Cricket Management Portal Using C

~Made By: Sarthak Garg

DOCUMENTATION

The Cricket Management Portal is a text-based application developed in C that handles player information using structures. It maintains detailed records of cricket players, including their personal information, match participation, playing status, and contract details. The system supports key operations such as adding new player entries, updating existing data, changing player statuses like injured, fit, or retired, and modifying contract terms. Players can also be filtered based on specific fields such as country, match format, or contract type. The portal keeps track of the total number of players and provides options to display all player details in a structured format. Internally, the program uses arrays of structures and menu-driven logic to manage the flow. Each section of the code focuses on a particular function, making the program modular and easy to understand. It offers a clear example of how data can be organized, modified, and displayed in C without relying on any external libraries or databases.

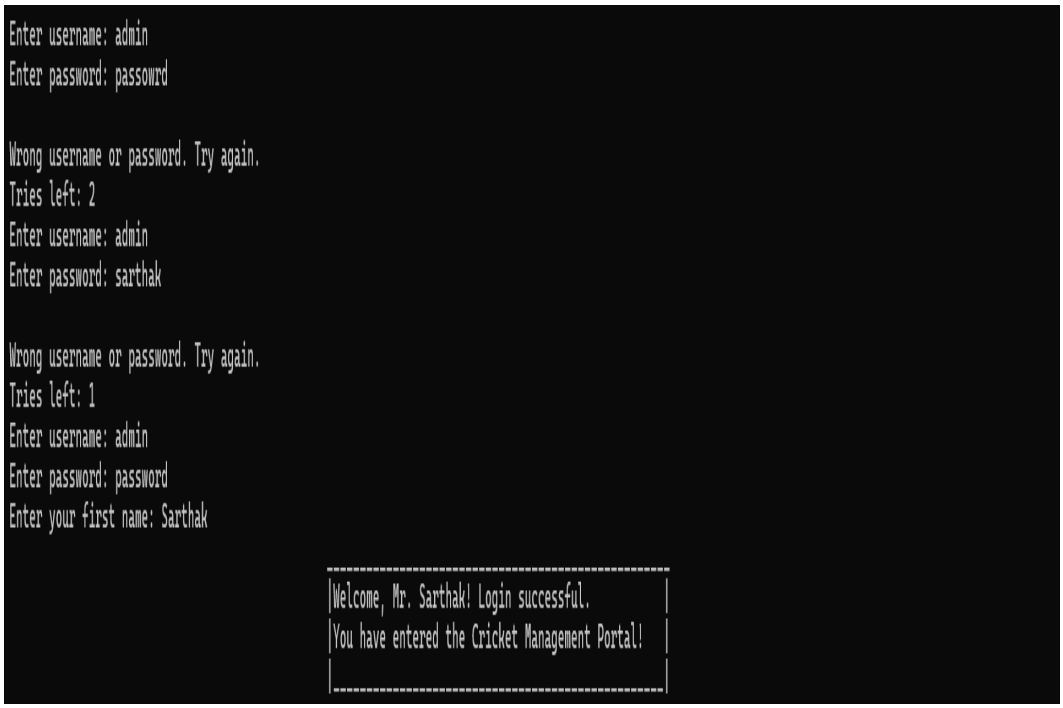
Table of Content:

Objective

The main objective of the Cricket Management Portal is to efficiently manage player data using the C programming language without relying on any external database system. The project is designed to store and organize all player-related information—such as personal details, match participation, current status (fit, injured, or retired), and contract agreements—in a structured and accessible format using C structures. It aims to make data handling simple by providing features to easily add, update, and modify records. By allowing filtering based on specific fields like player number, first name or contract status, the system ensures that users can quickly locate and study any player's data when needed. The program also maintains a count of the total number of players, helping in better tracking and team organization. All information is displayed in a neat, readable format through a menu-driven interface that promotes smooth navigation and ease of use. This project serves to show how powerful structure-based programming can be for building basic management systems, making it ideal for organizing and reviewing data in a simple yet effective manner.

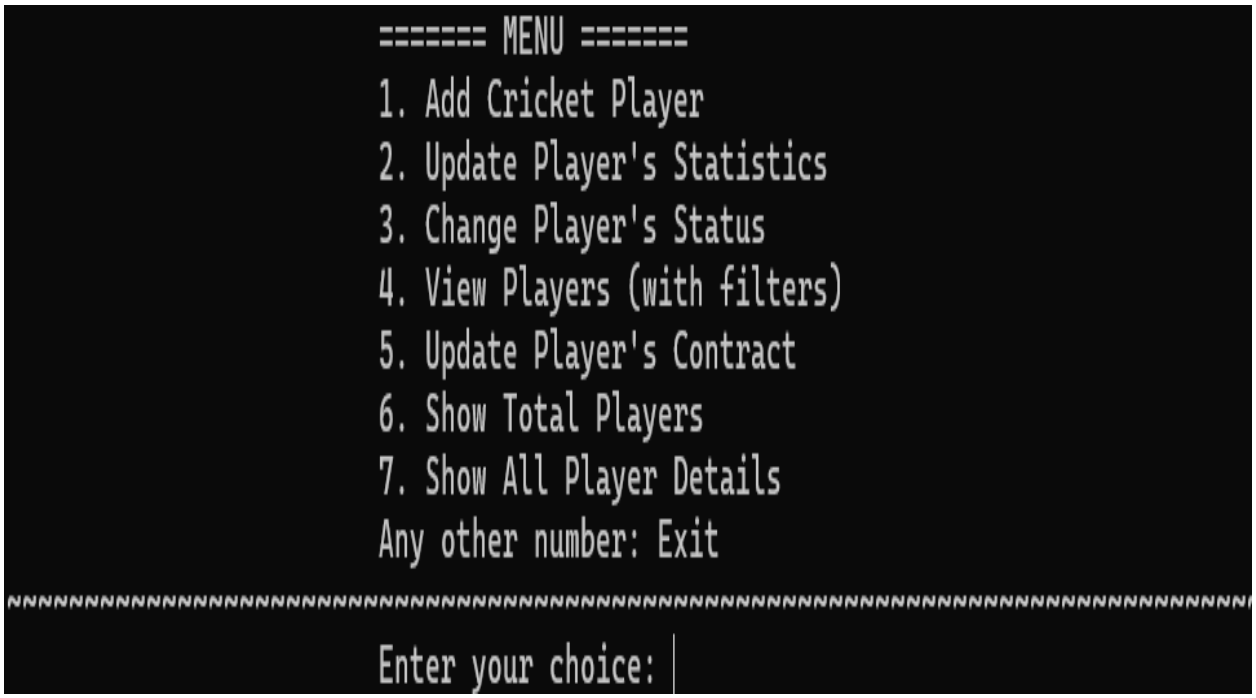
Login

Before accessing the Cricket Management Portal, users must pass through a login system that requires entering a valid username and password. This feature adds a basic layer of security to the application. The user is given three total attempts to enter the correct credentials. If the password is incorrect on the first try, the system notifies the user with "2 tries left," then "1 try left" after the second incorrect attempt. If all three attempts fail, access to the portal is denied, and the program exits. This helps ensure that only authorized users can interact with the player data in the portal.



Menu

The Cricket Management Portal features a menu-driven interface that allows users to perform various operations related to player data. The menu includes options to add new cricket players, update their statistics, and change their current status (such as fit, injured, or retired). Users can also filter and view player records, modify contract details, and check the total number of players stored in the system. An option is also available to display all player details in a structured format. If the user enters any number other than the listed options, the program will exit safely, ensuring a smooth and user-friendly experience.



Adding Player Details

The player addition process in this system begins when the user selects option 1, which activates the portal to add new player details. The first step is to enter a unique Player ID. If the entered ID matches an existing one, the system immediately notifies the user to assign a new, unique Player ID to avoid duplication. Once a valid ID is provided, the user is guided through a sequence of prompts to input various player details. The user starts by selecting the country code from a predefined list (India, England, Australia, etc.), followed by entering the player's first and last name. Next, the system asks for the type of player, offering choices such as All-Rounder, Bowler, Batter, or Wicketkeeper Batter (for any other number entered). The user is then prompted to indicate whether the player has participated in the India Country League, with "1" representing Yes and any other number indicating No. Subsequently, the player's current status is recorded—whether they are fit to play, injured, or retired—based on numerical input. The system also checks if the player has represented the national team, asking the user to confirm this with a binary choice (1 for Yes). Following this, the portal collects the date of debut, including day, month, and year. The next step involves selecting the player's contract type from available categories (A, B, C) or indicating no contract if applicable.

Finally, the system asks for the date when the player started playing cricket, again capturing the day, month, and year. Once all fields are successfully entered, the system validates the input and displays the message "Player added successfully!" confirming that the data has been stored without any issues. This structured and interactive process ensures a comprehensive and error-free entry of player profiles into the system.

```
Add Player Details:
-------------------
Enter the Code of Player: 1

1. India
2. England
3. Australia
4. Sri Lanka
5. South Africa
6. New Zeland
7. Others
Enter the Country Code: 1

Enter the First Name: Sarthak

Enter the Last Name: Garg

Enter the Type of Player(1. All Rounder  2. Bowler       3. Batter       Any other Number. Wicketkeeper Batter): 2

Played India Country League?(1. Yes     Any Other Number. No): 1

1. Fit to play
2. Injured
Any Other Number. Retired
Enter the Status of Player: 1

Has the Player Played for the Country(1=Yes; Any Other Number=No): 1

Enter the Date of Debut: 22

Enter the Month of Debut: 02

Enter the Year of Debut: 2024

1. A
2. B
3. C
Any Other Number. No Contract for Money
Enter the Contract Number: 1

Enter the Date of Start of Cricket: 22

Enter the Month of Start of Cricket: 02

Enter the Year of Start of Cricket: 2012
```

Adding Player Statistics

Once the user successfully executes choice 1 to add a new player, the system returns to the main menu and allows the selection of any other option, including choice 1 again for adding more players. When the user selects choice 2, the system transitions to the module for adding player statistics to an existing player profile. In this mode, the user is asked to enter key performance metrics such as the number of matches played, fifties, hundreds, double hundreds, highest individual score, number of five-wicket hauls, and the career strike rate. These inputs help in building a comprehensive statistical profile for the player. Before proceeding, the system prompts for the Player ID to ensure the statistics are assigned to the correct individual. If an invalid or non-existent Player ID is entered, the system immediately displays a warning message, alerting the user that the player was not found and preventing any incorrect data entry. This safeguards the integrity of the database and ensures that statistics are only linked to valid, registered players. Once the correct ID is confirmed and all data is entered, the statistics are successfully added to the player's profile. From this point on, whenever the player is searched in the system, both their personal information and updated performance statistics are displayed together in an organized and accessible format.

```
                                        ======= MENU =======
                                        1. Add Cricket Player
                                        2. Update Player's Statistics
                                        3. Change Player's Status
                                        4. View Players (with filters)
                                        5. Update Player's Contract
                                        6. Show Total Players
                                        7. Show All Player Details
                                        Any other number: Exit
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                        Enter your choice: 2
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


Update :
-----------------
Enter Player Number to add Milstone: 2

Player Not Found!

                                        ======= MENU =======
                                        1. Add Cricket Player
                                        2. Update Player's Statistics
                                        3. Change Player's Status
                                        4. View Players (with filters)
                                        5. Update Player's Contract
                                        6. Show Total Players
                                        7. Show All Player Details
                                        Any other number: Exit
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                        Enter your choice: 2
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


Update :
-----------------
Enter Player Number to add Milstone: 1

                                        Update all Statistics of the Player
                                        -------------------------------------
Enter the Number of Matches played by Sarthaj Garg: 23
Enter the Number of Fifties hit by Sarthaj Garg: 10
Enter the Number of Hundreds hit by Sarthaj Garg: 12
Enter the Number of Double Hundreds hit by Sarthaj Garg: 1
Enter the Career Strike Rate of Sarthaj Garg: 123.45
Enter the Number of 5 Wicket Haul taken by Sarthaj Garg: 22
Enter the Highest Score hit by Sarthaj Garg: 290
```

Updating Player Status

Similar to option 2, option 3 also requires that option 1 has been executed successfully beforehand—meaning the player must already exist in the system. When the user selects option 3 without a valid player having been added previously, the system immediately checks the entered Player ID. If the player is not found, it displays a warning message, informing the user that no such player exists and halts the update process to maintain data accuracy. However, if the player is found, the system proceeds to prompt the user to update the player's current status. The available options include marking the player as injured, retired, or fit to play. Based on the selection made by the user, the system then updates the player's status accordingly within their profile. This ensures that the most recent information about the player's availability is always reflected in the system. The update is done seamlessly, and the revised status becomes part of the player's record, viewable during any future searches or when accessing player details. This feature is essential for maintaining up-to-date and accurate tracking of a player’s playing condition.

```
                                        ======= MENU =======
                                        1. Add Cricket Player
                                        2. Update Player's Statistics
                                        3. Change Player's Status
                                        4. View Players (with filters)
                                        5. Update Player's Contract
                                        6. Show Total Players
                                        7. Show All Player Details
                                        Any other number: Exit
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                        Enter your choice: 3
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Change Status:
----------------
Enter Player Number to update status: 23

Player not found.

                                        ======= MENU =======
                                        1. Add Cricket Player
                                        2. Update Player's Statistics
                                        3. Change Player's Status
                                        4. View Players (with filters)
                                        5. Update Player's Contract
                                        6. Show Total Players
                                        7. Show All Player Details
                                        Any other number: Exit
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                        Enter your choice: 3
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Change Status:
----------------
Enter Player Number to update status: 1

What do you want the selected player status to be?:
1. Retire
2. Injured & Active
3. Fit & Active
Choose your Option (1-3): 1
Are you sure you want to continue with choice 1? (1: Yes; Any Other Number: No): 1
Status updated to Retired.
```

Viewing Players with applying Filters

Similar to the previous options, option 4 also depends on the successful execution of option 1, meaning at least one player must be added to the system beforehand. When the user selects option 4, the system first prompts for a Player ID to identify the player or set of players to be filtered. If the entered ID does not match any existing records, the system promptly displays a "Player not found" message and terminates the operation to avoid processing invalid data. If the player exists, the portal then proceeds to ask the user to specify a filter condition—this could include criteria such as player type, nationality, performance statistics, contract status, or playing condition (e.g., injured or retired). The system uses the selected filter to search through the list of registered players. If no players match the specified filter, the system returns a message indicating that no matching records were found. However, if one or more players meet the filter criteria, their full details are displayed on the screen. This feature helps users quickly locate players based on specific attributes or performance parameters, ensuring efficient data retrieval and management.

```
                                        ======= MENU =======
                                        1. Add Cricket Player
                                        2. Update Player's Statistics
                                        3. Change Player's Status
                                        4. View Players (with filters)
                                        5. Update Player's Contract
                                        6. Show Total Players
                                        7. Show All Player Details
                                        Any other number: Exit
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                        Enter your choice: 4
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

View Options:
---------------==== View Player by Filter ====
1. Player Number
2. First Name
3. Last Name
4. Contract Type
5. Player Type
6. Player Condition
Choose Option: 2
Enter First Name: Garg
No player matched your filter.
```

```
                                        ======= MENU =======
                                        1. Add Cricket Player
                                        2. Update Player's Statistics
                                        3. Change Player's Status
                                        4. View Players (with filters)
                                        5. Update Player's Contract
                                        6. Show Total Players
                                        7. Show All Player Details
                                        Any other number: Exit
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                        Enter your choice: 4
                    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

View Options:
---------------==== View Player by Filter ====
1. Player Number
2. First Name
3. Last Name
4. Contract Type
5. Player Type
6. Player Condition
Choose Option: 6
Enter Player Condition: Fit
No player matched your filter.
```

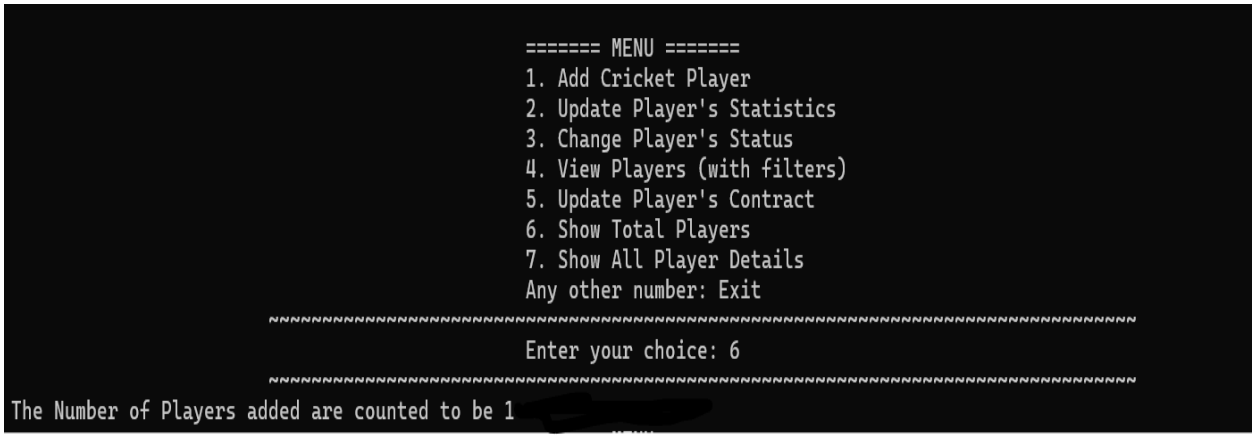
Updating Player Contract

Similar to the previous options, option 5 also requires the successful execution of option 1 to function correctly. This means that at least one player must already be added to the system using their unique Player ID. When the user selects option 5, the system first prompts for the Player ID to identify the player whose contract details are to be updated. If the entered Player ID is not found, the system immediately responds with a "Player not found" message, preventing any further action to maintain data consistency. However, if the Player ID is valid and the player exists in the system, the portal then asks the user to update the player's contract type. The user can input contract categories such as A, B, C, or specify None if the player is not under any contract. Once the new contract type is entered, the system updates the player's record accordingly, ensuring that the latest contract status is reflected in the player's profile. This feature is essential for keeping track of a player's professional agreements and ensures that contract information remains current and accessible whenever needed.

```
                                ======= MENU =======
                                1. Add Cricket Player
                                2. Update Player's Statistics
                                3. Change Player's Status
                                4. View Players (with filters)
                                5. Update Player's Contract
                                6. Show Total Players
                                7. Show All Player Details
                                Any other number: Exit
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                Enter your choice: 5
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


Update Contract:
------------------
Enter Player Number to update contract: 1

Select new contract:
1. A
2. B
3. C
Choice: 1
Confirm choice 1? (1: Yes): 1
Contract updated!

                                ======= MENU =======
                                1. Add Cricket Player
                                2. Update Player's Statistics
                                3. Change Player's Status
                                4. View Players (with filters)
                                5. Update Player's Contract
                                6. Show Total Players
                                7. Show All Player Details
                                Any other number: Exit
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                Enter your choice: 5
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


Update Contract:
------------------
Enter Player Number to update contract: 23
Player not found.
```

Displaying the Total Number of Players

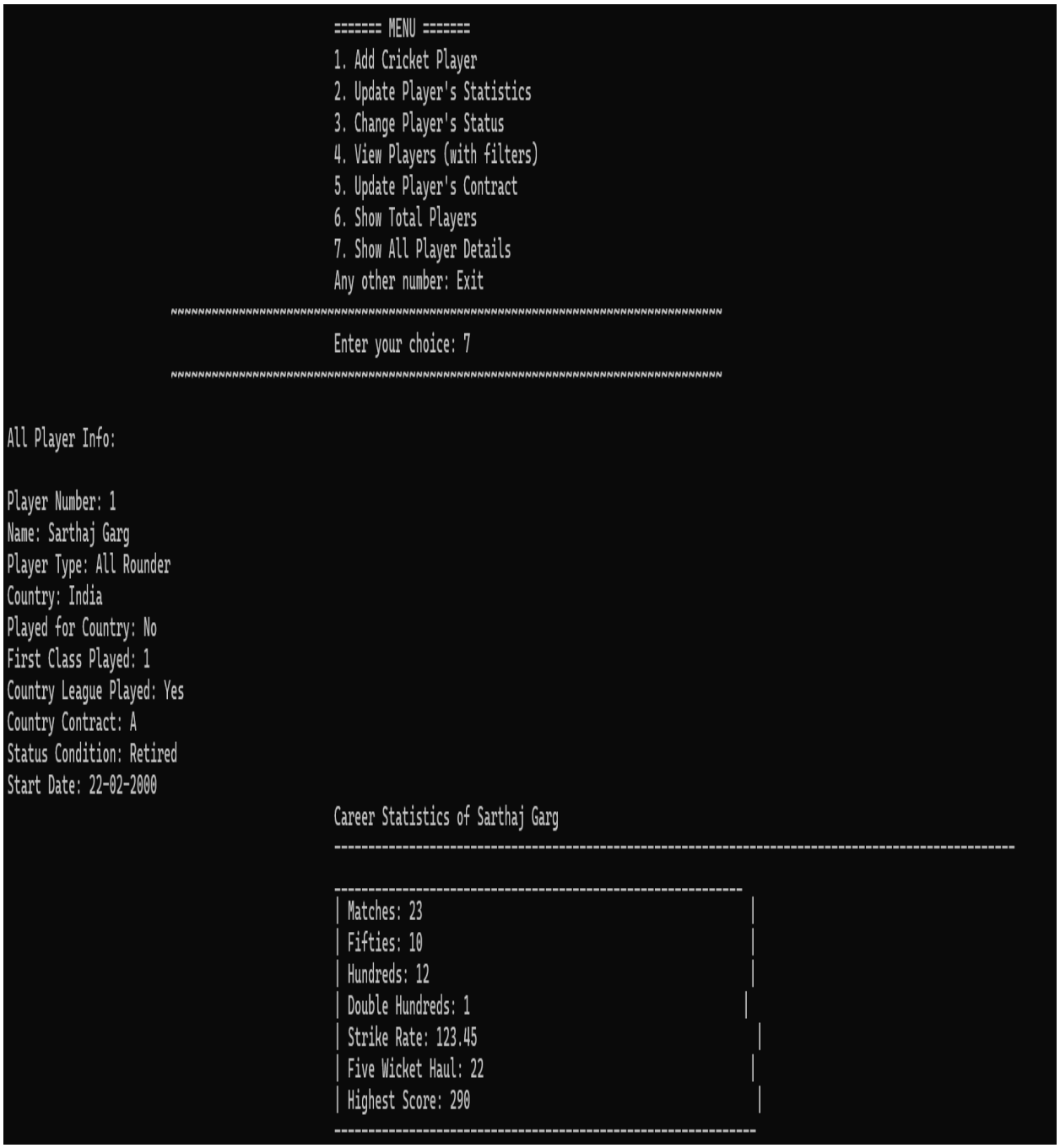
The 6th option, like the previous ones, requires that option 1 has been successfully executed at least once, meaning at least one player must be added to the system for it to work properly. If the user selects option 6 without any players registered, the system automatically displays a message indicating that no player data is available, ensuring that empty or invalid operations are not performed. However, if players have been added, option 6 functions as a summary feature. When executed, it displays the total number of players currently stored in the system. This provides a quick overview of how many players have been registered so far, serving as a useful reference for tracking database size or verifying successful additions. The process is simple, informative, and helps in managing the player database more effectively.



Displaying all Player Details

The 7th option is particularly brilliant and plays a crucial role in the overall functionality of the system. As with the previous options, it is dependent on the successful execution of option 1, meaning at least one player must have been added beforehand. If no players exist in the system, selecting this option will immediately trigger a message informing the user that no player data is available, thereby avoiding the display of an empty or meaningless output. When players are available, option 7 acts as a detailed overview module, displaying the complete set of details for every player registered in the system. This includes personal information such as name, nationality, and player type, along with any data added through other options like statistics (matches played, fifties, hundreds, etc.), contract status, player availability (fit, injured, retired), and more. It consolidates all this information into a well-structured format, giving the user a clear snapshot of all entries made so far. This feature is extremely useful from a data management and review perspective. It allows users to cross-check all entries in one place, making it easier to identify any inconsistencies, incomplete data, or errors that may have occurred during input. If any detail seems incorrect or missing, the user can quickly take action by navigating back to the relevant options to update or correct the information. In essence,

option 7 not only enhances transparency and accessibility but also ensures that the system remains organized, accurate, and up-to-date. It is a valuable tool for maintaining data integrity and gives users confidence that the information stored is both complete and correct.



Code

```
#include <stdio.h>


#include <string.h>


#include <stdlib.h>


struct Country {
  char country_name[100];
  char country_played[100];
  int country_firstclassplayed;
  char country_leagueplayed[100];
  char country_contract[100];
};
struct Status {
  char condition[100];
  int debut_date;
  int debut_month;
  int debut_year;
  int start_date;
  int start_month;
  int start_year;
  struct Country country;
};
struct Milstone {
```

```
  int no_of_fifties;
  int no_of_hundred;
  int no_of_doublehundred;
  int no_of_fivewickethaul;
  int no_of_matches;
  double strike_rate;
  int highest_score;
};
struct Player {
  int player_number;
  char player_firstname[100];
  char player_lastname[100];
  char player_type[100];
  struct Country country;
  struct Status status;
  struct Milstone milstone;
};
struct Player player[200];
int number_of_players = 0;

void AddPlayerDetail() {
  int i;
  int temp_player_number;
  printf("\nEnter the Code of Player: ");
  scanf("%d", & temp_player_number);
```

```
 for (i = 0; i < number_of_players; i++) {

  if (player[i].player_number == temp_player_number) {

   printf("\nThis Player Number already exists.");

   return;

  }

 }

 int w;

 player[number_of_players].player_number = temp_player_number;

   printf("\n1. India\n2. England\n3. Australia\n4. Sri Lanka\n5. South Africa\n6. New Zeland\n7. Others");

 printf("\nEnter the Country Code: ");

 scanf("%d", & w);

 switch (w) {

 case 1:

  strcpy(player[number_of_players].country.country_name, "India");

  break;

 case 2:

  strcpy(player[number_of_players].country.country_name, "England");

  break;

 case 3:

  strcpy(player[number_of_players].country.country_name, "Australia");

  break;

 case 4:

  strcpy(player[number_of_players].country.country_name, "Sri Lanka");

  break;

 case 5:
```

```
  strcpy(player[number_of_players].country.country_name, "South Africa");
  break;
 case 6:
  strcpy(player[number_of_players].country.country_name, "New Zeland");
  break;
 default:
  strcpy(player[number_of_players].country.country_name, "Others");
  break;
 }
 printf("\nEnter the First Name: ");
 scanf("%s", player[number_of_players].player_firstname);
 printf("\nEnter the Last Name: ");
 scanf("%s", player[number_of_players].player_lastname);
 int qw;
   printf("\nEnter the Type of Player(1. All Rounder\t 2. Bowler \t 3. Batter \t Any other Number. Wicketkeeper Batter): ");
 scanf("%d", & qw);
 switch (qw) {
 case 1:
  strcpy(player[number_of_players].player_type, "All Rounder");
  break;
 case 2:
  strcpy(player[number_of_players].player_type, "Bowler");
  break;
 case 3:
  strcpy(player[number_of_players].player_type, "Batter");
```

```
  break;
 default:
  strcpy(player[number_of_players].player_type, "Wicketkeepter Batter");
  break;
 }
 player[number_of_players].country.country_firstclassplayed = 1;
 int abcd;
 int asd;
        printf("\nPlayed %s Country League?(1. Yes \tAny Other Number. No): ",
player[number_of_players].country.country_name);
 scanf("%d", & abcd);
 switch (abcd) {
 case 1:
  strcpy(player[number_of_players].country.country_leagueplayed, "Yes");
  break;
 default:
  printf("Are you sure you have'nt played %s National League? (1. Yes \tAny Other Number. No ): ");
  scanf("%d", & asd);
  switch (asd) {
  case 1:
   strcpy(player[number_of_players].country.country_leagueplayed, "No");
   break;
  default:
   strcpy(player[number_of_players].country.country_leagueplayed, "Yes");
   break;
  }
```

```
}
printf("\n1. Fit to play\n2. Injured\nAny Other Number. Retired");
int omega;
printf("\nEnter the Status of Player: ");
scanf("%d", & omega);
if (omega == 1) {
 int alpha;
 int z;
 strcpy(player[number_of_players].status.condition, "Fit & Active");
 printf("\nHas the Player Played for the Country(1=Yes; Any Other Number=No): ");
 scanf("%d", & alpha);
 switch (alpha) {
 case 1:
  strcpy(player[number_of_players].country.country_played, "Yes");
  printf("\nEnter the Date of Debut: ");
  scanf("%d", & player[number_of_players].status.debut_date);
  printf("\nEnter the Month of Debut: ");
  scanf("%d", & player[number_of_players].status.debut_month);
  printf("\nEnter the Year of Debut: ");
  scanf("%d", & player[number_of_players].status.debut_year);
  printf("\n1. A\n2. B\n3. C\nAny Other Number. No Contract for Money");
  printf("\nEnter the Contract Number: ");
  scanf("%d", & z);
  switch (z) {
  case 1:
   strcpy(player[number_of_players].country.country_contract, "A");
```

```
    break;
   case 2:
    strcpy(player[number_of_players].country.country_contract, "B");
    break;
   case 3:
    strcpy(player[number_of_players].country.country_contract, "C");
    break;
   default:
    strcpy(player[number_of_players].country.country_contract, "NONE");
    break;
   }
  default:
   strcpy(player[number_of_players].country.country_played, "No");
   printf("\nEnter the Date of Start of Cricket: ");
   scanf("%d", & player[number_of_players].status.start_date);
   printf("\nEnter the Month of Start of Cricket: ");
   scanf("%d", & player[number_of_players].status.start_month);
   printf("\nEnter the Year of Start of Cricket: ");
   scanf("%d", & player[number_of_players].status.start_year);
   strcpy(player[number_of_players].country.country_contract, "D");
   player[number_of_players].country.country_firstclassplayed = 1;
   break;
  }
 } else if (omega == 2) {
  int beta;
  int z;
```

```
strcpy(player[number_of_players].status.condition, "Injured & Active");
printf("\nHas the Player Played for the Country(1=Yes; Any Other Number=No): ");
scanf("%d", & beta);
switch (beta) {
case 1:
 strcpy(player[number_of_players].country.country_played, "Yes");
 printf("\nEnter the Date of Debut: ");
 scanf("%d", & player[number_of_players].status.debut_date);
 printf("\nEnter the Month of Debut: ");
 scanf("%d", & player[number_of_players].status.debut_month);
 printf("\nEnter the Year of Debut: ");
 scanf("%d", & player[number_of_players].status.debut_year);
 printf("\n1. A\n2. B\n3. C\nAny Other Number. No Contract for Money");
 printf("\nEnter the Contract Number: ");
 scanf("%d", & z);
 switch (z) {
 case 1:
  strcpy(player[number_of_players].country.country_contract, "A");
  break;
 case 2:
  strcpy(player[number_of_players].country.country_contract, "B");
  break;
 case 3:
  strcpy(player[number_of_players].country.country_contract, "C");
  break;
 default:
```

```
    strcpy(player[number_of_players].country.country_contract, "NONE");
    break;
   }

  default:
   strcpy(player[number_of_players].country.country_played, "No");
   printf("\nEnter the Date of Start of Cricket: ");
   scanf("%d", & player[number_of_players].status.start_date);
   printf("\nEnter the Month of Start of Cricket: ");
   scanf("%d", & player[number_of_players].status.start_month);
   printf("\nEnter the Year of Start of Cricket: ");
   scanf("%d", & player[number_of_players].status.start_year);
   strcpy(player[number_of_players].country.country_contract, "D");
   player[number_of_players].country.country_firstclassplayed = 1;
   break;
  }
 } else {
  int gama;
  int z;
  strcpy(player[number_of_players].status.condition, "Retired");
  printf("\nHas the Player Played for the Country(1=Yes; Any Other Number=No): ");
  scanf("%d", & gama);
  switch (gama) {
  case 1:
   strcpy(player[number_of_players].country.country_played, "Yes");
   printf("\nEnter the Date of Debut: ");
```

```
 scanf("%d", & player[number_of_players].status.debut_date);
 printf("\nEnter the Month of Debut: ");
 scanf("%d", & player[number_of_players].status.debut_month);
 printf("\nEnter the Year of Debut: ");
 scanf("%d", & player[number_of_players].status.debut_year);
 printf("\n1. A\n2. B\n3. C\nAny Other Number. No Contract for Money");
 printf("\nEnter the Contract Number: ");
 scanf("%d", & z);
 switch (z) {
 case 1:
  strcpy(player[number_of_players].country.country_contract, "A");
  break;
 case 2:
  strcpy(player[number_of_players].country.country_contract, "B");
  break;
 case 3:
  strcpy(player[number_of_players].country.country_contract, "C");
  break;
 default:
  strcpy(player[number_of_players].country.country_contract, "NONE");
  break;
 }
 break;
default:
 strcpy(player[number_of_players].country.country_played, "No");
 printf("\nEnter the Date of Start of Cricket: ");
```

```
    scanf("%d", & player[number_of_players].status.start_date);
    printf("\nEnter the Month of Start of Cricket: ");
    scanf("%d", & player[number_of_players].status.start_month);
    printf("\nEnter the Year of Start of Cricket: ");
    strcpy(player[number_of_players].country.country_contract, "D");
    player[number_of_players].country.country_firstclassplayed = 1;
    break;
   }
  }
  number_of_players++;
  printf("\nPlayer added successfully!\n");
}

void MilestoneUpdate() {
  int i, j;
  int play, choice, found = 0;
  printf("\nEnter Player Number to add Milstone: ");
  scanf("%d", & play);
  for (i = 0; i < number_of_players; i++) {
   if (player[i].player_number == play) {
    found = 1;
    printf("\n\t\t\t\t\t\tUpdate all Statistics of the Player\n");
    printf("\t\t\t\t\t\t-----------------------------------");

        printf("\nEnter the Number of Matches played by %s %s: ", player[i].player_firstname,
player[i].player_lastname);
```

```
    scanf("%d", & player[i].milstone.no_of_matches);

            printf("Enter the Number of Fifties hit by %s %s: ", player[i].player_firstname, player[i].player_lastname);

    scanf("%d", & player[i].milstone.no_of_fifties);

            printf("Enter the Number of Hundreds hit by %s %s: ", player[i].player_firstname, player[i].player_lastname);

    scanf("%d", & player[i].milstone.no_of_hundred);

          printf("Enter the Number of Double Hundreds hit by %s %s: ", player[i].player_firstname, player[i].player_lastname);

    scanf("%d", & player[i].milstone.no_of_doublehundred);

              printf("Enter the Career Strike Rate of %s %s: ", player[i].player_firstname, player[i].player_lastname);

    scanf("%lf", & player[i].milstone.strike_rate);

          printf("Enter the Number of 5 Wicket Haul taken by %s %s: ", player[i].player_firstname, player[i].player_lastname);

    scanf("%d", & player[i].milstone.no_of_fivewickethaul);

              printf("Enter the Highest Score hit by %s %s: ", player[i].player_firstname, player[i].player_lastname);

    scanf("%d", & player[i].milstone.highest_score);

   } else if (!found) {

    printf("\nPlayer Not Found!\n");

   }

 }

}


void NumberofPlayers() {

 if (number_of_players == 0) {

  printf("First Enter Player Details.");

 } else {
```

```
   printf("The Number of Players added are counted to be %d", number_of_players);
  }
}
void ChangeStatus() {
  int i, play, choice, confirm;
  int found = 0;

  printf("\nEnter Player Number to update status: ");
  scanf("%d", & play);

  for (i = 0; i < number_of_players; i++) {
    if (player[i].player_number == play) {
      found = 1;

      printf("\nWhat do you want the selected player status to be?: \n");
      printf("1. Retire\n2. Injured & Active\n3. Fit & Active");
      printf("\nChoose your Option (1-3): ");
      scanf("%d", & choice);

      printf("Are you sure you want to continue with choice %d? (1: Yes; Any Other Number: No): ", choice);
      scanf("%d", & confirm);

      if (confirm != 1) {
        printf("Operation cancelled.\n");
        return;
      }
```

```
    switch (choice) {
    case 1:
      strcpy(player[i].status.condition, "Retired");
      printf("Status updated to Retired.\n");
      break;

    case 2:
      strcpy(player[i].status.condition, "Injured & Active");
      printf("Status updated to Injured & Active.\n");
      break;

    case 3:
      strcpy(player[i].status.condition, "Fit & Active");
      printf("Status updated to Fit & Active.\n");
      break;

    default:
      printf("Invalid choice.\n");
      break;
    }
    break;
  }
}

if (!found)
```

```
    printf("\nPlayer not found.\n");
}
void UpdateContract() {
  int i, play, choice, confirm, found = 0;

  printf("\nEnter Player Number to update contract: ");
  scanf("%d", & play);

  for (i = 0; i < number_of_players; i++) {
    if (player[i].player_number == play) {
      found = 1;
      printf("\nSelect new contract:\n1. A\n2. B\n3. C\nChoice: ");
      scanf("%d", & choice);
      printf("Confirm choice %d? (1: Yes): ", choice);
      scanf("%d", & confirm);
      if (confirm != 1) {
        printf("Cancelled.\n");
        return;
      }

      if (choice == 1) strcpy(player[i].country.country_contract, "A");
      else if (choice == 2) strcpy(player[i].country.country_contract, "B");
      else if (choice == 3) strcpy(player[i].country.country_contract, "C");
      else {
        printf("Invalid input.\n");
        return;
```

```
    }

    printf("Contract updated!\n");
    return;
   }
  }

  if (!found) printf("Player not found.\n");
}

void ViewingOptions() {
  int choice, i, found = 0, num_input;
  char str_input[100];
  if (number_of_players == 0) {
   printf("No players added yet.\n");
   return;
  }
  printf("==== View Player by Filter ====\n");
   printf("1. Player Number\n2. First Name\n3. Last Name\n4. Contract Type\n5. Player Type\n6. Player
Condition\nChoose Option: ");
  scanf("%d", & choice);
  switch (choice) {
  case 1:
   printf("Enter Player Number: ");
   scanf("%d", & num_input);
   for (i = 0; i < number_of_players; i++)
```

```
    if (player[i].player_number == num_input) {
      found = 1;
      printf("\n#%d: %s %s | Type: %s | Country: %s | Contract: %s | Status: %s\n",
        player[i].player_number, player[i].player_firstname, player[i].player_lastname,
        player[i].player_type, player[i].country.country_name,
        player[i].country.country_contract, player[i].status.condition);
    }
  break;
case 2:
  printf("Enter First Name: ");
  scanf("%s", str_input);
  for (i = 0; i < number_of_players; i++)
    if (strcmp(player[i].player_firstname, str_input) == 0) {
      found = 1;
      printf("\n#%d: %s %s | Type: %s | Country: %s | Contract: %s | Status: %s\n",
        player[i].player_number, player[i].player_firstname, player[i].player_lastname,
        player[i].player_type, player[i].country.country_name,
        player[i].country.country_contract, player[i].status.condition);
    }
  break;
case 3:
  printf("Enter Last Name: ");
  scanf("%s", str_input);
  for (i = 0; i < number_of_players; i++)
    if (strcmp(player[i].player_lastname, str_input) == 0) {
      found = 1;
```

```
      printf("\n#%d: %s %s | Type: %s | Country: %s | Contract: %s | Status: %s\n",
        player[i].player_number, player[i].player_firstname, player[i].player_lastname,
        player[i].player_type, player[i].country.country_name,
        player[i].country.country_contract, player[i].status.condition);
    }
  break;
case 4:
  printf("Enter Contract Type (A/B/C/D/NONE): ");
  scanf("%s", str_input);
  for (i = 0; i < number_of_players; i++)
    if (strcmp(player[i].country.country_contract, str_input) == 0) {
      found = 1;
      printf("\n#%d: %s %s | Type: %s | Country: %s | Contract: %s | Status: %s\n",
        player[i].player_number, player[i].player_firstname, player[i].player_lastname,
        player[i].player_type, player[i].country.country_name,
        player[i].country.country_contract, player[i].status.condition);
    }
  break;
case 5:
  printf("Enter Player Type: ");
  scanf(" %[^\n]s", str_input);
  for (i = 0; i < number_of_players; i++)
    if (strcmp(player[i].player_type, str_input) == 0) {
      found = 1;
      printf("\n#%d: %s %s | Type: %s | Country: %s | Contract: %s | Status: %s\n",
        player[i].player_number, player[i].player_firstname, player[i].player_lastname,
```

```
        player[i].player_type, player[i].country.country_name,
        player[i].country.country_contract, player[i].status.condition);
    }
   break;
  case 6:
   printf("Enter Player Condition: ");
   scanf(" %[^\n]s", str_input);
   for (i = 0; i < number_of_players; i++)
    if (strcmp(player[i].status.condition, str_input) == 0) {
      found = 1;
      printf("\n#%d: %s %s | Type: %s | Country: %s | Contract: %s | Status: %s\n",
        player[i].player_number, player[i].player_firstname, player[i].player_lastname,
        player[i].player_type, player[i].country.country_name,
        player[i].country.country_contract, player[i].status.condition);
    }
   break;
  default:
   printf("Invalid Option.\n");
   return;
  }
  if (!found) printf("No player matched your filter.\n");
}

void AllPlayerInfo() {
  int i;
  if (number_of_players == 0) {
```

```
  printf("No players added yet.\n");
  return;
 }

 for (i = 0; i < number_of_players; i++) {
  printf("\nPlayer Number: %d\n", player[i].player_number);
  printf("Name: %s %s\n", player[i].player_firstname, player[i].player_lastname);
  printf("Player Type: %s\n", player[i].player_type);

  printf("Country: %s\n", player[i].country.country_name);
  printf("Played for Country: %s\n", player[i].country.country_played);
  printf("First Class Played: %d\n", player[i].country.country_firstclassplayed);
  printf("Country League Played: %s\n", player[i].country.country_leagueplayed);
  printf("Country Contract: %s\n", player[i].country.country_contract);

  printf("Status Condition: %s\n", player[i].status.condition);

  if (strcmp(player[i].country.country_played, "Yes") == 0) {
      printf("Debut Date: %02d-%02d-%d\n", player[i].status.debut_date, player[i].status.debut_month,
player[i].status.debut_year);
  }
  if (strcmp(player[i].country.country_played, "No") == 0) {
        printf("Start Date: %02d-%02d-%d\n", player[i].status.start_date, player[i].status.start_month,
player[i].status.start_year);
  }

  printf("\t\t\t\t\t\tCareer Statistics of %s %s\n", player[i].player_firstname, player[i].player_lastname);
```

```
printf("\t\t\t\t\t\t----------------------------------------------------------------------------------------------------\n
");


printf("\t\t\t\t\t\t_____________________________________________________________________________________________
");

   printf("\n\t\t\t\t\t\t| Matches: %d                                  |", player[i].milstone.no_of_matches);

   printf("\n\t\t\t\t\t\t| Fifties: %d                                 |", player[i].milstone.no_of_fifties);

   printf("\n\t\t\t\t\t\t| Hundreds: %d                                 |", player[i].milstone.no_of_hundred);

        printf("\n\t\t\t\t\t\t|  Double  Hundreds:  %d                                                                  |",
player[i].milstone.no_of_doublehundred);

   printf("\n\t\t\t\t\t\t| Strike Rate: %.2lf                           |", player[i].milstone.strike_rate);

        printf("\n\t\t\t\t\t\t|  Five  Wicket  Haul:  %d                                                                |",
player[i].milstone.no_of_fivewickethaul);

   printf("\n\t\t\t\t\t\t| Highest Score: %d                            |", player[i].milstone.highest_score);

   printf("\n\t\t\t\t\t\t-------------------------------------------");

 }

}


int main() {

 char username[100] = "admin";

 char password[100] = "password";

 char user[100];

 char pass[100];

 char name[35];

 int j, choice, o;


 for (j = 1; j <= 3; j++) {

  printf("Enter username: ");
```

```
scanf("%s", user);

printf("Enter password: ");
scanf("%s", pass);

if (strcmp(user, username) != 0 || strcmp(pass, password) != 0) {
  if (j != 3) {
    printf("\nWrong username or password. Try again.\n");
    printf("Tries left: %d\n", 3 - j);
  } else {
    printf("\nYou tried too many times. Access Denied.\n");
  }
} else {
  printf("Enter your first name: ");
  scanf("%s", name);

  printf("\t\t\t\t\t\t_____________________________________________________________");
  printf("\n\t\t\t\t\t\t|Welcome, Mr. %s! Login successful.          |\n", name);
  printf("\t\t\t\t\t\t|You have entered the Cricket Management Portal!   |\n");
  printf("\t\t\t\t\t\t|_____________________________________________________________|");

  while (1) {
    printf("\n\t\t\t\t\t\t======= MENU =======\n");
    printf("\t\t\t\t\t\t1. Add Cricket Player\n");
    printf("\t\t\t\t\t\t2. Update Player's Statistics\n");
    printf("\t\t\t\t\t\t3. Change Player's Status\n");
```

```
        printf("\t\t\t\t\t\t4. View Players (with filters)\n");

        printf("\t\t\t\t\t\t5. Update Player's Contract\n");

        printf("\t\t\t\t\t\t6. Show Total Players\n");

        printf("\t\t\t\t\t\t7. Show All Player Details\n");

        printf("\t\t\t\t\t\tAny other number: Exit\n");

printf("\t\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~\n");

        printf("\t\t\t\t\t\tEnter your choice: ");

        scanf("%d", & choice);

printf("\t\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~\n");

        switch (choice) {

        case 1:

          printf("\nAdd Player Details: \n");

          printf("------------------");

          AddPlayerDetail();

          break; //Done

        case 2:

          printf("\nUpdate :\n");

          printf("----------------");

          MilestoneUpdate();

          break; //Done

        case 3:

          printf("\nChange Status:\n");

          printf("---------------");

          ChangeStatus();
```

```
  break; //Done
case 4:
  printf("\nView Options:\n");
  printf("--------------");
  ViewingOptions();
  break; //Done
case 5:
  printf("\nUpdate Contract:\n");
  printf("------------------");
  UpdateContract();
  break; //Done
case 6:
  NumberofPlayers();
  break; //Done
case 7:
  printf("\nAll Player Info:\n");
  AllPlayerInfo();
  printf("------------------");
  break; //Done
default:
  printf("Are you sure you want to exit(1:Yes;Any Number: No): ");
  scanf("%d", & o);
  switch (o) {
  case 1:
    printf("\t\t\t\t\t\t_____________________________________________________________________");
    printf("\n\t\t\t\t\t\t|Thankss, Mr. %s! Exit successful.           |\n", name);
```

```
            printf("\t\t\t\t\t\t|You have exited the Cricket Management Portal!    |\n");
            printf("\t\t\t\t\t\t|_____|");
            exit(0);
            break;
           default:
            printf("You can reuse the program");
            continue;
            break;
          }
        }
      }
    }
   return 0;
}
```

Further Uses and Conclusion

Further Uses:

The current player management system serves as a foundational platform for maintaining and updating player-related information such as personal details, match statistics, player status, contract classification, and filtered views. While the existing command-line interface offers essential functionality, there is substantial scope for enhancement and expansion of this system in the future. Below are the key areas where this system can be extended:

Database Integration

1. At present, the system does not maintain persistent data storage. By integrating with relational databases such as MySQL or PostgreSQL, or NoSQL solutions like MongoDB, the data can be preserved beyond the current session. This will support scalability, multi-user access, and more efficient data retrieval.

Enhanced Search, Filter, and Sort Functionalities

2. Advanced search and filter functionalities can be introduced, including range-based filtering (e.g., strike rate range, score thresholds) and sorting (e.g., top performers, most matches played). These features

will assist users in deriving more meaningful insights from the data.

Player Performance Analytics

3. The system can be enhanced to provide statistical analysis and insights into player performance over time. Libraries such as Pandas, Matplotlib, and Seaborn can be used to generate performance trends, comparisons, and predictive analyses to support decision-making.

Integration with Live Match Data

4. Integration with live match data APIs (such as CricAPI) can automate the updating of player statistics in real time. This feature is particularly useful for coaches, analysts, and selectors monitoring player performance during active seasons.

Multi-Team and Tournament Management

5. The system can be extended to support multiple teams and full tournament structures. Features such as inter-team transfers, tournament tracking, and aggregated statistics can be implemented to manage larger datasets and more complex requirements.

Conclusion

The player management portal developed in this project provides a systematic and interactive platform for handling key aspects of cricket player data. Through a series of well-defined options, users can add player details, update individual statistics, modify player status and contract information, filter data based on custom conditions, and view a consolidated list of all entered player profiles. The portal ensures that each operation is performed only after the successful registration of player data, thereby preserving accuracy and consistency throughout the process. Clear user prompts, structured input validation, and informative system messages contribute to an efficient and user-friendly experience. The modular design ensures that each feature functions independently while maintaining logical coherence with the underlying dataset. Although currently designed as a console-based application, the portal establishes a strong foundation for future development. Potential enhancements include integration with persistent databases, graphical interfaces, analytics modules, and mobile accessibility, which would significantly expand its usability in real-world scenarios such as sports academies, coaching centers, and performance monitoring environments. In conclusion, this portal serves as a practical demonstration of how programming principles

can be applied to build an effective, real-time data handling solution within a sports context.