Assignment 1

1 Functional requirements

1.1 Must haves

- Player can register an account
- Player can login
- Player can logout
- Player can start a game
- Levels consists of a grid
- A grid has walls on the border
- Levels have a laser.
- Walls block the laser
- Mirrors should reflect lasers at a 90 degree angle
- A level has a goal for the laser
- When the goal is hit the player wins the level
- Every object has a design
- The game has a scoreboard which shows the top 5 scores that have ever been recorded.

1.2 Should haves

- Games have multiple levels
- Levels have walls in the grid
- Mirrors can be rotated
- Mirrors can be moved
- Start menu
- Settings menu
- Mirrors can reflect lasers different angles
- Multiple goals in a level

1.3 Could haves

- Rotate the laser
- Achievements
- Stars
- Shop with power-ups
- Sound effects
- Bombs that have to be avoided
- Prism that splits the laser
- Multiple lasers in a level
- Custom level editor
- One way glass
- Lens refracts laser
- Level progress bar

1.4 Won’t haves

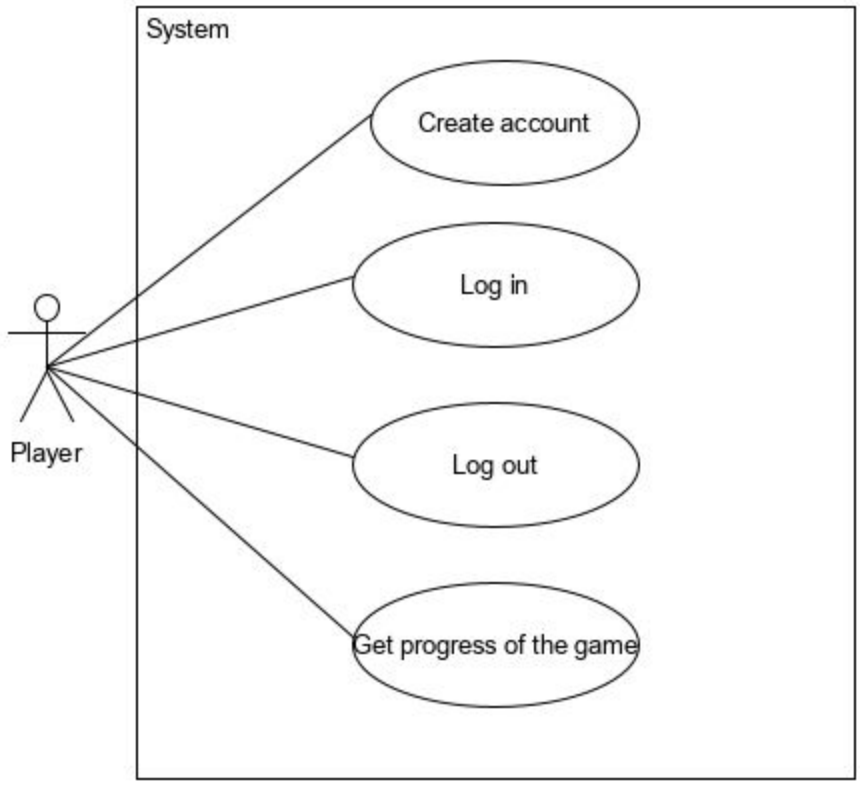
- Multiplayer

- Friends list
- 3D support

2 Non-Functional requirements

- Uses Java as programming language (version 12)
- Uses powerful game engine or libraries (LibGDX)
- Preferred test coverage at the end: 75
- Platforms supported: Windows
- Have MySQL database to store authentication info and game progress and use JDBC driver to control it.

Story 1: Player authentication



User story:

As a Player,
I want to be able to create an account and log in;
So that I can attribute my progress to my name.

Scenario S1.1: User registration.
Given no user is logged in;

When I click the “Register” button;
Then I can enter my preferred username and password and my account is created.

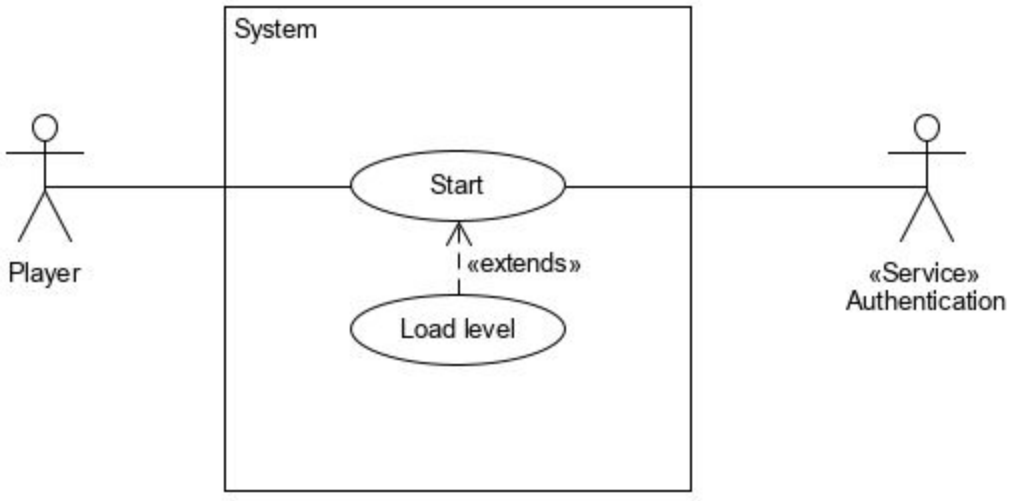
Scenario S1.2: Logging in.
Given no user is logged in;
And I have registered an account;
When I click the “Log in” button and enter my username and password;
Then I am logged in.

Scenario S1.3: Logging out.
Given I am logged in;
When I click the “Log out” button;
Then I am logged out and any running level stops.

Use case description:

Date: 26/11/2019
Purpose: Create an account so that one can play the game afterwards
Overview: A player presses the create account button to create an account. After that the user will fill in his or her credentials to create this account. If the creation is successful, then the player sees a confirmation that it’s successful and returns to the login screen. If the creation is unsuccessful, then an error message will be shown. The creation of an account is unsuccessful whenever an invalid name, password or email is entered during the creation.
Cross-references:

Story 2: Player can start a game



User story:

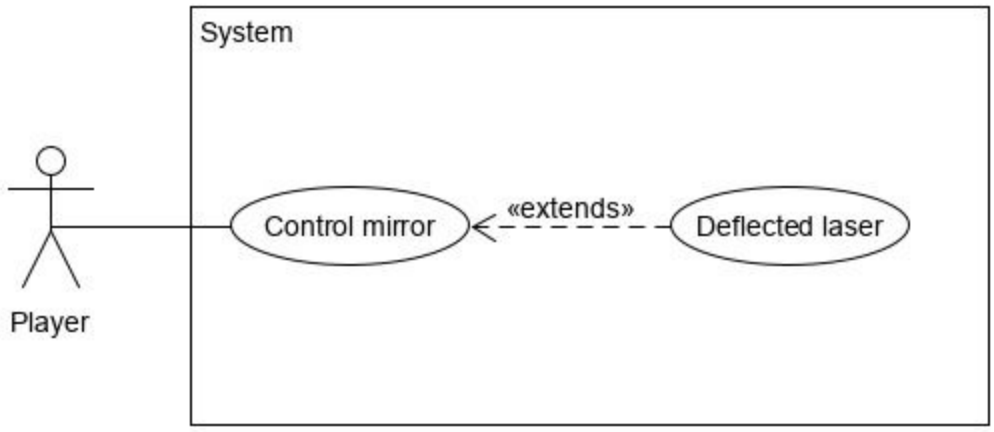
As a Player,
I want to be able to start the game;
So that I can play the game.

Scenario S2.1: Start the game.
Given the player has logged in;
When the player clicks the "Start" button;
Then the level is loaded.

Use case description:

Date: 26/11/2019
Purpose: Start the game
Overview: The player presses the “Play” button and switches to a screen with a level selection. Depending on the progression of the player, some of the level will be grayed out. The player will successfully start a game when clicking on one of levels which are not grayed out. If it is unsuccessful, then nothing happens and the player remains in the current window. It is unsuccessful whenever the player clicks on a level that is grayed out. At that point nothing will happen.
Cross-reference: Story 1
Player can register an account, Player can login, Player can logout.

Story 3: Mirrors should reflect laser



User story:

As a player,
When I control mirrors in the grid;
And when the laser hits the mirror;
Then the laser will reflect in a 90 degree angle;
So that I can reach the goal.

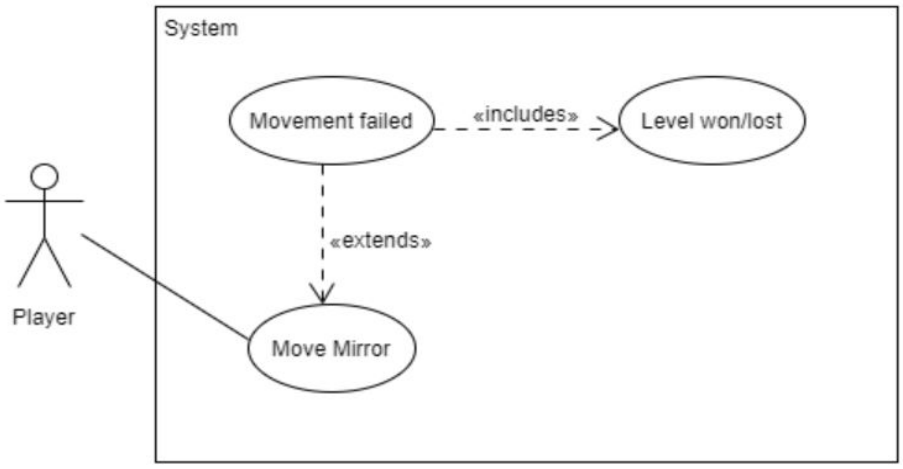
Scenario S3.1: hit mirror
Given the game has started;
When the laser hits a mirror;
Then it reflects in a 90 degree angle.

Scenario S3.2: miss mirror
Given the game has started;
When the laser misses a mirror;
It continues on its way.

Use case description:

Date: 26/11/2019
Purpose: Reflect the laser so that it can reach the goal of the level.
Overview: The player controls the mirrors in the grid. Whenever the player moves the mirror, the system will check if the mirror collides with the laser. If successful, then the mirror collides with the laser and the mirror will reflect the laser in a 90 degree angle. If unsuccessful, then the mirror does not collide with the laser and the laser will continue on it's current path without being reflected.
Cross-references: Story 1, 2 and 4
Player can register an account, Player can login, Player can logout, Mirrors can be moved, Player can start a game.

Story 4: Mirrors can be moved



User story:

As a player,
I want to be able to move the mirror to whatever the position I want;
So that I can reflect the laser to any direction and position I want.

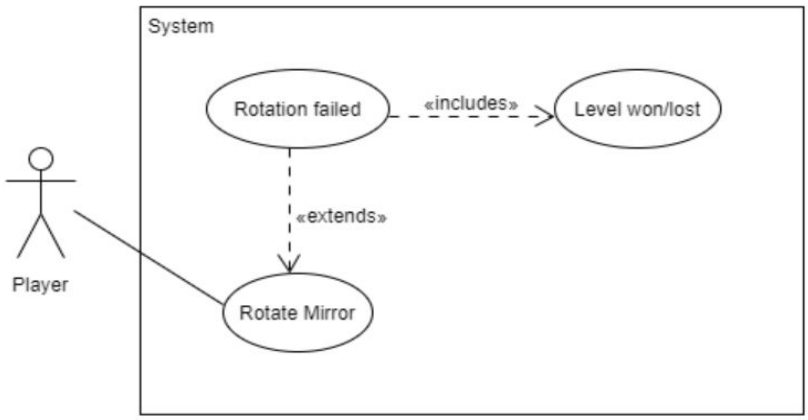
Scenario S4.1: Move the mirror during the game.
Given the game has started and mirrors are placed somewhere in the grid;
When the player hovers the mouse on the mirror and moves the mouse;
Then the mirror should be moved to where the mouse stops.

Scenario S4.2: Cannot move the mirror when the player loses/wins the level or the game is paused.
Given the game has suspended or the player loses/wins the level;
When the player hovers the mouse on the mirror and moves the mouse;
Then the mirror should not be moved.

Use case description:

Date: 26/11/2019
Purpose: Move a mirror in the grid
Overview: The player clicks on the mirror in the grid and can move it anywhere on the grid. If the move is successful, then the mirror will be positioned on this new position where the player drops it. If it is unsuccessful, then the mirror will stay on it's current position. A move is unsuccessful if a mirror is dragged outside of the grid, onto other objects or dragged while the game is already suspended/finished.
Cross-references: Story 1 and 2
Player can register an account, Player can login, Player can logout, Player can start a game.

Story 5: Mirrors can be rotated



User story:

As a player,
I want to be able to rotate the mirror to different angles;
So that I can reflect the laser to any direction and position I want.

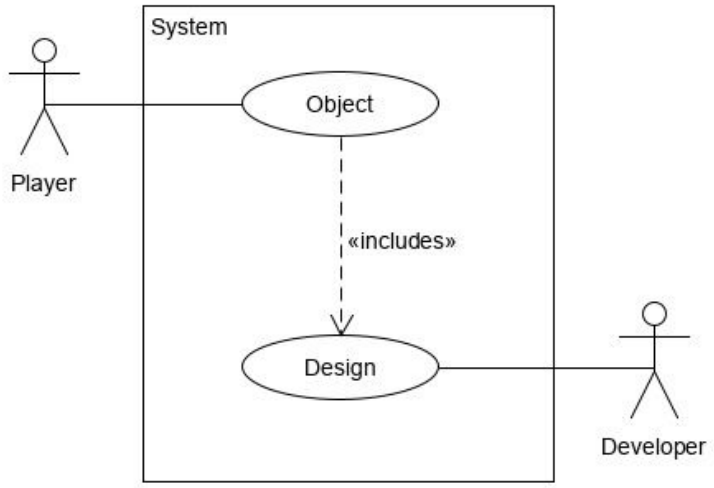
Scenario S5.1: Rotate the mirror during the game.
Given the game has started and mirrors are placed somewhere in the grid;
When the player hovers the mouse on the mirror and clicks the mouse;
Then the mirror should be rotated to where the player wants.

Scenario S5.2: Cannot rotate the mirror when the player loses/wins the level or the game is paused.
Given the game has suspended or the player loses/wins the level;
When the player hovers the mouse on the mirror and clicks the mouse;
Then the mirror should not be rotated.

Use case description:

Date: 26/11/2019
Purpose: Rotate a mirror so that a laser can be reflected in different angles.
Overview: The player clicks on a mirror in the grid to attempt to rotate this mirror. If this action is successful, then the mirror will rotate 45 degrees clockwise. If this action is unsuccessful, then the mirror will remain on its current position with its current angle. This action is unsuccessful when the game is already suspended/finished and the player attempts to rotate the mirror.
Cross-references: Story 1 and 2.
Player can register an account, Player can login, Player can logout, Player can start a game.

Story 6: Every object has a design



User story:

As an object,
I want to have a unique design;
So that the player can tell me apart from other objects.

Scenario S6.1: Object visible.
Given the game has started;
When I am part of the current level;
Then my corresponding sprite is rendered at my location.

Use case description:

Date: 26/11/2019
Purpose: Make discerning between different objects visible.
Overview: The player wants to be able to discern between the different objects in the grid. Therefore whenever a new object gets put on the grid, it will display the texture created for the object.
Cross-references: Story 1 and 2.
Player can register an account, Player can login, Player can logout, Player can start a game.