Final project

Football league

1) General idea of the project:

It is about a database of a league where you can handle the teams.

At the beginning you have the details of a given league or a team in a file and using that file you will control the data there.

All teams are ranked in the league from the first team to the last one based on points or based on votes.

You can add a team to a league or delete a team from a league.

You can search for a team with the number of points.

Other functions may be added later.

There are two main users for our system: the first is a normal user and the second is the admin.

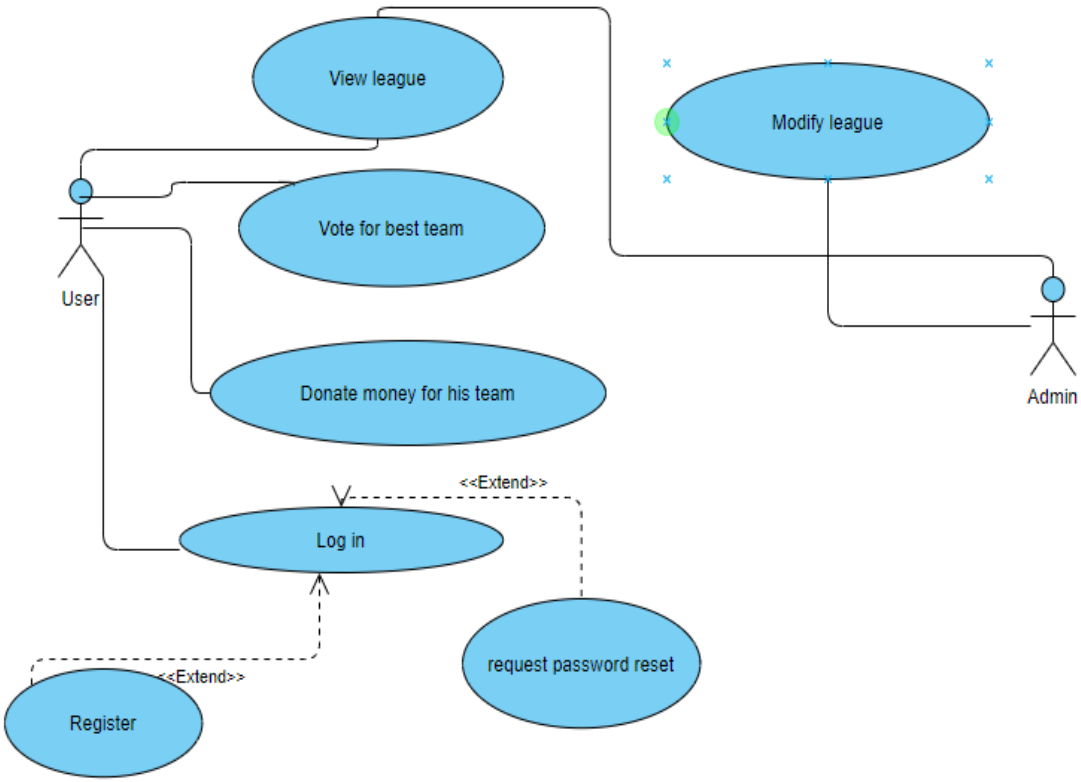
The normal user must sign in for first time in order to access the information of the league. He can reset his password as well

The normal user can only view the information of the league. Vote for the best team , donate money for the team he wants. He can also see the teams ranked by points or ranked by votes.

The admin as well must sign in for first time in order to access the information of the league. He can reset his password as well .

The admin can modify the information in the league. For example, add a team or delete one. update all the information of teams.

2) Use case diagram:



2) Description:

Title	View league
Actor	User
Description	The player can see all the information about the league, teams ,points and the rank

Title	Vote for best team
Actor	User
Description	The player can vote for his best team. The team which gets the most votes is the best team

Title	Donate money for his team

Actor	User
Description	The player can donate money for his team

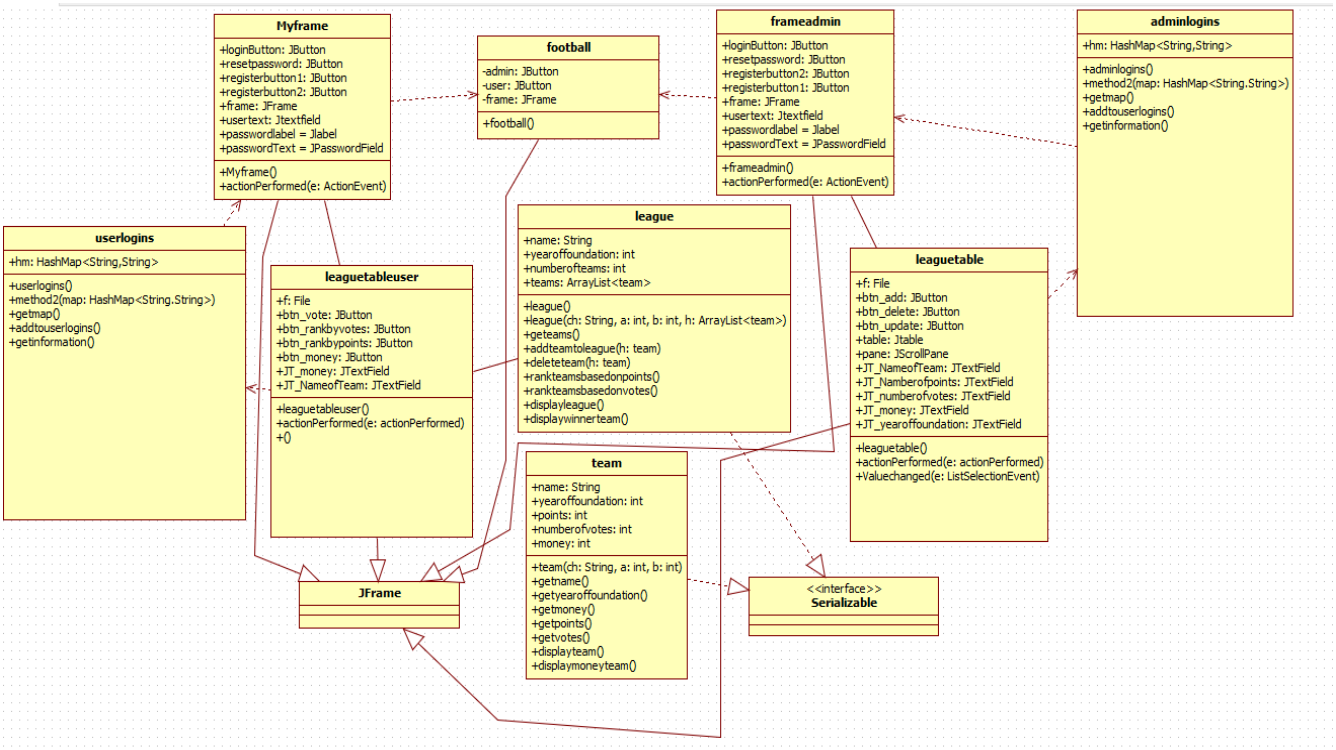
Title	Log in
Actor	User
Description	The player can log in: 1) If hasn't an account, he must register 2) If he forget is password he can reset it.

Title	Log in
Actor	Admin
Description	The admin can log in: 1) If hasn't an account, he must register 2) If he forget his password he can reset it.
Title	View league
Actor	Admin
Description	The admin can see all the information about the league ,teams ,points and the rank

Title	Modify league
Actor	Admin

Description	The player can modify all the information about the league ,teams ,points and the rank.

Class Diagram:

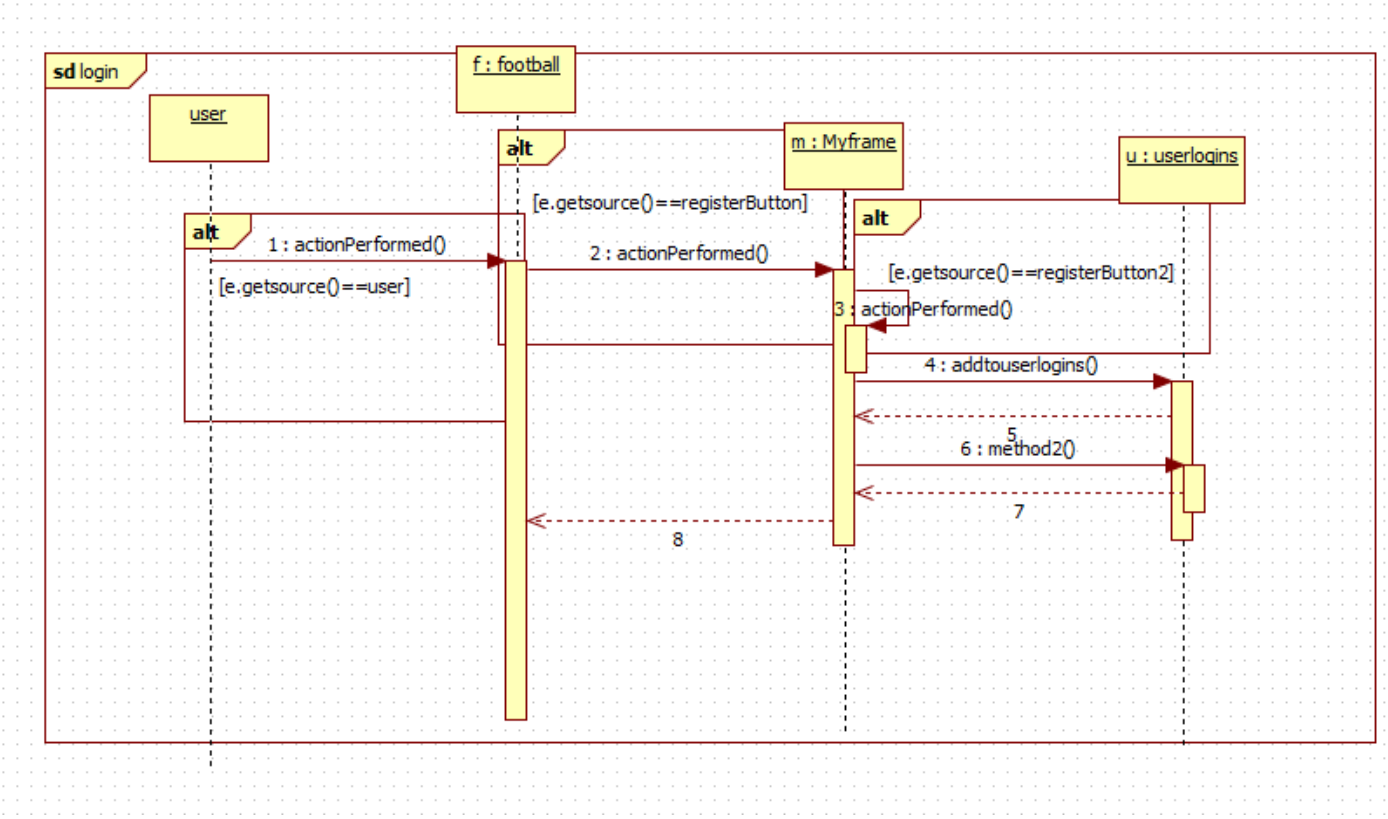


Sequence Diagrams :

1/Login for a normal user:

The user should press on user first then press on register button then put his username and password and press on registerbutton2 and then we call the function addusertologins in order to add the information to our userlogins and save them .

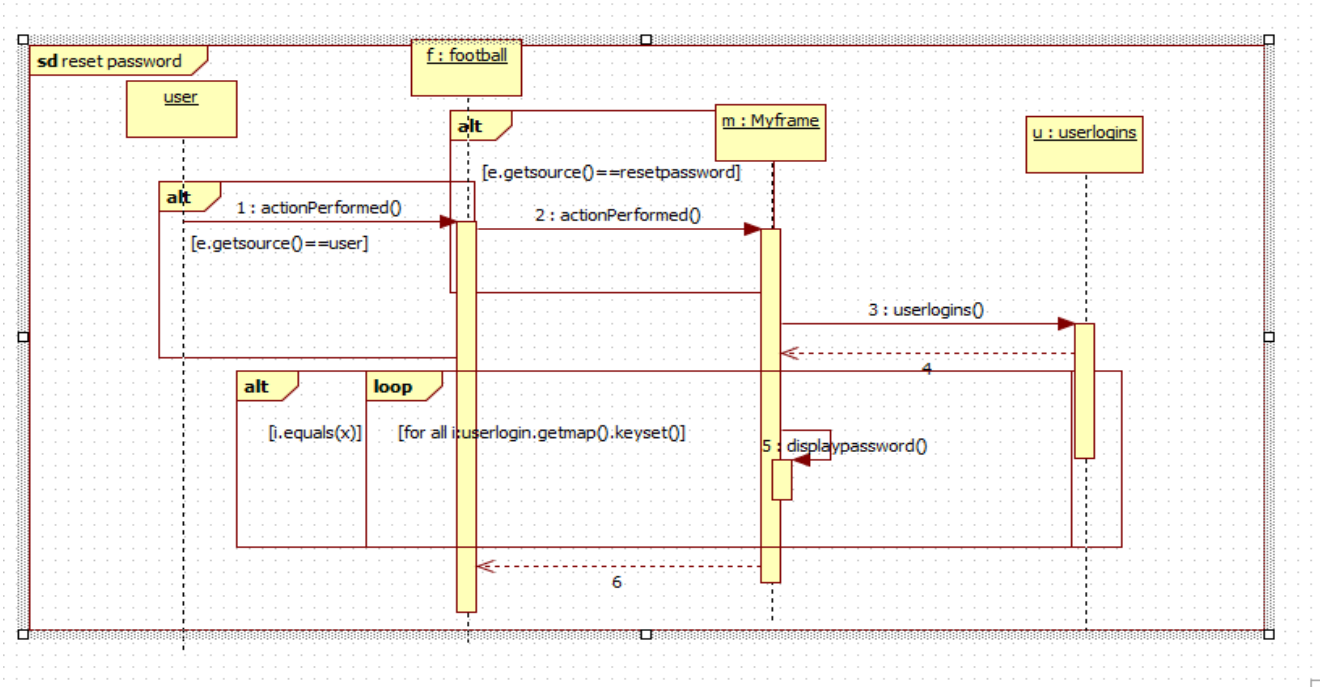
The same thing is for admin.



2/reset password for a normal user:

The user should press on user first then write his username and select resetpassword button which will call the userlogins() where we find the credentials of all users then we need to check whether the name of the user is there or no . if it is there we display its password .

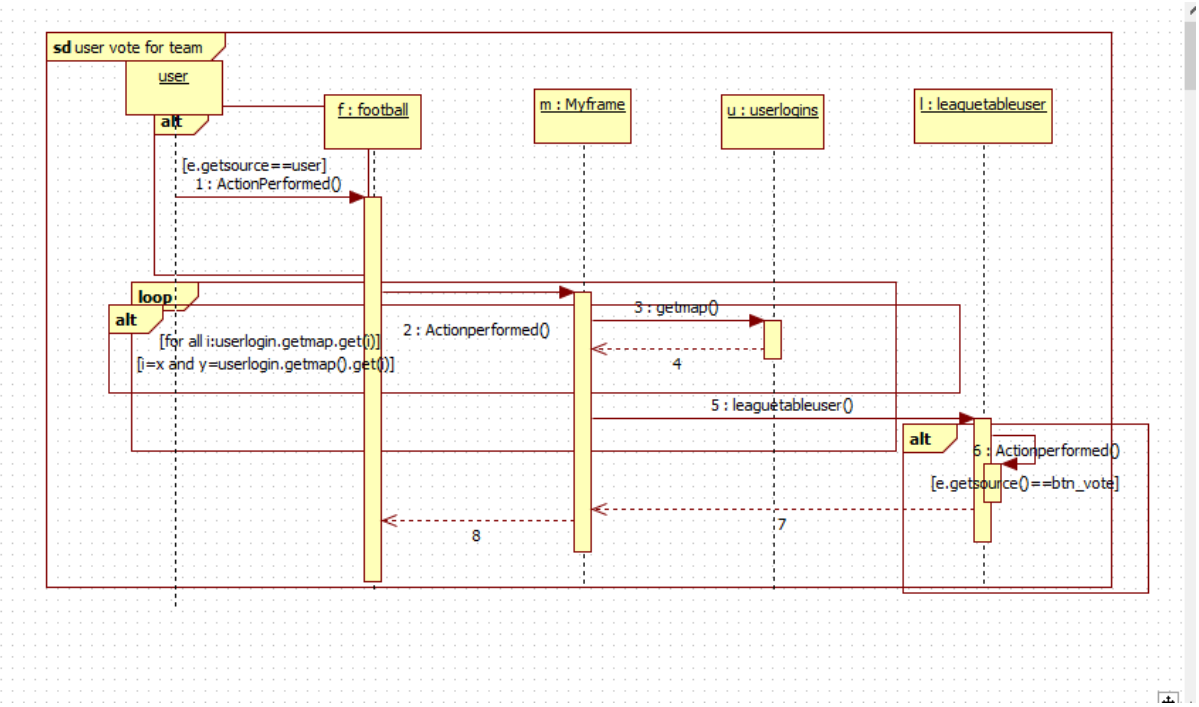
The same thing is done for the admin.



3/A user vote for his team:

The user should first log in then . we have to call userlogins() where there is the credentials of our users and compare a username and his password the credentials which we have in our file . if there is a match so log in was done successfully then a user just vote for his team.

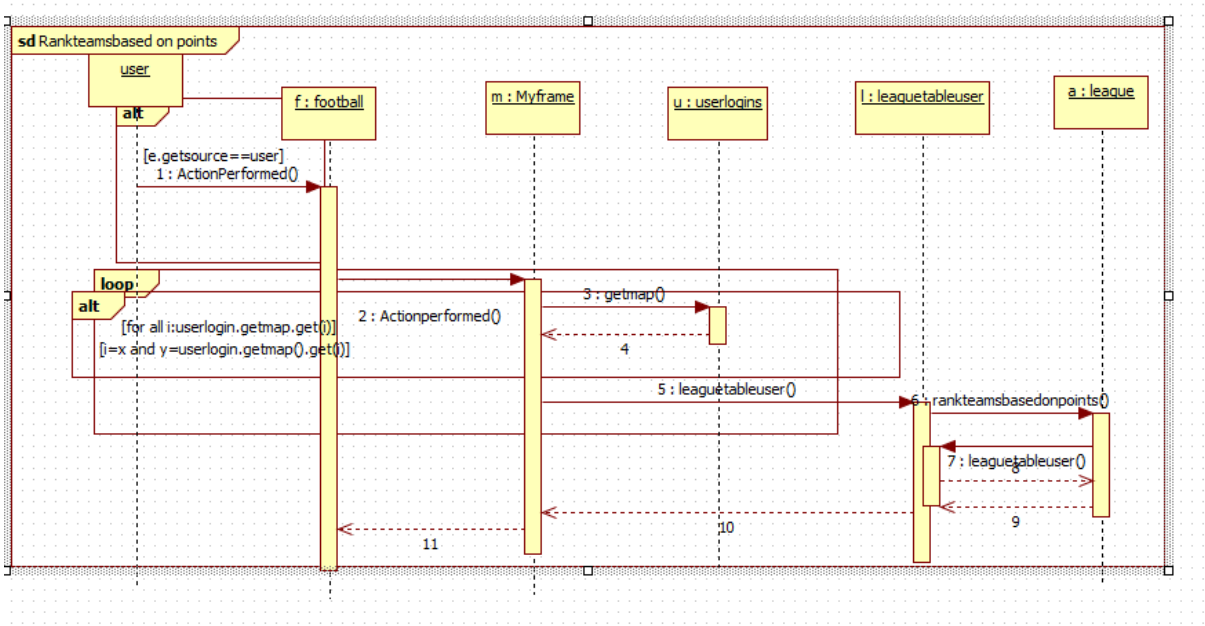
The same procedure is for donating money



4/A user rank the teams based on the number of points:

The user log in successfully then he press on the button rankteamsbypoints this button will call the function rankteams bypoints from the class league and it will rank the teams and after that we call the function leaguetableuser() which will display the teams .

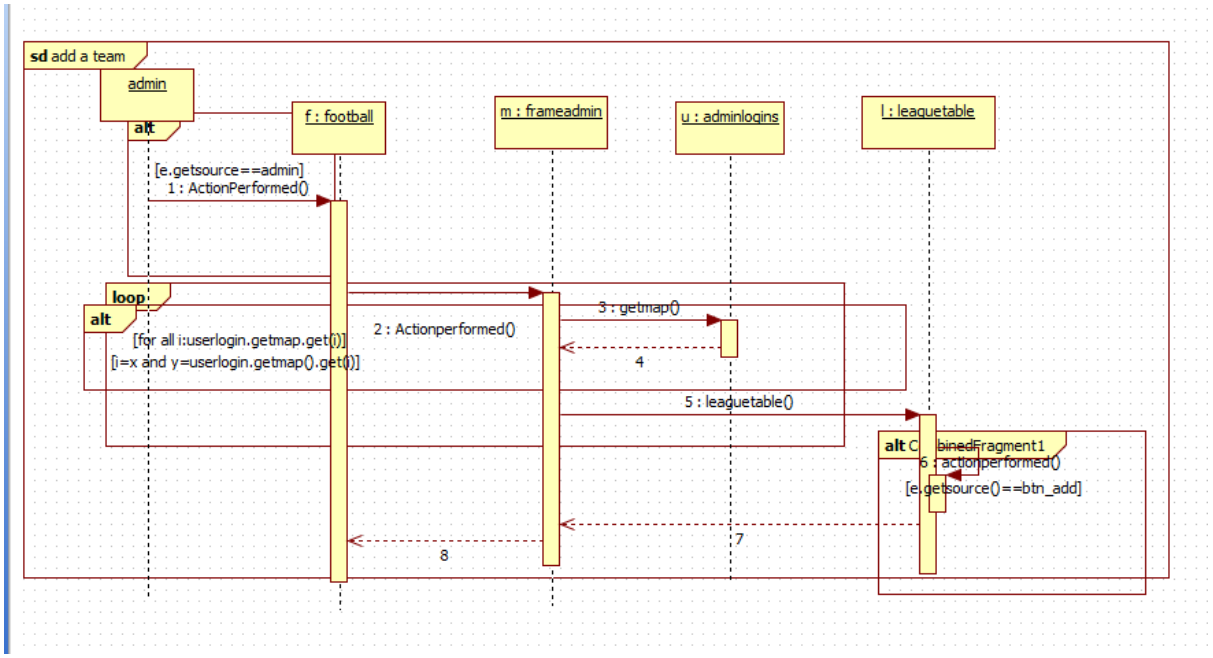
The same procedure is for rankbyvotes.



5/an admin add a team to the league:

The admin log in successfully then he fill the information of the teams and press on the button btn_add which will add the teams to the file and display it .

The same procedure is for modify and delete.



Unit tests:

1/test on adding a team to the league:

```
public class leagueTest {

    @Test
    public void testAddteamtoleague() {
         team t1= new team("oussa",12,6,20,300);
            team t2= new team("lili",20,12,30,2000);
            team t3= new team("lili",9,3,25,4100);
            ArrayList<team> teams = new ArrayList<team>();


        league l=new league("spain",1930,3,teams) ;
          l.addteamtoleague(t1);
          l.addteamtoleague(t2);
          l.addteamtoleague(t3);

            assertEquals(t1, l.getteams().get(0));
            assertEquals(t2,  l.getteams().get(1));
            assertEquals(t3,  l.getteams().get(2));


            assertEquals("second attempt", t1, l.getteams().get(0));   // make sure I can get them a second time
            assertEquals("second attempt", t3, l.getteams().get(2));
    }

    @Test
```

2/test on deleting a team from the league:

```
@Test
public void testDeleteteam() {

     team t1= new team("oussa",12,6,20,300);
        team t2= new team("lili",20,12,30,2000);
        team t3= new team("lili",9,3,25,4100);
        ArrayList<team> teams = new ArrayList<team>();


    league l=new league("spain",1930,0,teams) ;
      l.addteamtoleague(t1);
      l.addteamtoleague(t2);
      l.addteamtoleague(t3);

      assertEquals(l.numberofteams, l.getteams().size());
        l.deleteteam(t3);

        l.deleteteam(t2);
        assertEquals(l.numberofteams, l.getteams().size()) ;


        assertEquals(l.numberofteams, l.getteams().size()) ;// make sure I can get them a second time

}
```

2/test on ranking a team based on number of points:

```
@Test
public void testRankteamsbasedonpoints() {
     team t1= new team("oussa",12,6,20,300);
        team t2= new team("lili",20,12,30,2000);
        team t3= new team("lili",9,3,25,4100);
        ArrayList<team> teams = new ArrayList<team>();


    league l=new league("spain",1930,3,teams) ;
      l.addteamtoleague(t1);
      l.addteamtoleague(t2);
      l.addteamtoleague(t3);
      l.rankteamsbasedonpoints();
      assertEquals(t2, l.getteams().get(0));

        assertEquals(t1, l.getteams().get(1)) ;


        assertEquals(t3, l.getteams().get(2)) ;

}
```

4/test on ranking a team based on number of votes:

```
@Test
public void testRankteamsbasedonvotes() {
     team t1= new team("oussa",12,6,20,300);
        team t2= new team("lili",20,12,30,2000);
        team t3= new team("lili",9,3,25,4100);
        ArrayList<team> teams = new ArrayList<team>();


    league l=new league("spain",1930,3,teams) ;
      l.addteamtoleague(t1);
      l.addteamtoleague(t2);
      l.addteamtoleague(t3);
      l.rankteamsbasedonvotes();
      assertEquals(t2, l.getteams().get(0));

        assertEquals(t3, l.getteams().get(1)) ;


        assertEquals(t1, l.getteams().get(2)) ;

}
```

The tests were all successful