Requirement Analysis

Mariana Agudelo Salazar, Vanessa Sánchez Morales, Natalia Vargas

Case:

Client **User**	Karthik Naru (Owner of PipeMania) (*see Google Play*) Pipe Mania players
Functional Requirements	- RF1: Start a new game - RF2: View scores - RF3: Place pipe line - RF4: Verify sewer system
Problem Context	*The Pipe Mania game consists of a sewer system simulation. In this game, the player can locate three different types of "pipes" within an 8x8 board, with the objective of connecting the "water source" to the "draining pipe". The users can also view a players' ranking according to the scores gained, with each of the players' names.*
Non Functional requirements	- The program must only use linked lists and tree structures. - The program must make use of recursion. - The scores table is deleted once the program closes. - The project must use a version control software such as git.

Identifier and Name	*RF1: Start a new game*		
Summary	*The system must start a new game when the user selects option 1 on the main menu. The system will ask the user for their nickname and save it to showcast the game scores later. Then, the system will display an 8x8 board with an "F" and a "D", each one representing the water source and the draining pipe respectively. Besides an option menu to play the game will be shown.*		
Input	**Input name**	**Data type**	**Valid condition**
	nickName	String	*Can't be empty.*
Result or Postcondition	After the system receives the data entered by the user, it will show the game options (place pipe, verify and exit) and the 8x8 board game, randomly locating an "F" and a "D", each one representing the water source and the draining pipe respectively. It also takes the time in which the player started the game in order to calculate its game time.		
Output	**Output name**	**Data type**	**Format**
	gameBoard	DoubleLinkedList	*Example:* *[x x x x x x x x x* *x x F x x x x x x*

			x x x x x x x x x *x x x x x x D x x …]*
	gamePlayMenu	String	*1. Place a pipe* *2. Verify* *3. Return*

Identifier and Name	*RF2: View scores*		
Summary	*The system must show, in descending order, the final scores of the players that have played and finished a game. This will happen if the user chooses the "View scores" option (2) in the menu.*		
Input	**Input name**	**Data type**	**Valid condition**
	N/A	N/A	*N/A*
Result or Postcondition	After each successful game, the system calculates the user's score and adds it to a Binary Search Tree. The system will calculates the player´s final score using the formula: $Score = usedPipes * 100 - (60 - timeInSeconds) * 10$ and will show the scores of the players if the user chooses the "View Score" option. The time each player takes to play the game will be calculated during the game.		
Output	**Output name**	**Data type**	**Format**
	finalScoreList	Tree	*Message by console showing the scores* *1. A xxx points* *2. B xxx points* *…*

Identifier and Name	*RF3: Place pipes*		
Summary	*After entering the option 1 on the Start a new game menu (see RF1), The system must allow the user to locate a "pipe" in an specific position of the 8x8 board, by asking for the coordinates in which the new pipe will be located and the type of pipe.*		
Input	**Input name**	**Data type**	**Valid condition**
	xCoordinate	int	*Must be a Integer between [0,7]*

	yCoordinate	int	*Must be an Integer between [0,7]*
	pipeType	int	*Must be a number between [1,3]. Each of these options corresponds to a different type of pipe:* *1. Horizontal (=)* *2. Vertical (| |)* *3. Circular (o)*
Result or Postcondition	The system searches for the coordinate that the player chose. If said coordinate is located within the possible range of the board and it isn´t occupied by an "F" or "D", the type of pipe that the player chose will be displayed in the [x,y] coordinate of the board. Else, the board will appear with no changes made.		
Output	**Output name**	**Data type**	**Format**
	gameBoard	DoubleLinkedList	*8x8 board of "x" characters, contains an F (water source), a D (draining pipe) and the pipes ("=","| |","o")*

Identifier and Name	*RF4: Verify sewer system*		
Summary	*The system must verify that the pipes' solution provided by the user is valid, meaning, the water source ("F") is connected to the draining pipe ("D") with a correct usage of the pipes ("=","||","o"). Then, the user will see if their option is correct or not, and depending on that, the game will close.*		
Input	**Input name**	**Data type**	**Valid condition**
	N/A	N/A	N/A
Result or Postcondition	The system checks that the pipes are located correctly according to their type and direction. The "F" and "D" must be connected, allowing the water flow, with the "=" pipes going one next to the other, and the "| |" pipes going one under the other. Besides, an "o" cannot be next to another "o" or to the "F" and "D", and it can only be used to do a 90° spin with the pipes. If the solution is correct, the game is closed , the time is taken, calculated and the score saved; else, the game continues and the menu and board will be displayed again.		
Output	**Output name**	**Data type**	**Format**
	message	String	*It shows whether the user's options were*

			correct or not. It can be: ● *"The solution is correct"* ● *"The sewer system is not working"*