Requirements Air Hockey

Game created by:
Matei Anton, Ben Provan-Bessel,Shaan Hossain,
Iona Savu, Krzysztof Garbowicz

Contents:

1. Functional Requirements
    1.1 Must Haves
    1.2 Should Haves
    1.3 Could Haves
    1.4 Won’t Haves
2. Non-functional Requirements

1.Functional Requirements

For the game Air-Hockey the functionality and service requirements are grouped under the Functionality Requirements.
Within these functional requirements four categories can be identified using the MoSCoW model for prioritizing requirements.

1.1 Must Haves:

- When the program starts the user is presented with an authentication window in which :
    - The user is able to register a new account with a unique username and password.
    - The user is able to type in his/her username and password to log in.
- The usernames and passwords are stored in a database.
- The passwords stored in the database are safely stored.
- After authentication the user sees the main menu from which he/she can choose to :
    - Start a new game
    - See the scoreboard
    - Exit the game
- When the user starts a new game he/she is taken to a new game session.
- The game shows a board with one puck and two paddles before the game starts.
- The game's board has a clear layout with two goals, one for each player.
- The game shall let the puck slide to one of the player's halves when the game starts.
- The game's clock should start when the game starts.
- The game's score should reset when the game starts.
- The game shall randomly slide the puck into one of the players halves when the game starts.
- The player shall be able to move his/her paddle in all directions.

- The paddle of player one moves:
    - To the left when the player one presses ‘A’ on the keyboard.
    - To the right when the player one presses ‘D’ on the keyboard.
    - Down when the player one presses ‘S’ on the keyboard.
    - Up when the player one presses ‘W’ on the keyboard.
- The paddle of player two moves:
    - To the left when the player two presses “left arrow key” on the keyboard.
    - To the right when the player two presses “right arrow key” on the keyboard.
    - Down when the player two presses “down arrow key” on the keyboard.
    - Up when the player two presses “up arrow key” on the keyboard.
- The game shall not allow the player to move his/her paddle outside of the board.
- The game shall not allow the puck to move outside of the board.
- The game shall not allow for the puck and the paddle to overlap.
- The player shall not be able to move his/her paddle beyond his/her half of the board.
- The player shall be able to touch the puck with his/her paddle.
- The game shall move the puck in the appropriate direction after the paddle-puck collision.
- The game shall move the puck in the appropriate direction after the collision of the puck and the side of the board.
- The game shall add one point to the score of the player when the puck crosses the goal line of the other player.
- The game shall reset the paddles and the puck to the initial state after the goal.
- The game shall slide the puck into the half of the player that lost a point.
- The game is finished when:
    - The player wins the game:

        - The player scores eleven points before the other player.
        - The player has more points than the other player, when the clock stops.
        - The player is still in the game after the other player left the current game session.
    - The players tie the game:
        - The clock runs out and both players have the same amount of points.
    - The player loses the game:
        - The opposing player scores eleven points.
        - The opposing player has more points than the player when the clock stops.
        - The player left the game while the game was still being played.
- The game shall not allow any movement on the board after the winner was announced.
- At the end of each play, the user should be able to enter his/her name together with the recorded score. The data is stored in the database.
- At the end of each play, the game should show the top 5 scores that have ever been recorded
- The program shall take the user back to the main menu after the game ends.
- After the user presses the Exit button the program closes.

1.1 Should Haves:

- When the program starts the user is presented a welcome screen with authentication options.
- The program has a verifying mechanism for safe registration.
- The program gives feedback in case of an unsuccessful try to log in.
- The program should take the user to the main menu after a successful login.
- The main menu has a settings tab with options the user can adjust.
- The player is able to start a game from the main menu.
- The player is able to pause or quit the game during its progress.
- The games board, the puck and the paddles shall be easily distinguishable and good-looking.
- The player is able to control his/her paddle with a keyboard/mouse controls.
- The game outputs sound effects during the game.
    - At the beginning and the end of the game.
    - On collision of the puck and a paddle and the puck and a side of the board.
- The game shall adjust the puck's momentum after a collision.

1.1 Could Haves:

- The user can select different game modes:
    - The user may choose to play a game against another user locally.
    - The user may choose to play a game against another player online.
    - The user may choose to play a game against AI.
- The user may choose to play a nonstandard version of air-hockey.
- The user has an option to play his/her own music in the background of the program.

1.1 Won't Haves:

- 3D Graphics
- Complex AI
- iOS version of the game

2.Non-Functional Requirements

Besides the provided functionality and services, design constraints need to be included in the requirements specification as well. These requirements do not indicate what the system should do, but instead indicate the constraints that apply to the system or the development process of the system.

- The game shall be playable on Windows10 and MacOS (10.13 (High Sierra) and up)
- The program shall be implemented in Java13
- The program shall be implemented according to the pull-based development principles
- The program makes use of libraries:
    - LibGDX
    - PostgresJDBC
- The program should use prepared statements in Java to avoid code-injection vulnerabilities.
- The program should work smoothly
- The game should bring joy to anyone who plays it