**Assignment: Designing a Software Solution for Battleship**

**Overview:**

For this assignment, we're diving into the exciting world of software development by creating a digital version of the classic game Battleship. We'll take it step by step, starting from a simple idea and building it into a comprehensive plan. This journey will mimic the software development lifecycle, emphasizing the importance of gathering requirements, documenting our progress, and crafting a user-friendly experience. In this initial phase, we'll sketch out the game's concept, pinpoint our target audience, and clarify the problem our software is designed to tackle. Our primary objective is to identify the essential requirements for the Battleship game, which will guide our development efforts as we proceed.

**Part 1: Requirements Gathering via an Interview**

**Interview Simulation:**

To give a taste of how we gather requirements in the real world, we're going to have a mock interview with a Battleship expert. This will help us pinpoint what’s needed for a digital version of the game, focusing on the mechanics, what users want, and the hurdles we might face in software development.

**Interviewer**: “Thank you for taking the time to speak with me today. To start off, could you explain the core mechanics of Battleship?”

**Expert**: “Absolutely! Battleship is a game for two players, and each player has their own grid. They secretly position five ships of varying lengths on this grid. The ships can be arranged either horizontally or vertically, but they can’t overlap. The main objective is to find and sink all of your opponent’s ships by calling out specific grid coordinates.”

**Interviewer**: “So, players take turns guessing the locations of the opponent’s ships. How are the shots tracked during the game?”

**Expert**: “Yes, that’s right. When a player calls out a coordinate, the opponent checks their grid to see if a ship occupies that spot. If there’s a ship at that location, it’s a ‘hit.’ If not, it’s a ‘miss.’ Each player marks their own grid with red pegs for hits and white pegs for misses. The game continues until all of one player’s ships are sunk.”

**Interviewer**: “Great. In terms of ship placement, what are the rules I need to be aware of for the digital version?”

**Expert**: “Ships can only be placed horizontally or vertically on the grid, and they must remain entirely within the grid boundaries. You can’t have a ship overlapping another, and once the game starts, you can’t move the ships. We want to keep the placement process intuitive and simple, while also enforcing these rules automatically.

**Interviewer**: “Got it. Regarding the digital version, what kind of user interface features do you think would be essential?”

**Expert**: “Well, the interface should be straightforward, with two grids: one for the player’s ships and another for tracking the opponent’s moves. Players should be able to drag and drop ships onto their grid before the game starts. Once the game begins, the system must update after each turn, displaying the result of each shot. Additionally, visual feedback for hits and misses will be crucial.

**Interviewer**: “What challenges do players typically face during a game of Battleship?”

**Expert**: “One challenge is keeping track of which coordinates have already been called and which ones remain unknown.” Players also need to stay strategic about where they place their ships and how they guess the opponent’s shots. The game involves a lot of memory and planning.”

**Interviewer**: “Are there any additional features you think would improve the digital version of the game?”

**Expert**: “In addition to tracking hits and misses, it could be helpful to include a status display showing which ships are still afloat and which ones have been sunk. It would also be nice to have a timer or turn indicator so players know whose turn it is and how much time they have left to make a move.”

**Interviewer**: “That’s very helpful, thank you. Finally, how does the game end, and how do you know when one player has won?”

**Expert**: “The game ends when all of the opponent’s ships have been sunk.” The system should notify both players when this happens. There should be a message indicating the game’s end, and the winner should be announced right away.”

**Summary Report:**

From the mock interview, we gained insight into the core mechanics and rules of Battleship that will be essential in developing the digital version of the game. Key takeaways include:

* **Ship Placement**: Ships must be placed horizontally or vertically, with no overlap, and stay within the grid.
* **Gameplay**: Players take turns guessing coordinates to locate the opponent’s ships. Hits are tracked with red pegs, and misses with white pegs.
* **User Interface**: The game needs two grids—one for each player—and must support drag-and-drop ship placement. The system should track hits/misses and provide clear visual feedback.
* **End of Game**: The game concludes when all the opponent's ships have been sunk. The winner is declared immediately.

**Part 2: User Stories**

**Development:**

Using the information gathered from the interview, we can now convert the requirements into user stories. Each story is written from the player's perspective, focusing on the functionality that needs to be implemented for a seamless experience.

**User Story 1: Ship Placement**

* **As a player**, I want to place my ships on the grid so that I can prepare for the game.
* **Acceptance Criteria**:
  + The player can drag and drop ships onto the grid.
  + Ships can only be placed horizontally or vertically.
  + Ships must fit entirely within the grid, without overlapping other ships or extending beyond the grid.
  + If a ship placement is invalid, the system should visually indicate it (e.g., with a red highlight).
  + Once the game starts, ships cannot be moved.

**User Story 2: Calling Coordinates**

* **As a player**, I want to call out coordinates on the opponent's grid so that I can try to hit their ships.
* **Acceptance Criteria**:
* The player can input a coordinate in the format of a letter (A-J) and a number (1-10).
* After calling out a coordinate, the game checks if a ship is located at that position and notifies the player with a "hit" or "miss" message.
* The grid will update to reflect the result, marking hits with red pegs and misses with white pegs.

**User Story 3: Tracking Moves**

* **As the red player**, I want a grid to track my moves so that I know which points on the blue player’s grid I have called, and which are still unknown.
* **Acceptance Criteria**:
  + The grid will display the results of all previous moves, showing red pegs for hits and white pegs for misses.
  + The grid will update in real-time after each move, allowing players to easily track their own and the opponent’s shots.

**User Story 4: Ship Sinking**

* **As a player**, I want to be notified when I have sunk a ship, so I know I have successfully destroyed one of my opponent’s ships.
* **Acceptance Criteria**:
  + The game will notify the player when a ship has been completely sunk.
  + The sunk ship will be marked on the player’s grid, and a message will appear indicating which ship has been destroyed.
  + Both players will be informed when a ship is sunk.

**User Story 5: End Game Notification**

* **As a player**, I want to be notified when the game ends so that I know whether I won or lost.
* **Acceptance Criteria**:
  + The game will automatically declare a winner once all of the opponent’s ships are sunk.
  + A "Game Over" message will appear, displaying the winner’s name and the game's status.
  + Players can view a summary of the game, including which ships were sunk.

**Portfolio Submission:**

This portfolio contains the following:

1. **Interview report**: A detailed summary of the questions and answers from the mock interview with a Battleship expert.
2. **User stories with acceptance criteria**: A collection of user stories outlining the requirements and functionality for the Battleship software, with corresponding acceptance criteria to ensure each requirement is met.

This portfolio serves as the foundation for the Battleship software solution, ensuring that both user needs and game rules are accurately implemented in the digital version of the game.