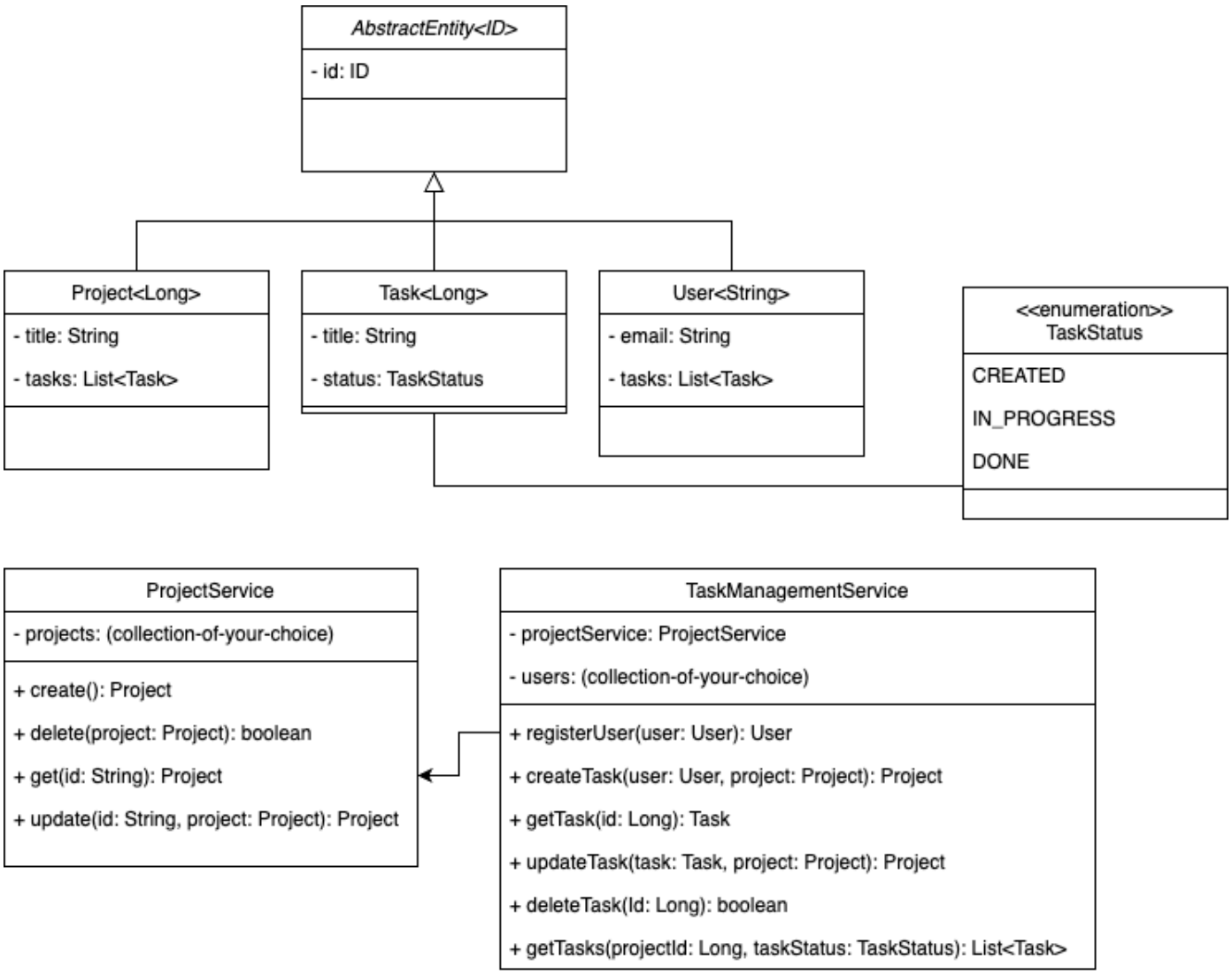
A. (1p for attendance) Task management system

Your requirement for today would be to design an implement a task management system. The overall idea would be that a user could create and work with projects and tasks in your application. This application would help users to keep track of the work they have to perform no matter what their job is. The application should provide some methods in order to fulfill flows like: user registration, projects and tasks CRUD (create, read, update, delete) actions or specific actions like getting a list of all open tasks.

Figure 1: Guideline UML diagram



Requirements

1. (2p) Read and understand everything presented below + the UML diagram. Start by creating the classes structure presented above. Feel free to add attributes/classes as you wish in order to fulfill the upcoming requirements. Getters and setters were not added in the diagram but they should be there.

2. (3p) Implement `ProjectService` methods as follows:

- pick a collection of your choice for storing the projects data;
- on creating a new project make sure it gets a unique id. The id must be unique application wise (if there is a project with id 23L there must not be any other project or task with this id). Throw an unchecked exception `DuplicateProjectNameException` in case there already exists a project with the **exact** same name;
- get method should return the project with the provided id or throw an unchecked exception `ProjectNotFoundException`
- update method should update station identified by the provided id or throw `ProjectNotFoundException` if we provide an id of an non-existent project.

3. (1p) If you try to delete a project you should take into account the following scenarios:

- if the project doesn't exist throw a `ProjectNotFoundException`
- if there are still tasks having the status `IN_PROGRESS` don't delete the project and return false.
- in case it passed more than 6 months since the project was created then it can be deleted even though there are tasks having the status `IN_PROGRESS`. You might need to store information on when the project was created.

4. (2p) Implement `TaskManagementService` methods as follows:

- pick a collection of your choice for storing the users data
- `registerUser` must assign a unique id to the provided user object and store it to the users collection. Throw a checked exception `EmailAlreadyTakenException` if there is a user with the same email already registered
- `createTask, getTask, updateTask and deleteTask` should work in a similar way as you already implemented for projects. Throw an unckecked exception if trying to access a task that doesn't exist.. just as you did with projects.

5. (1p) Implement some specific methods as follows:

- `getTasks` must return a list of tasks from a specific project identified by the provided id that have the exact status as the one provided as parameter to the method;
- implement a different `getTasks` method (you decide what signature it has) that would receive a parameter of type `TaskStatus` and would return a collection of your choice. The method must return a collection of all tasks from all projects which have the provided status. From the response of the method you must be able to identify to which project each task belongs to.