ASSIGNMENT 1

GROUP 8

28.11.2019

EXERCISE 1: REQUIREMENT ENGINEERING

Must Haves:

Functional:

- The player is able to move the snake using the WASD-keys
- The user can register a new account with a chosen password and username
- Authentication using a username and a password, via a database
- The user is able to start a new game by clicking a button/pressing a key
- The game shall place the snake at the bottom-left of the screen when a new game starts
- When a new game is started the snake initially moves upwards if no user-input is provided
- The initial length of the snake shall be *9* amount of squares
- A snack will spawn in a random location outside of the snake
- The score of the player is increased by 10 each time the head of the snake is on the same square as a snack
- The length of the snake is increased by one square each time the head of the snake is on the same square as a snack
- The user is able to view his/her score while playing the game
- The game will end if the head of the snake hits one of the outer walls
- The game will end if the head of the snake hits the body
- After each play, the player can enter his/her username together with the score
- After each play, the top 5 global scores are shown
- The game shows a 16x16 grid on the board that has 256 squares

Non-functional:

- The game is written in Java
- The authentication of username and password is done by communicating with a database
- The scores of each individual player are stored in the database
- A SQL Database is used to store the user information
- The JDBC Driver is used to interact with the database
- When communicating with the database prepared statements are used to prevent code-injection

Should Haves:

Functional:

- The game ends after the player dies
- The user is able to start a new game after the first game is over by clicking a button or pressing a key
- The level will be reset after a game ends
- The user is able to pause the current game by pressing a key
- If the game is paused, the user is able to resume the current game by pressing a key
- There is a clear indication which part of the snake is the head and which part of the snake is the tail
- When no key has been pressed during the previous frame, the snake will keep moving in the direction in which it was going the previous frame
- The player is not able to move the snake in the opposite direction of which it is currently going
- The snake is visibly moved every *166.6667* amount of (milli)seconds

Non-functional:

- There is a >75% meaningful test coverage
- Test-Driven Development is used

Could Haves:

Functional:

- There is a menu in which the user can change settings (e.g. sounds off, grid visibility, traversal through walls)
- The player is able to toggle sound
- The player is able to toggle grid visibility
- Background music is played when the user is in the main menu/game screen
- There is a sound indication when the player collects a snack
- There is a sound indication when the player dies
- The player can choose whether or not the snake can go through walls and wind up on the other side of the board
- The player is able to unlock a new skin for the snake which changes the appearance of the snake
- There is a static(not moving) enemy that impedes the movement of the user, if the player collides with the enemy then the player will die
- Two players can play at the same time, one plays with the WASD-keys and the other player plays with the arrow keys. If one of the heads of the snakes collides with the body of the other, that snake will die
- After a certain amount of points (1000) the background image of the game changes, a new image, texture or colour is displayed.
- There are several power-ups which change something about the game (e.g. the snake speeds up, the snacks are worth double points)
- There is an AI which the player can compete against for a snack. This is a new game mode.

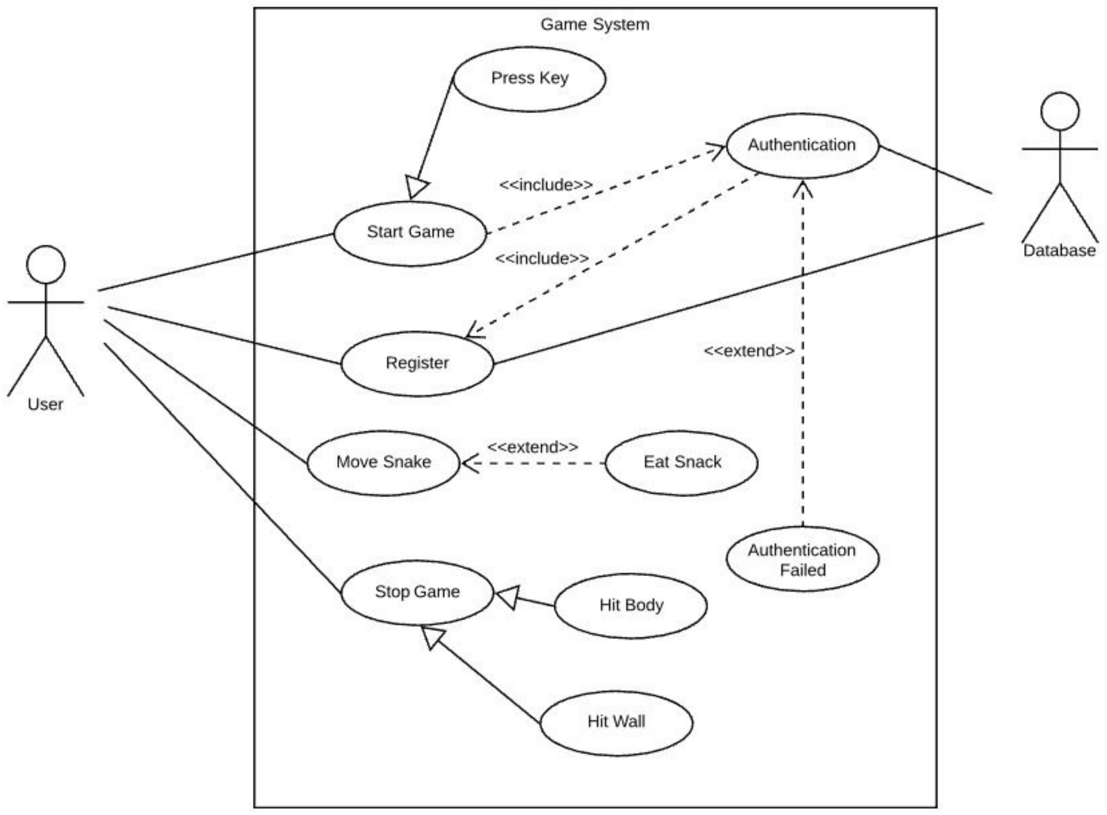
Won’t Haves:

Functional:

- The player is able to participate against other players online
- 3D support for the game
- A leaderboard containing all players
- Dynamic background for the game
- Story mode
- The player is able to control the movement of the snake using the mouse

EXERCISE 2: MODELLING USE CASES

The use case diagrams for the chosen requirements



Natural language description of the use case scenario

- Use Case: Register
- Author: Chaiwon Park
- Date: 27/11/2019
- Purpose: The user can register a new account with a chosen password and username
- Overview: The user enter the username and password so that the new account is created. The database validates if the username is duplicated or not. If the validation is successful, the username and password will be saved in the database; otherwise (unsuccessful validation) a validation failure message is shown, user needs to enter another username, enter the password again.
- Cross-reference:Requirements R2, R3
- Actors: User and Database
- Pre-conditions:
    - The system should be connected to database
- Post-conditions:
    - No duplicated username
    - Username and password are successfully saved in database
- Flow of events:

User Action	System Action	Database Action
1. User starts the program	2. System loads the register screen	
3. User enters username and password		4. Validate the username, in case there is any duplicated username
		5.Username and password saved in database

- The alternative flow of events:
    - Step 4: The validation failed. An error message is displayed, the user needs to enter another username and enter the password again.

- Use Case: Start Game
- Author: Chaiwon Park
- Date: 27/11/2019
- Purpose: The user is able to start a new game by clicking a button/pressing a key
- Overview: The start screen with the start button is provided to the user, so he/she can click on it. After clicking on it, the system should load screen, with the snake moving upwards
- Cross-reference: Requirements R3, R4, R5, R6
- Actors: User
- Pre-conditions:
    - The user is authenticated
    - The system is started
- Post-conditions:
    - The game screen should be ready so that user can move snake
- Flow of events:

User Action	System Action
	1. System loads the start screen
2. User clicks a button/presses a key	3. The game starts, loads the game board

- Use Case: Move
- Author: Johan Hensman
- Date: 27/11/2019
- Purpose: Being able to move the snake with WASD-keys
- Overview: After the user has logged in and started the game, the snake will automatically move upwards, but the user can move the snake. If the player presses D, the snake will move to the right. If he/she presses A, it will move the left. If he/she presses W, it will move upwards. If he/she presses S, it will move downwards. If the player doesn't press anything or presses other keys the snake will keep moving in the direction of the last pressed button.
- Cross-reference: Requirement R1, R5
- Actors: The player
- Pre-Conditions:
    - The system is started
    - The player has already authenticated
    - The WASD-keys must be working

- Post-Conditions:
    - The snake moves in the correct direction
- Flow of events

User Actions	System Actions
	1. Starts the game
2. User presses D	3. The snake moves to the right
4. User presses W	5. The snake moves upwards
6. User presses A	7. The snake moves to the left
8. User presses S	9. The snake moves downwards

- The alternative flow of events:
    - Step 2: The user presses a random key (not WSAD). The direction of the snake doesn't change.

- Use Case: Authentication
- Author: Johan Hensman
- Date: 27/11/2019
- Purpose: Authentication using a username and a password, via a database
- Overview: After the player has registered, the player is able to enter his/her login credentials at the start of the game. The database validates the combination by comparing the credentials in the database with the input. If the player correctly enters their username and password, he/she can start playing the game. If either the username or password is wrong, an error message is shown.
- Cross-reference: Requirement R3, R19
- Actors: The player
- Pre-Conditions:
  - The computer is able to perform snake
  - The player has already registered
  - The database is accessible
- Post-Conditions:
  - The player is logged in
  - The player can start the game
- Flow of events:

User Actions	System Actions	Database Actions
1. The user enters his login credentials		2. Validates the credentials
	3. Let the user past the login screen to the game screen	

- The alternative flow of events:
  - Step 3: The validation failed. An error message is displayed, the user needs to enter the correct username and/or password and enter the password again.

- Use Case: Eat snack
- Author: Marina Wiemers
- Date: 28/11/2019
- Purpose: As an extension to move, the user can move the snake to collect snacks
- Overview: While a game is ongoing, the user can use the WASD keys to navigate the snake through the board to collect snacks. When the snake has collected a snack, the length of the snake will increase by one unit and the game continues as before. Furthermore, the score increased whenever a snack has been collected.
- Cross-reference: Requirements R10, R11
- Actors: The player
- Pre-conditions:
  - A game is ongoing
  - The player is authenticated
- Post-conditions:
  - The snake has increased by one unit.
- Flow of events:

User Action	System Action
	1. Starts game
2. User moves snake and collects a snack	3. The length of the snake increases by one unit.
	4. The score of the current game increases.

- Use Case: Stop Game
- Author: Marina Wiemers
- Date: 28/11/2019
- Purpose: The user can end the game by having the game hit the outer wall or itself
- Overview: After the start of a game, the user can move the snake and end the game by hitting one of the bounding outer walls. Alternatively, the game can be stopped if the snake's head hits its own body. If the user does not take any action using the WASD keys and the snake's head hits either a wall or itself, the game will end as well.
- Cross-reference: Requirements R13, R14
- Actors: The player
- Pre-conditions:
  - A game is on-going
- Post-conditions:
  - The game has ended
- Flow of events:

User Action	System Action
	1. Starts game
2. User lets snake hit wall or itself	3. Ends game

- The alternative flow of events:
  - If the user does not take an action or uses any of the non-functional keys leading to the fulfillment of one of the conditions, then the system will end the game.