# Synoptic Project B
# Maze Game

## Design

Amy O'Toole

**Specification**

The gaming company 'Olde Worlde Phunne' wished to increase the number of traffic on their website. It was therefore decided that offering a maze game to any potential customer who signed up to the website would be a way in which to do this. The ask on my side was to design, create and test the initial version of the maze game.

After looking at the requirements of this project, I decided that it would be best to create this component as a website rather than a stand-alone program that needed to be downloaded. I also decided to create the application using React.js since it suited the requirements of the project and I have previous knowledge surrounding it.

I took the decision that my version of the Maze Game would be comprised of five individual levels, which as the user progresses through them the difficulty of the maze will increase. The user will be able to navigate through each maze by using the mouse rather than using keyboard inputs. Each maze level will also have its own time limit that will increase in accordance to the difficulty of the level. I have decided to store the mazes within a 'Maze' array, which will be hosted within a JSON file. This makes the data required for the application to render each maze easily accessible.

Throughout the game, a score will be generated for each level the user completes and will be displayed at the bottom of the form along with the time remaining counter. The score achieved for a level will be generated from the time remaining upon completion of the maze in milliseconds. For example, to complete level 1 the user will have a total of 500 milliseconds and therefore a total score of 500 can be achieved. Thus, if the user takes 3 seconds to complete level 1 then they will achieve a score of 200 since 500-300= 200. Once the user has completed the level, the time remaining for that level will be added to the score and their current total score will be displayed.

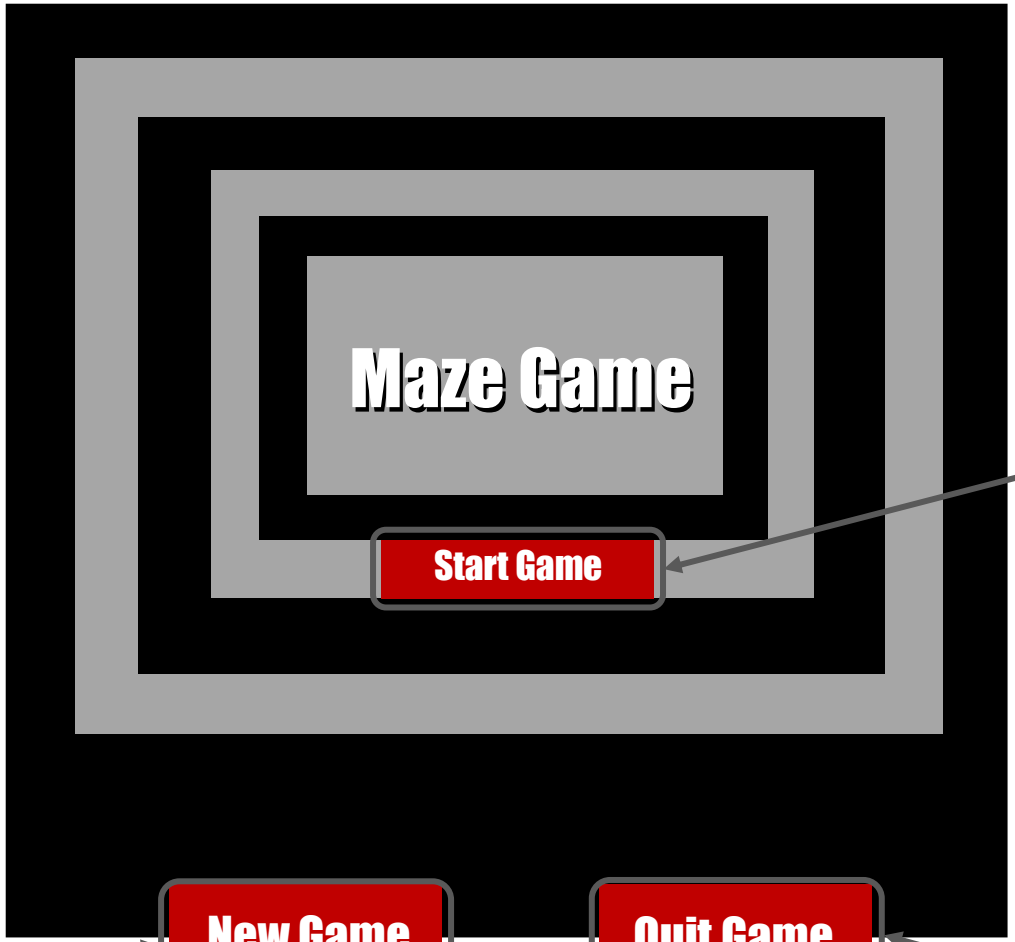
The application will run through a series of pages which will be triggered by either a user input or certain defined events. To navigate the pages I have decided to use a Hash Router in conjunction with a switch case. Logic used throughout the application will determine which window location the Hash Router needs to point to. For example, if the user completes a level then the application will check if that level was level 5 and if so the 'Completed' screen needs to be loaded instead of the 'Winner' screen.

For my version of the 'Maze Game' I have made the following assumptions:

- The user will be running the game on a desktop computer rather than a mobile
- The user will have access to and the ability to use either an external mouse or trackpad

## Form Designs

**Title Screen**




Start Game button – when clicked this will load the 'Start' screen

New Game button – when clicked this will restart the game by loading the 'Title' screen, setting the current maze back to the first maze, setting the timer to 5 seconds and setting the score to 0

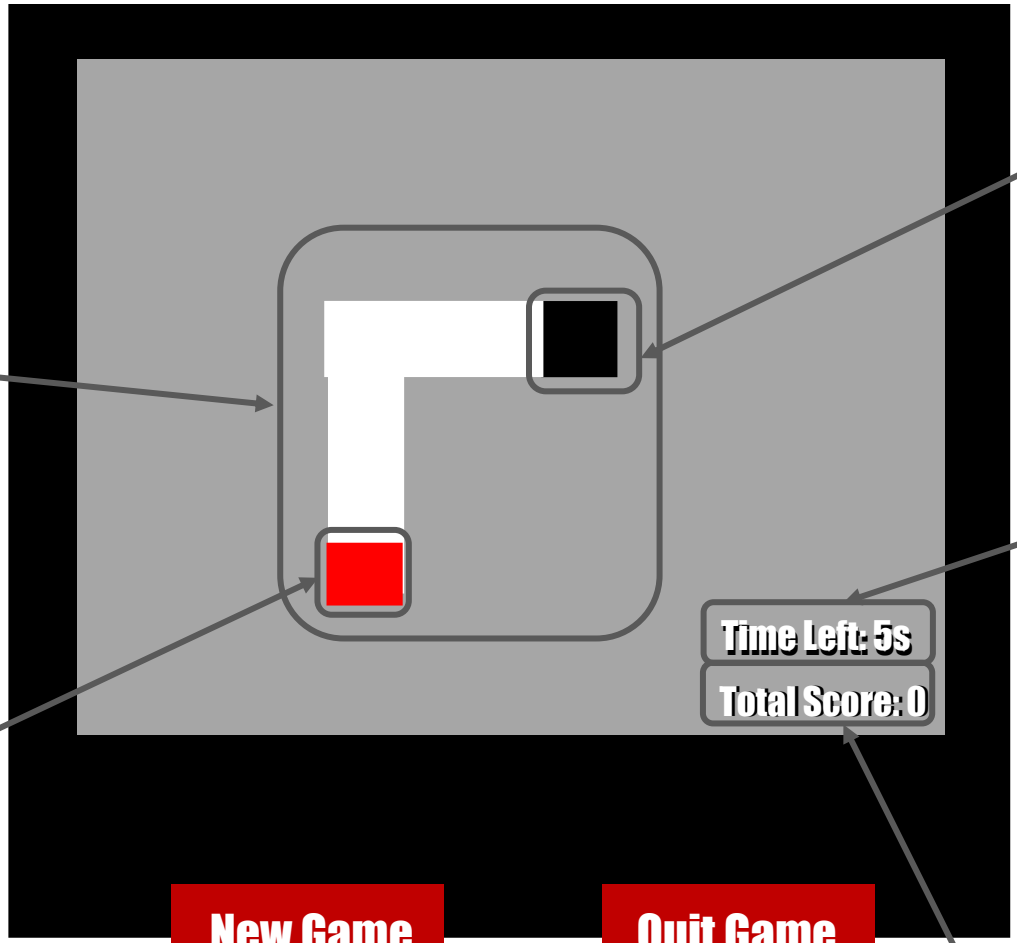
Quit Game button – when clicked this will load the 'Quit' screen with the user's total score

**Start Screen**


Level text – the number will be a variable that will display which level the user is about to start

Start Maze button- when clicked this will load the 'Maze' screen with the first maze and start the timer

**Maze #1 Screen**




Maze – The maze for each level will be made out of individual SVG rectangles pulled from an array of mazes hosted within a JSON file.

First Rectangle – this will be the starting area for the maze

Timer – this will display the time the user has remaining to complete the maze. The number of seconds provide will increase per maze in accordance to that level's difficulty

Last Rectangle - The user will complete the maze once they have reached this rectangle. This will be the last rectangle within the current level's array

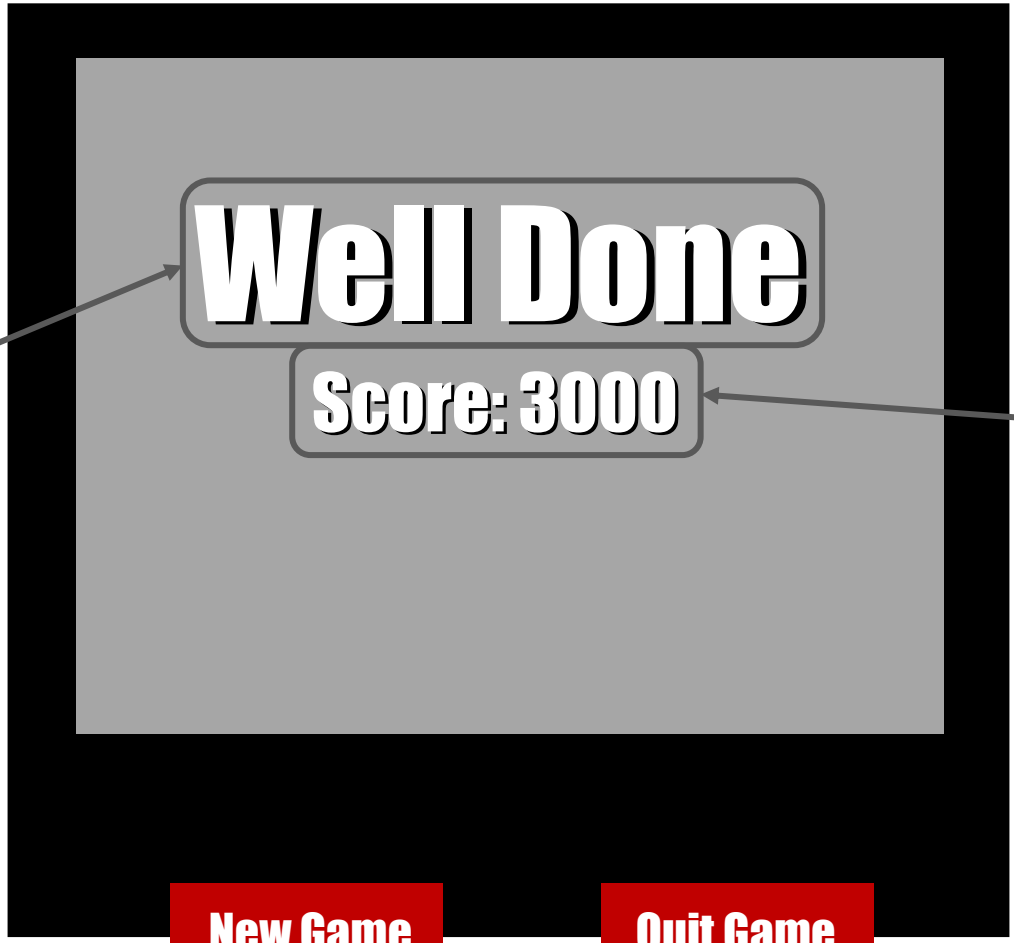
Score – this will display the user's running score total and will be updated once a user has completed a level. The score awarded to the user will dependant on the time remaining, as any remaining milliseconds will be the score awarded for that level

**Winner Screen**


Next Level button – when clicked this button will load the 'Maze' screen as well as setting the maze and the timer to the next object within their respective arrays

Completed text – this will load as part of the 'Winner' screen, the level number displayed will be a variable and will change dependant on which level the user has just completed

**Completed/Quit Screen**




Done-text – the user will see this banner once they either complete all three levels or click on the 'Quit Game' button

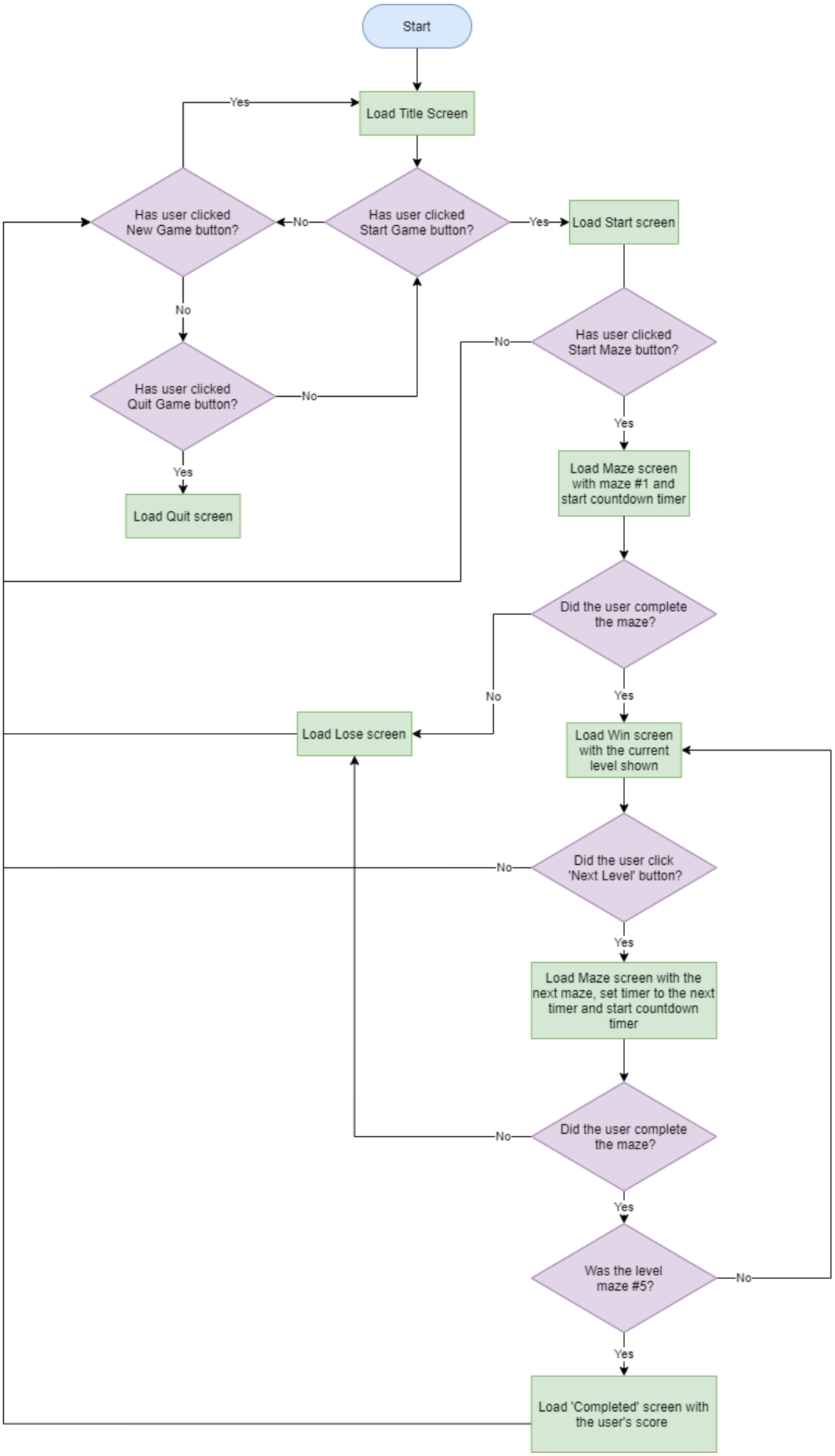
Score-text – this will display the user's total score

**Lose Screen**


Game Over text- this will be displayed as part of the 'Lose' screen to inform the user that they have lost the game

Lose screen – this will automatically be displayed when the user either runs out of time or goes outside the maze walls

## Flow Diagram



## Inputs, Outputs & Processing

**Inputs:**

- onClick 'Start Game' button
- onClick 'New Game' button
- onClick 'Quit Game' button
- onClick ' Next Level' button
- onClick 'Start Maze' button
- Mouse movement to navigate maze

**Processing:**

- If the mouse goes outside the maze boundaries then change the hash location to "#/lose" and set the timer to 0
- If the timer reaches 0 on any maze then change the hash location to "#/lose"
- If the current level = level 1 then set the timer to 500 milliseconds
- If the current level = level 2 then set the timer to 1500 milliseconds
- If the current level = level 3 then set the timer to 2500 milliseconds
- If the current level = level 4 then set the timer to 3500 milliseconds
- If the current level = level 5 then set the timer to 6000 milliseconds
- If the user completes the current level within the allotted time then add the remaining time from the countdown timer to the score
- If the user completes the 5$^{th}$ level then change the hash location to "#/completed" and display the user's total score
- If the user clicks the 'New Game' button then change the hash location to "#/title", reset the timer to 5 seconds, reset the score to 0 and reset the current maze array to 0
- If the user clicks the 'Start Game' button then change the hash location to "#/start"
- If the user clicks the 'Quit Game' button then change the hash location to "#/quit" and load the user's score
- If the user clicks the 'Start Maze' button then change the hash location to "#/maze" using the first object within the maze array and start the countdown timer
- If the user clicks the 'Next Level' button then set the timer to the next object within the timer array, set the maze to current maze + 1 from the maze array and change the hash location to "#/maze"

**Outputs:**

- When the user clicks on the 'Start Game' button then the 'Start' screen is loaded
- When the user clicks on the 'New Game' button then the 'Title' screen is displayed, the score will be set to 0 and the timer will be reset to 500 milliseconds

- When the user clicks on the 'Quit Game' button then the 'Quit' screen will be loaded along with the user's score
- When the 'Next Level' button is clicked the 'Maze' screen will be loaded with the next maze within the maze array being shown
- When the 'Start Maze' button is clicked then the 'Maze' screen will be loaded with the first maze within the maze array loaded
- Display the user's score at the bottom of the form
- Display the time remaining in seconds at the bottom of the form