Individual Programming Hand-in Assignment

Hockey team management software

Date Finished- 3/13/24

Kushmi Anuththara

Table of Contents

01-Introduction

In this Python assignment, a software application for managing a "Hockey team" was developed. The primary objective was to create a system facilitating information management for the team. The report outlines the technical aspects employed and provides user guidance for the Hockey team management system.

The system encompasses a designed interface with the specified features from the assignment brief. Additionally, details regarding the required user information are included. The process begins with the user entering the team's name, followed by specifying the team type based on gender. Subsequently, the payment status is recorded, and finally, the payment for participating in the hockey competition is entered in Swedish Krona (SEK).

1.1 Main Features added in the System.

The system introduces a comprehensive set of features to manage event teams efficiently. Users can create, view, update, and delete teams, with options to display all teams, filter by gender, and access analytics. Additionally, features include cancelling team participation, saving and loading records from a file, and the ability to exit the system. These functionalities collectively ensure a seamless and organized team management experience.

1. Create New Team
2. Display Existing Team Information By Team ID
3. Update Existing Team By Team No
4. Delete Existing Team By Team No
5. Display All Teams
6. Display Girls Teams Only
7. Display Boys Teams Only
8. Display Analytics
9. Cancel Team Participation For the Event
10. Save Entered Records to the File

11. Load from the File

12.Exit/Quit

1.2 Tools that used to the development.

Mainly used the PyCharm but checked the code also using Visual Studio Editor.

1.3 Why PyCharm?

PyCharm provides substantial benefits for the development of the Hockey team management system in Python. Its intelligent coding assistance enhances efficiency by suggesting and completing code, reducing errors. The integrated debugger and testing tools ensure seamless debugging and reliable code quality. PyCharm's version control integration facilitates collaborative development, enabling efficient team coordination. The comprehensive set of built-in tools streamlines the development process, offering code analysis, project navigation, and a rich ecosystem of plugins.

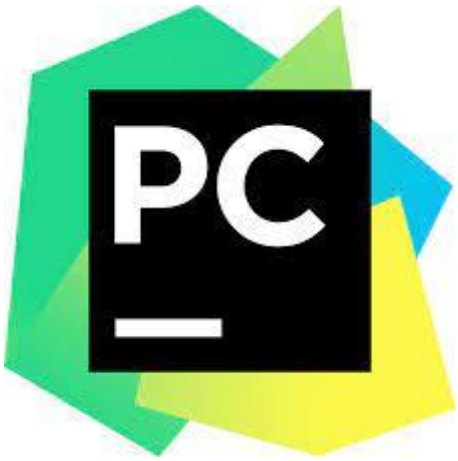

Figure 1- PyCharm Logo.

02-Description (The solution with classes defined and used).

The Hockey team management system employs several crucial classes, each serving a distinct purpose and shouldering specific responsibilities. The "Team" class lies at the system's heart, responsible for managing individual team instances. It encapsulates attributes like team ID, name, type, and fee-related details. Collaborating with the "InputValidatorService" class, it ensures the validation of user inputs during team creation, promoting data integrity. The "HockeyCupService" class acts as the overarching service orchestrator, overseeing the entire system's functionality. It collaborates with the "Team" class for tasks like team creation, modification, and analytics, fostering efficient communication and seamless coordination between different system components. This modular and organized class structure enhances the system's flexibility and maintainability.

- Making it easy for users to interact with the system through a clear menu.
- Storing data in memory for quick access and also saving it in text files for future use.
- Organizing the system in a neat, modular way using object-oriented design.
- Checking user inputs to make sure they follow the rules we set.
- Planning for the future by building a system that can easily grow.
- Handling errors and system messages the way everyone expects.
- Writing code that's easy to understand, reuse, and follow good practices.
- Making sure the code has helpful comments and is logically structured.

2.1 Used Python Libraries.

The code utilizes two Python libraries, "os" and "datetime," to enhance its functionality. The "os" library facilitates interaction with the operating system, providing methods for file operations and directory management. Specifically, it enables checking file existence, ensuring the data source file is available before attempting to read or write. On the other hand, the "datetime" library aids in handling date and time-related operations. In this context, it helps capture the current date for team creation and allows formatting for consistent storage in the data source file. These libraries enhance the overall robustness and precision of the Hockey Team Management System by incorporating essential operating system interactions and efficient date handling.

The "sys" library provides access to some variables used or maintained by the Python interpreter and functions that interact with the interpreter, supporting command-line arguments. In this system, it ensures smooth program execution by allowing the handling of command-line inputs effectively.

Additionally, the "re" library, short for regular expressions, aids in string manipulation and pattern matching. It specifically assists in validating team names, ensuring they adhere to specified patterns. By integrating these libraries, the Hockey Team Management System achieves enhanced command-line interaction and robust string validation, contributing to its overall reliability and user-friendly design.

2.2 User Input Validations.

The system employs several input validation methods to ensure accurate and meaningful user inputs:

- **Team ID Validation:** Ensures the entered Team ID is a valid integer.
- **Team Name Validation:** Verifies that the team name follows a specified pattern, allowing alphanumeric characters, spaces, and hyphens.
- **Team Type Validation:** Checks that the team type input is either "G" for Girls or "B" for Boys.
- **Participation Fee Validation:** Validates the participation fee input as a positive numerical value.
- **Fee Status Validation:** Verifies that the fee status is either "Y" for Yes or "N" for No.
- **Cancellation Date Validation:** Ensures the cancellation date follows the "YYYY-MM-DD" format.
- **User Confirmation Validation:** Confirms user choices, accepting either "Y" for Yes or "N" for No.

These validations collectively guarantee that the user inputs align with predefined rules, maintaining data integrity and the system's functionality.

2.3 Exception Handling.

1. **ValueError Handling:** Used this to catch ValueErrors, like when users enter invalid data types or negative participation fees.

2. **FileNotFoundError Handling:** This helps manage situations where the system can't find the data source file during loading operations.
3. **General Exception Handling:** I've included measures to capture unforeseen errors during file operations, making the program stable and user-friendly.
4. **Try-Except Blocks:** Throughout critical parts of the code, we use these blocks to gracefully handle errors, preventing the program from abruptly stopping.

2.4 System Design for Hockey Teams.

The system design for this project is all about making things easy for users. I have a simple menu system that's easy to navigate. The data is stored in a way that makes it quick to access, and we also keep a backup in text files just in case. Each part of the system has its own job, keeping everything organized. I make sure that the data entered by users follows the rules we set, so it stays reliable. The design allows for future improvements and handles errors in a way that won't confuse users. The code is well-organized, making it easy to understand and maintain.

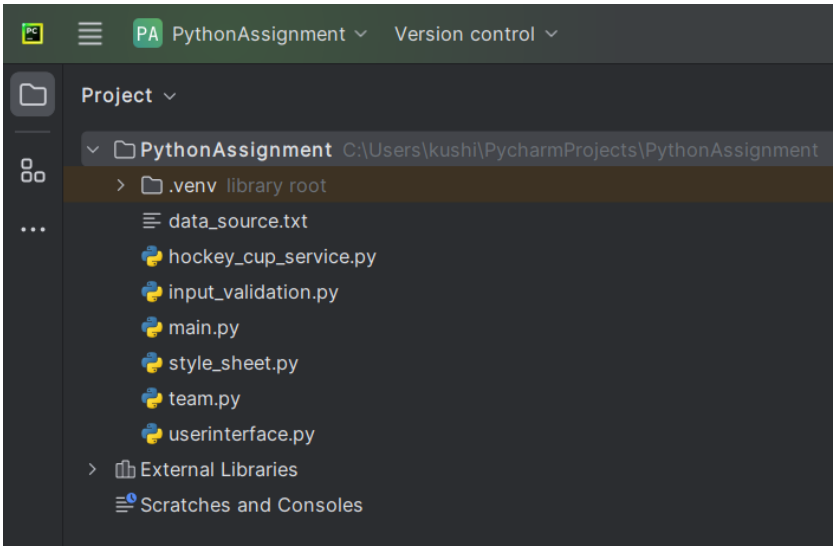

Figure 2- System design for the Hockey team.

2.4.1 User Interface Class.

The UserInterface class is like the control center for the Hockey Cup Team Management System. It helps users do different things with their teams, like creating new ones, updating info, or even canceling participation. The menu is straightforward, giving clear options from creating teams to saving data. It makes the whole process easy and user-friendly.

User Interface Class Description:

The UserInterface class acts as the bridge between users and the Hockey Cup Team Management System. It facilitates user interaction by presenting a menu system with various options. The class utilizes the functionalities provided by the HockeyCupService, InputValidatorService, and StyleSheet classes. It ensures ease of use for individuals unfamiliar with the program, offering options to create, display, update, delete, and manage teams efficiently. The menu options cover a range of functionalities, from viewing analytics to saving and loading records from files. The UserInterface class plays a crucial role in providing a seamless experience for users navigating the hockey team management system.

------Menu Options-----

1. Create New Team
2. Display Existing Team Information By Team ID
3. Update Existing Team By Team No
4. Delete Existing Team By Team No
5. Display All Teams
6. Display Girls Teams Only
7. Display Boys Teams Only
8. Display Analytics
9. Cancel Team Participation For the Event
10. Save Entered Records to the File
11. Load from the File
12. Exit/Quit

2.4.2 Team Class.

To initialize the attributes here I have used “Init” constructor and this has designed for create new objects with in team class.

The Team class has several private attributes:

1. `__id`: Represents the unique identifier for each team.

2. `__date`: Represents the date the team is created or modified.

3. `__name`: Represents the name of the team.

4. `__type`: Represents the type of the team (e.g., Girls-G, Boys-B).

5. `__fee`: Represents the participation fee for the team in SEK.

6. `__fee_paid`: Represents whether the participation fee has been paid (True or False).

7. `__cancellation_date`: Represents the date on which the team canceled their participation.

List of methods:

methods associated with attribute access in the Team class:

1. **Getter Methods (@property):**
    - **id()**: Retrieves the Team ID.
    - **date()**: Retrieves the Team creation date.
    - **name()**: Retrieves the Team name.
    - **type()**: Retrieves the Team type.
    - **fee()**: Retrieves the Team participation fee.
    - **fee_paid()**: Retrieves the Team fee payment status.
    - **cancellation_date()**: Retrieves the Team cancellation date.
2. **Setter Methods:**
    - **set_date(value)**: Sets the Team creation date.
    - **set_name(value)**: Sets the Team name.
    - **set_type(value)**: Sets the Team type.
    - **set_fee(value)**: Sets the Team participation fee.

- **set_fee_paid(value)**: Sets the Team fee payment status.
- **set_cancellation_date(value, date)**: Sets the Team cancellation date.

3. **Static Methods:**
    - **set_unique_id()**: Generates a unique ID for a Team.
    - **set_datetime_now()**: Generates the current date and time.
4. **Special Method:**
    - **__str__()**: Provides a string representation of the Team instance.

2.4.3 Service Class.

In the hockey_cup_service.py script has 2 service classes.

1. **HockeyCupService:**
    - Manages the main functionality related to the Hockey Cup Team Management System.
2. **InputValidatorService:**
    - Provides input validation services for various user inputs in the program.
    - Ensures that the entered data meets specified criteria, preventing invalid inputs.

Methods:

1. **__init__(self):**
    - Initializes the **HockeyCupService** class with an empty **team_array**.
2. **get_last_id_from_historical_data(self):**
    - Retrieves the last used team ID from historical data and updates the Team class counter accordingly.
3. **check_historical_data_availability(self):**
    - Checks if historical data is available and prompts the user to load it.

4. **create_team(self, name, team_type, fee, fee_paid):**
    - Creates a new team with the provided details and adds it to the **team_array**.
5. **read_team(self, id, option=0):**
    - Reads and displays information about a specific team based on the provided team ID.
6. **update_team(self, id, new_value, field, option=0):**
    - Updates specific details of a team (e.g., name, type, fee, fee_paid) based on the provided parameters.
7. **delete_team(self, id):**
    - Deletes a team from the **team_array** based on the provided team ID.
8. **list_teams_info(self):**
    - Lists information about all teams present in the **team_array**.
9. **list_girls_teams_Info(self):**
    - Lists information about girls' teams from the **team_array**.
10. **list_boys_teams_Info(self):**
    - Lists information about boys' teams from the **team_array**.
11. **show_team_analytics(self):**
    - Displays analytics related to the total number of teams, paid teams, canceled teams, and the percentage of paid teams.
12. **cancel_team_participation(self, id, date):**
    - Cancels a team's participation by setting the cancellation date.
13. **save_to_file(self, option=0):**
    - Saves team data to a text file (data_source.txt).
14. **load_from_file(self, option=0):**

- Loads team data from a text file (data_source.txt).

2.4.4 User Input Validation Class.

1. To manage the user inputs I created a separate class module. (input_validation.py)

   validate_team_id_input():

   - Validates user input for Team ID.
   - Returns: Validated Team ID.

2. **validate_team_name_input(str=''):**
   - Validates user input for Team Name using regular expressions.
   - Returns: Validated Team Name.
3. **validate_team_type_input(str=''):**
   - Validates user input for Team Type (Girls-G, Boys-B).
   - Returns: Validated Team Type.
4. **validate_participation_fee_input(str=''):**
   - Validates user input for Participation Fee (SEK) as a positive float.
   - Returns: Validated Participation Fee.
5. **validate_fee_status_input(str=''):**
   - Validates user input for Fee Status (Yes-Y, No-N).
   - Returns: True if fee is paid (Y), False otherwise.
6. **validate_cancelation_date_input(str=''):**
   - Validates user input for Cancellation Date (YYYY-MM-DD) using regular expressions.
   - Returns: Validated Cancellation Date.
7. **validate_user_confirmation_input(str=''):**

- Validates user input for user confirmation (Y-Yes, N-No).
- Returns: 'Y' for Yes, 'N' for No.

2.4.5 Separate Sheet for Style sheet module.

```
def print_menu_title():
    print(f"\n{'-' * 34}  Main Menu {'-' * 34}\n")
```

2.4.6 Main Module.

This has been created to initiate the system and show what are the menu options that added in to system.

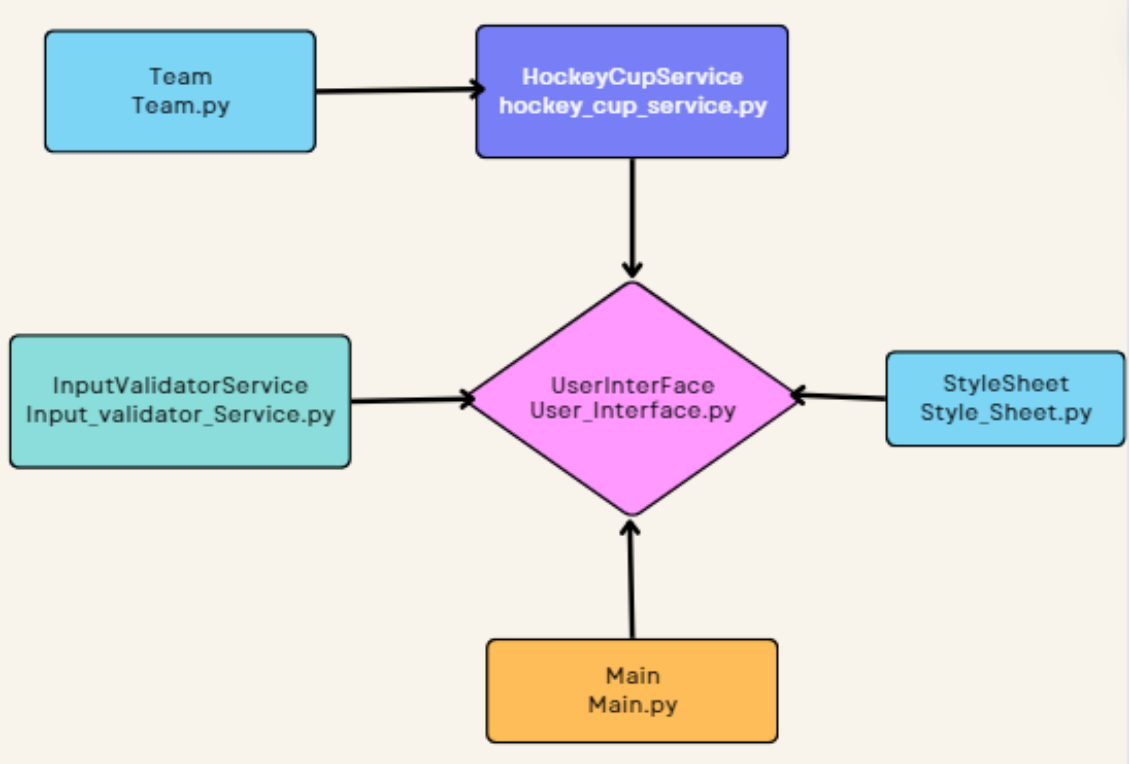

Figure 3- Main Module diagram.

03-User Guide. (How to Run the program?)

3.1 Software Installation and Compiling.

- The Hockey system requires Python version 3.12.
- You can install the Python 3.11 version from > https://www.python.org/downloads/
- In the terminal you need to navigate to the code file directory and use the command “Python main.py” to run the program.

3.2 User Instructions.

3.2.1 Start the program.

- Upon starting the program loads and checks for a data text file.
- Then it will reads the historical data, loads it into memory and sets the global variable ‘_id_counter’ for generating unique team IDs.

3.2.2 The Main Menu.

- After loading the main menu You will see the menu or it will prompt users to choose tasks numbered from 1 to 12.
- Invalid inputs generate error messages.

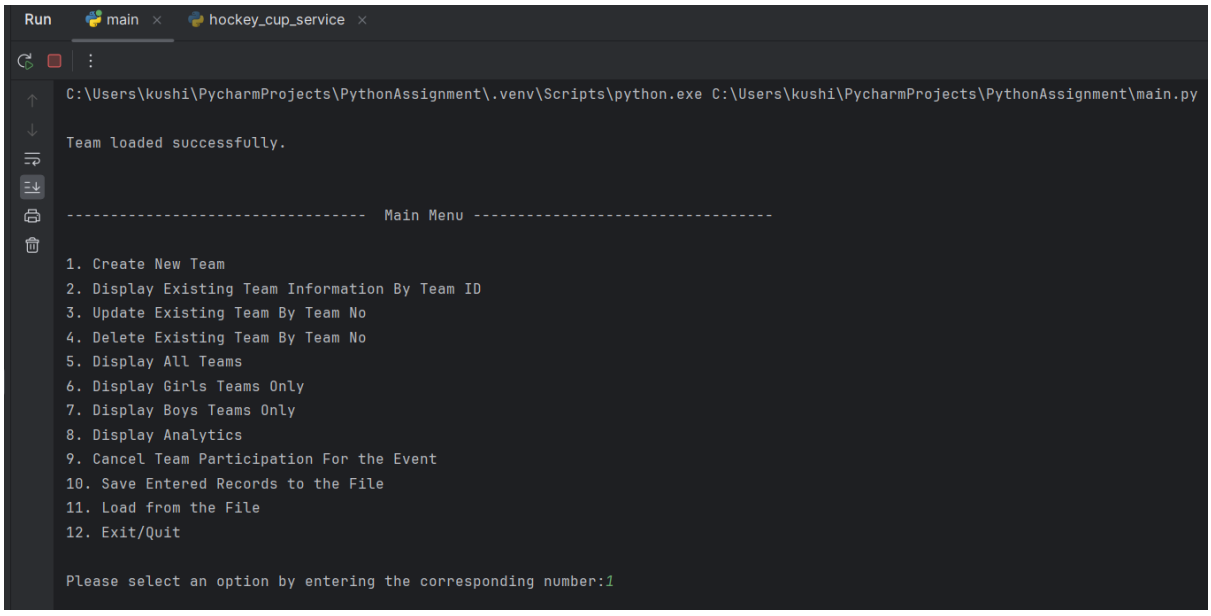

Figure 4- Main Manu result after Executing the code.

3.2.2.1 Add Team.

- Select Option 1 to enter the team name and other team information in to the system.

- Other information as in ensuring valid characters for name, correct team type input, fee status, and a positive fee amount.
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:1
1. Create New Team
Enter the  team name : Kushmi Test team
Enter the  team type (e.g., Girls-G, Boys-B) : G
Enter the  participation fee (SEK): 500
Enter the  fee paid status (e.g., Yes-Y, No-N): y
Invalid input. Please enter a valid fee status (Y, N).
Enter the  fee paid status (e.g., Yes-Y, No-N): Y

Team has created successfully with the Team Id: 3

Do you want to return to the Main Menu? Then kindly press the Enter key....
```

Figure 5-Add Teams.

3.2.2.2 Search by ID:

- Option 2 directs users to the "Search by ID" menu.
- Input a valid positive integer ID to view team information.
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:2
2. Show Existing Team Information By Team ID

Enter the Team ID: 3
Team ID: 3
Team Name: Kushmi Test team
Team Type: Girls
Fee (SEK): 500.0
Fee Paid: Yes

To return to the Main Menu, kindly press the Enter key..
```

Figure 6- Search by ID results.

3.2.2.3 Update Team:

- Option 3 leads to the "Modify Teams Menu."
- Input the team ID to edit, then update the name, type, fee status, and amount.

- Confirm changes to see updated information.
- Press "Enter" to return to the main menu.

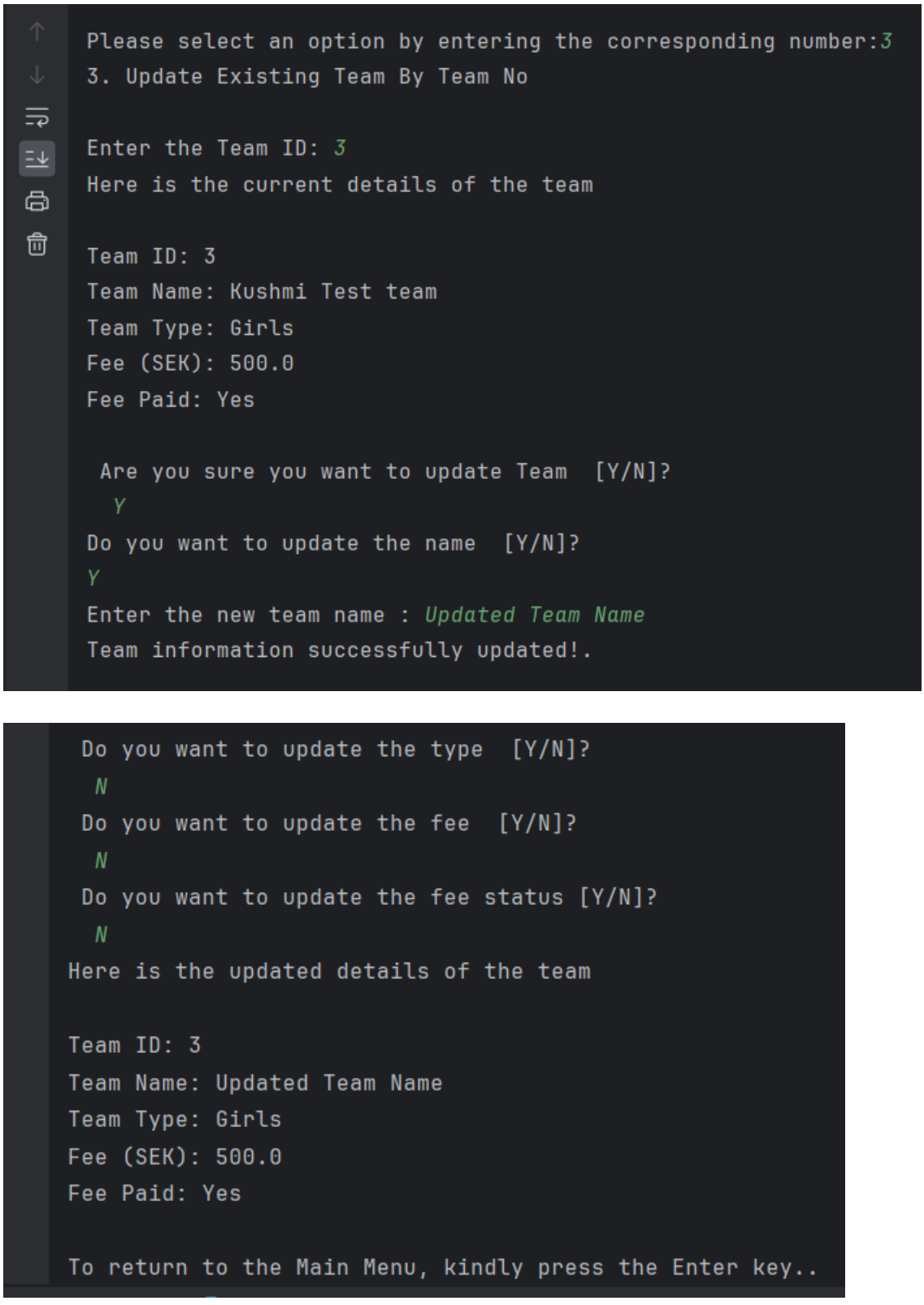

Figure 7- After updating the teams.

3.2.2.4 Delete Teams:

- Option 4 opens the "Delete Menu."
- Input the team ID for deletion, confirm with "Y," and press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:4
4. Delete Existing Team By Team No

Enter the Team ID: 2
 Are you sure you want to delete Team  [Y/N]?
  Y
Team information successfully deleted.

To return to the Main Menu, kindly press the Enter key..
```

Figure 8- After deleting a team result.

3.2.2.5 List All Records:

- Option 5 lists all teams, requiring no additional input.
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:5
5. List All Teams

Team ID: 1
Team Name: KUSHI
Team Type: Girls
Fee (SEK): 500.0
Fee Paid: Yes
Team has been canceled participation on: 1996-02-23
---------------------------------
Team ID: 3
Team Name: Updated Team Name
Team Type: Girls
Fee (SEK): 500.0
Fee Paid: Yes

---------------------------------
Do you want to return to the Main Menu? Then kindly press the Enter key....
```

Figure 9- List All records results.

3.2.2.6 Search by Team Type (Boys/Girls):

- Options 7 and 6 list girls' and boys' teams separately.
- No input needed; press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:7
7. List Boys Teams Only


Boys teams not found.

To return to the Main Menu, kindly press the Enter key..
```

Figure 10- In the list no boys team added.

```
Please select an option by entering the corresponding number:6. List Girls Teams Only

Team ID: 1
Team Name: KUSHI
Team Type: Girls
Fee (SEK): 500.0
Fee Paid: Yes
Team has been canceled participation on: 1996-02-23
---------------------------------
Team ID: 3
Team Name: Updated Team Name
Team Type: Girls
Fee (SEK): 500.0
Fee Paid: Yes

---------------------------------
To return to the Main Menu, kindly press the Enter key..
```

Figure 11- Girls team results.

3.2.2.7 Generate Analytics:

- Option 8 displays team analytics, including total teams, cancelled teams, paid teams, and the percentage of paid teams.
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:8
8. Analytics

Number of total teams : 2

Number of canceled team (Not Active Team Only): 1

Number of paid team: 1

Percentage of Paid Teams: 50.00%

To return to the Main Menu, kindly press the Enter key..
```

Figure 12- Generate Analytics.

3.2.2.8 Cancel Participation:

- Option 9 allows cancelling a team's participation.
- Input the team ID, confirm with "Y," and set the cancellation date.
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:9
9. Cancel Team Participation For the Event

 Are you sure you want to cancel the team's participation in the event? Please note that once canceled, the team cannot be re-activated. [Y/N]?
  Y
Enter the Team ID: 1
Enter the  cancelation date (YYYY-MM-DD): 2024-03-13

Team participation successfully canceled.

Do you want to return to the Main Menu? Then kindly press the Enter key....
```

Figure 13- Cancel participation.

3.2.2.9 Saving In Memory to Text File:

- Option 10 saves in-memory records to a text file.
- Confirm with "Y" to create or update "data_source.txt."
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:10
 Do you want to save data in the text file [Y/N]?
  Y
10. Save Records to the File


The data has been successfully saved to the text file (File name: Data_Source.txt)

Do you want to return to the Main Menu? Then kindly press the Enter key....
```

Figure 14- Text File saving.

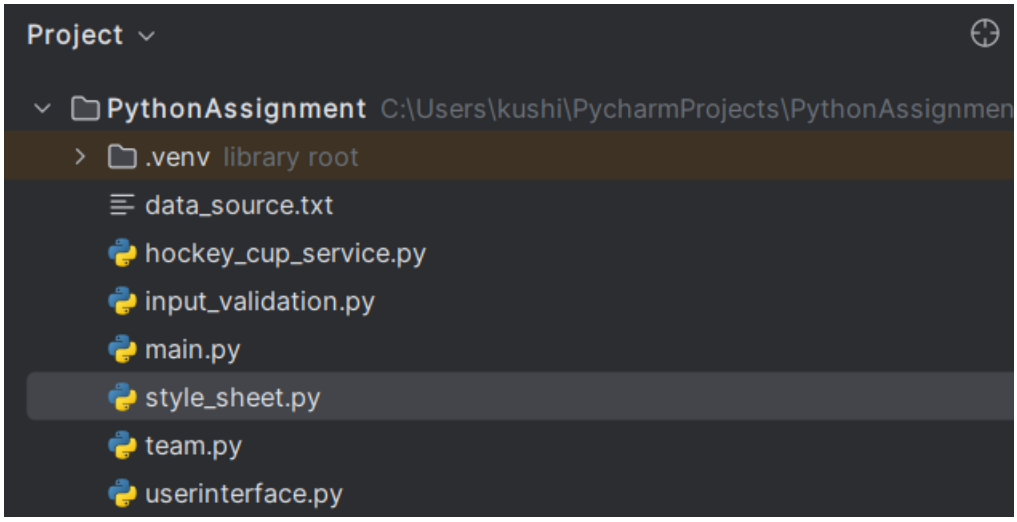

Figure 15 - After saving as a Text File result.

3.2.2.10 Load Data from Text File:

- Option 11 loads historical data from "data_source.txt."
- Confirm with "Y" to read and update in-memory records.
- Press "Enter" to return to the main menu.

```
Please select an option by entering the corresponding number:11
 Do you want to load data historical data from text file [Y/N]?
  Y
11. Load from File


Team loaded successfully.
```

Figure 16- Load data from the text file.

3.2.2.11 Exit from the Program:

- Option 12 exits the system, prompting to save changes to "data_source.txt" with "Y."
- The program gracefully exits after saving.

```
Please select an option by entering the corresponding number:12
12. Quit
 Would you like to save the changes to the text file for future use [Y/N]?
  Y

The data has been successfully saved to the text file (File name: Data_Source.txt)

Thank you for using Hockey Cup Team Management System

Process finished with exit code 0
```

Figure 17- Exit/ Quit results.