

**Software Engineering**

**AE2**

**Name of team:***YHRLLL*

**Team Members:**

*Chaoyi Yang 2724184Y*

*Haoshi Huang 2635088H*

*Shuai Ran 2633609R*

*Xintong Lin 2824115L*

*Yi Liu 2807941L*

*Zexuan Li 2815250L*

**Github link:** https://github.com/1RanShy/Software\_Engeneering\_Assement2.git

Table Contents

[User stories 3](#_Toc129902290)

[Class structure diagram (UML) 9](#_Toc129902291)

[Software structure and design patterns 10](#_Toc129902292)

[Sequence diagrams for each user story 12](#_Toc129902293)

[System operation results 20](#_Toc129902294)

[Administrator Test 20](#_Toc129902295)

[Part Time Teacher 27](#_Toc129902296)

[Output to the txt file 27](#_Toc129902297)

[Retrospective 28](#_Toc129902298)

# Team members:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Team Members** | **GUID** | **Contribution** | **Percentage Contribution** |
| **1** | Chaoyi Yang | 2724184Y | Staff.java  Administrator.java  PTT.java  Requirement.java | 16.7% |
| **2** | Haoshi Huang | 2635088H | Validator.java  Controller.java | 16.7% |
| **3** | Shuai Ran | 2633609R | View.java  Controller.java | 16.7% |
| **4** | Xintong Lin | 2824115L | FileName.java  FileIO.java | 16.7% |
| **5** | Yi Liu | 2807941L | Staff.java  ClassDirector.java  Course.java  Requirement.java | 16.7% |
| **6** | Zexuan Li | 2815250L | Controller.java  NumberGeneration.java | 16.7% |

# User stories

In this software engineering design exercise, the main task of our group is to develop a teaching requirements list and a training list, and design and develop a system that involves three roles: Class director, administrator, and part-time teacher (PTT). Class director can submit and modify teaching requests and view all teaching requirements in the system. Administrators can allocate teaching requests to part-time teachers and arrange training plans. Part-time teachers can view teaching tasks and training plans. The specific user descriptions are as follows:

Class director: can view all existing teaching tasks and submit new teaching requests. When submitting a request, they need to fill in course information and time. Class director can also modify their own submitted teaching requests in the system.

Administrator: can view all teaching requests in the system and allocate appropriate teaching requirements to part-time teachers based on course information and teaching time. Administrators can also assign training tasks to part-time teachers and set the information of the training task.

Part-time teacher: can view their own teaching tasks in the system and complete corresponding teaching work. At the same time, teachers can also view the training tasks based on the allocated teaching requirements.

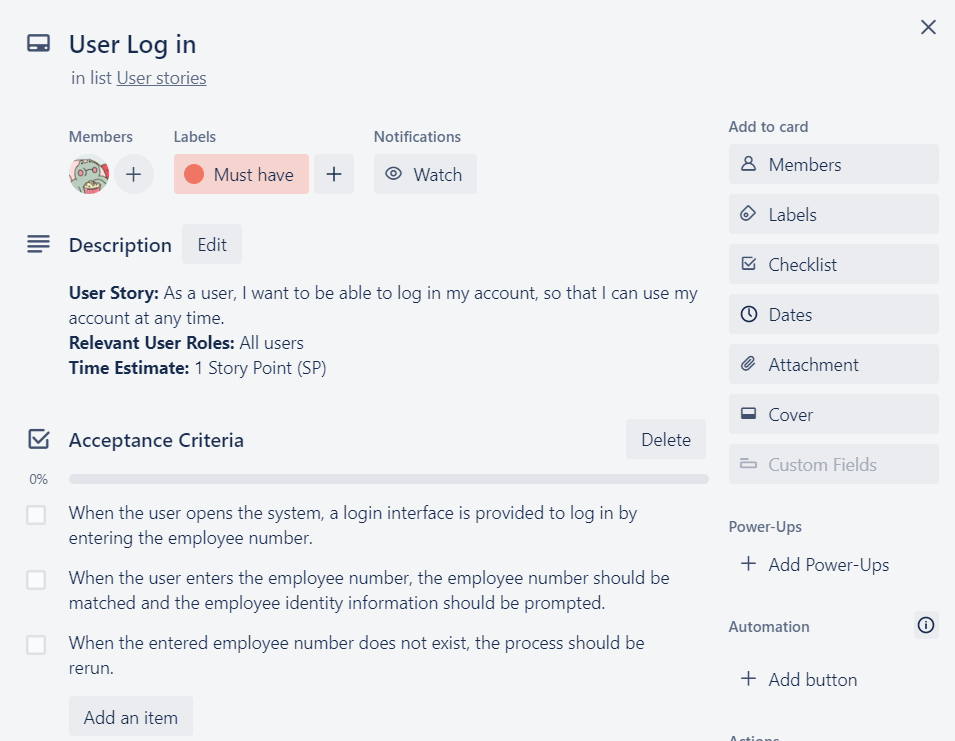


Figure 1.1: UserStory\_1

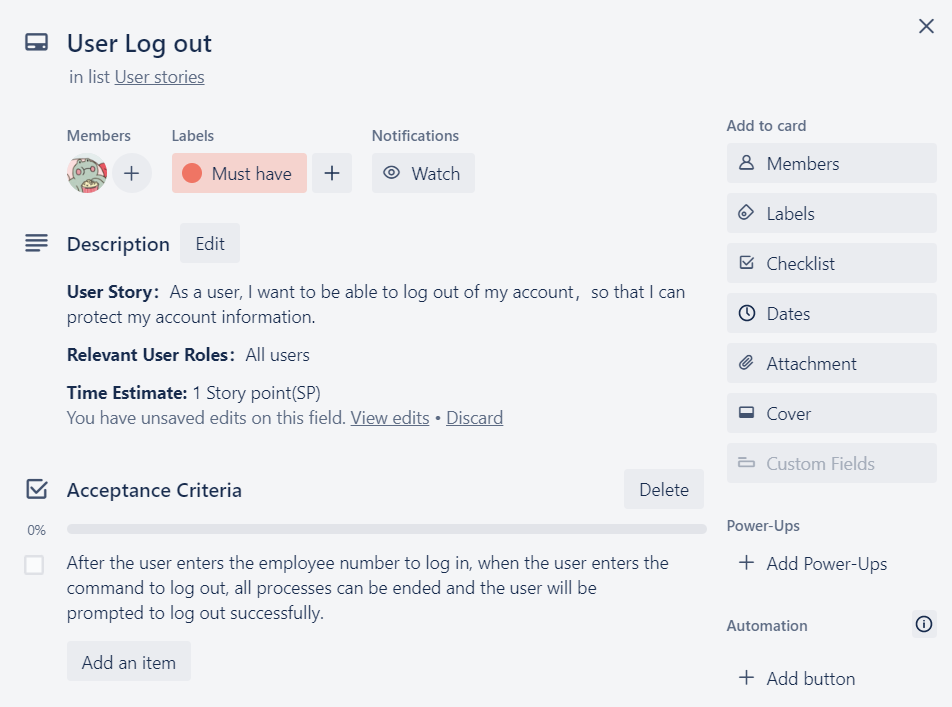


Figure 1.2: UserStory\_2

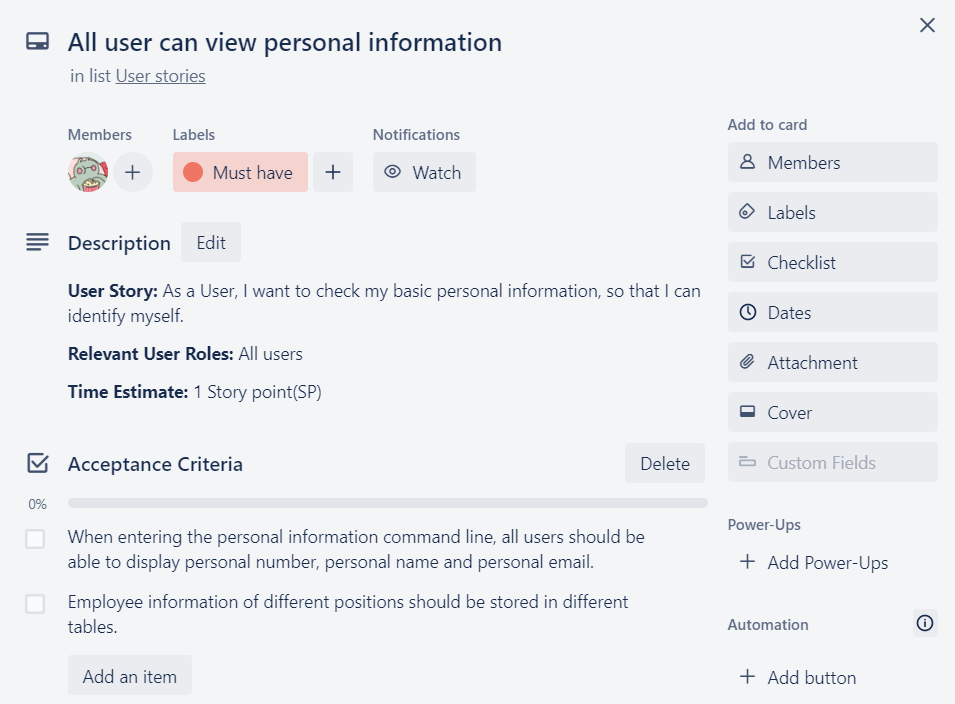


Figure 1.3: UserStory\_3

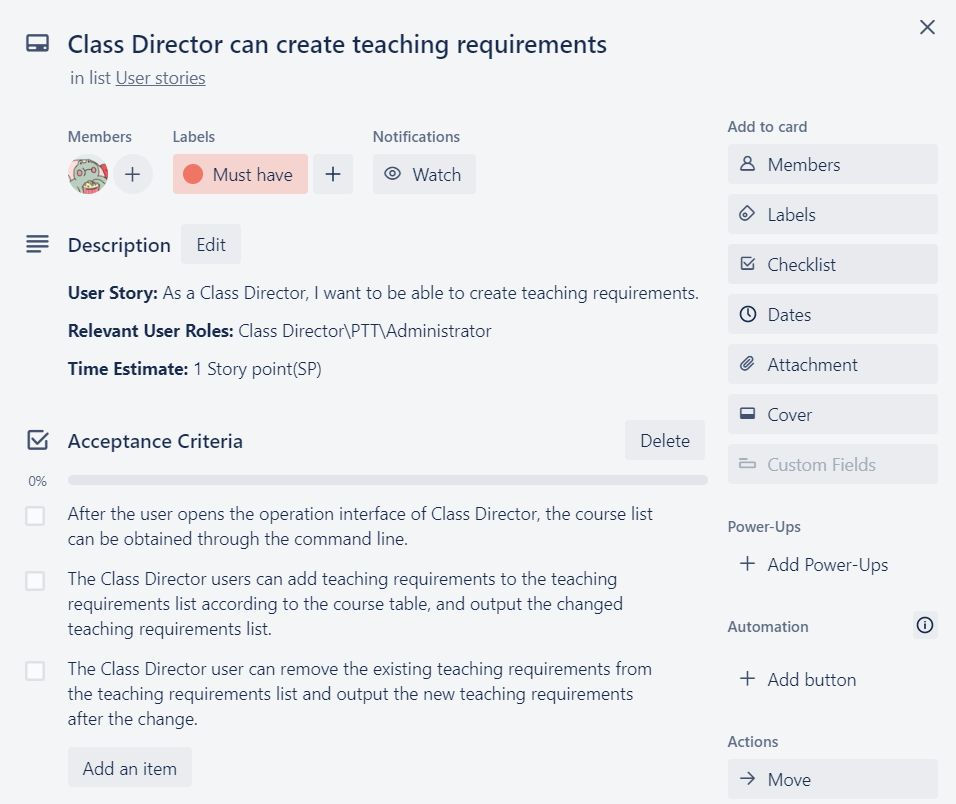


Figure 1.4: UserStory\_4

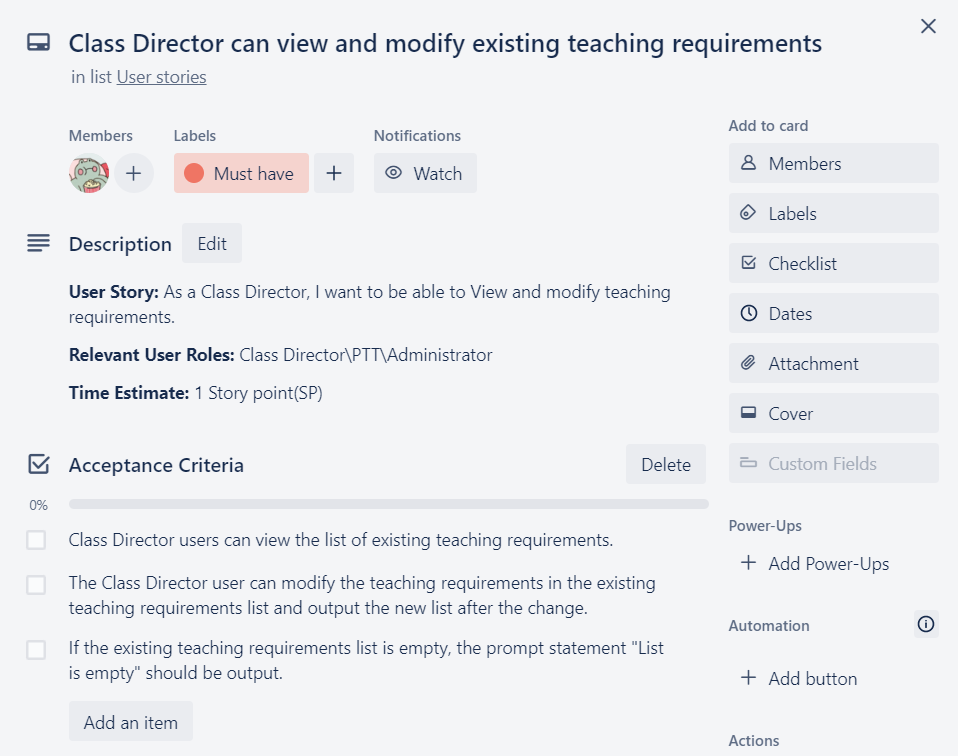


Figure 1.4: UserStory\_4

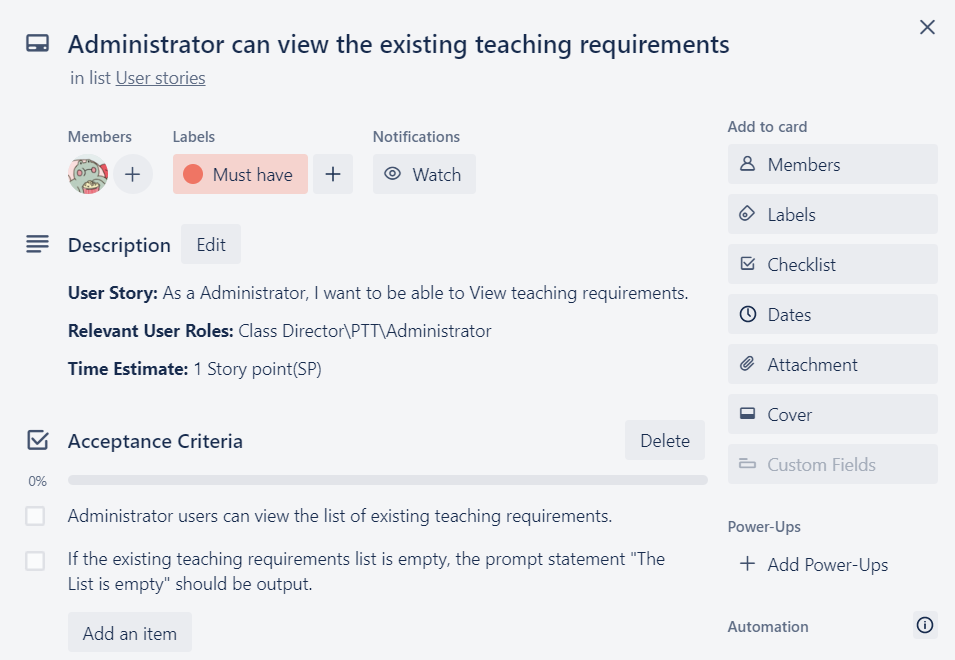


Figure 1.5: UserStory\_5

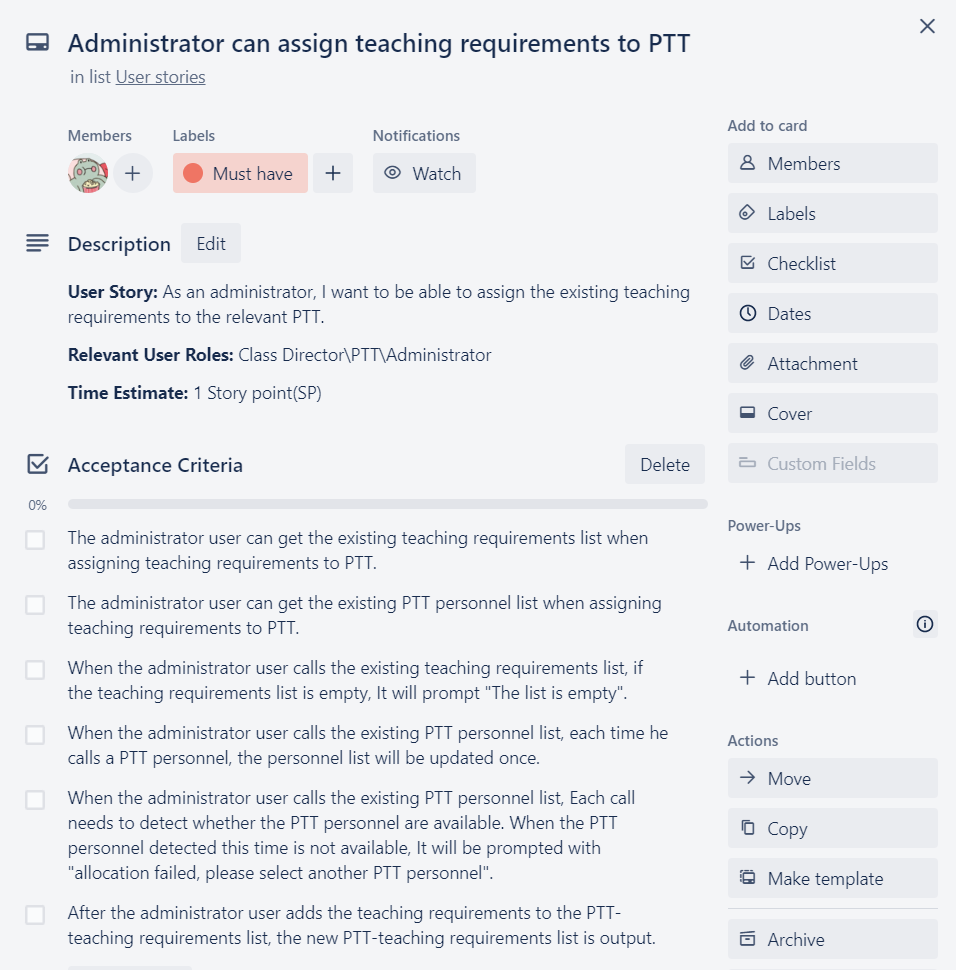


Figure 1.6: UserStory\_6

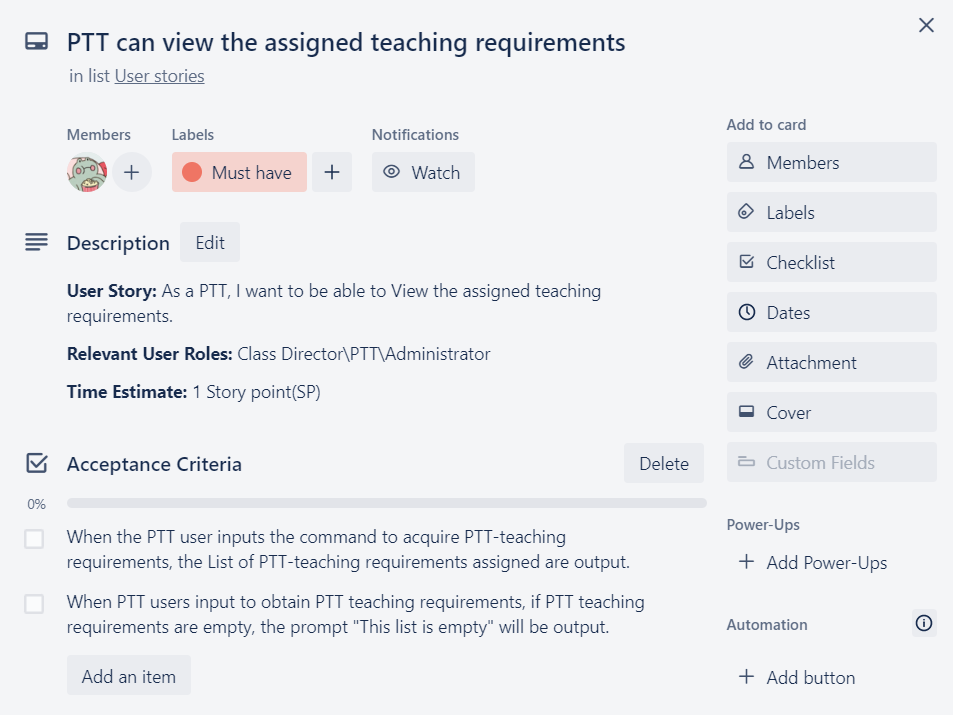


Figure 1.7: UserStory\_7

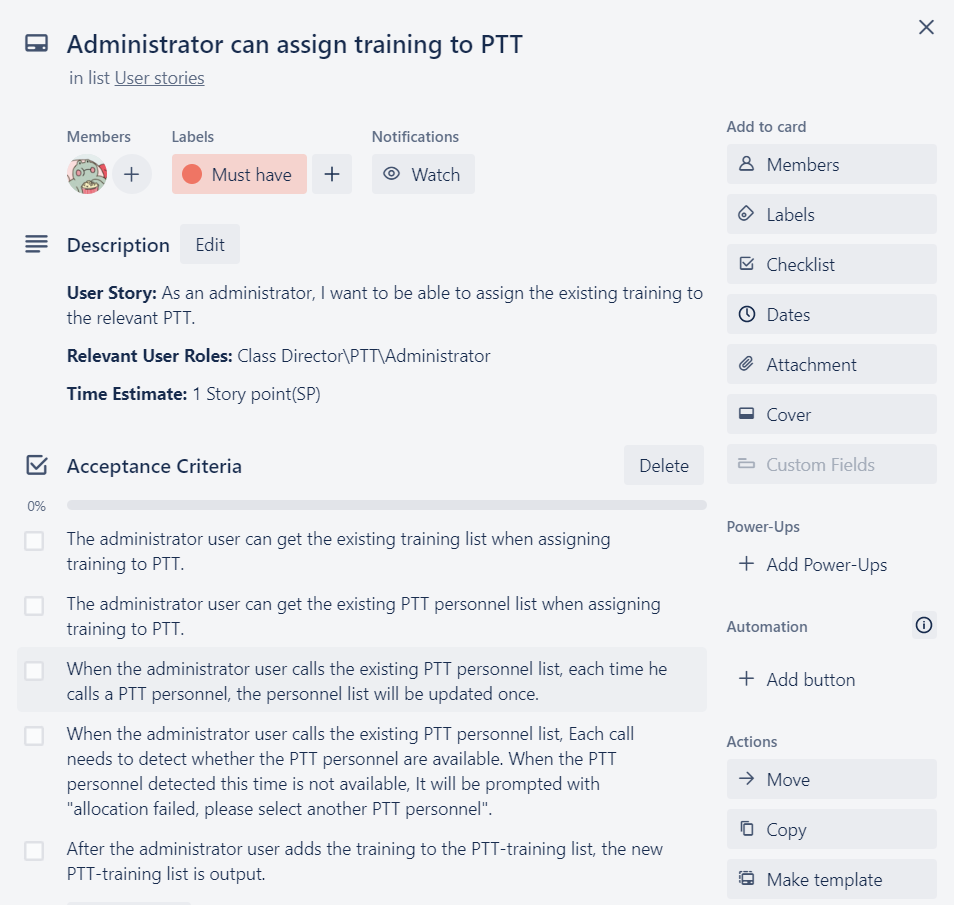


Figure 1.8: UserStory\_8

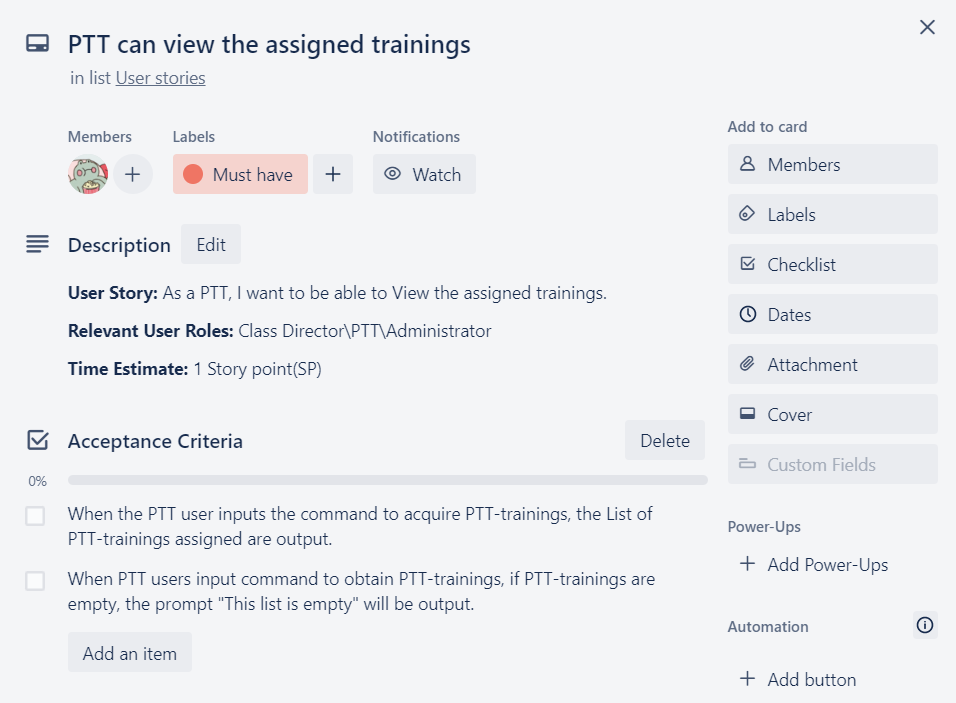


Figure 1.9: UserStory\_9

This is our user story, and we have basically completed most of the content required in the user story

# Class structure diagram (UML)

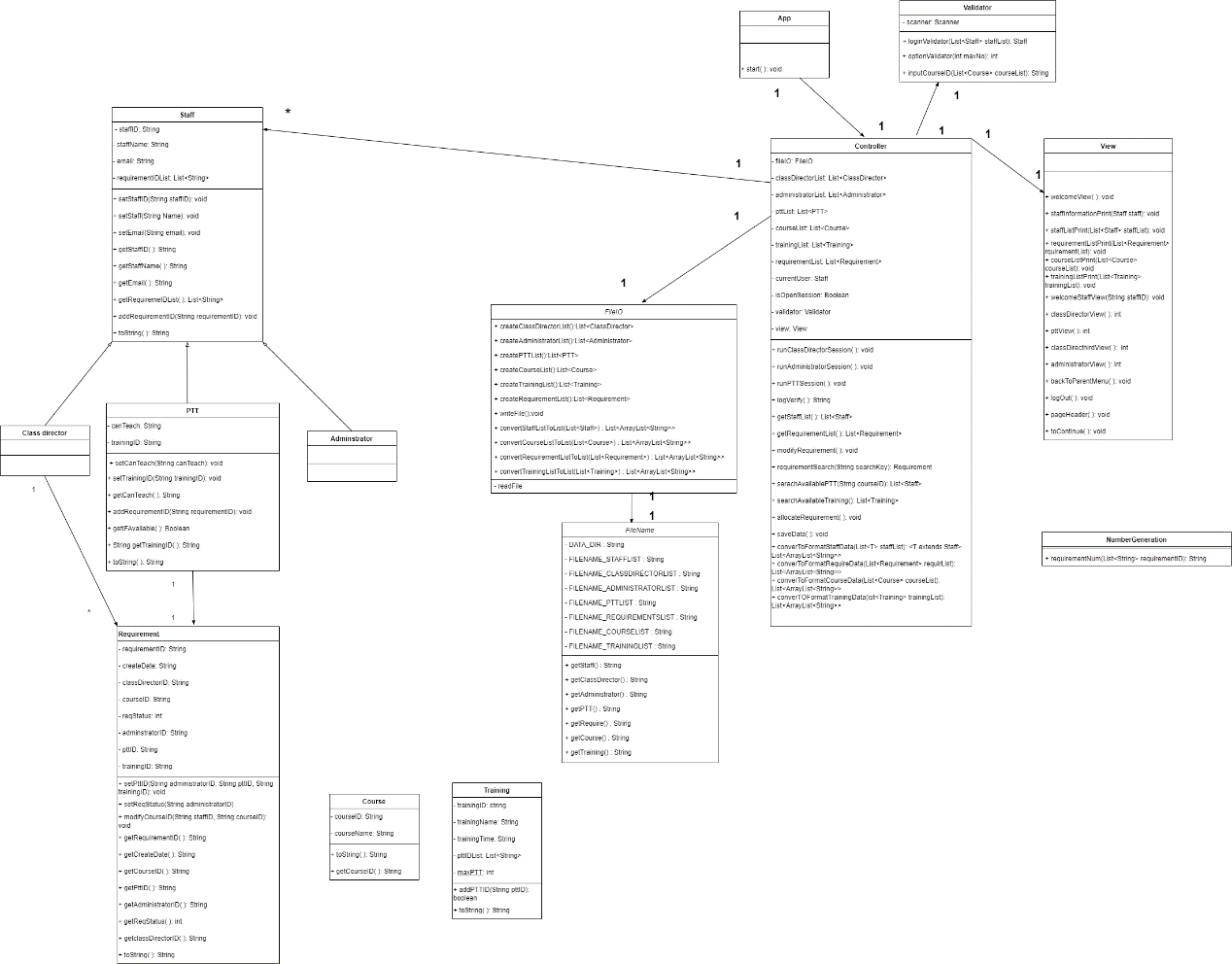


Figure 2.1: UML Diagram

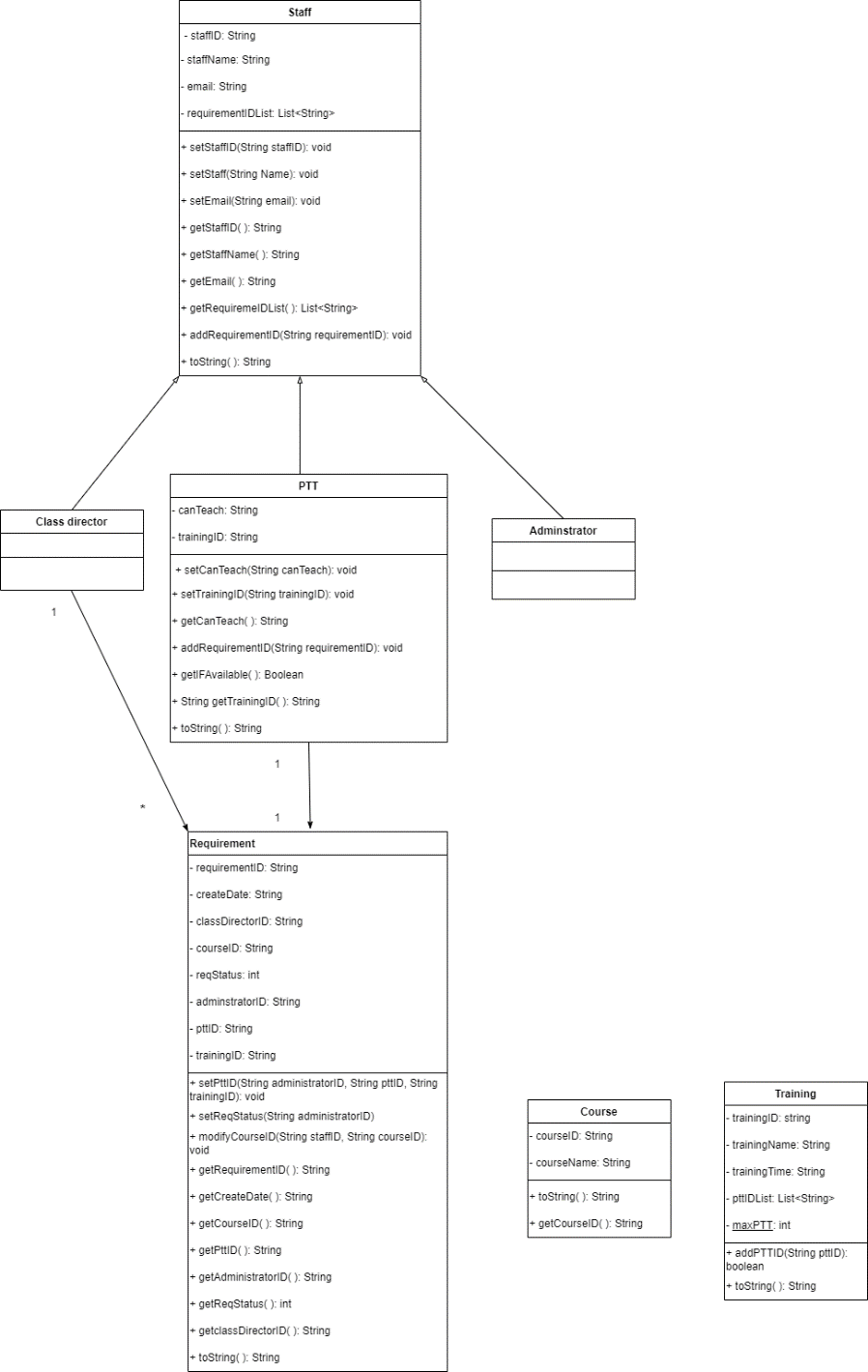


Figure 2.2: Detailed UML Diagram 2

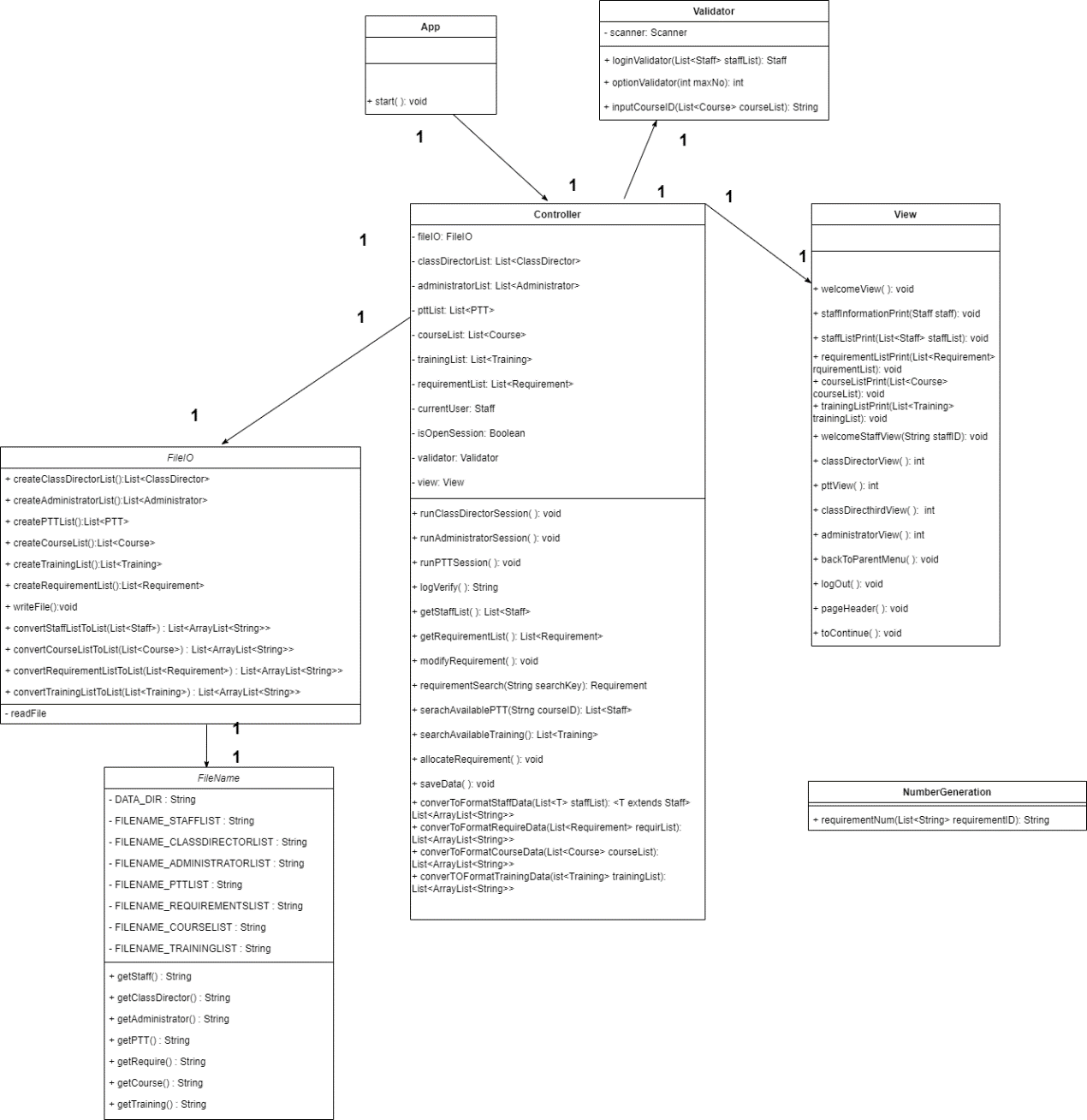


Figure 2.3: Detailed UML Diagram 3

# Software structure and design patterns

In our project design process, we chose MVC as our software architecture and divided the software application into three core parts: the Model, the View, and the Controller, each responsible for different tasks.

The Model is responsible for processing data and represents the business logic and data in the application, including operations such as data storage, retrieval, update, and deletion. The View is responsible for presenting data and is typically a part of the user interface, presenting data in a visual way to the user. The Controller is the interactive part of the application that receives input from the user and passes it on to the Model or View for processing. The Controller operates the Model and View based on the user's requests to ensure proper application operation.

In the MVC structure, each component can be developed and maintained independently, making MVC particularly suitable for large, complex applications because it improves code reusability, scalability, and maintainability.

The singleton pattern is a very useful creational design pattern that ensures a class has only one instance and provides a global access point for that instance, making it easy to access the unique instance. The singleton pattern is very useful when it is necessary to restrict the instantiation of a class, such as controlling resource access or ensuring that only one database connection instance exists.

In our design, we applied the singleton pattern to ensure that all function classes are singletons. This avoids recursion when users frequently switch between different functions, thus improving code performance and readability.

Additionally, we also use the singleton pattern to handle global variables, including lists of all Staff, all teaching requests, filenames for storing permanent information, and the current logged-in user (Staff). By treating these global variables as singletons, we ensure that they have only one instance and can be easily called by other modules and functions, thus improving code reusability and maintainability.

# Sequence diagrams for each user story

A sequence diagram for a user story is a graphical representation that displays the interaction process between objects in a user story. The use of sequence diagrams can help the development team better understand user requirements and expectations by displaying the interaction process between the user and the system. Team members can gain a clearer understanding of the user's actions and expected results through the sequence diagram, allowing for better development of system functionality that meets user needs. Therefore, we have designed a sequence diagram for each user story to guide and illustrate the program development process. The specific sequence diagram is shown below.

**User Story:** As a user, I want to be able to log in my account, so that I can use my account at any time.

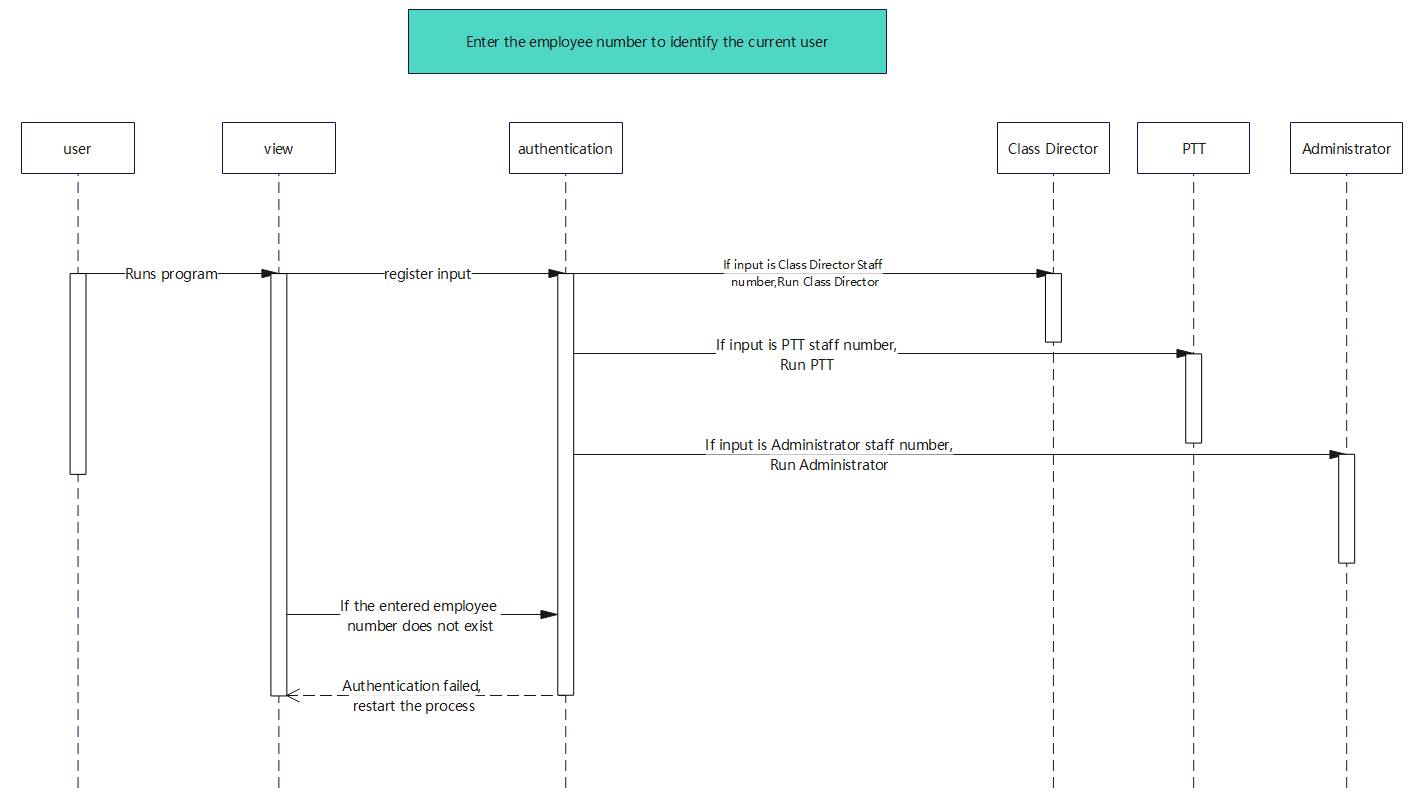


Figure 3.1

**User Story：**As a user, I want to be able to log out of my account，so that I can protect my account information.

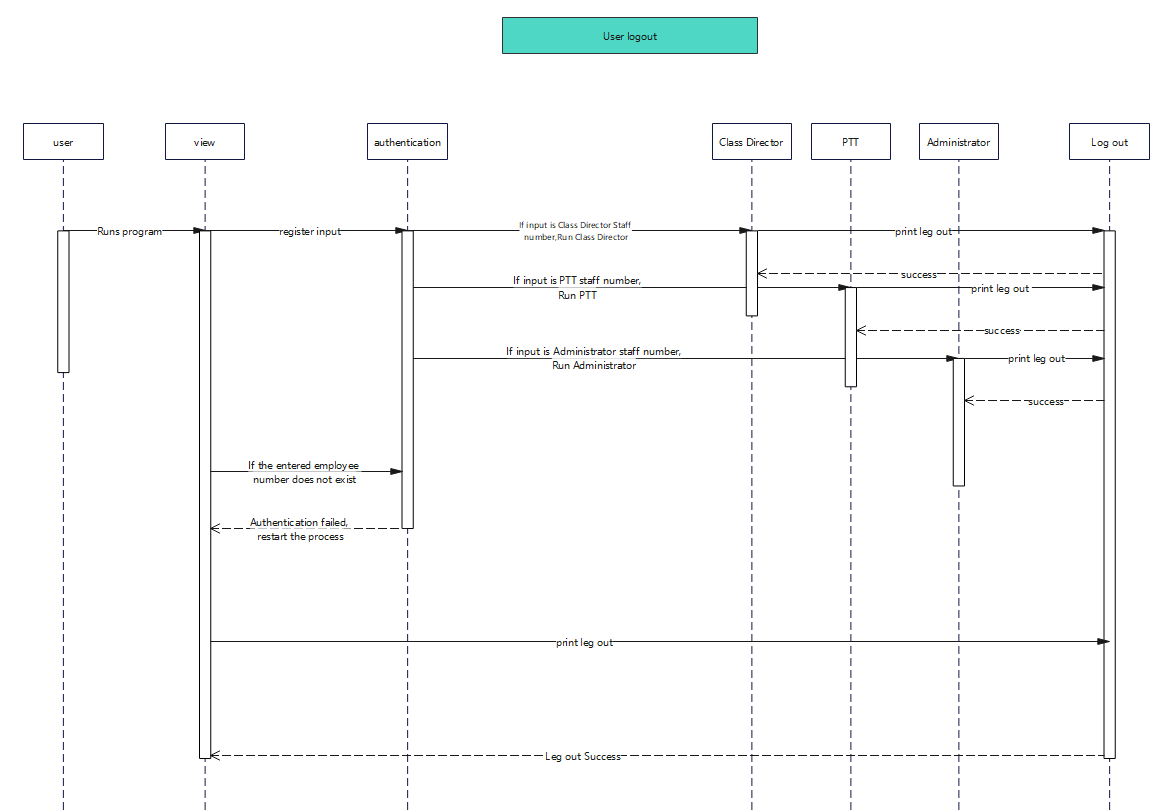
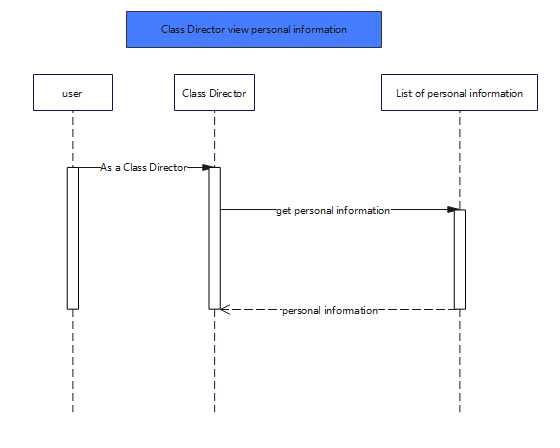


Figure 3.2

**User story：**As a User, I want to check my basic personal information, so that I can identify myself.



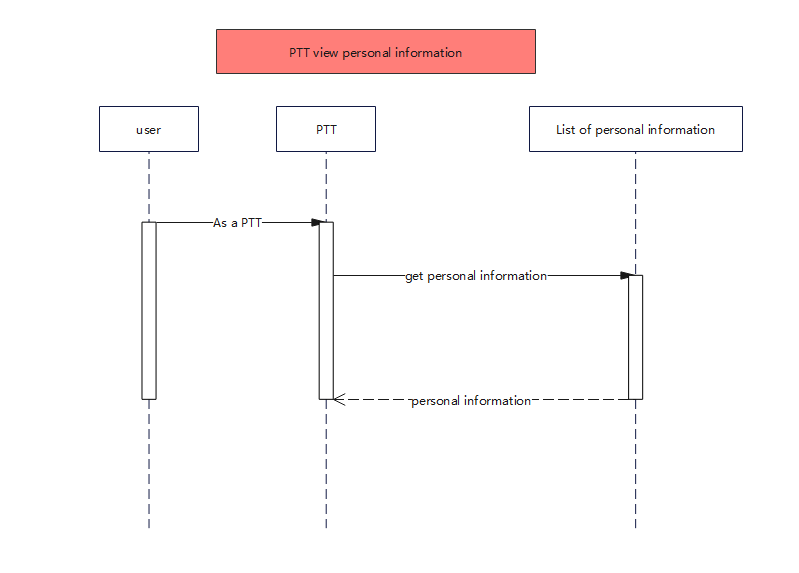


Figure 3.3

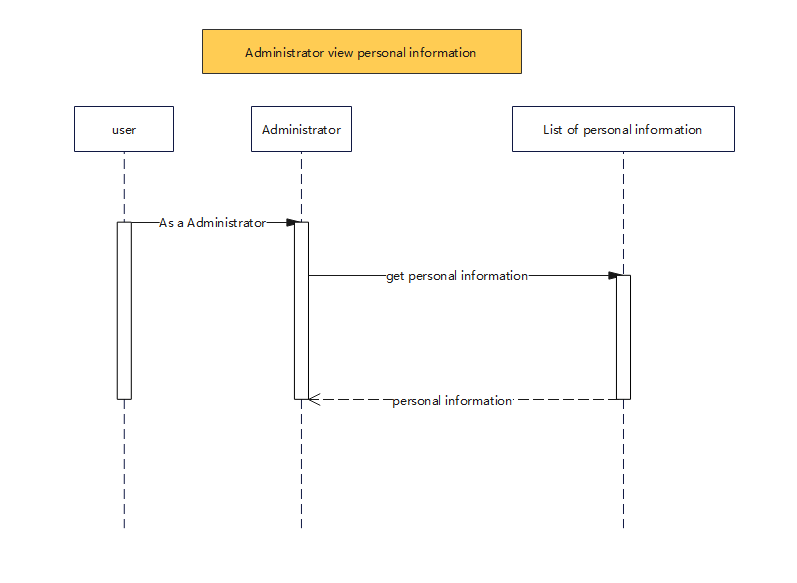


Figure 3.4

**User Story：**As a Class Director, I want to be able to create teaching requirements.

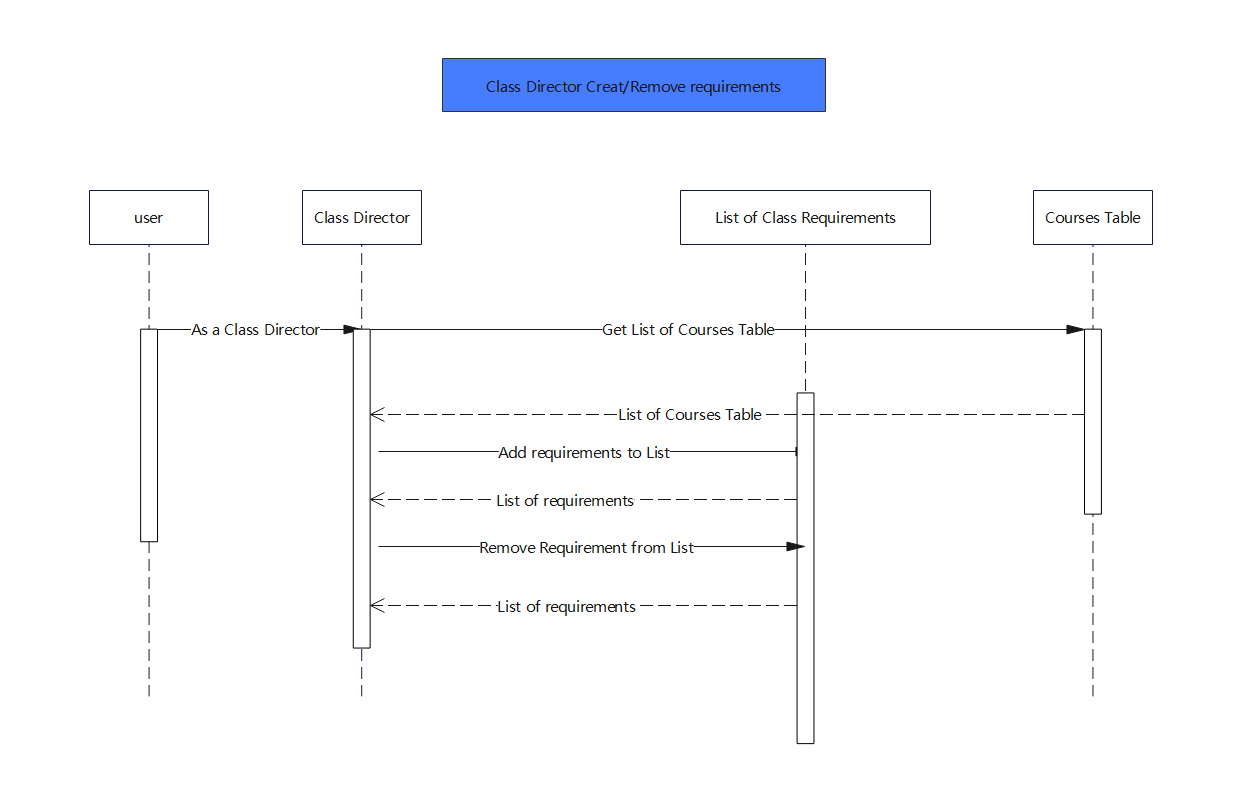


Figure 3.5

**User Story：**Class Director can view and modify existing teaching requirements

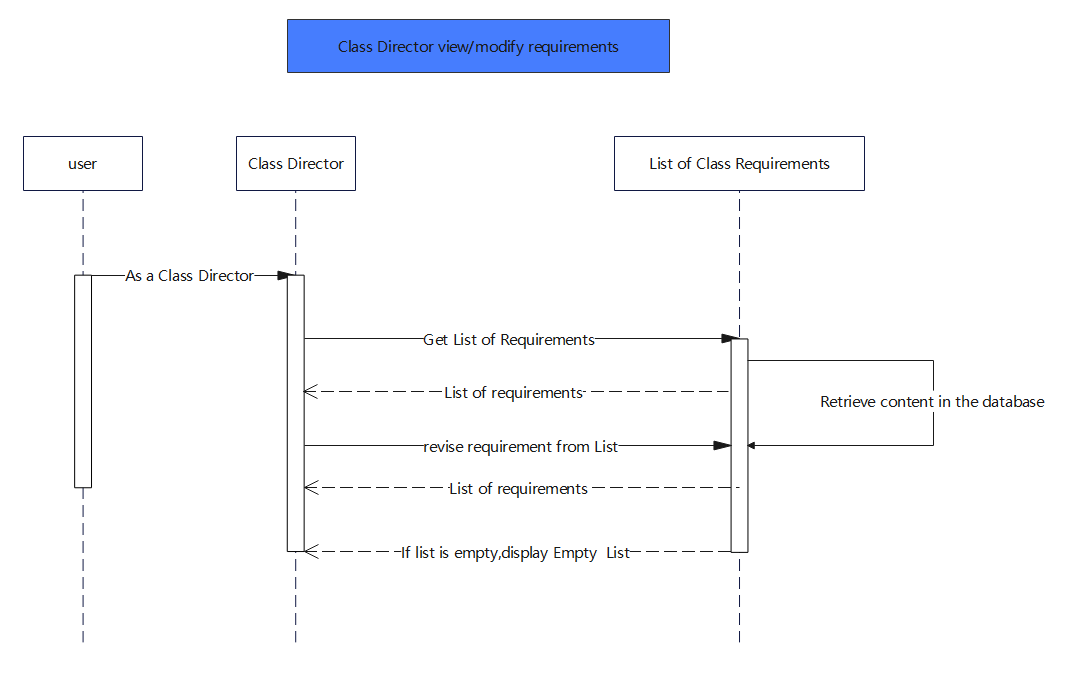


Figure 3.6

**User Story:** As a Administrator, I want to be able to View teaching requirements.

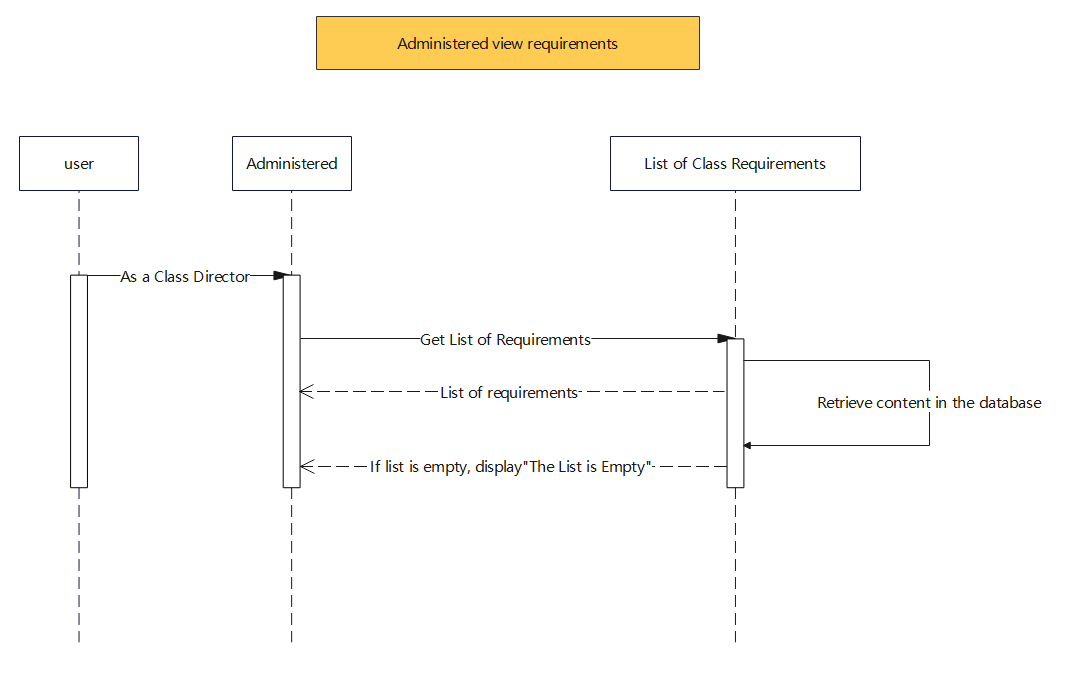


Figure 3.7

**User Story: As an administrator, I want to be able to assign the existing teaching requirements to the relevant PTT.**

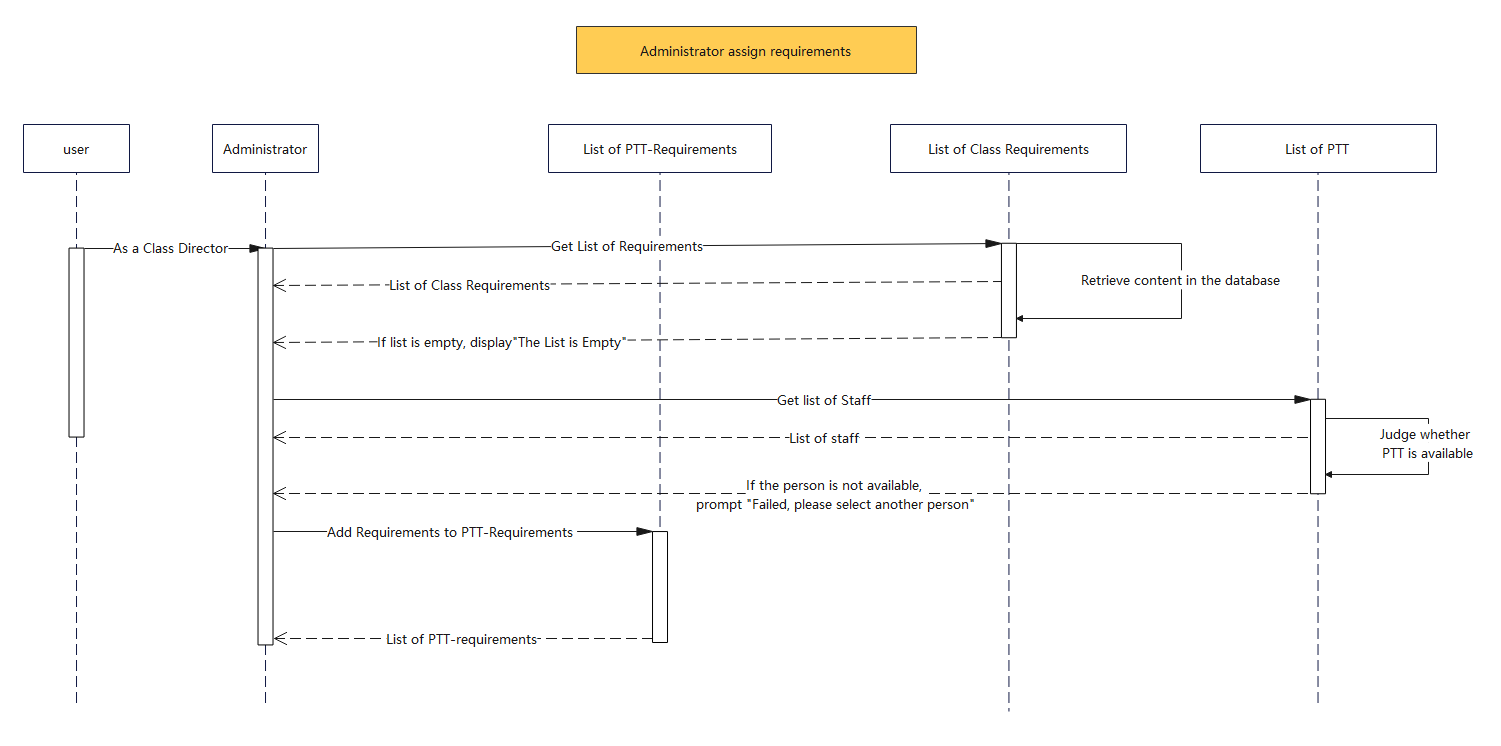


Figure 3.8

**User Story: As an administrator, I want to be able to assign the existing training to the relevant PTT.**

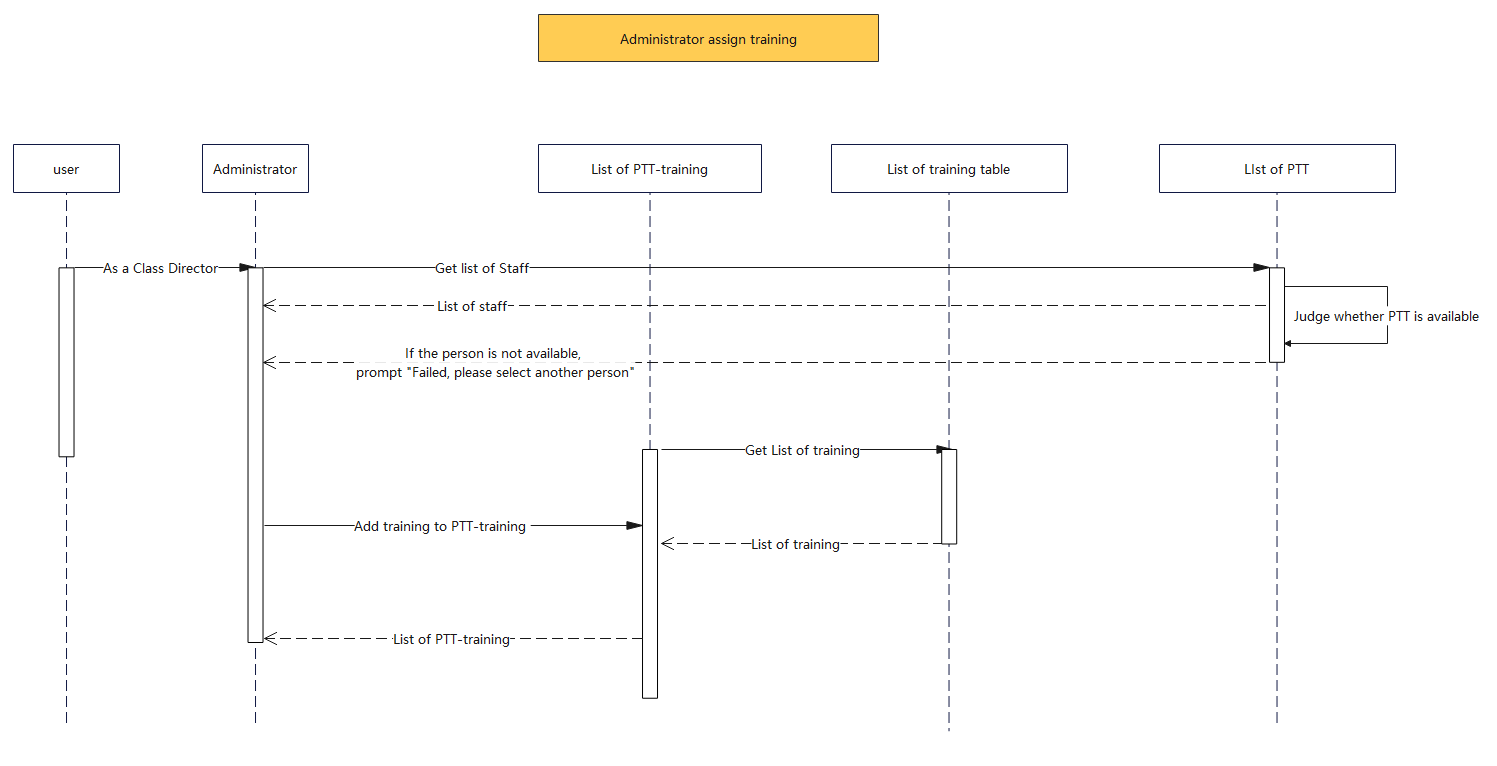


Figure 3.8

**User Story: As a PTT, I want to be able to View the assigned teaching requirements.**

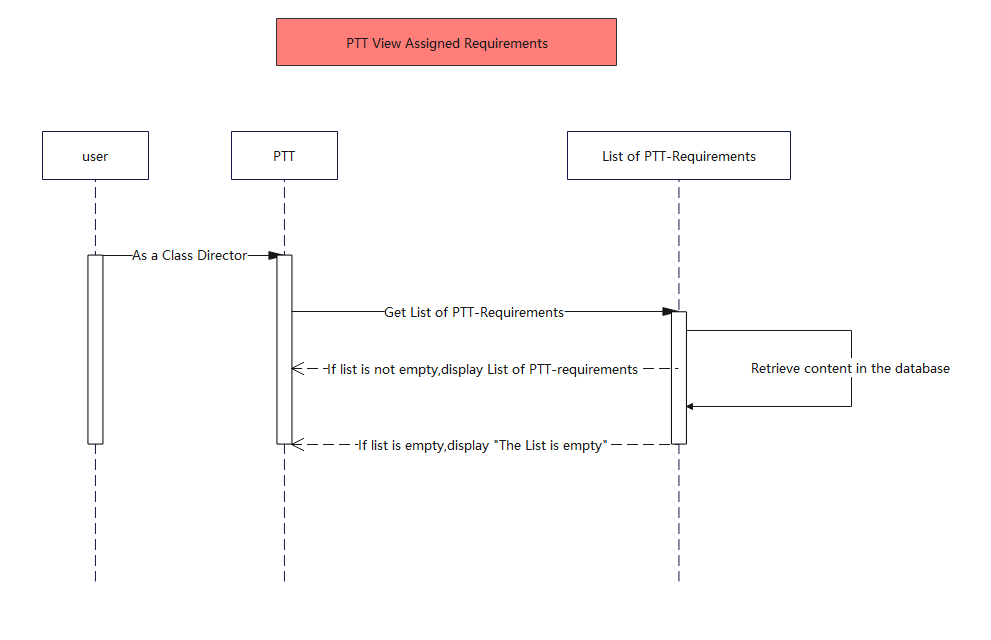


Figure 3.9

**User Story:** As a PTT, I want to be able to View the assigned trainings.

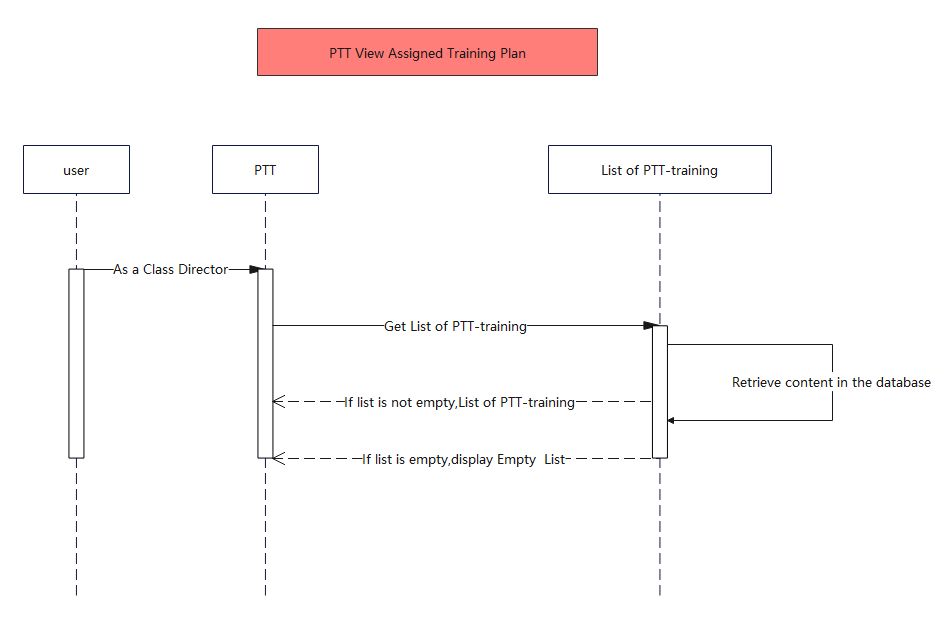


Figure 3.10

# System operation results

## Administrator Test

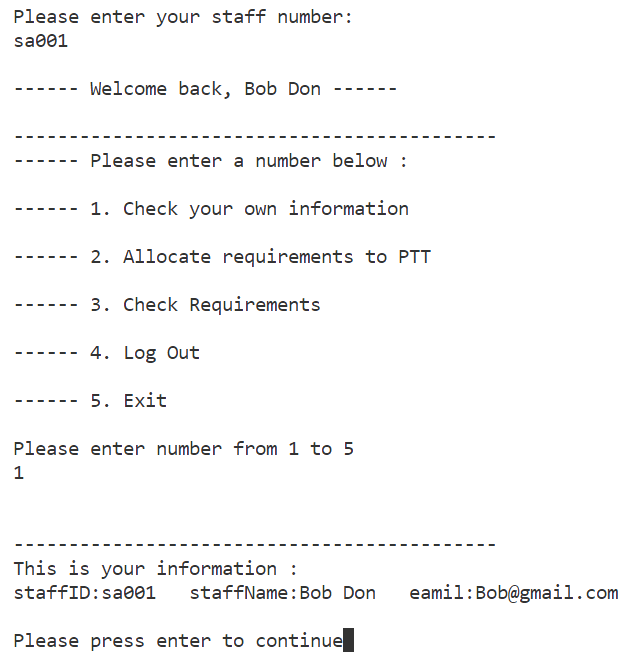
****

Figure: Administrator Test\_1

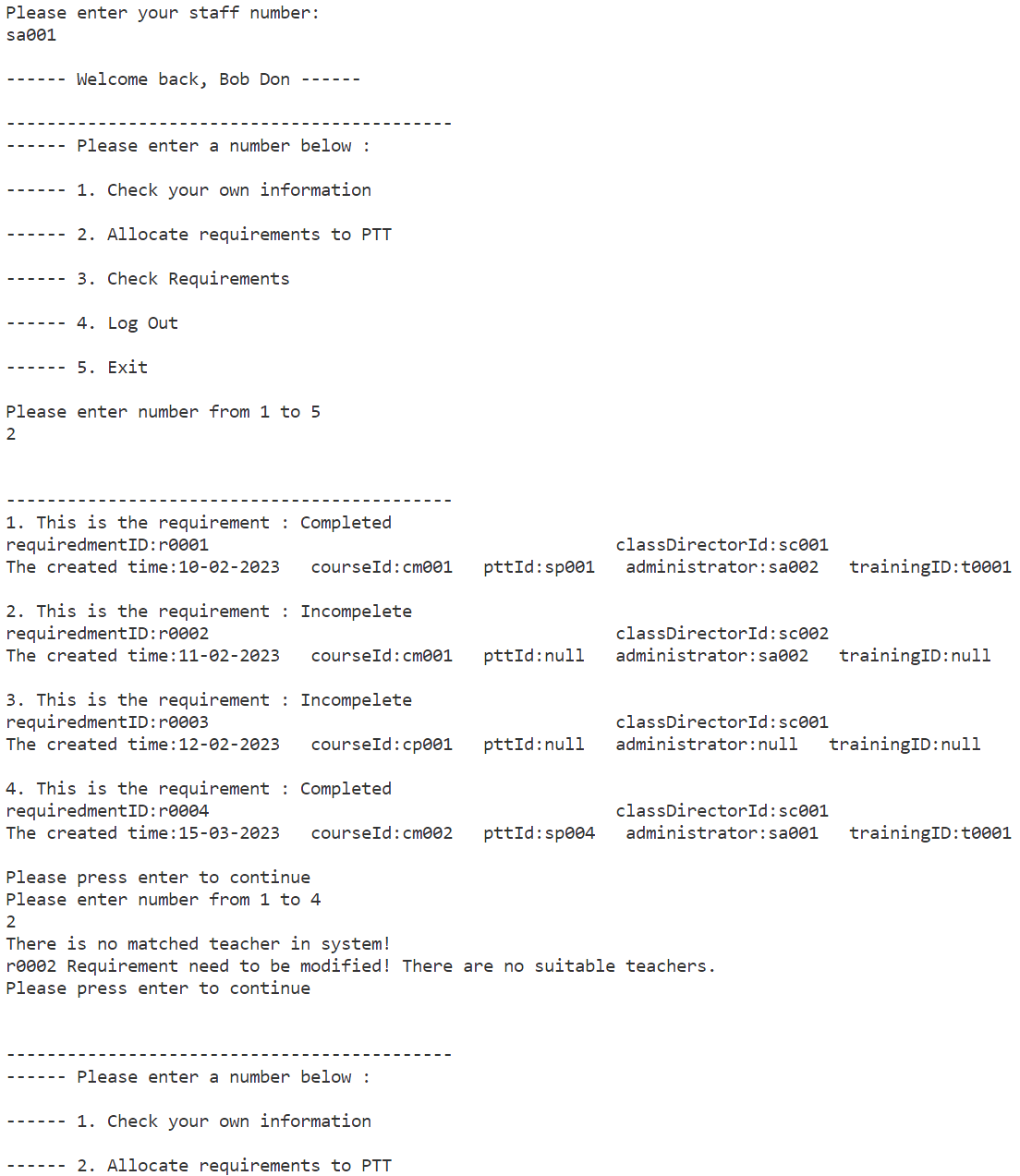


Figure: Administrator Test\_2

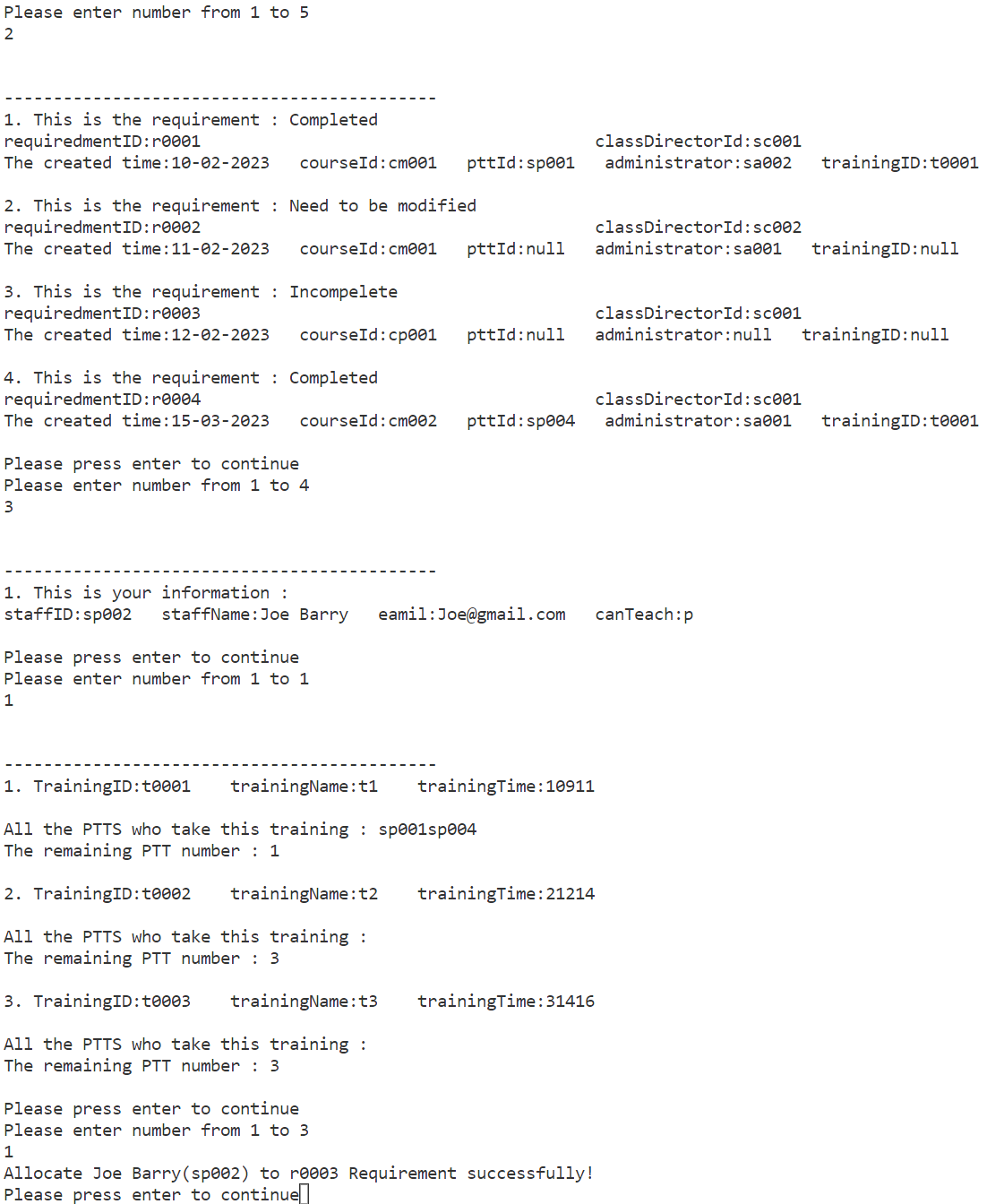


Figure: Administrator Test\_3

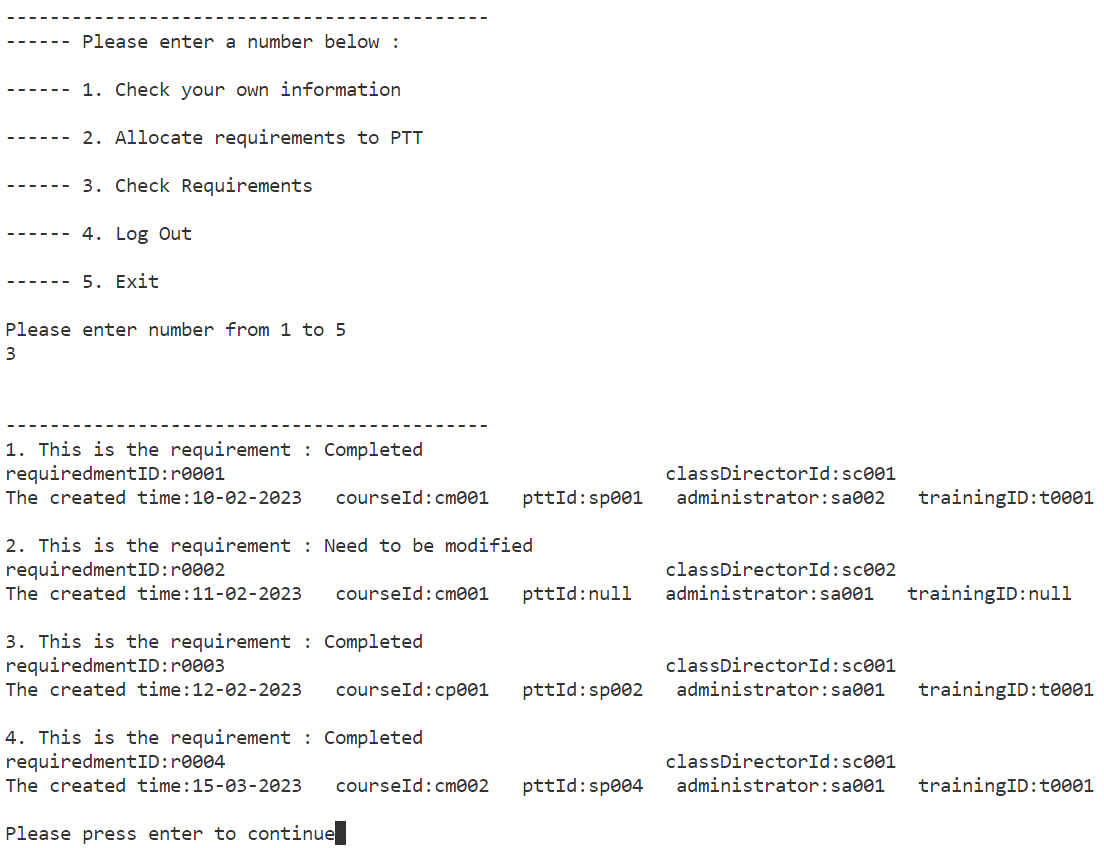


Figure: Administrator Test\_4

**Classdirector Test**

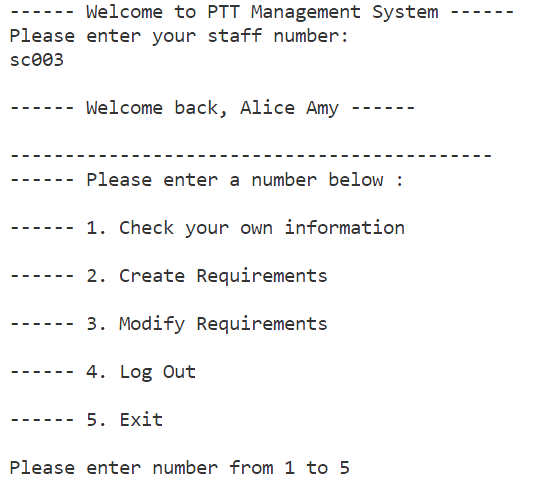
****

Figure : ClassDirector Test\_1

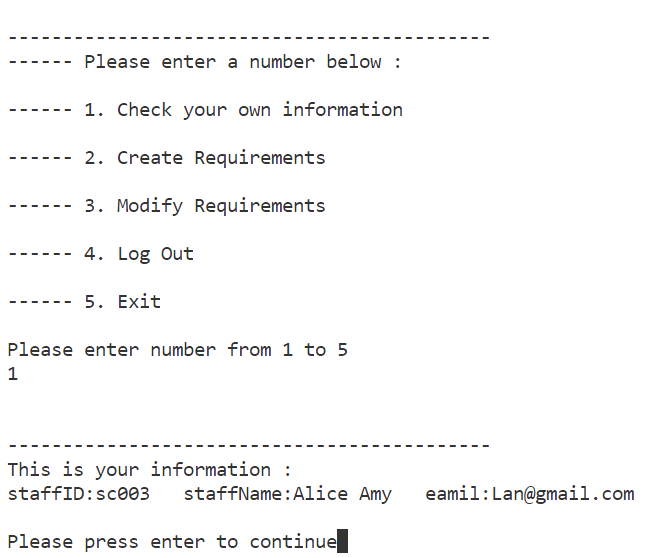
****

Figure : ClassDirector Test\_2

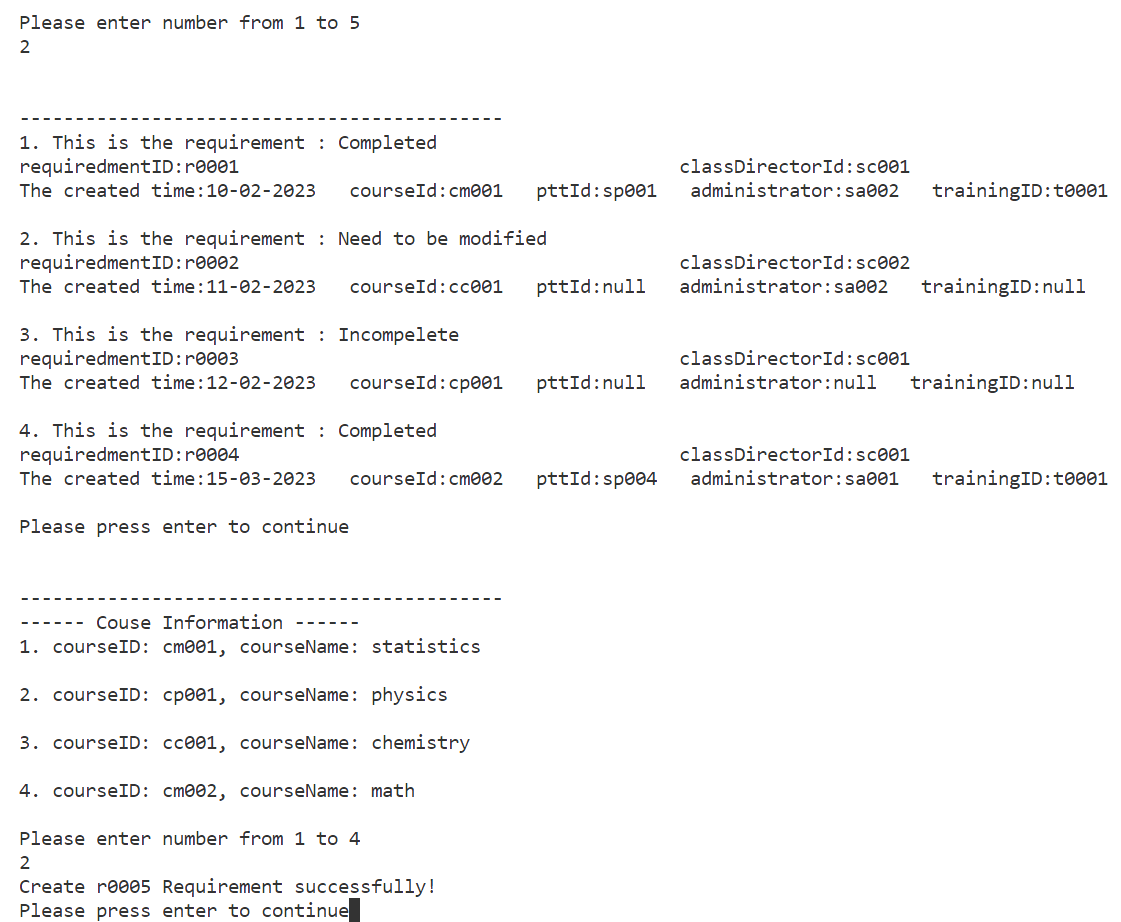


Figure : ClassDirector Test\_3

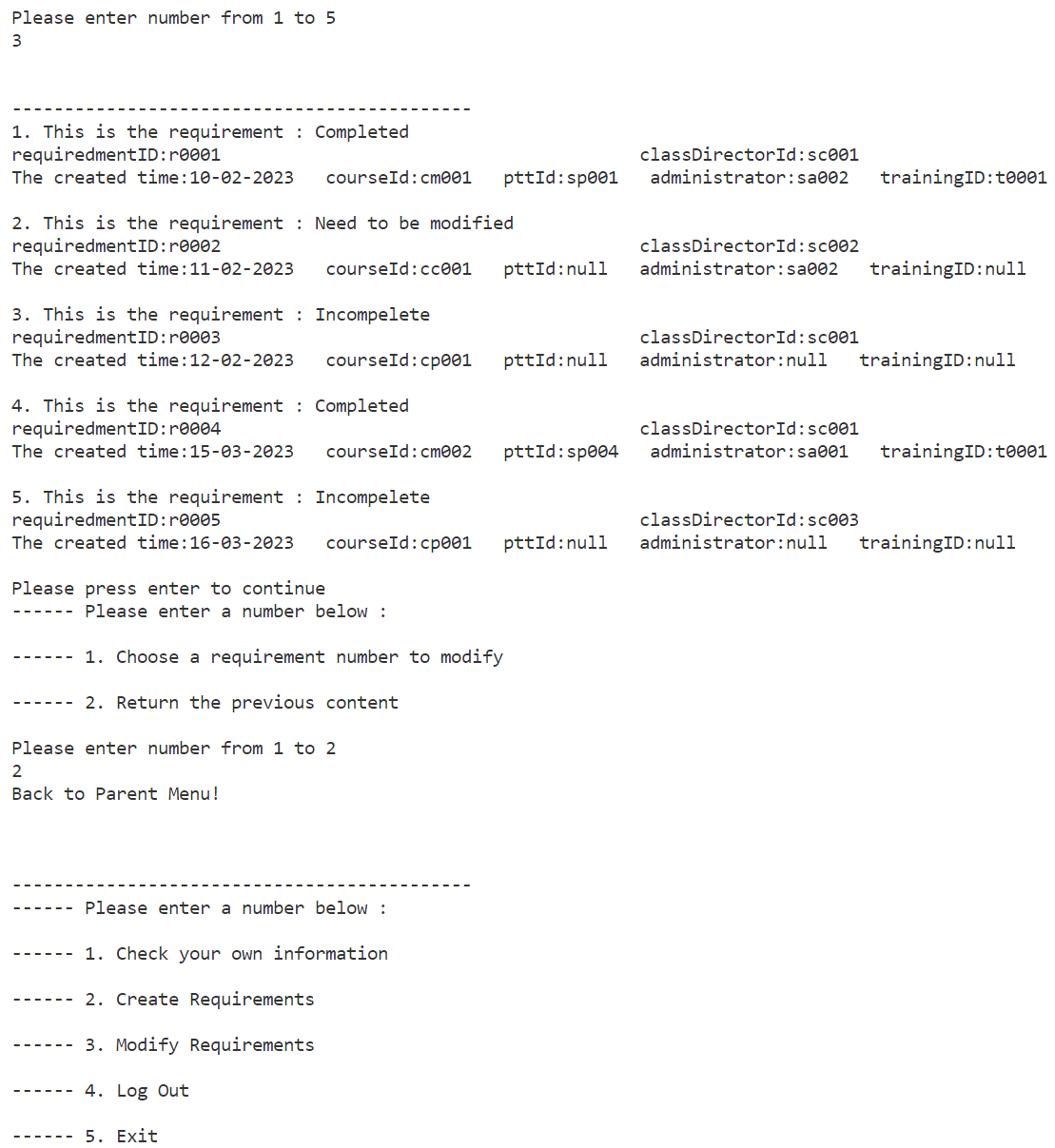


Figure : ClassDirector Test\_4

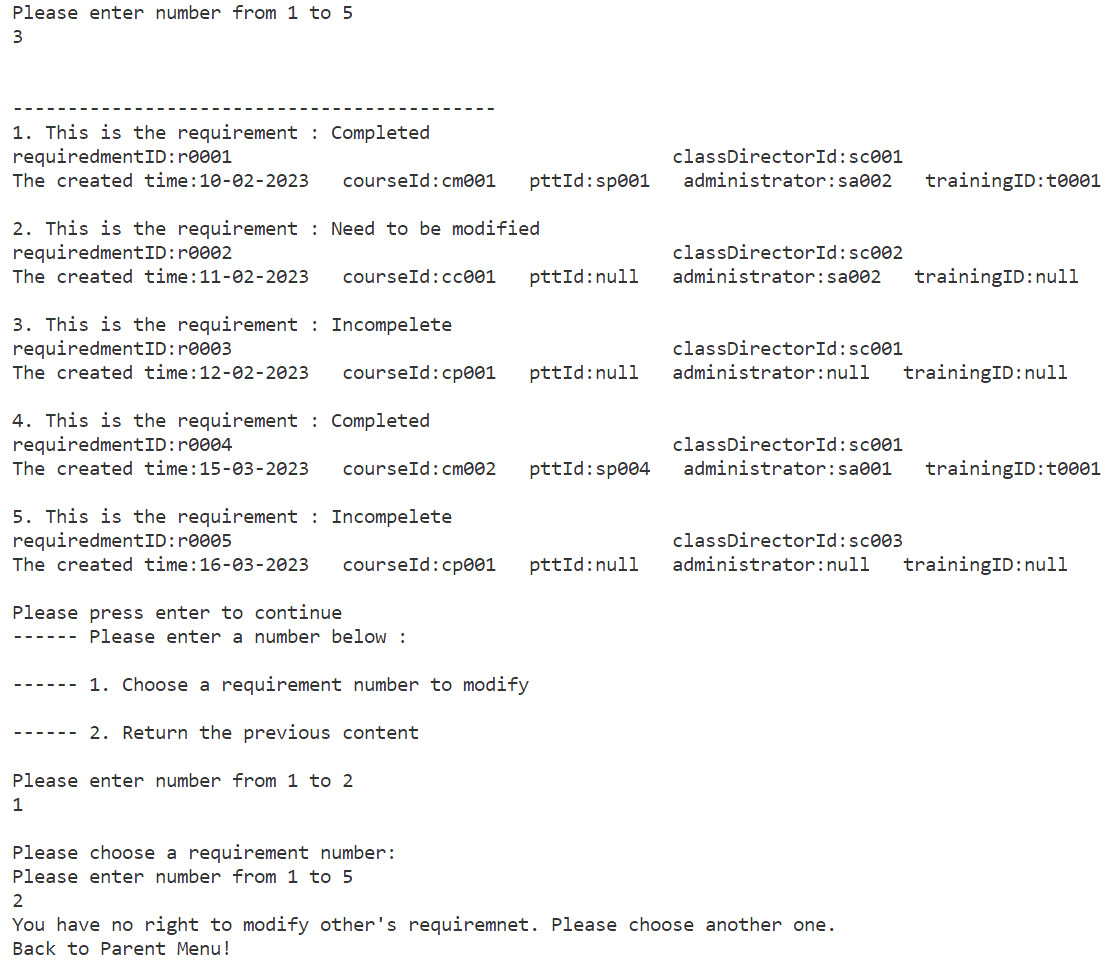


Figure : ClassDirector Test\_5

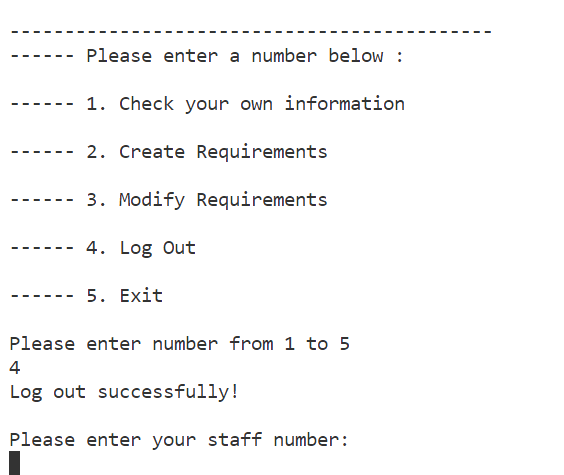


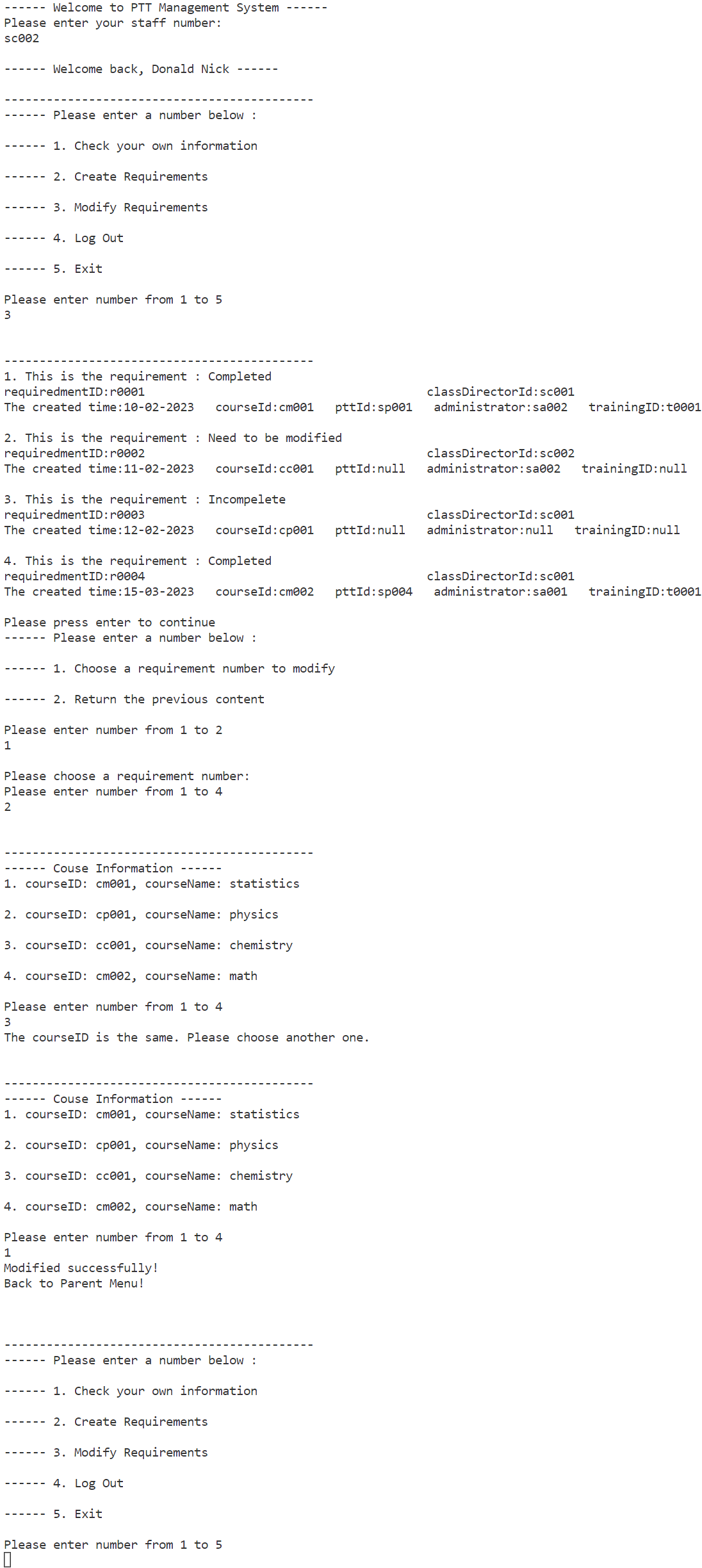
Figure : ClassDirector Test\_6

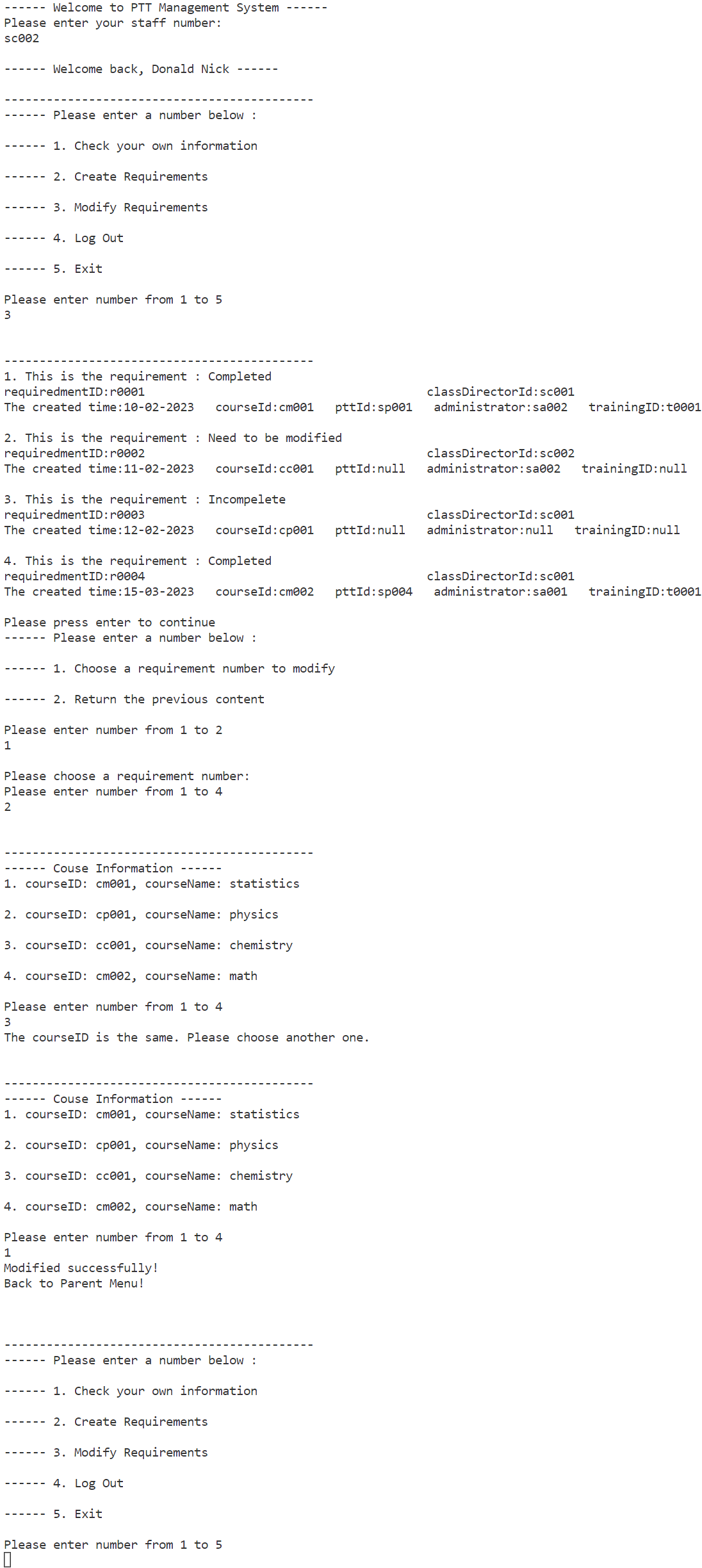
Figure : ClassDirector Test\_6

Figure : ClassDirector Test\_7

## Part Time Teacher

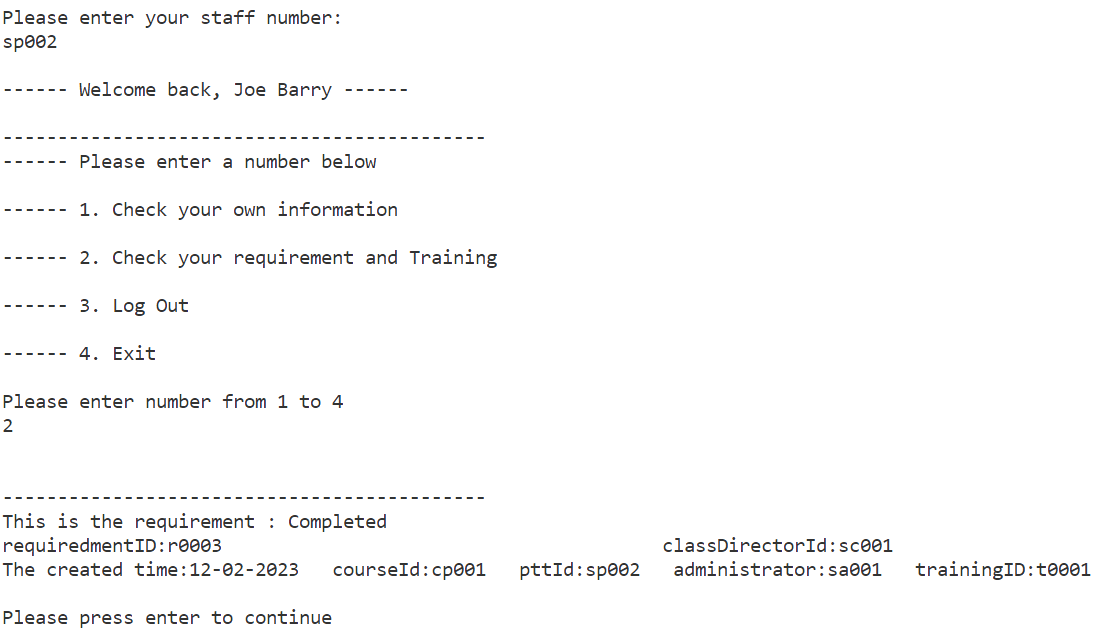
****

Figure : PTT Check\_Requirement Test

## Output to the txt file

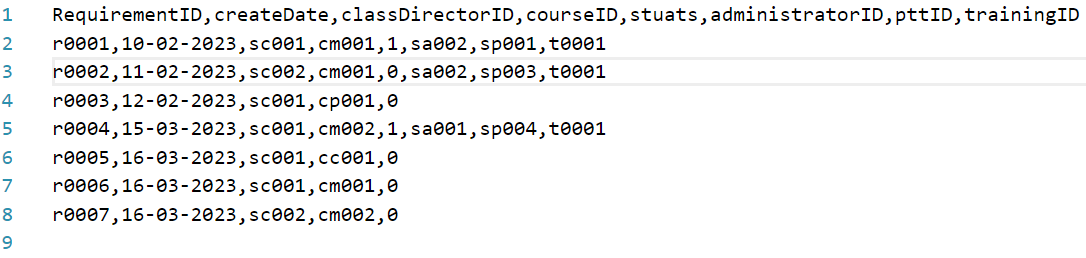
****

Figure : Data Output Test

**Each role class has a corresponding txt file which stores the relevant data for that class. First, we read the data from the txt file and create the corresponding class based on the data that was read. Then, we follow the prompts in the menu to select the corresponding options and test whether each menu option can provide the correct output.**

# Retrospective

Fortunately, everyone on our team is willing to share the work and be responsible for it. Therefore, our team is very cohesive and happy to cooperate. I think this is the main advantage of our team. However, it has to be mentioned that the user story is determined and conceived at the early stage of the project, and it is not clear how to implement the content of the project design according to the requirements at the early stage of the project design. Therefore, in the second stage of project design, we spent a lot of time discussing how to realize user stories, how to determine the final scale of the project, and how to filter our final user stories. After confirming the user story, there are some differences in how to implement each function. Some members want to use a more centralized and simple way to achieve it, while others want to ensure the diversity and reliability of the final product, and different ideas determine the amount of code in the implementation process.

Due to the size of the project, we encountered some difficulties in determining the distribution of each member's class. Therefore, we chose to actively hold a group meeting to ensure that the understanding and cooperation between our groups are strengthened and that the final task is smoothly distributed among members. Positive communication and cooperation saved us a lot of time for later modification and adaptation. When confirming the structure level of the project, due to the lack of discussion on the relationship between classes in the early stage of our project design, it is difficult and inconsistent to determine a reasonable design pattern and structure level.

The final code implements the basic functions of our selected user stories. In the actual future iterations of the plan, the basic functions may be enriched and improved, and the diversity will be increased.