Testing

# Test Plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Number | Test Description | Input | Expected Output | Actual Output | Comments |
| 1 | Attempt to register whilst not connected to the server. | UN: greeg  PW: abcdef!  CF: abcdef! | Error label should provide a relevant error message after 3 seconds. | See Fig.1 | Works as expected. |
| 2 | Attempt to register an account with a server connection | UN: greeg  PW: abcdef!  CF: abcdef! | Label should inform user of success, close the form and create records in the Login and UserStats tables. | See Fig.2 | Works as expected. All subsequent tests will be run with this account in the database. |
| 3 | Attempt to register a new account with a taken username | UN: greeg  PW: abcdef!  CF: abcdef!  Database not reset from test 2. | Label should inform user that the username is taken. | See Fig.3 | Works as expected. |
| 4 | Attempt to register an account with invalid password fields | PW: abcde  Then  PW: abcdefgh  Then  PW: abcdef!  CF: xyzxyz | Label should inform the user their password is too short, then that it has no special characters, then that their passwords do not match. | See Fig.4 | Works as expected. |
| 5 | Attempt to log in when not connected to the server. | UN: greeg  PW: abcdef! | Label should inform the user that they are not connected after 3 seconds. | See Fig.5 | Works as expected. |
| 6 | Attempt to log in with incorrect credentials. | UN: greeg  PW: xyzxyz | Label should give a message indicating that the credentials are incorrect. | See Fig.6 | Works as expected. |
| 7 | Attempt to login with correct credentials | UN: greeg  PW: abcdef! | Access is granted: the login form hides and the MazeParameter form opens. | See Fig.7 | Works as expected; the login must have worked since it is logged in the server and the parameter form greets the user in the form title. |
| 8 | Attempt to generate a recursive backtrack maze with odd numbered dimensions. | Algorithm: Recursive Backtrack  Width: 25  Height: 25  Exit: Centre  RemoveWalls: 0 | The maze should generate with the given parameters. The server should log the activity. |  |  |
| 9 | Attempt to solve the maze with depth-first search. | Algorithm: Depth First search | The maze should show a solution with a purple line. |  |  |
| 10 | Generate a maze with the growing tree algorithm with different width and height dimensions and many removed walls. | Algorithm: Growing Tree  Width: 25  Height: 40  Exit: Border  RemoveWalls: 40 | The maze should generate with the given parameters. The server should log the activity. |  |  |
| 11 | Attempt to solve the maze with Breadth-First search. | Algorithm: Breadth First search | The maze should display the best solution to the user. |  |  |
| 12 | Save the maze locally. | n/a | The client should display a file explorer window and allow the user to save an image of their maze. |  |  |
| 13 | Attempt to save the maze to the server with no name. | [no input] | The client should display an error message stating that the name field is required. |  |  |
| 14 | Attempt to correctly server save the maze. | Name: test | The client should display a message indicating that the maze has been saved. The server should log the activity. |  |  |
| 15 | Attempt to generate a Wilson’s maze with minimum sizes. | Algorithm: Wilson’s  Width: 2  Height: 2  Exit: Border  RemoveWalls: 0 | The maze should generate with the given parameters. |  |  |
| 16 | Attempt to solve the maze with the Maze-routing algorithm | Algorithm: Maze-Routing | The maze should show a solution with a purple line. |  |  |
| 17 | Attempt to load the maze saved as “test” from test 14. | Press Get Mazes. Then select the “test” maze and press Load Maze. | The maze should load from the server. |  |  |
| 18 | Attempt to delete the test maze from test 14 | Press Get Mazes and select the “test” maze. Press Delete Maze and then Get Mazes again. | The client should show that it found 0 mazes, indicating that the maze has been deleted. |  |  |
| 19 | Generate a small Wilson’s maze and use the keyboard controls to solve it. | WASD controls.  Algorithm: Wilson’s  Width: 5  Height: 5  Exit: Border  RemoveWalls: 0 | A timer should be displayed, which stops when the user reaches the exit. |  | This test will be repeated multiple times to record data for statistic testing. |
| 20 | Request local times from server | Stat type: Best Times  Global?: False | The times generated from test 19 should be displayed in order. |  |  |
| 21 | Request local mazes generated from server | Stat type: Mazes Generated  Global?: False | The pie chart should display W’s:RB:GT in a 2:1:1 ratio. |  |  |
| 22 | Create another account and generate 1 recursive backtrack maze and solve it manually. Then request each stat type. | UN: greeg2  PW: abcdef!  CF: abcdef!  Algorithm: Recursive backtrack  Width: 5  Height: 5  Exit: Border  RemoveWalls: 0  WASD controls.  Stat type: Best Times  Global?: False Stat type: Best Times  Global?: True Stat type: Mazes Generated  Global?: False Stat type: Mazes Generated  Global?: True | The local best times should have only 1 time, whilst the global best times should display the times set in the “greeg” account. The local mazes generated should be 1:0:0 in favour of recursive backtrack, while the global should now show W’s:RB:GT in a 2:2:1 ratio. |  |  |

# Screenshots

|  |  |
| --- | --- |
| Figure # | Screenshots |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |