESEARCH BEFORE PROCEEDING TO IMPLEMENTATION