**Project Sprint #2**

The SOS game is described in CS449HomeworkOverview.docx. You should read the description very carefully.

Your submission must include the GitHub link to your project and you must ensure that the instructor has the proper access to your project. You will receive no points otherwise.

**GitHub link:**

**https://github.com/rilesc555/sos\_game**

Implement the following features of the SOS game: (1) the basic components for the game options (board size and game mode) and initial game, and (2) S/O placement for human players ***without*** checking for the formation of SOS or determining the winner. The following is a sample interface. The implementation of a GUI is required. You should practice object-oriented programming, making your code easy to extend. It is required to separate the user interface code and the game logic code into different classes (refer to the TicTacToe example). xUnit tests are required.

|  |  |  |
| --- | --- | --- |
| SOS Icon  Description automatically generated Simple game Icon  Description automatically generated General game Board size  8 | | |
| Blue player  Icon  Description automatically generated S  Icon  Description automatically generated O | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | | O |  |  |  |  |  |  |  | |  |  | S | O | S |  |  |  | |  |  |  |  | S |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | S | | Red player  Icon  Description automatically generated S  Icon  Description automatically generated O |
|  | Current turn: blue (or red) |  |

Figure 1. Sample GUI layout of the Sprint 2 program

**Deliverables:**

1. **Demonstration (8 points)**

Submit a link to a video of no more than three minutes, clearly demonstrating that you have implemented the required features and written some automated unit tests. In the video, you must explain what is being demonstrated. No points will be given without a video link.

**YouTube/Panopto link: https://youtu.be/lE24sxxBJTw**

|  |  |
| --- | --- |
|  | **Feature** |
| 1 | Choose board size |
| 2 | Choose game mode |
| 3 | Start a new game of the chosen board size and game mode |
| 4 | “S” moves |
| 5 | “O” moves |
| 6 | Automated unit tests |

1. **Summary of Source Code (1 points)**

|  |  |  |
| --- | --- | --- |
| Source code file name | Production code or test code? | # lines of code |
| Src/app/page.tsx | production | 85 |
| Src/components/GameBoard.tsx | Production | 46 |
| Src/components/GameBoardPreview.tsx | Production | 42 |
| Src/components/GameOptions.tsx | Production | 140 |
| Src/game/SOSGame.ts | Production | 76 |
| Src/tests/SOSGame.test.ts | Test | 64 |
| Total | | 453 |

**You must submit all source code to get any credit for this assignment.**

1. **Production Code vs User stories/Acceptance Criteria (3 points)**

Update your user stories and acceptance criteria from the previous assignment and ensure they adequately capture the requirements. Summarize how each of the following user story/acceptance criteria is implemented in your production code (class name and method name etc.)

|  |  |
| --- | --- |
| **User Story ID** | **User Story Name** |
| 1 | Choose a board size |
| 2 | Choose the game mode of a chosen board |
| 3 | Start a new game of the chosen board size and game mode |
| 4 | Make a move in a simple game |
| 6 | Make a move in a general game |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **User Story ID and Name** | **AC ID** | **Class Name(s)** | **Method Name(s)** | **Status (complete or not)** | **Notes (optional)** |
| 1 Choose a Board size | 1.1 | GameOptions.tsx | const handleBoardSizeChange | Complete |  |
|  | 1.2 | See above | See above | Complete | Error message unimplemented (unnecessary) |
|  | 1.3 | See above | See above | Complete |  |
| 2 Choose the game mode of a chosen board | 2.1-3 | GameOptions.tsx |  | Complete |  |
| 3 Start a new game of the chosen board size and game mode | 3.1 | Page.tsx | useEffect(() => {  if (gameStarted) {  const game = new SOSGame(boardSize, gameMode);  setGameState(game);  }  }, [gameStarted, boardSize, gameMode]); | Complete |  |
| 4. Make a move in a simple game | 4.1 | Page.tsx | const handleCellClick = (row: number, col: number) => {  if (!gameState || gameState.getCell(row, col) !== '') return;    gameState.placeMove(row, col, selectedLetter, currentPlayer);  setGameState(Object.assign(new SOSGame(boardSize, gameMode), gameState));  setCurrentPlayer(currentPlayer === 1 ? 2 : 1);  }; | Complete |  |
| 5. Endgame conditions for simple game | 5.1 | Not implemented |  |  |  |
| 6. General Game Moves | 6.1 | Page.tsx | const handleCellClick = (row: number, col: number) => {  if (!gameState || gameState.getCell(row, col) !== '') return;    gameState.placeMove(row, col, selectedLetter, currentPlayer);  setGameState(Object.assign(new SOSGame(boardSize, gameMode), gameState));  setCurrentPlayer(currentPlayer === 1 ? 2 : 1);  }; | Complete |  |
|  | 6.2 | Not Implemented |  |  |  |
| 7 General endgame conditions |  | Not implemented |  |  |  |

1. **Tests vs User stories/Acceptance Criteria (3 points)**

Summarize how each of the user story/acceptance criteria is tested by your test code (class name and method name) or manually performed tests.

|  |  |  |
| --- | --- | --- |
| **User Story ID** | **User Story Name** |  |
| 1 | Choose a board size | Manual Test |
| 2 | Choose the game mode of a chosen board | Manual Test |
| 3 | Start a new game of the chosen board size and game mode | Automated test: In SOSGame.test.ts: “test('should initialize with empty board of correct size'…)” |
| 4 | Make a move in a simple game | Automated test: In SOSGame.test.ts: In test('should place S and O on the board correctly', ()… |
| 6 | Make a move in a general game | Manually performed tests checking if moves can be made in general game |

4.1 Automated tests directly corresponding to the acceptance criteria of the above user stories

You are required to use ChatGPT to create at least 2 unit tests. You also need to ensure that the generated user stories are correct, and refine them if not. At the end of the submission, provide the screenshots of your ChatGPT prompts and answers, along with errors ChatGPT made and you fixed. You may also use another LLM, including hosted locally. Points will be deducted if no screenshots are provided.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User Story ID and Name** | **Acceptance Criterion ID** | **Class Name (s) of the Test Code** | **Method Name(s) of the Test Code** | **Description of the Test Case (input & expected output)** |
| 3. Start a new game of the chosen board size and game mode | 3.1 | SOSGame.test.ts | test('should initialize with empty board regardless of game mode', | Input simple and general SOSGame components, each with size 5, output expected 5x5 empty board for each |
| 4. Make a move in a simple game | 4.1 | SOSGame.test.ts | test('should correctly place S and O moves on the board', | Input simple game component, 4 moves at various places, expected output 4 true moves and those spaces filled |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect. A black screen with white text

AI-generated content may be incorrect.