Task Planner System

Design and implement a task planner system

A task has the following details :

1. Title
2. Creator
3. Assignee (Optional)
4. Status
5. Type
6. Due date

A task can be of following types with additional information apart from what is mentioned above

1. Feature
    a. Feature summary
    b. Impact (Low, Moderate, High)
2. Bug
    a. Severity (P0, P1 or P2)
3. Story
    a. Story Summary
    b. Story Points

It should be **easy** to add a new task type to your application

The status can change from a state to any state.
Status field takes one of the following states:
Tasks => Open, In progress, Completed

A sprint is defined as a collection of tasks used to track progress. You can add or remove a task from sprints.
A task can be part of only one sprint at a time.

Your task planner should have the following functionalities:

1. Task
    a. Create a task of any type
    b. Change the status of the task
    c. Change assignee of the task

    d. Display tasks assigned to a user
2. Sprint
    a. Create/Delete a Sprint
    b. Add/remove task to/from sprint
    c. Display sprint snapshot: Show tasks part of the sprint.
3. **Bonus Question (Only if time permits) :-**
   The transition/change in the status should be based on allowed transitions
   eg :-
   For task, we might want to have only following allowed transitions :
   Open => In progress
   In progress => Completed

The examples below are just to understand the functionalities and **may not necessarily** be used in the same format as input to your driver program.

Let's say we want to add the following tasks to a sprint :

Tasks:

Title	**Creator**	**Assignee**	**Status**	**Due date**	**Type**	**Type attributes (comma separated)**	**Sprint**
Create Dashboard	Brad	Peter	Open	2019-04-12	Feature	Create console for debugging, Low	
Fix mysql issue	Ryan	Ryan	In progress	2019-04-14	Bug	P0	Sprint-1
Create a microservice	Amy	Ryan	Completed	2019-03-12	Story	Add logging to the feature	Sprint-1

Setup console	Ryan	Ryan	In progress	2019-04-14	Feature	Create console for debugging, High	
Console api	Ryan	Ryan	In progress	2019-04-14	Feature	Create api for console , High	

Display tasks assigned to a user:

Eg: For assignee Ryan
User => Ryan:
Task Type => Bug
Title => Fix mysql issue
Sprint => Sprint-1

Task Type => Feature
Title => Setup console
Sprint =>

Task Type => Feature
Title => Console api
Sprint =>

Task Type => Story
Title => Create a microservice
Sprint => Sprint-1

Eg: For assignee Peter
User => Peter:
Task Type => Feature:
Title => Create Dashboard
Sprint => Sprint Id/Sprint name

Display status of Sprint-1

Sprint title => Sprint-1

Fix mysql issue
Setup console
Create Dashboard
Create a microservice

Guidelines:

You are free to use the language of your choice.
Input can be read from a file or STDIN or hard-coded in a driver method. It should be easy to give new input or change existing input and test new cases.
Output can be written to a file or STDOUT.
Feel free to store all interim/output data in-memory. Use of any external persistent datastore is not required.
Restrict internet usage to looking up syntax or API references
Save your code/project by your name and email it to the email address provided by the interviewer. Your program will be executed on another machine. So, explicitly specify any dependencies in your email.

Expectations:

Minimum:
Code should be demo-able (very important).
Code should be functionally correct and complete.
At the end of this interview round, an interviewer will provide multiple inputs to your program for which it is expected to work.
Code should handle all edge cases properly and fail gracefully. Add suitable exception handling, wherever applicable.
Code should be readable, modular and testable. Use intuitive names for your variables, methods and classes.
Bonus:
Code should have good object-oriented design and be extensible.
There should be proper separation of concerns.
It should be easy to add/remove functionality without rewriting a lot of code.
It shouldn’t be monolithic.