CMSC 447
Software Test Report (STR)

Name	Role	Signature

1 Scope

This section shall be divided into the following paragraphs.

1.1 Identification

This document describes the testing of a web application that simulates a customizable Conway's Game of Life. This application will run on the current version of chrome, 73.0.3683.103 and the current version of Firefox ESR 67.0+.

1.2 System overview

The purpose of this system is to provide users a customizable version of Conway's game of life in the form of a web application. Additionally, the system shall provide the user with a means of customizing the appearance, speed, and functionality of Conway's Game of Life. This system will be developed and tested over a three-month period by a group of six. The operation of the system shall be accessible for the software sponsor, acquirer and user, Geoff Weiss, and Russell Cain. The development team does not have access to a support agency. This system is operable on any computer installed with the current releases of Chrome and Firefox ESR (defined in paragraph 1.1) and will be developed on the UMBC campus.

1.3 Document overview

This document will describe any relevant information about the results of each test listed in our Software Test Document (STD).

2 Referenced documents

This document will make references to our SRS and our STD. These documents will be provided to our customer and the distribution and/or accessibility of the documents will be determined by the customer.

3 Overview of test results

This section shall be divided into the following paragraphs to provide an overview of test results.

3.1 Overall assessment of the software tested

All required features work correctly. There were a few errors that we detected through testing, but all were fixed and tested again for the official release. The test results will be from the initial testing process.

3.2 Impact of test environment

The testing accounts for the standard usage of the web application. The testing environment however will not include a large amount of concurrent users, which makes our testing relevant for all cases except for wide-use.

3.3 Recommended improvements

To improve the testing process, there should be more tests regarding the usage of many features at once. Lastly, smaller unit tests should've been done on each feature, time constraints limited the amount of testing we could do.

4 Detailed test results

This section shall be divided into the following paragraphs to describe the detailed results for each test. Note: The word "test" means a related collection of test cases.

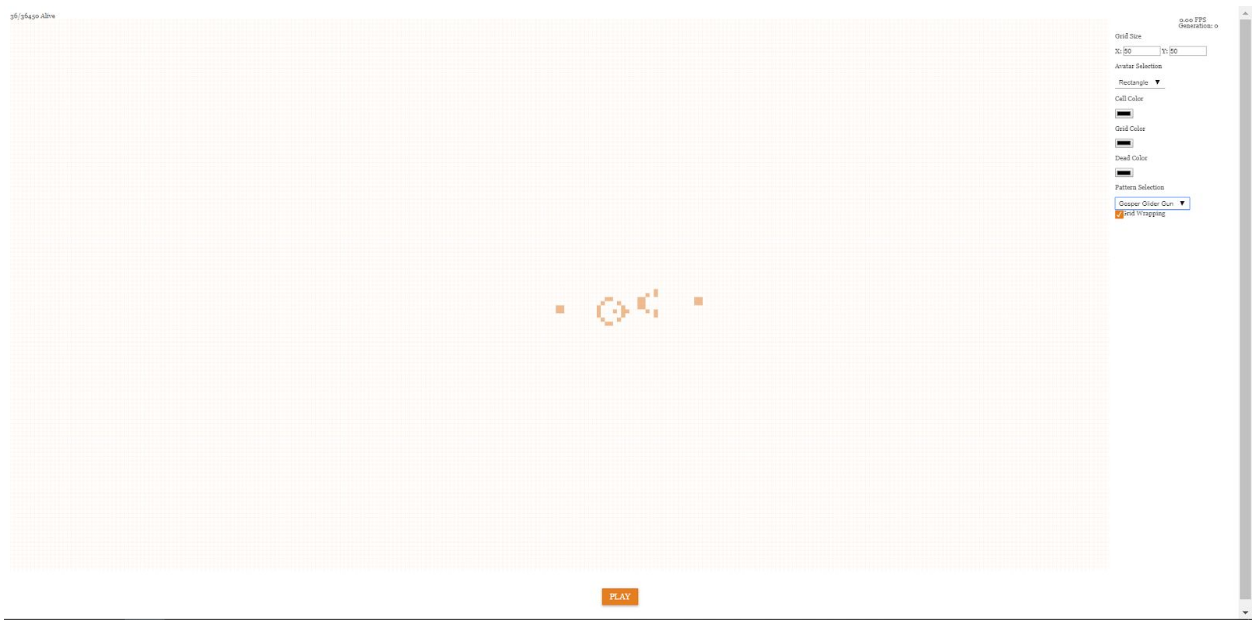
4.1 Webpage Environment Testing

Webpage Environment Testing will include all of the testing of the compatibilities between our Conway's Game of Life web application and the web browsers listed in section 2.1 of our SRS. The see the testing procedure, look in section 4.1.1.6 of our STD.

4.1.1 Summary of test results

The test evaluated all of these requirements to be a success. The Chrome and Firefox pages both include all of the relevant features without looking too far off visually.

The following is a comparison of both screenshots:
Chrome:



36/36450 Alive

0.00 FPS
Generation: 0

Grid Size

X: 50 Y: 50

Avatar Selection

Rectangle

Cell Color

■

Grid Color

■

Dead Color

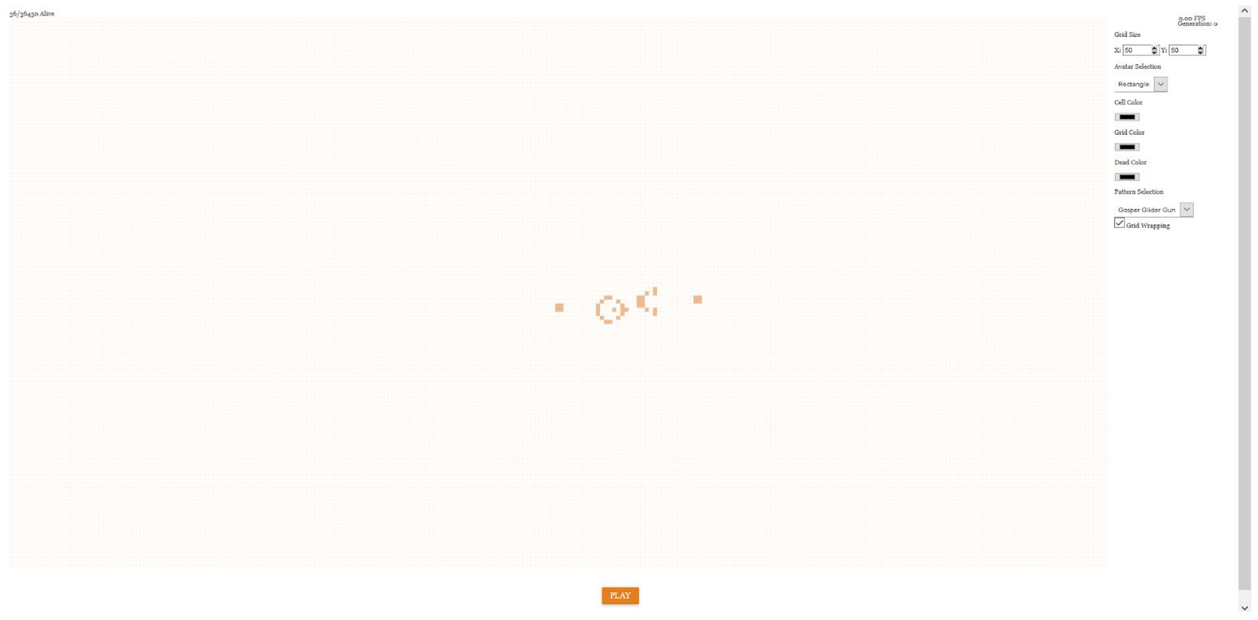
■

Pattern Selection

Gosper Glider Gun

✓ Grid Wrapping

Firefox:



36/36450 Alive

0.00 FPS
Generation: 0

Grid Size

X: 50 Y: 50

Avatar Selection

Rectangle

Cell Color

■

Grid Color

■

Dead Color

■

Pattern Selection

Gosper Glider Gun

✓ Grid Wrapping

Note: These screenshots may be of an earlier build compared to release.

4.1.2 Problems encountered

No problems were encountered during our testing process.

4.1.3 Deviations from test cases/procedures

No deviations to the testing process were made.

4.2 Pattern Customization Testing

Pattern Customization Testing will refer to the testing all requirements related to the customization of cells' starting position. The requirements are listed in section 2.2 of the SRS and the test procedures can be found in section 4.1.2.6 in the STD.

4.2.1 Summary of test results

All tests were evaluated as a success except for requirement 2.2.1.3. The following is a table of each requirement and their sub requirements. If a requirement is partially a success, then either a sub requirement or itself had a problem, or the feature was not fully implemented yet.

Requirement #	Success(s)/Fail(f)/Partial Success(p)	Notes
2.2.1	s	All sub requirements were successful.
2.2.1.1	s	The ten preset patterns change the board.
2.2.1.2	s	Randomly inserts acorns across the board. Each of the five tests of the random feature made a different board.
2.2.1.3	p	Clicking on the board will simply add a dot. Using ctrl and clicking will remove the dot. Using the draw feature while the game is playing does not cause any other error within the game. Clicking however is not

		accurate, and clicking whitespace outside of the grid can fill the grid.
2.2.1.4	p	Uploading a file changes the pattern. Uploading a non text file will not crash the site, uploading applications will notify the user that the file is invalid. Uploading a non text file will sometimes create dead cells for no apparent reason.

Retest results

File upload was fixed. The draw feature still is inaccurate, but the whitespace bug was fixed. All requirements that didn't score an "s" for their first tests were retested.

Requirement #	Success(s)/Fail(f)/Partial Success(p)	Notes
2.2.1.3	p	Clicking on the board will simply add a dot. Using ctrl and clicking will remove the dot. Using the draw feature while the game is playing does not cause any other error within the game. Clicking however is not accurate.
2.2.1.4	s	Uploading a file changes the pattern. Uploading a non text file will not crash the site, uploading applications will notify the user that the file is invalid. Uploading a non text will give the error message of invalid file type as it should.

4.2.2 Problems encountered

4.2.2.1 Mouse Pointer Accuracy

Drawing cells on the board tends to be inaccurate. The offset seems to be based on the size of the grid. The basic use test case in the STD detected this problem.

4.2.2.2 Parts of Site Draw on the Grid When Clicked

Using certain options will sometimes draw on the grid, this can make the user input certain cells they normally wouldn't. The basic use test case in the STD detected this problem.

4.2.2.3 Uploading non Text Files Creates Live and Dead Cells

The file uploader will allow you to upload any file if it is under 1MB. This allows for non text files to be uploaded. These files still create patterns despite not being a correct file type. This was detected with the test procedure as described in the STD. The website however

4.2.3 Deviations from test cases/procedures

No deviations from the testing process were made.

4.3 Rule Customization Testing

Rule Customization Testing will refer to the testing all requirements related to the customization of the rules of the game. The requirements are listed in section 2.3 of the SRS and the test procedures can be found in section 4.1.3.6 in the STD.

4.3.1 Summary of test results

All of our current implemented features are functional. No bugs were detected.

Requirement #	Success(s)/Failure(f)/Partial Success(p)	Notes
2.3.1	s	All sub requirements successful.
2.3.1.1	s	Cells with exactly three neighbors do come alive.
2.3.1.2	s	Cells with one or less neighbors die in the next generation.

2.3.1.3	s	Cells with four or more neighbors die in the next generation.
2.3.1.4	s	Live cells with two or three neighbors stay alive in the next generation.
2.3.2	s	All implemented sub-requirements Successful.
2.3.2.1	s	Upon setting all the rules to die, all the cells die.
2.3.2.2	s	Upon setting all rules to “nothing” cells do not grow or die.
2.3.2.3	s	Upon setting all rules to “grow” all cells that were uninhabited become live and current live cells stay live.
2.3.2.4	s	Using alt + click will set a cell that will always be dead, this will be set to be a clear color.
2.3.2.5	s	Using shift + click will set a cell that will always be alive, its color will be the current color of the live cells.

4.3.2 Problems encountered

No problems were encountered during testing.

4.3.3 Deviations from test cases/procedures

No deviations occurred during testing.

4.4 Graphics Customization Testing

Graphics customization testing refers to the testing of all of the requirements which change the visual appearance of the board and cells. To see the requirements that will be tested, refer to section 2.4 of the SRS and section 4.1.4.1 of the STD. The testing procedure that is correlated to this is in section 4.1.4.6 of the STD.

4.4.1 Summary of test results

All of the tests were successful. The following is a table of each requirement and their sub requirements. If a requirement is partially a success, then either a sub requirement or itself had a problem, or the feature was not fully implemented yet.

Requirement #	Success(s)/Fail(f)/Partial Success(p)	Notes
2.4.1	s	All sub requirements are successful.
2.4.1.1	s	Choosing square changes all cells to square. This does not break anything if changed while the game is running.
2.4.1.2	s	Choosing circle changes all cells to circles. This does not break anything if changed while the game is running.
2.4.1.3	s	Choosing triangle changes all cells to triangles. This does not break anything if changed while the game is running.
2.4.2	s	All sub requirements are successful.
2.4.2.1	s	Live cells change to green when set to green. This does not break anything if changed while the game is running.
2.4.2.2	s	Dead cells change to blue when set to blue. This does not break anything if changed while the game is running.
2.4.2.3	s	Grid changes to red when set to red. This does not break anything if changed while the game is running.

4.4.2 Problems encountered

No problems were encountered during testing.

4.4.3 Deviations from test cases/procedures

No deviations occurred while testing.

4.5 Gameflow Customization Testing

Gameflow customization testing refers to the testing of all of the requirements which change the visual output of the game. To see the requirements that will be tested, refer to section 2.5 of the SRS and section 4.1.5 of the STD. The testing procedure that is correlated to this is in section 4.1.5.6 of the STD.

4.5.1 Summary of test results

All features are successful. No bugs or deviations happened during testing.

Requirement #	Success(s)/Failure(f)/Partial Success(p)	Notes
2.5.1	s	Game pauses when pause is pushed
2.5.2	s	All sub requirements successful
2.5.2.1	s	When slider is set to the left, the board does not move.
2.5.2.2	s	When slider is set to the right the application operates at 60 frames (generation sets) per second.
2.5.3	s	Upon setting a change to speed, generation skip or other settings, the effects take place and are apparent in the next generation.
2.5.4	s	Setting the frame skip to zero will make it skip no frames. Upon setting it to 1, every

		other frame is skipped (alternating patterns look the same visually between sets).
2.5.5	s	The game will display a notification, then will start a new game when the board is empty.
2.5.6	s	Refreshing the page, changing the grid size, and emptying the board all let the user start a new game.
2.5.7	s	Choosing acorn with default sizes leads to a steady state at around generation 210.
2.5.7.1	s	When the steady state was reached, text is displayed in the top left which will notify the user.

4.5.2 Problems encountered

No problems occurred during testing.

4.5.3 Deviations from test cases/procedures

No deviations occurred during testing.

4.6 Grid Testing

Grid Testing refers to the testing of all of the requirements which change the size or nature of the grid itself. To see the requirements that will be tested, refer to section 2.6 of the SRS. The testing procedure that is correlated to this is in section 4.1.6.6 of the STD.

4.6.1 Summary of test results

The results of the test were successful. The following is a table of each requirement and their sub requirements. If a requirement is partially a success, then either a sub requirement or itself had a problem, or the feature was not fully implemented yet.

Requirement #	Success(s)/Failure(f)/Partial Success(p)	Notes

2.6.1	p	The starting size properly responds once the user hits submit. Testing with 0 and negative sizes makes the board empty as it should. Inputting large grid sizes will break the grid. Refer to test cases to see exact numbers.
2.6.2	s	Toggling the setting takes effect immediately as it should. Using this during a game does not break the system. Gosper's Glider gun preset shows the bullets travelling around the grid.

Retest results

We added in a maximum grid size in order to fix the large grid size bug. Requirements are only retested if they didn't score an "s" on the first test.

Requirement #	Success(s)/Failure(f)/Partial Success(p)	Notes
2.6.1	s	The starting size properly responds once the user hits submit. Testing with 0 and negative sizes makes the board empty as it should. Inputting large grid sizes will give us an error message that tells us the grid size is too large.

4.6.2 Problems encountered

This paragraph shall be divided into subparagraphs that identify each test case in which one or more problems occurred.

4.6.2.1 Large Grid Size Test

Large grid size test refers to the test case where we give our grid large sizes to see how it operates. If using a value of x that is around 670 or more, then the grid is not visible. If using a value of around 2550 or more for y then the grid will also not be visible. The value of y that breaks this is based in the size of x, generally the higher x is the higher y would need to be, x=50 and y = 2550 seemed to be one of the pairings for this. We believe the deviations occur due to our engine's limitations on rendering board sizes, also there may be limitations within the browsers. The largest issue with this though is how the site allows for unbounded amounts of input from the user. The complete version of the software will have a fix for this issue which will bound user input.

4.6.3 Deviations from test cases/procedures

No deviations during testing were made.

4.7 Information Display Testing

Information Display testing refers to the testing of all of the requirements which display information about the current game to the user. To see the requirements that will be tested, refer to section 2.7 of the SRS. The testing procedure that is correlated to this is in section 4.1.7.6 of the STD.

4.7.1 Summary of test results

Most features behave correctly if the user were to input correct data. All requirements but 2.7.3 fulfill their functionalities and work correctly assuming correct use of the system. 2.7.1 and 2.7.2 both did not pass some edge cases that were tested for.

Requirement #	Success(s)/Fail(f)/Partial success(p)	Notes
2.7.1	p	Normal inputs for the grid size are accurate. Counter behaves correctly during the game. Does not respond properly to edge case inputs though. Negative numbers make the counter display data that is either wrong or not a number.
2.7.2	p	Normal inputs for the grid size are accurate. Counter behaves correctly during the game. Does not respond properly to edge case inputs

		though. Negative numbers, no input, and the letter e/E all make the counter display data that is either wrong or not a number.
2.7.3	p	Counter behaves correctly, except for when the frame skip feature is used. Does not count skipped generations.

Retest results

All bugs with the live cell and total cell counters were fixed by adding in a minimum and maximum grid size. The generation counter now takes the skips into consideration for calculation. All requirements that didn't score an "s" during the first testing process were tested again after the fixes.

Requirement #	Success(s)/Fail(f)/Partial success(p)	Notes
2.7.1	s	Normal inputs for the grid size are accurate. Counter behaves correctly during the game.
2.7.2	s	Normal inputs for the grid size are accurate. Counter behaves correctly during the game.
2.7.3	s	Counter behaves correctly.

4.7.2 Problems encountered

This paragraph shall be divided into subparagraphs that identify each test case in which one or more problems occurred.

4.7.2.1 Negative Sizes

The Negative Sizes test case refers to the testing of the displayed information when one or both of the fields for board size has a negative number. Doing this will result in the total cell count to display a X/Y if one number is negative. Doing this with two negative numbers will make a positive board count. The rule board count should be zero in both cases. This is recreatable with either field being negative with any number. The finished version of this software does not include this error.

4.7.2.2 Empty Size Fields

The Empty Size Fields test case refers to the testing of the displayed information when one or both of the fields for board size are emptied. Doing this will result in the total cell count to display NaN. This is recreatable with either field being empty. The finished version of this software does not include this error.

4.7.2.3 Non-Number Sizes

The Non-Number Sizes test case refers to the testing of the displayed information when a non-number is inputted into one of the board field fields. Using the letter e or E in either x or y, will make the total cell count display NaN. This is recreatable with only the letter E/e. The finished version of this software does not include this error.

4.7.2.4 Non-zero Generation Skip Value

The non-zero generation skip value test involves changing the value in the field to anything but zero. The generation counter does not take this skip into account, for example, setting it to 100 would not change the amount the generation counter would increase with each visual change between set of iterations that are viewable. To recreate this, change the generation skip field value to any number that is not zero. The finished version of this software will not include this error.

4.7.3 Deviations from test cases/procedures

This paragraph shall be divided into subparagraphs that identify each test case in which deviations from test case/test procedures occurred.

4.7.3.1 Non-zero Generation Skip Value

This deviation refers to the test case in section 4.7.2.4 of this STR. Originally, the testing procedures did not account for testing out different generation skip values. This deviation was made to ensure the generation skip counter will work as it should. The change in testing was the addition of checking how skipping the visual for generations effects. The exact procedure addition involved changing the generation skip to 1, a negative number, blank field, and a non integer character. All of these tests resulted in no change to the generation counter. This should still make our original test cases valid for what they tested, just not complete.

5 Test log

Test Name	Date - Time - Location	Notes
4.1 Webpage Environment Testing	5/7/19 - 6:00pm - UMBC	Testing performed on laptop, website hosted on github.
4.2 Pattern Customization Testing	5/7/19 - 6:00pm - UMBC	Testing performed on laptop, website hosted on github.
4.3 Rule Customization Testing	5/11/19 - 2:30pm - UMBC	Testing performed on laptop, website hosted on github.
4.4 Graphics Customization Testing	5/7/19 - 6:00pm - UMBC	Testing performed on laptop, website hosted on github.
4.5 Gameflow Customization Testing	5/11/19 - 2:30pm - UMBC	Testing performed on laptop, website hosted on github.
4.6 Grid Testing	5/9/19 - 10:00pm - UMBC	Testing performed on laptop, website hosted on github.
4.7 Information Display Testing	5/9/19 - 10:00pm - UMBC	Testing performed on laptop, website hosted on github.

6 Notes

6.1 Abbreviations

SRS	Software Requirement Specification
STD	Software Test Document
Firefox ESR	Extended Support Release

A. Appendixes

Appendixes may be used to provide information published separately for convenience in document maintenance (e.g., charts, classified data). As applicable, each appendix shall be referenced in the main body of the document where the data would normally have been provided. Appendixes may be bound as separate documents for ease in handling. Appendixes shall be lettered alphabetically (A, B, etc.).