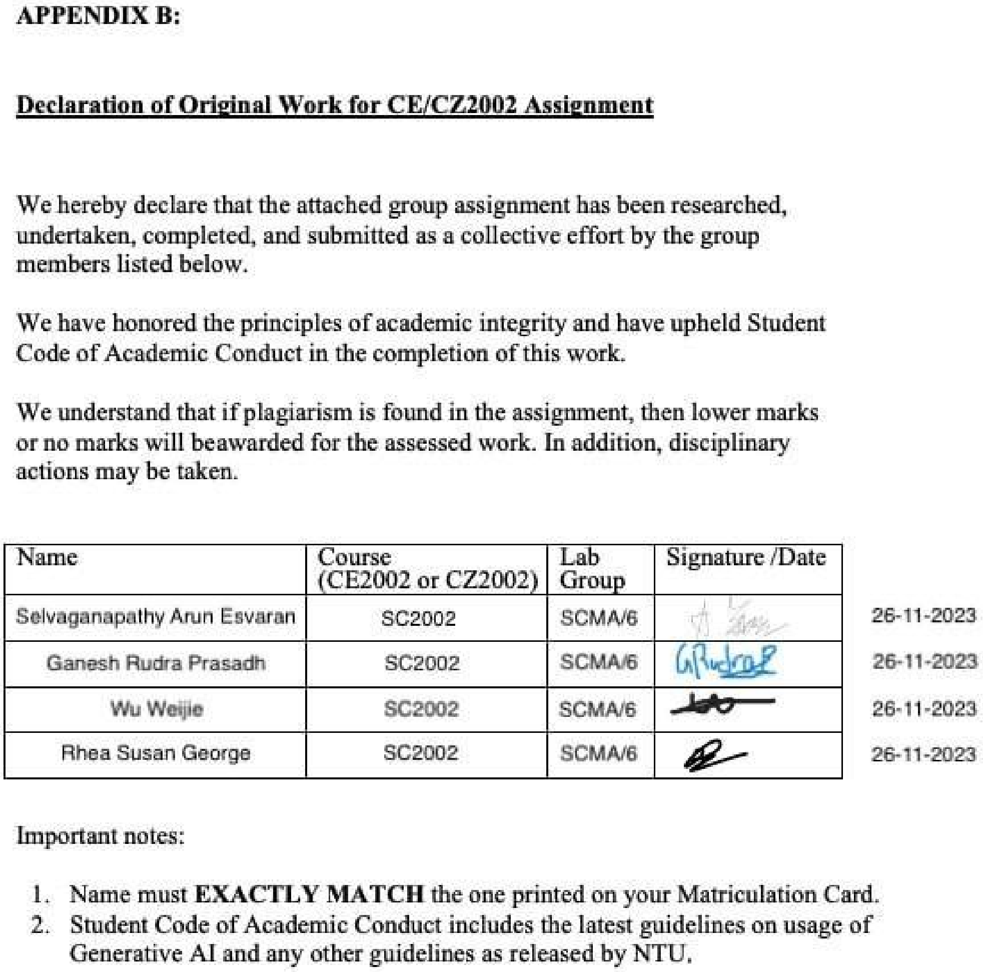
SC2002 FINAL REPORT

CAMS (CAMP APPLICATION AND MANAGEMENT SYSTEM)

TUTORIAL SCMA - GROUP 6

26TH NOVEMBER 2023



APPENDIX B:

Declaration of Original Work for CE/CZ2002 Assignment

We hereby declare that the attached group assignment has been researched, undertaken, completed, and submitted as a collective effort by the group members listed below.

We have honored the principles of academic integrity and have upheld Student Code of Academic Conduct in the completion of this work.

We understand that if plagiarism is found in the assignment, then lower marks or no marks will beawarded for the assessed work. In addition, disciplinary actions may be taken.

Name	Course (CE2002 or CZ2002)	Lab Group	Signature /Date	
Selvaganapathy Arun Esvaran	SC2002	SCMA/6		26-11-2023
Ganesh Rudra Prasadh	SC2002	SCMA/6	GRudraP	26-11-2023
Wu Weijie	SC2002	SCMA/6		26-11-2023
Rhea Susan George	SC2002	SCMA/6		26-11-2023

Important notes:

1. Name must **EXACTLY MATCH** the one printed on your Matriculation Card.
2. Student Code of Academic Conduct includes the latest guidelines on usage of Generative AI and any other guidelines as released by NTU.

Table of Contents

1. Introduction

The Camp Application and Management System (CAMs) is an intricate piece of software designed to facilitate the management of camps, student registrations, and operations within an educational institution. This report elucidates the adherence to **SOLID** design principles within the CAMs system, which is illustrated through the examination of various components of its codebase. In addition to implementing **SOLID** principles, the system ensures effective usage of loose coupling, high cohesion, reusability of code, and generalization relationships, which are salient features of good object-oriented programming (OOP) design.

2. Design Considerations

2.1 SOLID Design Principles

2.1.1 Single Responsibility Principle (SRP)

The SRP advocates for a class to have a singular responsibility, and this is well exemplified in a variety of classes. The *Initialize* class possesses the responsibility of setting up the initial state of the system, including loading and initializing student and staff data from the external files. The *Getter* class has the responsibility of extracting and processing the usernames from given inputs. The *Search* class searches and retrieves specific student or staff objects based on the given usernames, centralizing the search logic within the system. The *Login* class ensures the user authentication process is correctly validated during the login process. Finally, the *password* class takes charge of password-related operations: validation, changing passwords, and password policy enforcement.

2.1.2 Open/Closed Principle (OCP):

CAMs leverages the OCP through interfaces like *CampManagerInterface.java*, which *CampManager.java* implements. This allows new functionalities, such as different types of camp management, to be added without modifying existing code. The system can introduce new classes that implement CampManagerInterface, thus extending the application's capabilities without altering the classes that already implement the interface. *CampViewer* interface acts as a flexible blueprint for viewing camp details by allowing various user types such as *StudentViewer*, *StaffViewer*, and *CampCommMemberViewer* to extend their functionalities without altering the interface.

2.1.3 Liskov Substitution Principle (LSP):

In CAMs, LSP is maintained through the class hierarchy involving *User.java*, *Student.java*, and *Staff.java*. As Student and Staff extend User, they can be used interchangeably wherever a User instance is expected. Moreover, the *CampCommitteeMember*, which is a subclass of Student, adheres to the LSP rules by extending the functionalities of Student without altering the behavior of the Student class itself. This ensures that it can be utilized wherever a Student is expected without jeopardizing the system's integrity. This interchangeability allows for derived classes from User to be substituted without affecting the program's functionality.

2.1.4 Interface Segregation Principle (ISP):

The ISP is applied in CAMs with interfaces like *CampViewer.java* and *EnquiryManager.java*, which are specific to certain functionalities. For instance, CampViewer outlines methods for viewing camp details, and EnquiryManager provides an abstraction layer for managing enquiries. This enables classes to implement only the interfaces relevant to their use case, thus adhering to ISP.

2.1.5 Dependency Inversion Principle (DIP):

The high-level modules in CAMs, such as *StaffEnquiryManager.java* and *StudentEnquiryManager.java*, depend on the *EnquiryManagerInterface* rather than on concrete implementations, aligning with the DIP. Moreover, the high-level controller classes such as *CampCommController* and *StaffController*, along with their respective *Switch* classes, demonstrate DIP by orchestrating the application's flow and interacting with lower-level modules through abstract interfaces or base classes, thereby promoting modular design.

3. Demonstration of Good OOP Practices

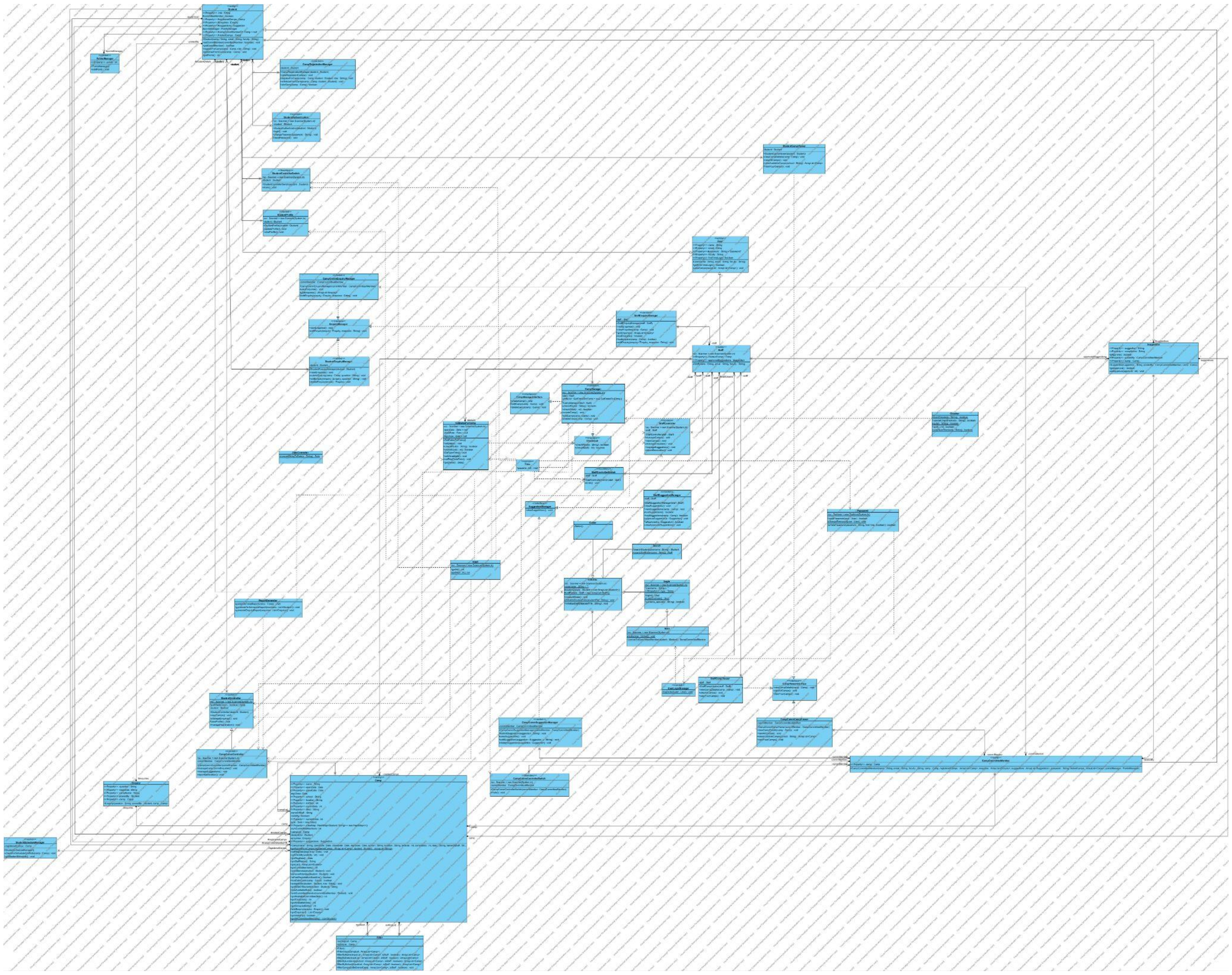
3.1 Loose Coupling and High Cohesion:

Loose coupling is evident in CAMs through the use of interfaces such as *CampManagerInterface* and *EnquiryManager*. Classes like *CampManager* and *StaffEnquiryManager* implement these interfaces, which means the system's components interact with each other through abstractions rather than concrete implementations. This reduces the dependencies between classes, allowing changes to be effectively made in one part of the system without significantly affecting others. *High cohesion* within CAMs is achieved by ensuring that classes like *CampRegistrationManager* and *ReportGenerator* are tightly focused on their responsibilities. CampRegistrationManager manages all aspects of registration, while ReportGenerator is solely dedicated to generating various reports. This focus makes the classes more manageable and maintainable.

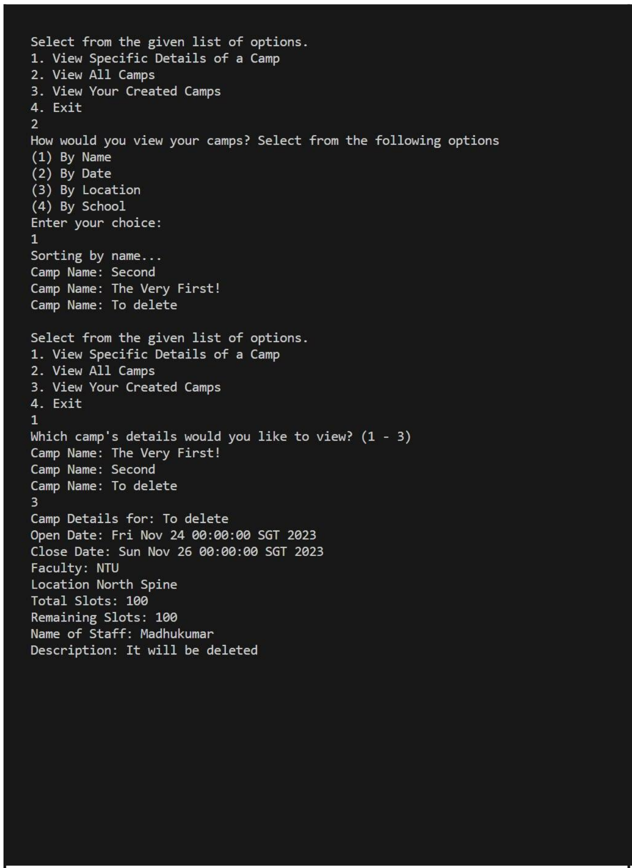
3.2 Reusability of Code:

The generalization relationships in CAMs contribute to the reusability of code. For instance, the *Student* and *Staff* classes extend the *User* class, which means they inherit common attributes and behaviors from User, reducing code duplication. Furthermore, interfaces like *CampViewer* and *SuggestionManager* allows different classes to implement the same methods in various ways, promoting reusability across the system.

4. Detailed UML Class Diagram

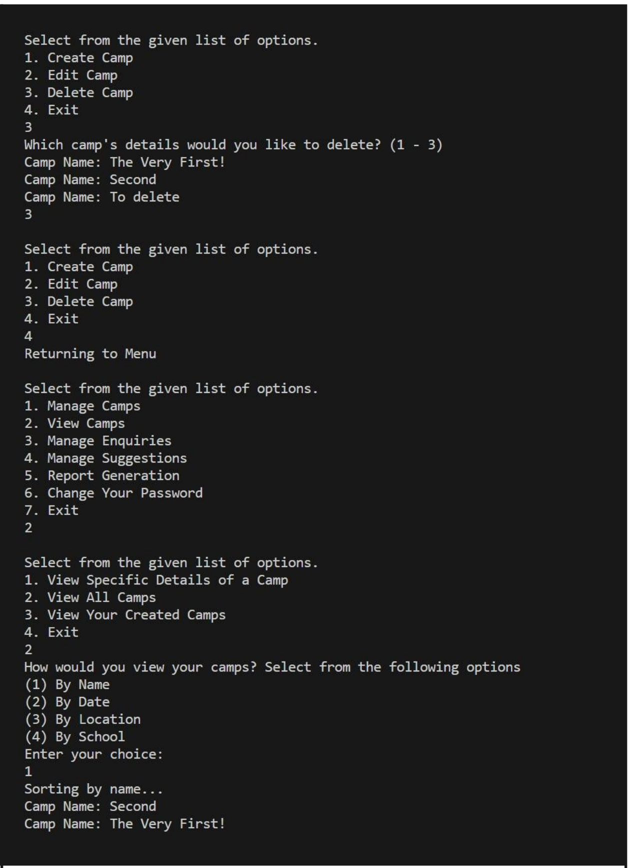


5. Testing



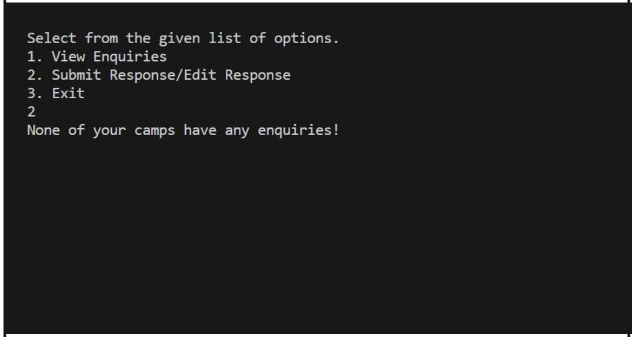
```
Select from the given list of options.
1. View Specific Details of a Camp
2. View All Camps
3. View Your Created Camps
4. Exit
2
How would you view your camps? Select from the following options
(1) By Name
(2) By Date
(3) By Location
(4) By School
Enter your choice:
1
Sorting by name...
Camp Name: Second
Camp Name: The Very First!
Camp Name: To delete

Select from the given list of options.
1. View Specific Details of a Camp
2. View All Camps
3. View Your Created Camps
4. Exit
1
Which camp's details would you like to view? (1 - 3)
Camp Name: The Very First!
Camp Name: Second
Camp Name: To delete
3
Camp Details for: To delete
Open Date: Fri Nov 24 00:00:00 SGT 2023
Close Date: Sun Nov 26 00:00:00 SGT 2023
Faculty: NTU
Location North Spine
Total Slots: 100
Remaining Slots: 100
Name of Staff: Madhukumar
Description: It will be deleted
```

Viewing specific details of a certain camp chosen by the user.



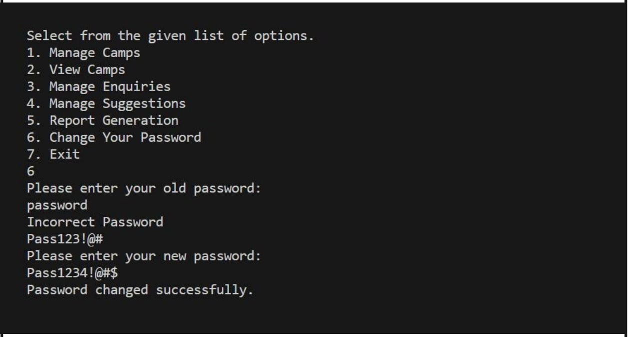
```
Select from the given list of options.
1. Create Camp
2. Edit Camp
3. Delete Camp
4. Exit
3
Which camp's details would you like to delete? (1 - 3)
Camp Name: The Very First!
Camp Name: Second
Camp Name: To delete
3

Select from the given list of options.
1. Create Camp
2. Edit Camp
3. Delete Camp
4. Exit
4
Returning to Menu

Select from the given list of options.
1. Manage Camps
2. View Camps
3. Manage Enquiries
4. Manage Suggestions
5. Report Generation
6. Change Your Password
7. Exit
2

Select from the given list of options.
1. View Specific Details of a Camp
2. View All Camps
3. View Your Created Camps
4. Exit
2
How would you view your camps? Select from the following options
(1) By Name
(2) By Date
(3) By Location
(4) By School
Enter your choice:
1
Sorting by name...
Camp Name: Second
Camp Name: The Very First!
```

Allowing the user to filter by a variety of different characteristics (Name, Date, Location, School).



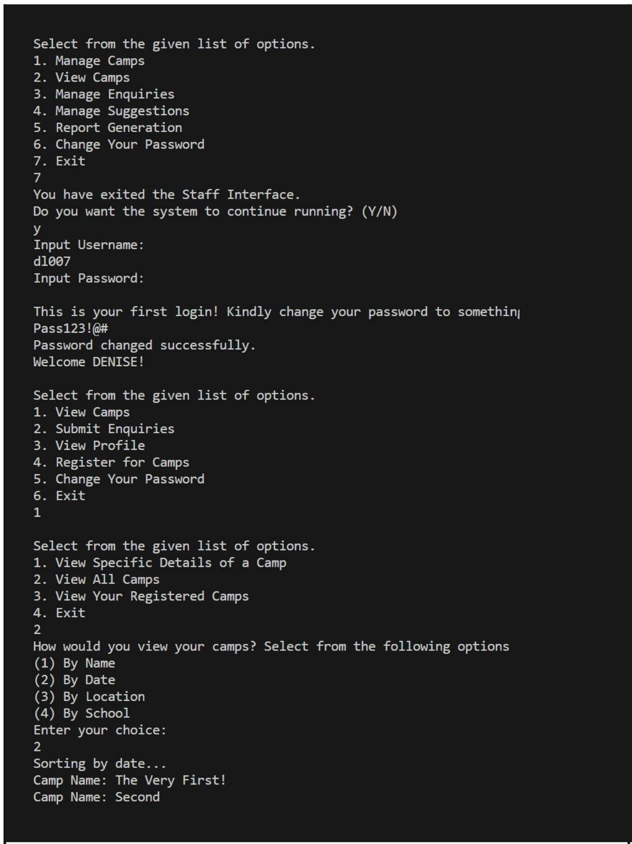
```
Select from the given list of options.
1. View Enquiries
2. Submit Response/Edit Response
3. Exit
2
None of your camps have any enquiries!
```

None of the staff's camps have any enquiries when they try to view enquiries.



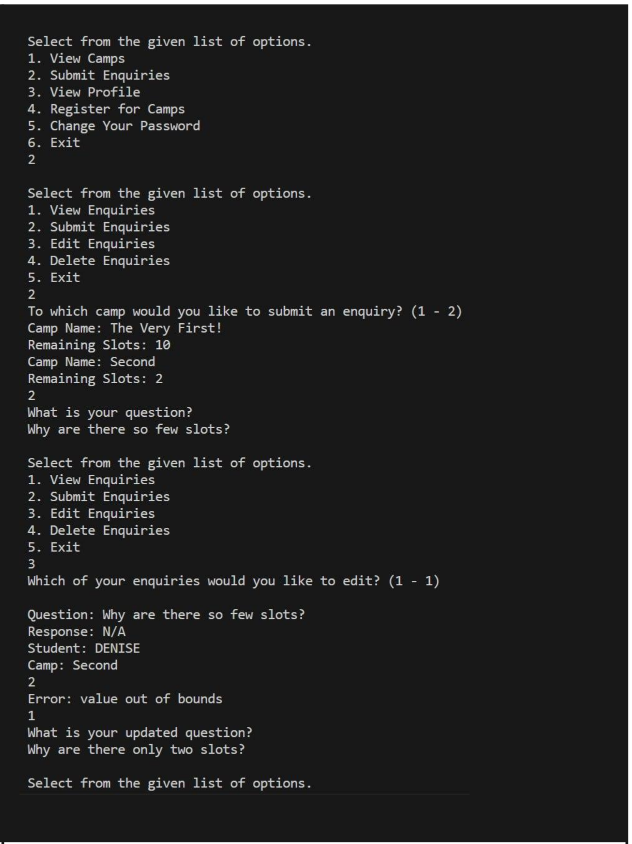
```
Select from the given list of options.
1. Manage Camps
2. View Camps
3. Manage Enquiries
4. Manage Suggestions
5. Report Generation
6. Change Your Password
7. Exit
6
Please enter your old password:
password
Incorrect Password
Pass123!@#
Please enter your new password:
Pass1234!@#$
Password changed successfully.
```

The password is incorrect and then, when changing, satisfy the given conditions.



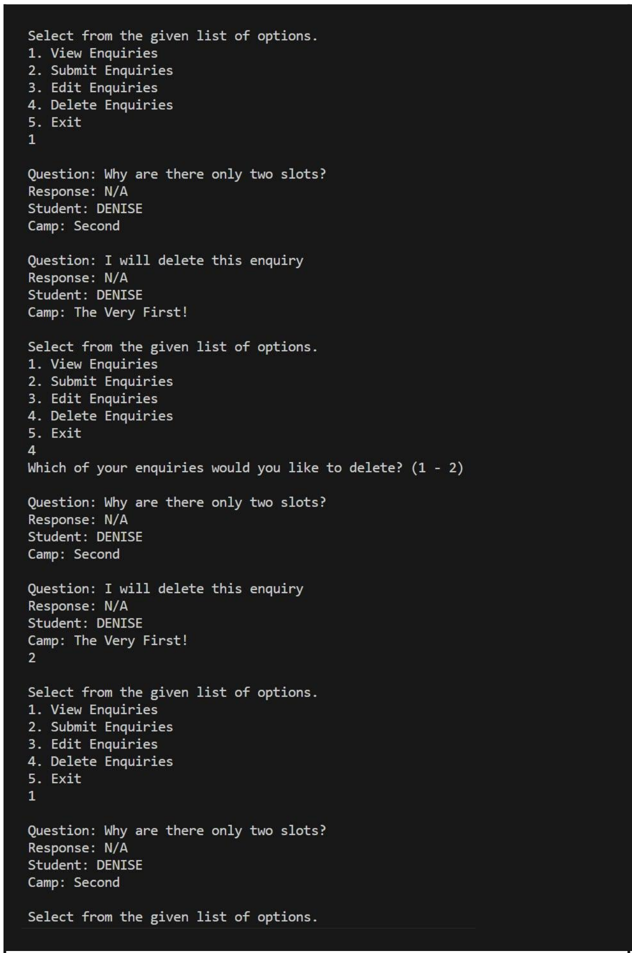
```
Select from the given list of options.
1. Manage Camps
2. View Camps
3. Manage Enquiries
4. Manage Suggestions
5. Report Generation
6. Change Your Password
7. Exit
7
You have exited the Staff Interface.
Do you want the system to continue running? (Y/N)
y
Input Username:
dl007
Input Password:

This is your first login! Kindly change your password to somethin
Pass123!@#
Password changed successfully.
Welcome DENISE!

Select from the given list of options.
1. View Camps
2. Submit Enquiries
3. View Profile
4. Register for Camps
5. Change Your Password
6. Exit
1

Select from the given list of options.
1. View Specific Details of a Camp
2. View All Camps
3. View Your Registered Camps
4. Exit
2
How would you view your camps? Select from the following options
(1) By Name
(2) By Date
(3) By Location
(4) By School
Enter your choice:
2
Sorting by date...
Camp Name: The Very First!
Camp Name: Second
```

Logout and re-login. User can view camps by whichever filter they choose.



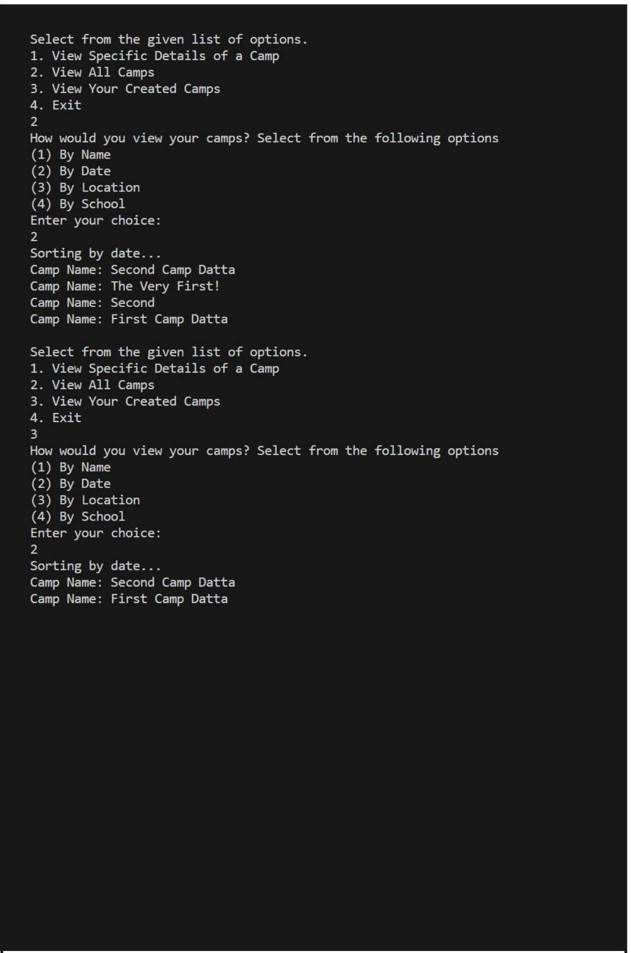
```
Select from the given list of options.
1. View Camps
2. Submit Enquiries
3. View Profile
4. Register for Camps
5. Change Your Password
6. Exit
2

Select from the given list of options.
1. View Enquiries
2. Submit Enquiries
3. Edit Enquiries
4. Delete Enquiries
5. Exit
2
To which camp would you like to submit an enquiry? (1 - 2)
Camp Name: The Very First!
Remaining Slots: 10
Camp Name: Second
Remaining Slots: 2
2
What is your question?
Why are there so few slots?

Select from the given list of options.
1. View Enquiries
2. Submit Enquiries
3. Edit Enquiries
4. Delete Enquiries
5. Exit
3
Which of your enquiries would you like to edit? (1 - 1)

Question: Why are there so few slots?
Response: N/A
Student: DENISE
Camp: Second
2
Error: value out of bounds
1
What is your updated question?
Why are there only two slots?

Select from the given list of options.
```

None of the staff's camps have any enquiries when they try to view enquiries.



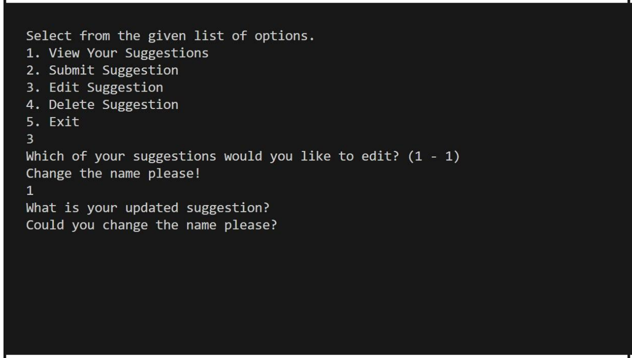
```
Select from the given list of options.
1. View Enquiries
2. Submit Enquiries
3. Edit Enquiries
4. Delete Enquiries
5. Exit
1

Question: Why are there only two slots?
Response: N/A
Student: DENISE
Camp: Second

Question: I will delete this enquiry
Response: N/A
Student: DENISE
Camp: The Very First!

Select from the given list of options.
1. View Enquiries
2. Submit Enquiries
3. Edit Enquiries
4. Delete Enquiries
5. Exit
4
Which of your enquiries would you like to delete? (1 - 2)

Question: Why are there only two slots?
Response: N/A
Student: DENISE
Camp: Second

Question: I will delete this enquiry
Response: N/A
Student: DENISE
Camp: The Very First!
2

Select from the given list of options.
1. View Enquiries
2. Submit Enquiries
3. Edit Enquiries
4. Delete Enquiries
5. Exit
1

Question: Why are there only two slots?
Response: N/A
Student: DENISE
Camp: Second

Select from the given list of options.
```

The user is able to delete enquiries that exist within their list of enquiries.



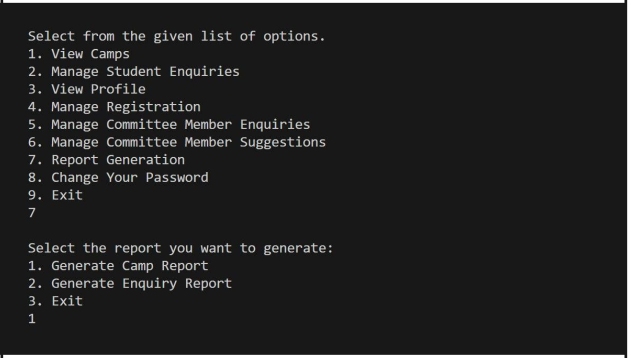
```
Select from the given list of options.
1. View Specific Details of a Camp
2. View All Camps
3. View Your Created Camps
4. Exit
2
How would you view your camps? Select from the following options
(1) By Name
(2) By Date
(3) By Location
(4) By School
Enter your choice:
2
Sorting by date...
Camp Name: Second Camp Datta
Camp Name: The Very First!
Camp Name: Second
Camp Name: First Camp Datta

Select from the given list of options.
1. View Specific Details of a Camp
2. View All Camps
3. View Your Created Camps
4. Exit
3
How would you view your camps? Select from the following options
(1) By Name
(2) By Date
(3) By Location
(4) By School
Enter your choice:
2
Sorting by date...
Camp Name: Second Camp Datta
Camp Name: First Camp Datta
```

The user is able to view their created camps, using the given filters.



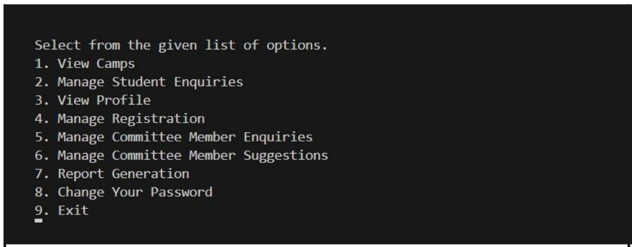
```
Select from the given list of options.
1. View Your Suggestions
2. Submit Suggestion
3. Edit Suggestion
4. Delete Suggestion
5. Exit
3
Which of your suggestions would you like to edit? (1 - 1)
Change the name please!
1
What is your updated suggestion?
Could you change the name please?
```

The user is able to update their suggestions.



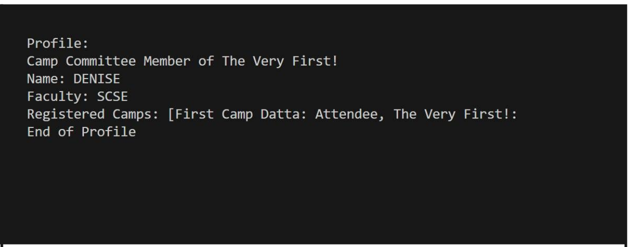
```
Select from the given list of options.
1. View Camps
2. Manage Student Enquiries
3. View Profile
4. Manage Registration
5. Manage Committee Member Enquiries
6. Manage Committee Member Suggestions
7. Report Generation
8. Change Your Password
9. Exit
7

Select the report you want to generate:
1. Generate Camp Report
2. Generate Enquiry Report
3. Exit
1
```

The user is able to generate a camp report.



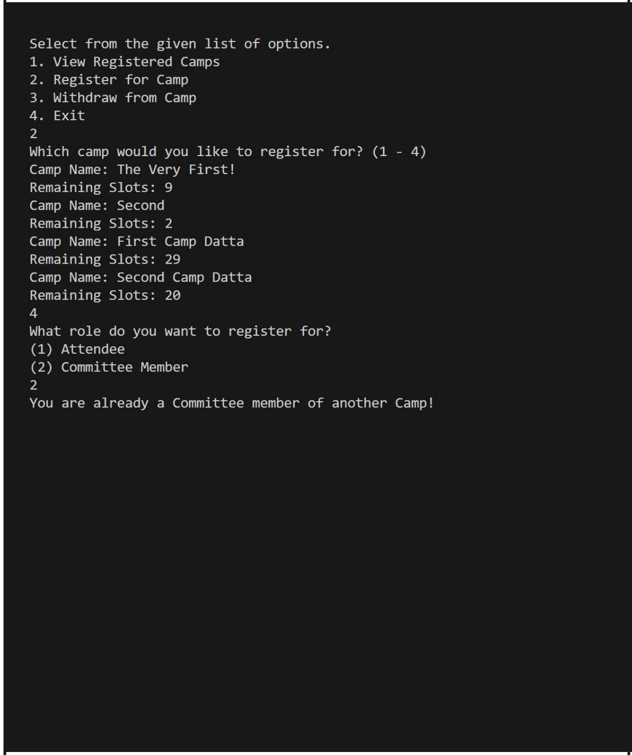
```
Select from the given list of options.
1. View Camps
2. Manage Student Enquiries
3. View Profile
4. Manage Registration
5. Manage Committee Member Enquiries
6. Manage Committee Member Suggestions
7. Report Generation
8. Change Your Password
9. Exit
```

This simply displays the list of options for camp committee members.



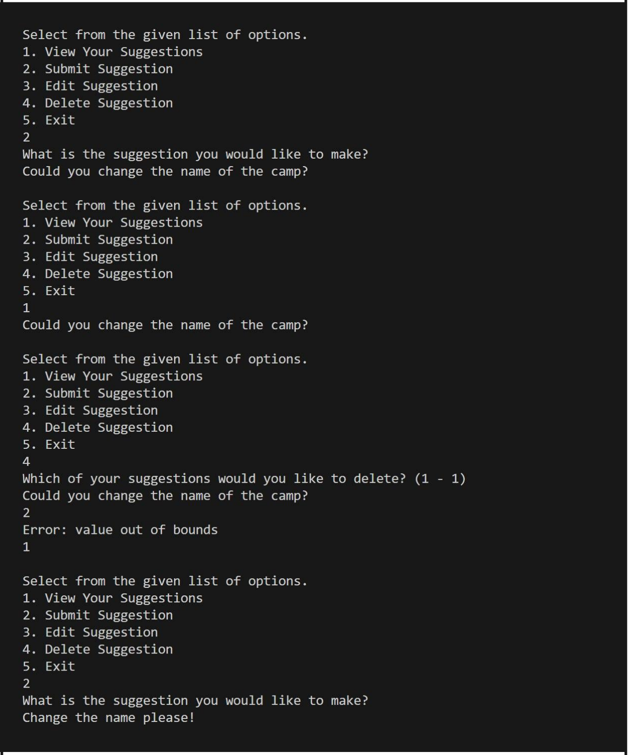
```
Profile:
Camp Committee Member of The Very First!
Name: DENISE
Faculty: SCSE
Registered Camps: [First Camp Datta: Attendee, The Very First!:
End of Profile
```

This displays the profile for the user.



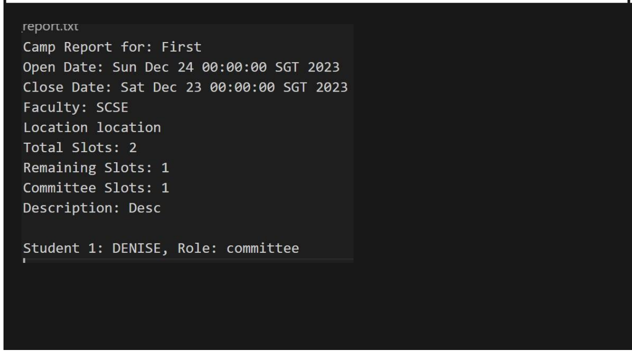
```
Select from the given list of options.
1. View Registered Camps
2. Register for Camp
3. Withdraw from Camp
4. Exit
2
Which camp would you like to register for? (1 - 4)
Camp Name: The Very First!
Remaining Slots: 9
Camp Name: Second
Remaining Slots: 2
Camp Name: First Camp Datta
Remaining Slots: 29
Camp Name: Second Camp Datta
Remaining Slots: 20
4
What role do you want to register for?
(1) Attendee
(2) Committee Member
2
You are already a Committee member of another Camp!
```

You cannot be a camp committee member of more than one camp.

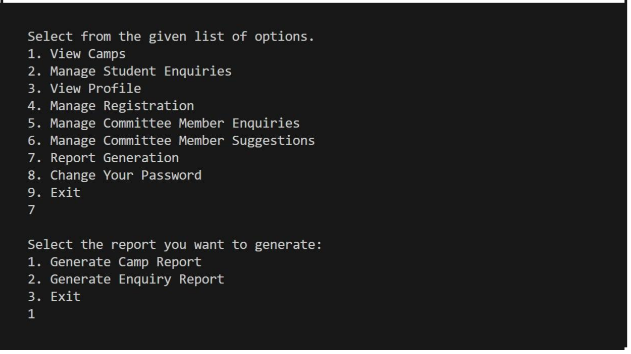


```
Select from the given list of options.
1. View Your Suggestions
2. Submit Suggestion
3. Edit Suggestion
4. Delete Suggestion
5. Exit
2
What is the suggestion you would like to make?
Could you change the name of the camp?

Select from the given list of options.
1. View Your Suggestions
2. Submit Suggestion
3. Edit Suggestion
4. Delete Suggestion
5. Exit
1
Could you change the name of the camp?

Select from the given list of options.
1. View Your Suggestions
2. Submit Suggestion
3. Edit Suggestion
4. Delete Suggestion
5. Exit
4
Which of your suggestions would you like to delete? (1 - 1)
Could you change the name of the camp?
2
Error: value out of bounds
1

Select from the given list of options.
1. View Your Suggestions
2. Submit Suggestion
3. Edit Suggestion
4. Delete Suggestion
5. Exit
2
What is the suggestion you would like to make?
Change the name please!
```

This displays error handling for the suggestions function for camp committee.



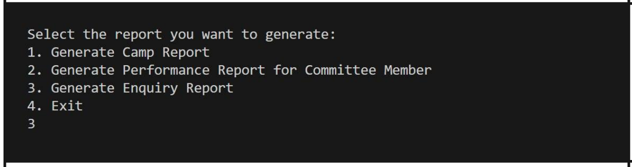
```
report.txt
Camp Report for: First
Open Date: Sun Dec 24 00:00:00 SGT 2023
Close Date: Sat Dec 23 00:00:00 SGT 2023
Faculty: SCSE
Location location
Total Slots: 2
Remaining Slots: 1
Committee Slots: 1
Description: Desc

Student 1: DENISE, Role: committee
```



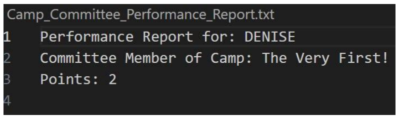
```
Select from the given list of options.
1. View Camps
2. Manage Student Enquiries
3. View Profile
4. Manage Registration
5. Manage Committee Member Enquiries
6. Manage Committee Member Suggestions
7. Report Generation
8. Change Your Password
9. Exit
7

Select the report you want to generate:
1. Generate Camp Report
2. Generate Enquiry Report
3. Exit
1
```

This displays the camp report generated by the camp committee member, listing all the students (in this case, only one).

This generates the camp performance report as shown below.



```
Select the report you want to generate:
1. Generate Camp Report
2. Generate Performance Report for Committee Member
3. Generate Enquiry Report
4. Exit
3
```

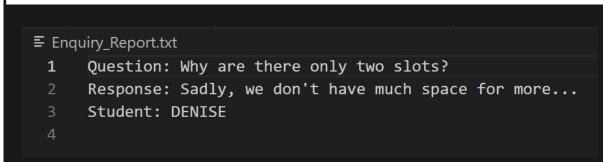


Camp_Committee_Performance_Report.txt

```
Performance Report for: DENISE
Committee Member of Camp: The Very First!
Points: 2
```

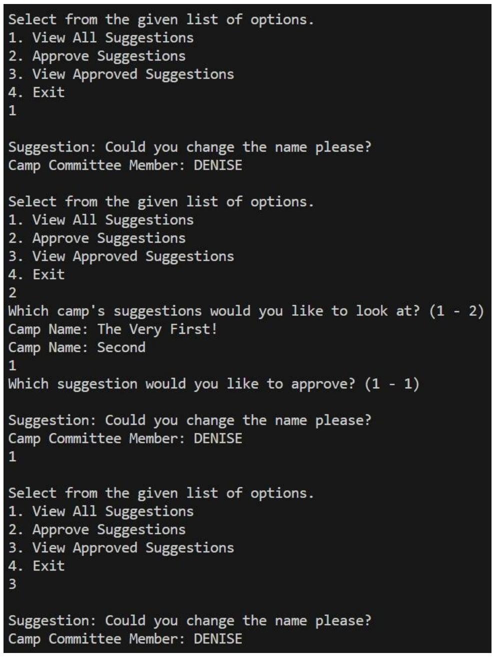
This displays the generation of the enquiry report.

This displays the performance report for the camp committee members.



Enquiry_Report.txt

```
Question: Why are there only two slots?
Response: Sadly, we don't have much space for more...
Student: DENISE
```



```
Select from the given list of options.
1. View All Suggestions
2. Approve Suggestions
3. View Approved Suggestions
4. Exit
1

Suggestion: Could you change the name please?
Camp Committee Member: DENISE

Select from the given list of options.
1. View All Suggestions
2. Approve Suggestions
3. View Approved Suggestions
4. Exit
2
Which camp's suggestions would you like to look at? (1 - 2)
Camp Name: The Very First!
Camp Name: Second
1
Which suggestion would you like to approve? (1 - 1)

Suggestion: Could you change the name please?
Camp Committee Member: DENISE
1

Select from the given list of options.
1. View All Suggestions
2. Approve Suggestions
3. View Approved Suggestions
4. Exit
3

Suggestion: Could you change the name please?
Camp Committee Member: DENISE
```

This shows a possible enquiry report generated by the user.

This displays how the staff member interacts with suggestions within the system.

6. Reflection

The Camp Application and Management System (CAMs) project was a collaborative effort aimed at developing a comprehensive system to manage camp activities within an educational setting. The project presented a multifaceted learning opportunity, not only in software development but also in teamwork and project management. Our objective was to deepen our understanding of SOLID principles within a practical setting, with the aim of developing a robust, user-friendly system. Throughout the project, these goals were our compass, guiding our design decisions and collaboration strategies.

We encountered a few challenges, such as differing opinions on design patterns and difficulties in integrating the modules. However, through open communication and our regular meetings, we were able to align our visions and integrate our separate contributions into a cohesive system. Our first major setback occurred when we planned on coding the system without the base of SOLID principle rules. What we soon came to realize was that the code was extremely rigid and hard to make changes to, especially when there were a few of us working on it simultaneously.

This setback opened our eyes to the lack of communication and progress we had within the group, which led to us adopting Scrum: an incremental output delivery approach through constant feedback and collaborative decision-making. We structured our project according to our weekly meetups, ensuring consistent communication and alignment of goals. Each week, before our designated meetings, we were assigned a task to complete. Scrum's iterative process allowed us to incorporate feedback rapidly and adjust our trajectory effectively. At the end of each meeting, we would discuss our progress and any roadblocks we encountered, which proved invaluable for continuous improvement. Moreover, not only did this enhance individual accountability but it also strengthened our collaboration, as everyone had a clear understanding of their responsibilities and the collective goal. In conclusion, this project was far more than just an academic exercise as it presented us with a real-world understanding of the software development industry, which will be undoubtedly valuable for our future endeavors.