3-leveled maze Java-based video game – Product Backlog

After a while of deliberation and many different concepts, we eventually arrived at the conclusion to create a simple, 3-leved, maze game, in which the main goal of the player is to find the right path to escape the dangerous labyrinthine as soon as it is possible. We motivated our final decision to create a maze-like video game with the opportunity to venture into many diverse java concepts which are not necessarily strictly related to each other. Our primary focus during the process will be devoted to advanced object-orienting programming (the interaction and connection between different methods) and game design/ user experience. We will try to prioritize using various design patterns, which will greatly contribute towards creating well maintained code and environment. Subsequently, we are willing to gain significant experience in working with GitHub and Git, as the main tools for cooperation while creating the game project together, as well as the primary link between diverse ideas. With a view to learning about the management of our own GitHub repository, our project is going to be thoroughly documented and highly dependent on the connection between various branches. The game provides clear tutorial through movement and goals and provides feedback by showing the score and time during game play, traps and time limit act as punishment when player fails, fruits and time both encourage player to attempt for a higher score. For each of the following we assigned one of the 3 priority categories.

NAME	DEMO	NOTE	PRIORITY
Main Game Window	Main window for the game pops up after the application is run	The window will have set dimensions, however it	High

		should be resizable	
Maze Display Layout	A specific maze shows up after the user selected the level difficulty	All elements (interactable and non-interactable) should be visible	High
Player Character image	The character stays the same throughout the game, appears at start after running game	A character behaves according to the user's actions	High
Exit	The exit (game's goal) is visible for the player after	The player's goal is to reach the exit within the specific time	High
Timer GUI display	The timer of the current game is shown at the top of the GUI.	Player can see how long has the game run	Medium
Score display	After the successful attempt the score is shown	Player sees the score of the particular attempt.	Medium
Start and (pause?) buttons	User stops/starts the game by pressing the appropriate button.	Ability to influence the game behavior	High
Level Selection	Users select a level difficulty after starting a new game	3 different level difficulties free to choose at the beginning of each game	High

Pop up message	End the game to see an according message	A message depending on the game outcome	Medium
Introductory tutorial	Press a correct button to see the game information and possible moves	The user can see the goal and rules of the game	Medium
Player Movement	Use arrows to move the character	The character moves according to the user's behavior	high
Collision detection	Try moving into walls, corners movement does not succeed	The character cannot move into the walls and corners	Medium
Trap detection	Enter the trap	Entering the trap ends the game with failure.	High
Maze level easy	Select the easy difficulty level start the option of the maze	The easiest maze, straightforward to beat	High
Maze level medium	Select the medium difficulty level to start the second option of the maze	More challenging compared to the first one	High
Maze level Hard	Select the hard difficulty to display the third option of the maze	The hardest maze, many trips, little time	High
Level completion	End the game successfully by reaching the exit or fail the game	Shows the appropriate message accordingly to	Medium

	by entering a trap or running out of time	the outcome of the game	
Single path Verification	There is only one possible exit	Prevent accidental multiple exits	Medium
Background Music	BGM plays in menu and game, stop when game end	Use Java sound API	Low
Difficulty Game time	Different game time with different difficulties	Game over when reach 0	Medium
Score multiplier (Time Left)	Win with time left: score=fruits score * time-based multiplier	Multiplier formula = (time left/total time) *(maximum multiplier 4)	Medium
Fruits	Collect fruit for points	Fruit disappears, score increases, collect sound play	High
Fake Fruits	Collect fake fruit: control reverse for 10s, sound play	Need a countdown for reverse key-bind duration	Medium
Sound Effects	Play under following events: wall hit, real fruit, fake fruit, trap, lose win	Use Java sound API	Medium
Game Over/Victory Screen	Lose: "Game Over" screen. Win: "Victory" screen with score	Lose: play "lose" sound effect Victory: play "win" sound	low

		effect, score calculated	
Character animation	Move the character	Character moves in a fluid way	low