INFS2044 Supplementary Assignment Case Study

In this assignment, you will be developing a simple ToDo reminder system. In this system, users can create and search for "To Do" tasks with an associated due date, and the system will remind the user of upcoming tasks. We will consider a simplistic version of such a software that is restricted to a command line interface.

Use Cases

The system supports the following use cases:

- UC1 Add Task: The user enters a title, due date, and notes, and the system creates a ToDo task and stores it.
- UC2 Search Task: The user enters keywords, and the system displays a list of tasks and their due dates that include all the keywords anywhere in the task title or in its notes.
- UC3 Remind User: Once a task is due in less than 24 hours, the system prints a reminder to the console.

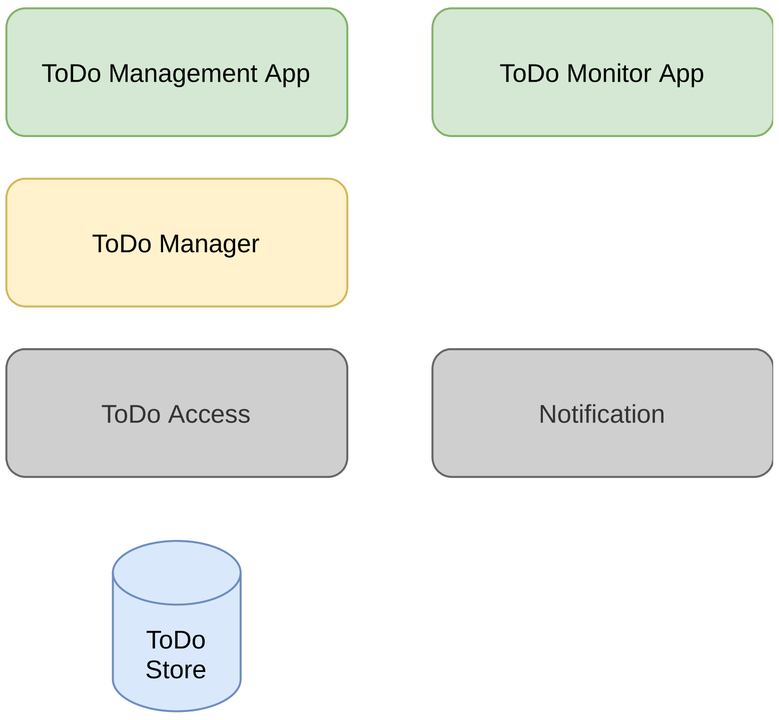
Other Requirements

- R01: The system shall interact with the user through a text console (Terminal on MacOS/Linux and Command Prompt on MS Windows).
- R02: The system shall store tasks in JSON format in file "tasks.json".
- R03: The system shall continuously monitor the tasks, no less than once per second.
- R04: Keyword search shall be case insensitive.
- R05: The system must load any tasks given in tasks.json when the application starts.
- R06: The json format for tasks.js must be as follows (the list of tasks can be arbitrarily long):

```
[
    {
        "id":1,
        "title":"First Task",
        "due":"2021-07-30T11:00:00.000000",
        "notes":"The first task is due at the end of the month."
    },
    {
        "id":2,
        "title":"Another Task",
        "due":"2021-07-23T12:00:00.000000",
        "notes":"Another task that is due earlier than First Task"
    }
]
```

Decomposition

You must use the following component decomposition as the basis for your implementation design:



The responsibilities of the components are as follows:

Component	**Responsibilities**
ToDo Management App	Interact with the user to create & search tasks
ToDo Monitor App	Issue reminders to the user
ToDo Manager	Orchestrates the use cases
ToDo Access	Retrieves and stores tasks
Notification	Issues notifications to the user
ToDo Store	Holds all the task information

Assumptions

This decomposition has been created based on the volatility assumptions that in future:

- tasks may be stored in a database
- a GUI / API front-end may be added
- users may be notified of upcoming tasks via email and/or text messages

Scope

Your implementation must respect the boundaries defined by the above decomposition and include classes for each of the components in this decomposition (except the store component, which is realised as a file as per requirements R02).

The implementation must:

- run on python 3, and
- correctly implement the given use cases, and
- it must function correctly with any tasks given in tasks.json in the correct format (you can assume that the entire content of this file fits into main memory); and
- it must include a comprehensive unit test suite using pytest.

Focus your attention on the *quality* of your code.
Speed and memory efficiency are not primary concerns for purposes of this assignment.

Implementation Notes

Use the python `datetime` package to parse the “due” field in the json document.

The ToDo Management App shall be run by issuing the command

```
python manager_app.py
```

The ToDo Reminder App shall be run by issuing the command

```
python reminder_app.py.
```

This application shall keep running and continuously monitor the tasks and issue reminder alerts as defined in the requirements.