Assignment Report

Information

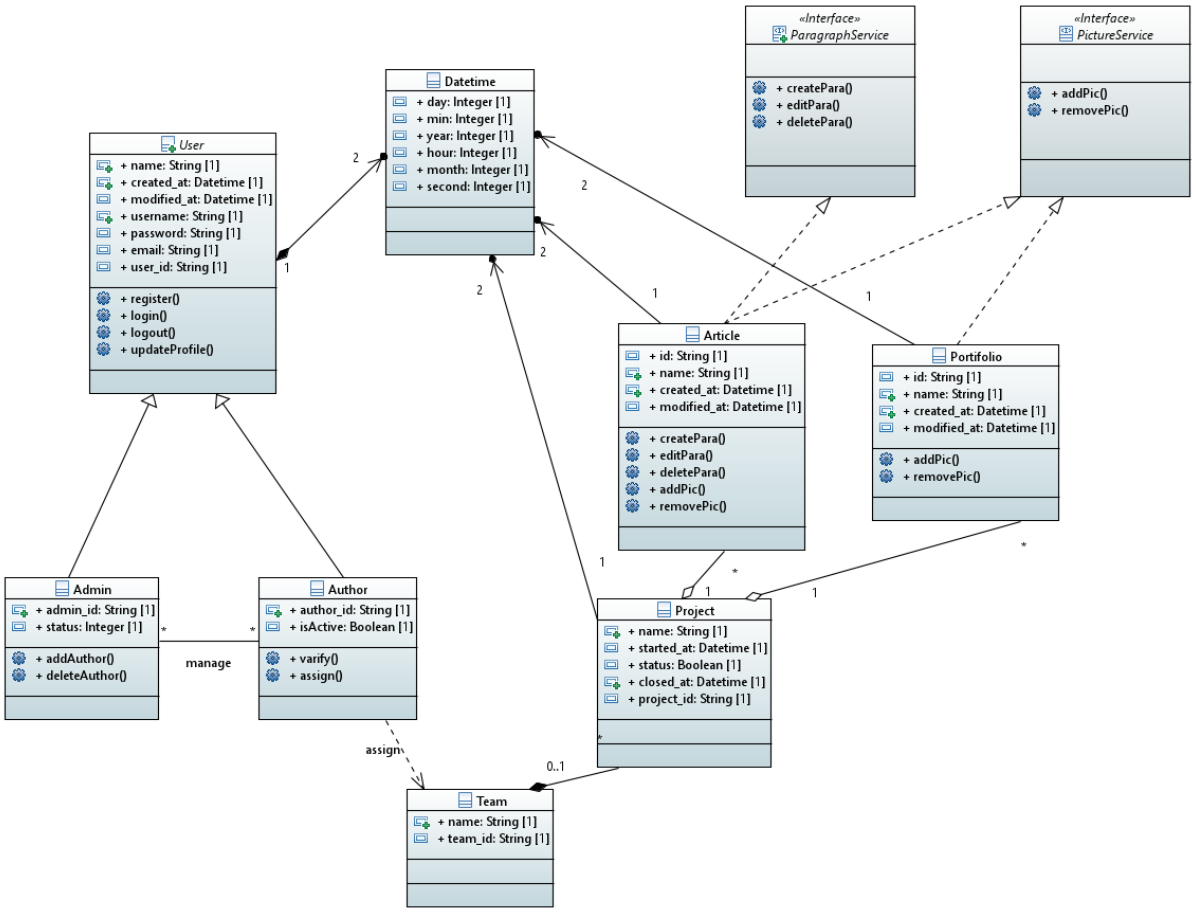
- Name: Cheng Bao
- ID: 1335784

Description of System

The system designed for this assignment is a content management system (CMS) that allows users to publish their projects online.

The users who sign in to this system will treated as an “author”, and the administrator of the system will approve the author’s requirement of registration and put them into a team. Meanwhile, the authors can create/edit/delete a project, which is the space for all the articles and portfolios that are related. The words and pictures can be represented in articles, while the portfolios only allowed to contain pictures.

For more details, you can see the class diagram of the system below:



Relationships and Cardinalities Explanation

As the class diagram shows above, there are several relationships between each class, they are:

1. The *Admin* and the *Author* are generated from the *User*, which makes the *User* the base class
2. The *Admin* manages the *Author*, so the cardinality is many-to-many
3. The *Team* is created only when there is an *Author* left, so the *Team* is dependent on the *Author* by assigning the *Author* to the *Team*
4. The *Team* can have many *Project*s, but one *Project* is assigned to only one *Team*. By system setting, there shouldn't be a standalone *Project*, so the relationship between the *Team* and The *Project* is composited, and the cardinality is one-to-many(or zero-to-many)
5. In the *Project*, there is the *Article* which can represent both paragraphs and pictures, and the *Portfolio* which only contains pictures, the relationship

between the *Article*/the *Portfolio* and the *Project* is shared, and the cardinality is one-to-many

6. Two interfaces are created to abstract the functions of the *Article* and the *Portfolio*. One is the *ParagraphService*, which indicates the functions and operations of paragraphs, and another is the *PictureService*, which indicates the functions and operations of pictures. The *Article* implements these two interfaces, while the *Portfolio* implements only the *PictureService*.
7. The *User*, *Project*, *Article*, and *Portfolio* all have properties called *create_at* and *modified_at*, which store the information of date and time. So the *Datetime* is created to allow them to use, the cardinality of them are all one-to-two

Challenges of Assignment

Although the system and diagram don't have many elements or complex functions, the whole process of designing and implementation cost more time than I expected. The challenges I faced during this assignment are listed below, from easiest to hardest:

1. Using Papyrus to model. Barely used Papyrus before, even in class I'm trying to use PlantUML instead of Papyrus to draw the diagram. So the time of playing Papyrus takes a lot than I expected.
2. Using Papyrus to generate codes. After completing my diagram, I can't generate code using Eclipse Java Developer, not sure if it's my setting problem or chose the wrong edition of the software. It took me almost a day to search for a solution through the Internet. Finally, I generated the code of my diagram just using Papyrus by installing a plugin called the Papyrus Software Designer inside the Papyrus. Hoping the code is generated correctly.
3. Some unclear knowledge to clarify. I think it's also the purpose of this assignment, to review the relationship & cardinality definition, which I think is the core concept of the class diagram. To be honest, there are still some points I'm not clear about the relationship & cardinality of the class diagram, and I have to search for some online tutorials to help me through the diagramming.