Approach C prompt for requirements engineer (written by project manager AI instance (ChatGPT))

Project Outline

We are developing a classic Tetris game using Python. The project will involve multiple stakeholders, each playing a crucial role in the development process. Your role as the Requirements Engineer is to plan and document the requirements for the game. These requirements will guide the development process and ensure that the final product meets the necessary standards and expectations.

Role of the Requirements Engineer

As the Requirements Engineer, your primary responsibilities include:

- Requirements Gathering: Identify and document all the necessary requirements for the Tetris game.
- Requirements Documentation: Create a comprehensive list of both functional and non-functional requirements.
- Collaboration: Work closely with other stakeholders, such as the Game and UI Designer, Software Architect, Developer, Code Reviewer, and Technical Writer, to ensure that all requirements are clearly understood and met.

Objectives

- Define the functional requirements that describe what the game should do.
- Define the non-functional requirements that describe the qualities and constraints of the game.

Functional Requirements

Functional requirements are specific behaviors or functions of the game. For Tetris, consider the following aspects:

Game Mechanics

- The game should allow players to start, pause, and reset the game.
- The game should feature the classic Tetris gameplay, where shapes (tetrominoes) fall from the top of the screen and can be rotated and moved horizontally.
- The game should detect when a row is completely filled and remove it, awarding points to the player.
- The game should progressively increase in difficulty as the player advances, typically by increasing the speed of falling tetrominoes.

User Interface

- The game should display the current score and level.

- The game should show the next tetromino that will fall.
- The game should provide visual feedback when rows are cleared.
- The game should display a game over screen when the player loses.

Controls

- The game should support keyboard controls for moving, rotating, and dropping tetrominoes.
- The game should allow players to customize control keys if desired.

Sound and Music

- The game should include sound effects for actions such as moving tetrominoes, clearing rows, and game over.
- The game should have background music that can be toggled on and off by the player.

Non-Functional Requirements

Non-functional requirements define the system's quality attributes and constraints. For Tetris, consider the following:

Performance

- The game should run smoothly at a consistent frame rate on a typical computer.
- The game should handle increasing game speed without performance degradation.

Usability

- The game should have an intuitive and easy-to-use interface.
- The game should provide instructions or a help menu for new players.

Compatibility

- The game should be compatible with the latest version of Python.
- The game should run on major operating systems (Windows, macOS, Linux).

Reliability

- The game should handle unexpected inputs gracefully without crashing.
- The game should save the player's progress and high scores.

Maintainability

- The game code should be well-documented to facilitate future maintenance and updates.
- The game should be modular to allow easy addition of new features or modifications.

Security

- The game should ensure that any saved data (like high scores) is securely stored.
- The game should not expose the system to any security vulnerabilities.

Next Steps

Please generate a comprehensive list of functional and non-functional requirements for the Tetris game, expanding on the points provided above. Once you have documented these requirements, we will review them with the other stakeholders to ensure alignment and completeness.