# Hands On Progress Assessment - 18th August 2021

Create an application that interacts with the user to add football players into a data repository and generates a list of players to form the football team based on the ranking. The football team in the output should comprise 11 players distributed as - 4 Defenders, 3 Mid fielders, 3 Forwards and 1 Goal Keeper.

The application consists of two parts/modules (use 1 package per module) – **User Interface Module** and **Player Repository Module**.

**User Interface Module**
When the application is launched, the user interface module should display the following menu to the user:

```
Welcome! Please select the action you want to perform
   1. Add Players to repository
   2. Form team
```

If the user selects “Add players to the repository” option:

- The application should display the following message and accept the input (n)

```
Enter the number of players that you would like to add
```

- After taking the input, the application should accept the details for each player. For each player the application must capture the following details
  - Name (The name of the player)
  - Category (One of Defender, Mid fielder, Forward or Goal keeper)
  - Ranking (The ranking for the player - a lower number is better)

- After accepting the input, the details that are entered must be passed onto the repository module to store the data.

If the user selects “Form Team”, then the user interface must display the list of the 11 players with their details as follows (Note: only partial output is shown)

```
Team formed with 11 players:
S No., Player Name, Category, Rank
1, XYZ, Goal Keeper, 7
2, ABC, Forward, 12
...
```

**Player repository module**

This module contains a repository of player information (implemented using the data structures available through the collections framework in Java)
It provides 2 functionalities and is accessed by the user interface module:

- `addPlayer(Player p)` which contains the code to add the Player into the repository
- `formTeam()` which contains the code to form the team using the 4-3-3-1 combination

The following exceptions must be raised when appropriate:

- `TeamNotFormedException` – raised when the required number of players in the formation are not available.

**Note:** Make any other necessary assumptions pertaining to the problem statement wherever required.

**Other guidelines:**

1. Follow the layered approach, create appropriate interfaces and implementation classes as you deem fit
2. Handle any exceptions that are raised and display a user friendly message in the user interface layer.
3. Adhere to the best practices and conventions as you develop the application
4. Ensure to upload your code into the GitHub repository in a timely manner. Any code commits submitted after the assessment end time would not be considered valid.