**SOFTWARE ENGINEERING PROBLEM SPECIFICATION TABLE, IDENTIFYING THE FOLLOWING ELEMENTS**

|  |  |
| --- | --- |
| CUSTOMER | Icesi University |
| USER | Customers of the Todo List app |
| FUNCTIONAL REQUIREMENTS | The software must allow:  RF0: To store reminders.  RF1: Modify the reminders.  RF2: Delete Reminders  RF3: Manage the priority of reminders.  RF4: Display reminders  RF5: Allow the user to undo an action. |
| CONTEXT | The problem consists of the need to design and develop a task and reminder management system that allows users to efficiently add, organize, and manage their to-dos and personal reminders. For this reason, the program has to be able to manage priority tasks and have an intuitive user interface. |
| NON-FUNCTIONAL REQUIREMENTS | The software must have:  RFN1: Time complexity analysis  RFN2: Spatial Complexity  RFN3: TAD implementation  RFN4: Class Diagram Design  RFN5: Test case design  RFN6: Intuitive User Interface. |

| Name or identifier | RF0: Store reminders | | |
| --- | --- | --- | --- |
| Summary | Users must be able to store new reminders in the system. Each reminder should include information such as title, description, deadline date, and priority. Reminders are organized based on their priority and arrival order. | | |
| Input | Entry name | Data type | Selection or repetition condition |
| Title | String | REQUIRED |
| Description | String | REQUIRED |
| DeadLine | Date | The date must be grater than the actual date |
| Priority | Enum | It has to be either (Priority or Not a priority) |
| Result or postcondition | The system stores the reminder based on its priority and arrival order. In the corresponding data structure. | | |
| Outputs | Entry name | Data type | Selection or repetition condition |
| Confirmation\_Message | String | A message indicating the stage of the transaction |

| Name or identifier | RF1: Modify reminders | | |
| --- | --- | --- | --- |
| Summary | Users must be able to modify the information of existing reminders. This includes the ability to change the title, description, deadline, and priority of a reminder. | | |
| Input | Entry name | Data type | Selection or repetition condition |
| Reminder\_ID | String | The reminder\_ID must exist in the current reminders |
| New\_Title | String | N/A |
| New\_Description | String | N/A |
| New\_deadline | Date | It has to be greater than the actual date |
| New\_priority | Enum | It has to be either (Priority or Not a priority) |
| Result or postcondition | The system updates the reminder information. | | |
| Outputs | Entry name | Data type | Selection or repetition condition |
| Confirmation\_Message | String | A message indicating the stage of the transaction |

| Name or identifier | RF2: Delete Reminders | | |
| --- | --- | --- | --- |
| Summary | Users must be able to delete reminders from the system when they are no longer needed. | | |
| Input | Entry name | Data type | Selection or repetition condition |
| Reminder\_ID | String | The reminder\_ID must exist in the current reminders |
| Result or postcondition | The system removes the reminder from storage. | | |
| Outputs | Entry name | Data type | Selection or repetition condition |
| Confirmation\_Message | String | A message indicating the stage of the transaction |

| Name or identifier | RF3: Manage the priority of reminders | | |
| --- | --- | --- | --- |
| Summary | The system must categorize reminders into two categories: "Priority" and "Not a priority."  Reminders must be sorted by priority using a heap sort algorithm. | | |
| Input | Entry name | Data type | Selection or repetition condition |
| Reminder\_ID | String | N/A |
| Priority\_Category | Enum | It has to be either (Priority or Not a priority) |
| Result or postcondition | The system assigns reminders to the specified priority category and sorts reminders by priority. | | |
| Outputs | Entry name | Data type | Selection or repetition condition |
|  |  |  |

| Name or identifier | RF4: Display Reminders | | |
| --- | --- | --- | --- |
| Summary | Users must be able to view a list of all their reminders. They should have the option to sort this list by deadline or priority. | | |
| Input | Entry name | Data type | Selection or repetition condition |
| Structure | ¿? | Th structure passed must not be empty for the program to show something |
|  |  |  |
|  |  |  |
| Result or postcondition | The system displays the list of reminders sorted according to the user's choice. | | |
| Outputs | Entry name | Data type | Selection or repetition condition |
| List of reminders | ¿? | N/A |

| Name or identifier | RF5: Allow the user to undo an action | | |
| --- | --- | --- | --- |
| Summary | Implement a function that allows users to undo the last action performed in the system. This includes the ability to undo adding, modifying, or deleting reminders. | | |
| Input | Entry name | Data type | Selection or repetition condition |
| Action\_to\_undo | Enum | It can be “add reminder”, “modify reminder” and “delete reminder” |
| Result or postcondition | The system reverses the last user action based on the information stored in the undo stack. | | |
| Outputs | Entry name | Data type | Selection or repetition condition |
|  |  |  |