SOFTWARE REQUIREMENTS SPECIFICATIONS

1- Functional requirements:

Game rules:

- **Players:** The game is played by two player or one player VS AI opponent (X and O) represented by the variables: player one and player two for two players mode.
- **Board:** 3 x 3 grid represented by the array symbols[3][3] is drawn by the function drawBoard
- **Winning conditions:** The checkWin function checks for four cases: game in progress, draw, player one wins or player two wins.
- **Turn order:** players alternate turns, it is managed by do while and their existing sequence.

User interactions:

- **Starting a Game:** The game starts after player log in to the game or make a profile if he does not have, then choose playing mode, AI or two players mode.
- **Making a Move:** Players input their move by choosing the suitable block.
- **Restarting the Game:** The game can be restarted the game by clicking reset button, it will return the board to the initial state.
- **Winning Message:** Displayed by checking if there is a winner and printing the winner.
- **Log-in / sign-up:**

  Sign-up: New users can create an account by providing a username and password.

  Login: Existing users can log in using their credentials.
- **AI Opponent:** Option to play against an AI opponent.

System Behavior:

- **invalid information:** The game send an alert for invalid user name or password or for empty information in sign up.
- **State Management:** Maintain the state of the game (which cells are filled, whose turn it is, etc.).
- **Error Handling:** Prevent invalid moves, such as placing a marker in an already occupied cell.
- **Game Records:** Optionally store game history for logged-in users.

2- Non-Functional Requirements

Performance

- **Responsiveness:** The game should respond to user actions within a fraction of a second.
- **Efficiency:** The game logic should efficiently handle move validation, win/draw detection, and AI decision-making without significant delays.

Usability

- **User Interface:** The game should have a simple, intuitive graphical interface that is easy to understand and use.

Reliability

- **Stability:** The game should be stable and not crash during normal gameplay.
- **Data Integrity:** The game should accurately maintain and display the state of the game at all times.
- **Security:** User information should be stored securely in the database.

Maintainability

- **Code Structure:** The code should be well-organized and commented to facilitate maintenance and future updates.

Other Considerations

Security

- **Data Security:** Ensure that user data is handled securely and is not exposed to unauthorized parties.
- **Authentication:** Implement secure login and sign-up mechanisms.

Dependencies

- **Libraries/Frameworks:** windows form as a GUI, SQL lite for data base.