IPlayer Interface

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TeamManagement
{
    public interface IPlayer
    {
        int PlayerId {  get; }
        string Name { get; }
        int Age {  get; }
    }
}
```

ITeam Interface

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TeamManagement
{
    public interface ITeam
    {
        bool AddPlayer(IPlayer player);
        bool RemovePlayer(int playerId);
        IPlayer GetPlayerById(int playerId);
        IEnumerable<IPlayer> GetPlayersByName(string playerName);
        IEnumerable<IPlayer> GetAllPlayers();
    }
}
```

CricketTeam Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TeamManagement
{
    public class CricketTeam:ITeam
    {
        public List<IPlayer> players = new List<IPlayer>();

        public bool AddPlayer(IPlayer player)
        {
            if (players.Count >= 11)
            {
                Console.WriteLine("Cannot add more than 11 players to the 
team.");
                return false;
            }

            players.Add(player);
            return true;
        }
        public bool RemovePlayer(int playerId)
        {
            IPlayer playerToRemove = players.FirstOrDefault(p => p.PlayerId ==
playerId);
            if (playerToRemove != null)
            {
                players.Remove(playerToRemove);
                return true;
            }

            Console.WriteLine("Player not found.");
            return false;
        }

        public IPlayer GetPlayerById(int playerId)
        {
            return players.FirstOrDefault(p => p.PlayerId == playerId);
        }

        public IEnumerable<IPlayer> GetPlayersByName(string playerName)
        {
            return players.Where(p => p.Name.Equals(playerName,
StringComparison.OrdinalIgnoreCase));
        }

        public IEnumerable<IPlayer> GetAllPlayers()
        {
            return players;
        }
    }
}
```

CricketPlayer Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TeamManagement
{
    public class CricketPlayer:IPlayer
    {
        public int PlayerId { get; }
        public string Name { get; }
        public int Age { get; }

        public CricketPlayer(int playerId, string name, int age)
        {
            PlayerId = playerId;
            Name = name;
            Age = age;
        }
    }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TeamManagement
{
    internal class Program
    {
        static void Main(string[] args)
        {
            ITeam team = new CricketTeam();
            bool exit = false;

            do
            {
                Console.WriteLine("Choose an option:");
                Console.WriteLine("1. Add a player");
                Console.WriteLine("2. Remove a player");
                Console.WriteLine("3. Get player details by ID");
                Console.WriteLine("4. Get player details by name");
                Console.WriteLine("5. Get all players");
                Console.WriteLine("6. Exit");

                int choice = int.Parse(Console.ReadLine());


                switch (choice)
                {
                    case 1:
                        Console.WriteLine("Enter Player ID:");
                        int playerId = Convert.ToInt32(Console.ReadLine());

                        Console.WriteLine("Enter Player Name:");
                        string playerName = Console.ReadLine();

                        Console.WriteLine("Enter Player Age:");
                        int playerAge = Convert.ToInt32(Console.ReadLine());

                        IPlayer newPlayer = new CricketPlayer(playerId,
playerName, playerAge);
                        team.AddPlayer(newPlayer);
                        break;

                    case 2:
                        Console.WriteLine("Enter Player ID to remove:");
                        int playerToRemoveId =
Convert.ToInt32(Console.ReadLine());
                        team.RemovePlayer(playerToRemoveId);
                        break;

                    case 3:
                        Console.WriteLine("Enter Player ID to get details:");
                        int playerIdToGet = Convert.ToInt32(Console.ReadLine());
                        IPlayer playerById = team.GetPlayerById(playerIdToGet);
                        if (playerById != null)
                        {
                            Console.WriteLine($"Player found by ID:
{playerById.Name}, Age: {playerById.Age}");
                        }
                        else
                        {
                            Console.WriteLine("Player not found.");
                        }
                        break;

                    case 4:
                        Console.WriteLine("Enter Player Name to get details:");
                        string playerNameToGet = Console.ReadLine();
                        IEnumerable<IPlayer> playersByName = 
team.GetPlayersByName(playerNameToGet);
                        if (playersByName.Any())
                        {
                            foreach (var player in playersByName)
                            {
                                Console.WriteLine($"Player found by Name: 
{player.Name}, Age: {player.Age}");
                            }
                        }
                        else
                        {
                            Console.WriteLine("Player not found.");
                        }
                        break;

                    case 5:
                        IEnumerable<IPlayer> allPlayers = team.GetAllPlayers();
                        foreach (var player in allPlayers)
                        {
                            Console.WriteLine($"Player: {player.Name}, Age: 
{player.Age}");
                        }
                        break;

                    case 6:
                        exit = true;
                        break;

                    default:
                        Console.WriteLine("Invalid choice. Please enter a number 
between 1 to 6.");
                        break;
                }
            } while (!exit);
            Console.ReadKey();
        }

    }
}
```