**Scenario: To-Do List Application**

**1. Introduction**

* **Purpose**: This SRS document outlines the requirements for a To-Do List Application designed to allow users to manage their tasks by adding, editing, and deleting them.
* **Scope**: The system will be a desktop application that helps users track and manage their tasks with features like task addition, editing, and deletion. The tasks are saved in the application and displayed to the user in a list format.

**2. Overall Description**

* **Product Perspective**: This application is a desktop-based task management tool using Python's Tkinter library. It provides a graphical user interface (GUI) where users can interact with the system to manage their tasks.
* **Product Functions**:
  + Task creation and display
  + Task editing and modification
  + Task deletion
  + Basic input validation (date format and priority selection)

**3. Specific Requirements**

**3.1 Functional Requirements**

* **Task Creation**:
  + Users must be able to add new tasks by providing a description, due date, and priority.
  + The application should validate the input fields to ensure that they are filled and follow the correct formats (date and priority).
  + The task should be displayed in the list with its description, due date, and priority.
* **Task Editing**:
  + Users can edit existing tasks by selecting a task from the list.
  + The selected task's data will be shown in the entry fields, allowing users to update it.
  + The updated task will be reflected in the list after saving the changes.
* **Task Deletion**:
  + Users can delete a selected task from the list.
  + A confirmation prompt will be shown to confirm the deletion action.
  + Once confirmed, the task is deleted from the list.

**3.2 Non-Functional Requirements**

* **Performance**: The system should handle the display and management of up to 100 tasks without performance issues.
* **Security**: No sensitive user data is handled, but basic error handling should be included to avoid application crashes.
* **Usability**: The application must be user-friendly with a simple interface for task management. The buttons and input fields should be clearly labeled.

**4. Use Cases**

* **Use Case 1**: **Add a New Task**
  + User enters task description, due date, and priority.
  + The application validates the input and adds the task to the list.
  + The task is displayed in the list with its details.
* **Use Case 2**: **Edit an Existing Task**
  + User selects a task from the list to edit.
  + The task's current details are displayed in the entry fields.
  + User makes changes and saves them, and the task is updated in the list.
* **Use Case 3**: **Delete a Task**
  + User selects a task from the list to delete.
  + The application confirms the deletion action.
  + The task is removed from the list.

**5. Assumptions and Dependencies**

* The application assumes the user has basic knowledge of interacting with a GUI-based application.
* The application depends on the Tkinter library for the GUI and Python for the logic.