Fantasy Points IPL

Design a fantasy league score application which has the following services
1. player-service
2. fantasy-league-service

1. Player Service

Below are the four end points in the player-service

i. **POST** /player

Example Input:

```
{
   "name": "Virat Kohli",
   "id" : 1,
   "team": "RCB"
}
```

Expected Result: Store the given input in an in memory data structure / database for later reference. Respond with success / failure accordingly keeping in mind the best practices of HTTP.

ii. **POST** /player/{id}/score

Example Input Request Body:

```
{
  “match”:“1”,
 “runs”: 60,
 “wickets”: 0
}
```

Expected result: Store the given input in a data store/ database along with their scores for later reference

iii. **GET** /players

Returns list of players with names , ids and teams

Example output:


```
{
 "players": [
 {
    "name": "Virat Kohli",
    "id": 1,
    "team": "RCB"
 },
 {
    "name": "M S Dhoni",
    "id": 2,
    "team": "CSK"
 }
 ]
}
```


iv. **GET** /players/scores

Example output:


```
{
   "playerscores": [
     {
        "id": 1,
        "scores": [
           {
              "match": 1,
              "runs": 20,
              "wickets": 0
           },
           {
              "match": 2,
              "runs": 12,
              "wickets": 1
           }
        ]
     },
     {
        "id": 2,
        "scores": [
           {
              "match": 2,
```

```
            "runs": 43,
            "wickets": 0
        },
        {
            "match": 2,
            "runs": 20,
            "wickets": 1
        }
    ]
  }
 ]
}
```


2. Fantasy League Service

fantasy-league-service has to fetch the details and scores from the player-service - This is a must

Fantasy League service has the following endpoints

i. **GET** /cap-holders

Prints the players with maximum wickets and maximum runs

purpleCap : Player with maximum wickets
orangeCap : Player with maximum runs

* If there is a clash in terms of maximum runs or wickets print any one of the players

Example Output:


```
{
  "purpleCap" : "Yuzvendra Chahal",
  "orangeCap" : "Jos Butler"
}
```


ii. **GET** /fantasy-scores

Fantasy scores are calculated according to the following rules:

1. Each wicket taken fetches 10 points
2. If a player has taken more than 5 wickets in a match he gets extra 50 points i.e if a player has taken 6 wickets he gets 60 + extra 50 = 110 points
3. Runs greater than 30 in one match fetches 20 points
4. Runs greater than 50 in a match fetches 50 points , if someone scores a 50 , he gets 20 + 50 = 70 points

5. Each 100 runs scored in a match fetches 100 points , if someone scores a 100 , he gets 20 + 50 + 100 = 170 points

6. Scores below 30 fetch no points

Example output:

Fetch the scores from player-service and calculate the results according to above rules and display the scores of players in the following format in the descending order of their fantasyScores


```
{
 "fantasyScores":[
  {
   "name" : "Jos Butler",
   "fantasyScore" : 580
  },
  {
   "name": "Yuzvendra Chahal",
   "fantasyScore":  430
  }
 ]
}
```


Please make reasonable assumptions in case of doubts and document your assumption

Develop the application in Golang with the below things in mind.

- Follow all the clean code principles
- Write Automated Unit Test cases
- Upload both the projects in separate repositories in Github and follow the commit guidelines as and when you finish a part of the code.
- Dockerize the application (nice to have)
- Implement Logging & Authentication wherever applicable
- Wherever there are questions make a reasonable assumption on the requirements and comment in the source code as to what assumption you are making.