**Documentation**

**for**

**To-Do List App**

**Author:** Denislav Georgiev   
**Language:** English

**(English)**

**1. Description**

The To-Do List program allows users to manage their daily tasks by adding, viewing, and removing tasks. Additionally, the program implements file handling to save tasks when the program exits and reload them when it starts. The program provides an interactive menu where users can choose different options to manage their tasks efficiently.

### **2. Code Structure**

The program is divided into three main files:

* **"main.cpp"** – The main file that initializes and runs the program.
* **"ToDoList.h"** – Header file that defines the ToDoList class, which manages the list of tasks.
* **"ToDoList.cpp"** – Implementation of the ToDoList class methods, including task management and file handling.

### **3. Description of Classes and Functions**

#### ****ToDoList Class****

This class is responsible for managing tasks and interacting with the user.

**Methods:**

* void showMenu() – Displays the menu options for the user.
* void addTask() – Allows the user to add a new task to the list.
* void viewTasks() – Displays all tasks currently in the list.
* void removeTask() – Removes a task from the list based on user input.
* void saveTasks() – Saves tasks to a file (tasks.txt) before exiting.
* void loadTasks() – Loads tasks from a file (tasks.txt) when the program starts.
* void run() – Handles the main loop of the program, processing user input and executing corresponding actions.

### **4. Program Usage**

1. Run the program from the terminal or an IDE.
2. The program will automatically load any previously saved tasks.
3. The user will see a menu with the following options:
   * **1. Add Task** – Enter a description to add a new task.
   * **2. View Tasks** – Displays all existing tasks.
   * **3. Remove Task** – Select a task to remove by its number.
   * **4. Exit** – Saves tasks to a file and exits the program.
4. The user selects an option by entering the corresponding number.
5. The program continues running until the user selects **Exit**.

### **5. Errors and Remarks**

* **Invalid Input Handling:** The program prevents invalid menu selections and non-numeric input errors.
* **Empty Task List:** If no tasks exist, the program informs the user when viewing or removing tasks.
* **Task Removal Validation:** The program ensures that users can only remove existing tasks.
* **File Handling Limitations:** If the tasks.txt file is missing or corrupted, the program will start with an empty task list.
* **No Task Editing Feature:** Currently, tasks cannot be edited after being added.

### **6. Future Improvements**

* **Add a Graphical User Interface (GUI)** to improve user experience.
* **Allow Task Editing** to modify existing tasks.
* **Add Task Due Dates & Priorities** for better task management.
* **Implement Long-Term Storage Options** such as databases instead of plain text files.