**1. Introduction**

The Task Manager project is a robust application designed to help users effectively manage their tasks and responsibilities in an increasingly busy world.

* **Domain Problem**: Individuals struggle to keep track of various commitments.
* **Significance**: Enhances productivity and organization.
* **Core Functionality**:
  + Create, modify, and delete tasks.
  + Set deadlines and prioritize workload.
  + Categorize tasks with tags.

**2. Objective**

The objective of the Task Manager project is to create a user-friendly platform for efficient task management.

* **Key Goals**:
  + Quickly add new tasks.
  + Easily modify existing tasks.
  + Remove tasks that are no longer needed.
  + Set deadlines for timely completion.
  + Prioritize tasks to focus on what matters most.
  + Categorize tasks using tags.

**3. Functionality**

The Task Manager includes several key features and functionalities:

* **Task Creation and Management**:
  + Add tasks with title, description, deadline, priority, and tags.
* **Recurring Tasks**:
  + Mark tasks as recurring for automatic scheduling.
* **Search Functionality**:
  + Search tasks by title or keywords in the description.
* **Task Display**:
  + Clear display of tasks with relevant details.
* **Undo/Redo Actions**:
  + Undo or redo last actions for flexibility.
* **Task Persistence**:
  + Save and load tasks from a text file.
* **Data Insights**:
  + Future visualizations for task completion rates and productivity metrics.

**4. Technology Stack**

The Task Manager project utilizes the following technologies:

* **Frontend**:
  + C++ for core application logic with a console-based interface.
* **Backend**:
  + C++ (standalone console application, no separate backend).
* **Database**:
  + File-based storage using text files for task persistence.
* **Libraries**:
  + Standard libraries such *as <iostream>, <string>, <vector>, <fstream>, and <sstream>* for data handling.

**5. Data Structures Used**

The project employs the following data structures to enhance functionality:

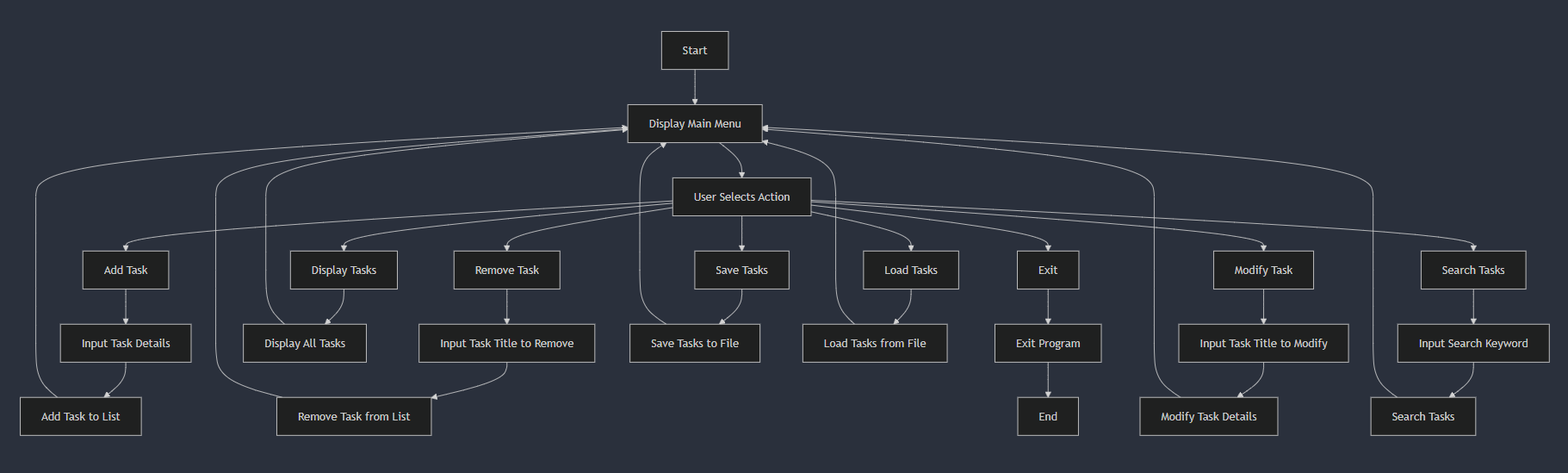
* **Linked List**:
  + **Purpose**: Used to manage tasks dynamically.
  + **Implementation**: The *TaskList* class utilizes a linked list where each node is represented by a Task structure. Each Task object contains a pointer to the next Task, allowing for easy insertion and deletion based on deadlines.
* **Vector**:
  + **Purpose**: Used to store tags associated with each task.
  + **Implementation**: The tags field in the Task structure is a vector<string>, enabling dynamic resizing and manipulation of tags for effective categorization.
* **String**:
  + **Purpose**: Used for handling task attributes such as title, description, priority, and tags.
  + **Implementation**: The string type is used throughout the code for input and storage of textual data, facilitating easy manipulation and search operations.
* **Stack**:
  + **Purpose**: Used for undo and redo operations.
  + **Implementation**: The *UndoStack* class implements a stack data structure to keep track of actions (add, remove, modify) performed on tasks. Each action is stored as a *StackNode*, containing the task details and action type.
* **Stack Node (Custom Structure)**:
  + **Purpose**: Each node of the stack used in the undo functionality.
  + **Implementation**: The *StackNode* struct is defined within the *UndoStack* class, holding fields for the action type and a pointer to a Task.
* **Temporary Structures**:
  + **String Stream**:
    - **Purpose**: Used for parsing tags and input data.
    - **Implementation**: *stringstream* is used to split strings (e.g., tags input) into individual components for easier processing.

**6. Conclusion**

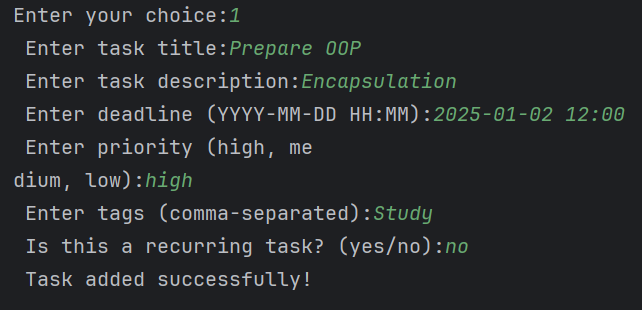
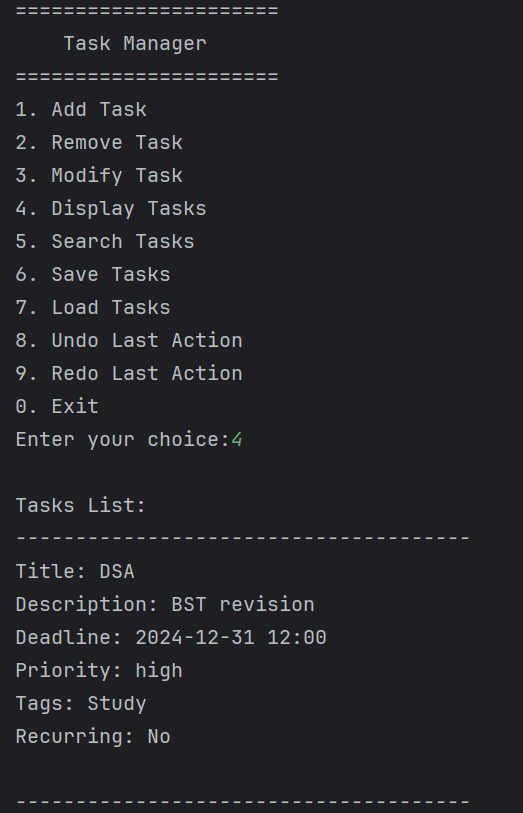
The Task Manager project effectively addresses the problem of task management by providing a streamlined solution for organizing responsibilities.

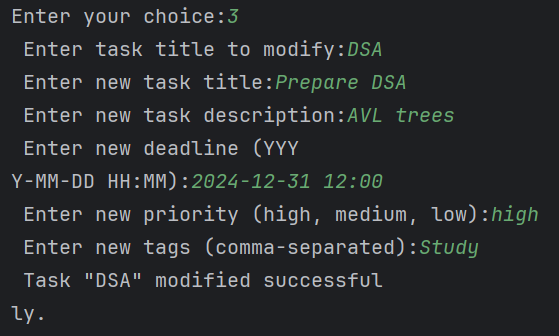
* **Benefits**:
  + User-friendly interface.
  + Robust functionality for time management.
  + Empowers users to focus on priorities, improving productivity and reducing stress.

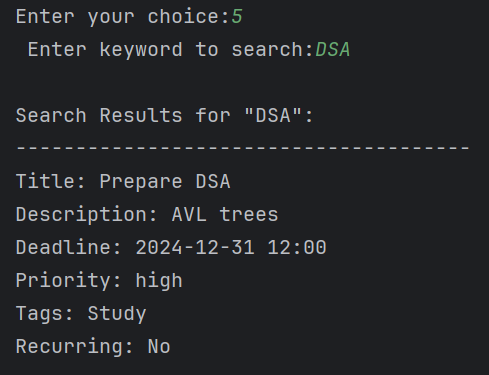
**7. Flowchart**



**8. Sample output**







A black screen with white text

Description automatically generated