

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using System.Data;
7 using System.Data.OleDb;
8 using System.Security.Cryptography;
9
10 namespace DAL
11 {
12     public static class userFunction
13     {
14         public static void AddUser(string username, string password)
15         {
16             string com = "insert into [users] ([username],[password],[isAdmin])
17                 VALUES ('" + username+"', '"+ AesCryp.encrypt(password)+"', 'n')";
18             OleDbHelper.Execute(com);
19         }
20         public static bool isAdmin(string username)
21         {
22             string com = "SELECT [isAdmin] FROM [users] where [username] = '" +
23                 username + "'";
24             DataTable dt = OleDbHelper.GetTable(com);
25             if (dt.Rows[0].ItemArray[0].ToString() == "y")
26                 return true;
27             return false;
28         }
29         public static void toggleAdmin(string username)
30         {
31             string com = "SELECT [isAdmin] FROM [users] where [username] = '" +
32                 username + "'";
33             DataTable dt = OleDbHelper.GetTable(com);
34             if (dt.Rows[0].ItemArray[0].ToString() == "y")
35             {
36                 string com2 = $"update [users] set [isAdmin]='n' where [username]
37                     =' {username}'";
38                 OleDbHelper.Execute(com2);
39             }
40             else
41             {
42                 string com2 = $"update [users] set [isAdmin]='y' where [username]
43                     =' {username}'";
44                 OleDbHelper.Execute(com2);
45             }
46         }
47         public static bool checkPassword(string username, string password)
48         {
49             string com = "SELECT [password] FROM [users] where [username] =
50                 '" + username + "'";
51             DataTable dt = OleDbHelper.GetTable(com);
52             string pass = dt.Rows[0].ItemArray[0].ToString();
53         }
54     }
55 }
```

```
47         if (AesCryp.Decrypt(pass) == password)
48             return true;
49         return false;
50     }
51     public static bool isUsername(string username)
52     {
53         string com = "SELECT * FROM [users] where [username] = '"+username
54             +"'";
55         DataTable dt = oledbhelper.GetTable(com);
56         if (dt.Rows.Count == 0)
57             return false;
58         else
59             return true;
60     }
61     public static void UpdatePassword(string username, string password)
62     {
63         string com = $"update [users] set [password]='{password}' where
64             [username]='{username}'";
65         oledbhelper.Execute(com);
66     }
67     public static void Deleteuser(string username)
68     {
69         string com = $"DELETE FROM [users] WHERE [username] = '{username}'";
70         oledbhelper.Execute(com);
71     }
72     public static string GetUsers()
73     {
74         string sql = "Select * from [users]";
75         return oledbhelper.printDataTable(sql);
76     }
77 }
78
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL
8 {
9     public static class predictor
10     {
11         public static void predict (List<prediction> ls)
12         {
13             foreach (prediction pred in ls)
14             {
15                 int hplay = FPLFunctions.getNumHPlayByTeamId(1);
16                 int hGF = FPLFunctions.getGFHByTeamId(1);
17                 int hGA = FPLFunctions.getGAHByTeamId(1);
18                 float hAttStrength = hGF / hplay;
19                 Console.WriteLine(hAttStrength);
20             }
21         }
22     }
23 }
24
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL
8 {
9     public class prediction
10    {
11        public int gameID { get; set; }
12        public int hteam { get; set; }
13        public int ateam { get; set; }
14
15        public int hwin { get; set; }
16        public int draw { get; set; }
17        public int awin { get; set; }
18
19        public prediction(int gameID ,int hteam, int ateam)
20        {
21            this.gameID = gameID;
22            this.hteam = hteam;
23            this.ateam = ateam;
24        }
25    }
26 }
27
```

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace DAL
9  {
10     public static class PackFunctions
11     {
12         public static string[] getByPackId(int Id)
13         {
14             string com = "SELECT * FROM [pack] where [packID] = " + Id;
15             DataTable dt = oledbhelper.GetTable(com);
16             int itmLength = dt.Rows[0].ItemArray.Length;
17             string[] mrow = new string[itmLength - 1];
18             for (int i = 1; i < itmLength; i++)
19             {
20                 mrow[i - 1] = dt.Rows[0].ItemArray[i].ToString();
21             }
22             return mrow;
23         }
24         public static void addPack(string username, int packID)
25         {
26             if (packID == 0)
27                 packID = 1;
28             string com = "insert into [packinventory] ([username],[PackID]) VALUES ⚡
29                 ('' + username + '' , " + packID + ")";
30             oledbhelper.Execute(com);
31         }
32         public static List<int> packsByUsername(string username)
33         {
34             List<int> ret = new List<int>();
35             string com = "SELECT * FROM [packinventory] where [username] = '' + ⚡
36                 username+''";
37             DataTable dt = oledbhelper.GetTable(com);
38             foreach (DataRow dr in dt.Rows)
39             {
40                 ret.Add(Convert.ToInt32(dr.ItemArray[2]));
41             }
42             return ret;
43         }
44         public static void deletePack(string username, int packID)
45         {
46             string com = "SELECT * FROM [packinventory] where [username]='' + ⚡
47                 username + '' AND [PackID] = " + packID + "";
48             string com2 = "DELETE * FROM [packinventory] where [ID] =" ⚡
49                 +Convert.ToInt32(oledbhelper.GetTable(com).Rows[0].ItemArray[0]);
50             oledbhelper.Execute(com2);
51         }
52     }
53 }

```

49 }

50

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Data;
6 using System.Data.OleDb;
7 using System.Web;
8
9 public static class oledbhelper
10 {
11     static OleDbConnection cn = new OleDbConnection(ConnectionString);
12
13     public static string ConnectionString
14     {
15         get
16         {
17             string path = @"C:\DB\football.accdb";
18             return @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + path;
19         }
20     }
21
22     //ושאילתות עדכון, מחיקה והוספת רשומות Database- שאילתות עדכון מבנה ה.
23
24     public static void Execute(string com)
25     {
26         //יצירת אובייקט מסוג Connection
27         //OleDbConnection cn = new OleDbConnection(ConnectionString);
28         //cn.Open();
29
30         if (cn.State != ConnectionState.Open)
31         {
32             cn.Open();
33         }
34         // יצירת אובייקט מסוג command
35         OleDbCommand command = new OleDbCommand();
36         command.Connection = cn;
37         command.CommandText = com;
38
39         try
40         {
41             command.ExecuteNonQuery();
42         }
43         catch (Exception)
44         {
45             throw;
46         }
47     }
48
49     public static DataTable GetTable(string com)
50     {
51         //יצירת אובייקט מסוג Connection
52         OleDbConnection cn = new OleDbConnection(ConnectionString);
```

```

53      // יצירת אובייקט מסוג command
54      OleDbCommand command = new OleDbCommand();
55      command.Connection = cn;
56      command.CommandText = com;
57
58      // יצירת אובייקט מסוג דטהסט - אוסף טבלאות בזיכרון
59      DataTable dt = new DataTable();
60      dt.TableName = "tbl";
61
62      // יצירת אובייקט אדפטר מטרתו לתאם בין הדטהסט לדטהבייס
63      OleDbDataAdapter adapter = new OleDbDataAdapter(command);
64
65      cn.Open();
66
67      try
68      {
69          // הפעולה פותחת את הדטהבייס ומחזירה את כל הנתונים לתוך טבלה חדשה בדטהסט
70
71          adapter.Fill(dt);
72      }
73      catch (Exception e)
74      {
75          throw;
76      }
77      finally
78      {
79          cn.Close();
80      }
81
82      return dt;
83  }
84  public static string printDataTable(string sql)
85  {
86      // הפעולה המקבלת מסד נתונים ושאייתה
87      // הפעולה מחזירה מחרזות השומרת את הטבלה בתור HTML
88      DataTable dt = GetTable(sql);
89
90      string printStr = "<table border='1'>";
91      printStr += "<tr><td></td><td name='test'>name</td><td>Password</td><td>is admin</td> </tr>";
92      foreach (DataRow row in dt.Rows)
93      {
94          printStr += "<tr>";
95          printStr += "<td><input type='radio' name='key' value='" + row.ItemArray[0] + "'>";
96          foreach (object myItemArray in row.ItemArray)
97          {
98              printStr += "<td>" + myItemArray.ToString() + "</td>";
99          }
100          printStr += "</tr>";
101      }

```



```
102     printStr += "</table>";
103
104     return printStr;
105 }
106 }
```

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using DAL.apiClasses;
7  using DAL;
8  using System.Web.UI.WebControls;
9
10 namespace DAL
11 {
12     public static class GlobalFunctions
13     {
14         public static string createCard(Card player, clubColour clr, Element els)
15         {
16             //create card
17             string card = "<div class='card'>";
18             card += "<div class='card__inner'>";
19
20
21
22             //front
23             card += "<div class='card__face card__face--front' style='background-
24                 color:' + clr.mcolour + "'>";
25
26             //create Rating text
27             card += "<div class='rating'><p>" + player.rating + "</p></div>";
28
29             //add gradient
30             card += "<div class='gradient' style='background: linear-gradient
31                 (0deg, " + clr.mcolour + " 0%, " + clr.mcolour + "CC 60%, " +
32                 clr.mcolour + "00 100%);'></div>";
33
34             //add player img
35             card += "<img class='player' src='" + player.img + "'>";
36
37             //add badge
38             card += "<div class='badgecont'>";
39             card += "<img class='Cbadge' src='images/badges/" + player.club +
40                 ".png'>";
41             card += "</div>";
42
43             //add name
44             card += "<div class='namecont'>";
45             string[] namesplit = player.name.Split(' ');
46             try { card += "<p class='fname'>" + namesplit[namesplit.Length - 2] +
47                 "</p>"; } catch { card += "<p class='fname' style='visibility:
48                 hidden;'>aa</p>"; }
49             card += "<p class='lname' style='color:' + clr.scolour + "'>" +
50                 namesplit[namesplit.Length - 1] + "</p>";
51             card += "</div>";
52

```

```

46
47         card += "</div>";
48
49
50         //back of card
51         card += "<div class='card_face card_face--back' style='background-  ↗
52             color:' + clr.mcolour + ">";
53         card += "<div class='mins'><p style='color:' + clr.scolour +  ↗
54             '>Minutes: " + els.minutes + "</p></div>";
55         if (player.pos != "Goalkeeper")
56         {
57             card += "<div class='goals'><p style='color:' + clr.scolour +  ↗
58                 '>Goals scored: " + els.goals_scored + "</p></div>";
59             card += "<div class='assits'><p style='color:' + clr.scolour +  ↗
60                 '>Assits: " + els.assists + "</p></div>";
61         }
62         else
63         {
64             card += "<div class='saves'><p style='color:' + clr.scolour +  ↗
65                 '>saves: " + els.saves + "</p></div>";
66             card += "<div class='cleansheet'><p style='color:' + clr.scolour  ↗
67                 + '>clean sheet: " + els.clean_sheets + "</p></div>";
68             card += "<div class='yellow'><p style='color:' + clr.scolour +  ↗
69                 '>yellow cards: " + els.yellow_cards + "</p></div>";
70             card += "<div class='red'><p style='color:' + clr.scolour +  ↗
71                 '>red cards: " + els.red_cards + "</p></div>";
72         }
73         card += "</div>";
74
75         //end
76         card += "</div>";
77         card += "</div>";
78
79
80         return card;
81     }
82     public static string createClubPrec(string[] a, int curnum, clubColour  ↗
83         clr)
84     {
85         string stylecalc = "stroke-dashoffset:calc(440 - (440 * "+ (100 *  ↗
86             curnum)/ Convert.ToInt32(a[1]) + ") / 100); stroke:" + clr.mcolour;
87
88         string club = "<div class='precent'>";
89         club += "<svg>";
90         club += "<circle style='" + stylecalc + "' cx='70' cy='70' r='70'></  ↗
91             circle>";
92         club += "<circle style='" + stylecalc + "' cx='70' cy='70' r='70'></  ↗

```

```

        circle>";
127     club += "</svg>";
128     club += "<div class='clubbdg'>";
129     club+= "<img src='images/badges/' + a[0] + ".png'>";
130     club += "</div>";
131     club += "</div>";
132     club += "<h2>" + (100 * curnum) / Convert.ToInt32(a[1]) + "%</h2>";
133
134     return club;
135 }
136 public static string CreateGame(Root game, Dictionary<int,string> clubs)
137 {
138     //creating vairables
139     string homeT = clubs[game.team_h];
140     string awayT = clubs[game.team_a];
141
142     DateTime time = game.kickoff_time ?? DateTime.Now.AddYears
143         (-19999999);
144
145     int? homeS = 0;
146     if (game.team_h_score != null)
147         homeS = game.team_h_score;
148     int? awayS = 0;
149     if (game.team_a_score != null)
150         awayS = game.team_a_score;
151
152     //create fixture
153     string fixture = "<div class='row'>";
154
155     fixture += "<div class='col - sm'>";
156     fixture += "<img src='images/badges/' + homeT + ".png'>";
157     fixture += "</div>";
158
159     fixture += "<div class='col - sm'>";
160     fixture += "<div class='row'>";
161     fixture += "<div class='col - sm'>";
162     fixture += "<p>" + homeT + "</p>";
163     fixture += "</div>";
164     fixture += "<div class='col - sm'>";
165     if (game.started == true)
166     {
167         fixture += "<p> (" + homeS + " - " + awayS + ") </p>";
168     }
169     else
170     {
171         fixture += "<p> " + time.ToString("dddd, dd MMMM h:mm tt") + " </
172             p>";
173         fixture+= "<asp:Button runat='server' class='btn btn-primary'
174             id='" + game.id + "' Text='bet' OnClick='bet_click' />";

```

```
135     }
136     fixture += "</div>";
137     fixture += "<div class='col - sm'>";
138     fixture += "<p>" + awayT + "</p>";
139     fixture += "</div>";
140     fixture += "</div>";
141     fixture += "</div>";
142
143     fixture += "<div class='col - sm'>";
144     fixture += "<img src='images/badges/' + awayT + ".png'>";
145     fixture += "</div>";
146
147     fixture += "</div>";
148     return fixture;
149 }
150 }
151 }
152
```

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace DAL
9  {
10     public static class GameFunctions
11     {
12         public static List<prediction> byGameweek(int gameweek)
13         {
14             string com = "SELECT * FROM [game] where [gw] = '" + (gameweek) + "'";
15             DataTable dt = oledbhelper.GetTable(com);
16             List<prediction> ls = new List<prediction>();
17             Console.WriteLine(dt.Rows.Count);
18             foreach (DataRow row in dt.Rows)
19             {
20                 ls.Add(new prediction(Convert.ToInt32(row.ItemArray[0].ToString
21                                     ()), Convert.ToInt32(row.ItemArray[1].ToString()),
22                                     Convert.ToInt32(row.ItemArray[2].ToString())));
23             }
24             return ls;
25         }
26         public static void addPrecent(double hteam, double draw, double ateam, int
27         gameID)
28         {
29             string com = "UPDATE game SET [hteamW] = " + Convert.ToInt32(hteam) +
30             ", [draw] = " + Convert.ToInt32(draw) + ", [ateamW] =" +
31             Convert.ToInt32(ateam) + " Where gameID = " + gameID;
32             oledbhelper.Execute(com);
33         }
34         public static int getPrecentbyMatchID(int gameID, int isH)
35         {
36             string com = "SELECT * FROM [game] where [gameID] = " + gameID;
37             DataTable dt = oledbhelper.GetTable(com);
38             int[] ret = new int[3];
39             ret[0] = Convert.ToInt32(dt.Rows[0].ItemArray[7]);
40             ret[1] = Convert.ToInt32(dt.Rows[0].ItemArray[8]);
41             ret[2] = Convert.ToInt32(dt.Rows[0].ItemArray[9]);
42             return ret[isH];
43         }
44     }
45 }

```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Data;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace DAL
9 {
10     public class FPLFunctions
11     {
12         public static string getByClubID(int ID)
13         {
14             string com = "SELECT * FROM [Clubs] where [ID] = "+ID;
15             DataTable dt = oledbhelper.GetTable(com);
16             string ret = dt.Rows[0].ItemArray[1].ToString();
17             return ret;
18         }
19         public static Dictionary<int, string> getdicOfClubs()
20         {
21             Dictionary<int, string> clubs = new Dictionary<int, string>();
22             string com = "SELECT * FROM [Clubs]";
23             DataTable dt = oledbhelper.GetTable(com);
24             foreach (DataRow dr in dt.Rows)
25             {
26                 clubs.Add(Convert.ToInt32(dr.ItemArray[0].ToString()),
27                             dr.ItemArray[1].ToString());
28             }
29             return clubs;
30         }
31         public static string getByCardID(int ID)
32         {
33             string com = "SELECT * FROM [card] where [CardID] = " + ID;
34             DataTable dt = oledbhelper.GetTable(com);
35             return dt.Rows[0].ItemArray[1].ToString();
36         }
37         public static int getGFHByTeamId(int ID)
38         {
39             string com = "SELECT homes FROM [game] where [hteam] = '" + ID + "'";
40             DataTable dt = oledbhelper.GetTable(com);
41             int total = 0;
42             foreach (DataRow dr in dt.Rows)
43             {
44                 if (dr.ItemArray[0].ToString() != "")
45                 {
46                     total += Convert.ToInt32(dr.ItemArray[0].ToString());
47                 }
48             }
49             return total;
50         }
51         public static int getGFABByTeamId(int ID)
52         {
```

```
52         string com = "SELECT ascore FROM [game] where [ateam] = '" + ID +  
53             """;  
54         DataTable dt = oledbhelper.GetTable(com);  
55         int total = 0;  
56         foreach (DataRow dr in dt.Rows)  
57         {  
58             if (dr.ItemArray[0].ToString() != "")  
59             {  
60                 total += Convert.ToInt32(dr.ItemArray[0].ToString());  
61             }  
62         }  
63         return total;  
64     }  
65     public static int getGAHByTeamId(int ID)  
66     {  
67         string com = "SELECT ascore FROM [game] where [hteam] = '" + ID +  
68             """;  
69         DataTable dt = oledbhelper.GetTable(com);  
70         int total = 0;  
71         foreach (DataRow dr in dt.Rows)  
72         {  
73             if (dr.ItemArray[0].ToString() != "")  
74             {  
75                 total += Convert.ToInt32(dr.ItemArray[0].ToString());  
76             }  
77         }  
78         return total;  
79     }  
80     public static int getGAAByTeamId(int ID)  
81     {  
82         string com = "SELECT homes FROM [game] where [ateam] = '" + ID + """;  
83         DataTable dt = oledbhelper.GetTable(com);  
84         int total = 0;  
85         foreach (DataRow dr in dt.Rows)  
86         {  
87             if (dr.ItemArray[0].ToString() != "")  
88             {  
89                 total += Convert.ToInt32(dr.ItemArray[0].ToString());  
90             }  
91         }  
92         return total;  
93     }  
94     public static int getNumHPlayByTeamId(int ID)  
95     {  
96         string com = "SELECT homes FROM [game] where [hteam] = '" + ID + """;  
97         DataTable dt = oledbhelper.GetTable(com);  
98         int total = 0;  
99         foreach (DataRow dr in dt.Rows)  
100         {  
101             if (dr.ItemArray[0].ToString() != "")  
102             {  
103                 total++;  
104             }  
105         }  
106         return total;  
107     }  
108 }
```



```
102     }
103     }
104     return total;
105 }
106 public static int getNumAPlayByTeamId(int ID)
107 {
108     string com = "SELECT ascore FROM [game] where [ateam] = '" + ID +
109         """";
110     DataTable dt = oledbhelper.GetTable(com);
111     return dt.Rows.Count;
112 }
113 public static int getallH()
114 {
115     string com = "SELECT homes FROM [game]";
116     DataTable dt = oledbhelper.GetTable(com);
117     int total = 0;
118     foreach (DataRow dr in dt.Rows)
119     {
120         if (dr.ItemArray[0].ToString() != "")
121         {
122             total += Convert.ToInt32(dr.ItemArray[0].ToString());
123         }
124     }
125     return total;
126 }
127 public static int getallA()
128 {
129     string com = "SELECT ascore FROM [game]";
130     DataTable dt = oledbhelper.GetTable(com);
131     int total = 0;
132     foreach (DataRow dr in dt.Rows)
133     {
134         if (dr.ItemArray[0].ToString() != "")
135         {
136             total += Convert.ToInt32(dr.ItemArray[0].ToString());
137         }
138     }
139     return total;
140 }
141 public static int numofgames()
142 {
143     string com = "SELECT ascore FROM [game]";
144     DataTable dt = oledbhelper.GetTable(com);
145     return dt.Rows.Count;
146 }
147 public static void numofplayers()
148 {
149     string com = "SELECT name FROM [Clubs]";
150     DataTable dt = oledbhelper.GetTable(com);
151     foreach (DataRow dr in dt.Rows)
152     {
153         string com2 = "SELECT CardID FROM [card] where [club]= '" +
```

```
        dr.ItemArray[0].ToString()+"'";  
153         DataTable dt2 = oledbhelper.GetTable(com2);  
154  
155  
156         string com3 = "UPDATE [Clubs] Set [Oplayers] = '" +  
            dt2.Rows.Count + "' Where [name] = '"+ dr.ItemArray[0].ToString  
            () + "'";  
157         oledbhelper.Execute(com3);  
158     }  
159 }  
160 }  
161 }  
162 }
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL
8 {
9     public class clubColour
10    {
11        public string mcolour { get; set; }
12        public string scolour { get; set; }
13        public clubColour(string mcolour, string scolour)
14        {
15            this.mcolour = mcolour;
16            this.scolour = scolour;
17        }
18    }
19 }
20
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Data;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace DAL
9 {
10     public static class cardInv
11     {
12         public static void Addplayer(string username, int cardId)
13         {
14             string com = "insert into [cardinventory] ([username],[cardID]) VALUES
15                 ('" + username + "'," + cardId + ")";
16             oledbhelper.Execute(com);
17         }
18         public static bool checkDuplicate(string username, int cardId)
19         {
20             string com = "SELECT [cardID] FROM [cardinventory] where [username] =
21                 '" + username + "' AND [cardID] = "+cardId+"";
22             DataTable dt = oledbhelper.GetTable(com);
23             if (dt.Rows.Count <= 0)
24                 return false;
25             return true;
26         }
27         public static string getAllcardId(string username)
28         {
29             string mrow = "";
30             string com = "SELECT [cardID] FROM [cardinventory] where [username] =
31                 '" + username + "'";
32             DataTable dt = oledbhelper.GetTable(com);
33             int itmLength = dt.Rows.Count;
34             for (int i = 0; i < itmLength; i++)
35             {
36                 mrow += dt.Rows[i].ItemArray[0].ToString();
37                 if (i != itmLength - 1)
38                     mrow += ",";
39             }
40             return mrow;
41         }
42     }
43 }
```

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace DAL
9  {
10     public class CardFunctions
11     {
12         public static List<Card> getByClub(string club)
13         {
14             List<Card> ret = new List<Card>();
15             string com = "SELECT * FROM [card] where [club] = '" + club + "'";
16             DataTable dt = oledbhelper.GetTable(com);
17             foreach (DataRow dr in dt.Rows)
18             {
19                 Card c = new Card(Convert.ToInt32(dr.ItemArray[0].ToString()),
20                                     dr.ItemArray[1].ToString(), dr.ItemArray[2].ToString(),
21                                     dr.ItemArray[3].ToString(), dr.ItemArray[4].ToString(),
22                                     Convert.ToInt32(dr.ItemArray[5].ToString()), dr.ItemArray
23                                     [6].ToString(), dr.ItemArray[7].ToString());
24
25                 ret.Add(c);
26             }
27             return ret;
28         }
29         public static List<string> getByClubNames(string club)
30         {
31             List<string> ret = new List<string>();
32             string com = "SELECT * FROM [card] where [club] = '" + club + "'";
33             DataTable dt = oledbhelper.GetTable(com);
34             foreach (DataRow dr in dt.Rows)
35             {
36                 ret.Add(dr.ItemArray[1].ToString());
37             }
38             return ret;
39         }
40         public static List<Card> getByNation(string nation)
41         {
42             List<Card> ret = new List<Card>();
43             string com = "SELECT * FROM [card] where [country] = '" + nation +
44                             "'";
45             DataTable dt = oledbhelper.GetTable(com);
46             foreach (DataRow dr in dt.Rows)
47             {
48                 Card c = new Card(Convert.ToInt32(dr.ItemArray[0].ToString()),
49                                     dr.ItemArray[1].ToString(), dr.ItemArray[2].ToString(),
50                                     dr.ItemArray[3].ToString(), dr.ItemArray[4].ToString(),
51                                     Convert.ToInt32(dr.ItemArray[5].ToString()), dr.ItemArray
52                                     [6].ToString(), dr.ItemArray[7].ToString());

```

```
44
45         ret.Add(c);
46     }
47     return ret;
48 }
49 public static Card getByRating(int ratinglow, int ratinghigh)
50 {
51     string com = "SELECT * FROM [card] where [rating] >= " + ratinglow + "
52         " and [rating] <=" + ratinghigh;
53     DataTable dt = oledbhelper.GetTable(com);
54     Random rnd = new Random();
55     int num = rnd.Next(0, dt.Rows.Count);
56     DataRow dr = dt.Rows[num];
57     int itemNum = dr.ItemArray.Length;
58     Card c = new Card(Convert.ToInt32(dr.ItemArray[0].ToString()),
59         dr.ItemArray[1].ToString(), dr.ItemArray[2].ToString(),
60         dr.ItemArray[3].ToString(), dr.ItemArray[4].ToString(),
61         Convert.ToInt32(dr.ItemArray[5].ToString()), dr.ItemArray
62         [6].ToString(), dr.ItemArray[7].ToString());
63     return c;
64 }
65 public static Card getByCardId(int id)
66 {
67     string com = "SELECT * FROM [card] where [cardID] = " + id;
68     DataTable dt = oledbhelper.GetTable(com);
69     DataRow dr = dt.Rows[0];
70     int itemNum = dr.ItemArray.Length;
71     Card c = new Card(Convert.ToInt32(dr.ItemArray[0].ToString()),
72         dr.ItemArray[1].ToString(), dr.ItemArray[2].ToString(),
73         dr.ItemArray[3].ToString(), dr.ItemArray[4].ToString(),
74         Convert.ToInt32(dr.ItemArray[5].ToString()), dr.ItemArray
75         [6].ToString(), dr.ItemArray[7].ToString());
76     return c;
77 }
78 public static List<Card> getALLByCardId(string Id)
79 {
80     try
81     {
82         string com = "SELECT * FROM [card] where [cardID] IN " + Id;
83         DataTable dt = oledbhelper.GetTable(com);
84         List<Card> lc = new List<Card>();
85         for (int i = 0; i < dt.Rows.Count; i++)
86         {
87             DataRow dr = dt.Rows[i];
88             int itemNum = dr.ItemArray.Length;
89             lc.Add(new Card(Convert.ToInt32(dr.ItemArray[0].ToString()),
90                 dr.ItemArray[1].ToString(), dr.ItemArray[2].ToString(),
91                 dr.ItemArray[3].ToString(), dr.ItemArray[4].ToString(),
92                 Convert.ToInt32(dr.ItemArray[5].ToString()), dr.ItemArray
93                 [6].ToString(), dr.ItemArray[7].ToString()));
94         }
95     }
96     return lc;
97 }
```

```
83         }
84         catch { return new List<Card>(); }
85     }
86     public static Dictionary<string, clubColour> getcolours()
87     {
88         var dic =new Dictionary<string, clubColour>();
89         string com = "SELECT * FROM [Clubs]";
90         DataTable dt = oledbhelper.GetTable(com);
91         foreach (DataRow dr in dt.Rows)
92         {
93             dic.Add(dr.ItemArray[1].ToString(),new clubColour(dr.ItemArray  ➤
94                 [2].ToString(), dr.ItemArray[3].ToString()));
95         }
96         return dic;
97     }
98     public static List<string[]> getClubsTotal()
99     {
100         var li = new List<string[]>();
101         string com = "SELECT * FROM [Clubs]";
102         DataTable dt = oledbhelper.GetTable(com);
103         foreach (DataRow dr in dt.Rows)
104         {
105             li.Add(new string[] {dr.ItemArray[1].ToString(),dr.ItemArray  ➤
106                 [4].ToString() });
107         }
108         return li;
109     }
110 }
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL
8 {
9     public class Card
10    {
11        public int id { get; set; }
12        public string name { get; set; }
13        public string img { get; set; }
14        public string country { get; set; }
15        public string club { get; set; }
16        public int rating { get; set; }
17        public string pos { get; set; }
18        public string type { get; set; }
19        public Card()
20        {
21
22        }
23        public Card(int id, string name, string img, string country, string club,
24                    int rating, string pos, string type)
25        {
26            this.id = id;
27            this.name = name;
28            this.img = img;
29            this.country = country;
30            this.club = club;
31            this.rating = rating;
32            this.pos = pos;
33            this.type = type;
34        }
35    }
36 }
```



```

1  using DAL.apiClases;
2  using System;
3  using System.Collections.Generic;
4  using System.Data;
5  using System.Globalization;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9
10 namespace DAL
11 {
12     public static class BettingFunctions
13     {
14         public static void AddBet(string username, int GameID, string winner,
15             string score, string scorer)
16         {
17             string com = "insert into [bets] ([username],[GameID],[winner],
18                 [score],[scorer],[didClaim]) VALUES ('" + username + "' , " + GameID
19                 + " ,'" + winner + "' ,'" + score + "' ,'" + scorer + "' , 0)";
20             OleDbHelper.Execute(com);
21         }
22         public static bool didBet(string username, int GameID)
23         {
24             string com = "SELECT [GameID] FROM [bets] where [username] = '" +
25                 username + "' AND [GameID] = " + GameID + "'";
26             DataTable dt = OleDbHelper.GetTable(com);
27             if (dt.Rows.Count <= 0)
28                 return false;
29             return true;
30         }
31         public static void claimed(string username, int GameID)
32         {
33             string com = "UPDATE [bets] Set [didClaim] = 1 WHERE [username] = '" +
34                 username + "' AND [GameID]= " + GameID + "'";
35             OleDbHelper.Execute(com);
36         }
37         public static List<Bet> getAllBets(string username)
38         {
39             string com = "SELECT * FROM [bets] where [username] = '" + username +
40                 "'";
41             DataTable dt = OleDbHelper.GetTable(com);
42             int itmLength = dt.Rows.Count;
43             List<Bet> lb = new List<Bet>();
44             for (int i = 0; i < itmLength; i++)
45             {
46                 Bet b = new Bet(dt.Rows[i].ItemArray[2].ToString(), dt.Rows
47                     [i].ItemArray[3].ToString(), dt.Rows[i].ItemArray[4].ToString(),
48                     dt.Rows[i].ItemArray[5].ToString(), Convert.ToInt32(dt.Rows
49                     [i].ItemArray[6]));
50                 lb.Add(b);
51             }
52             return lb;
53         }
54     }
55 }

```

```
44     }
45     public static List<Root> getAllGamestoPage(string gw)
46     {
47         string com = "SELECT * FROM [game] where [gw] = '" + gw + "'";
48         DataTable dt = oledbhelper.GetTable(com);
49         int itmLength = dt.Rows.Count;
50         List<Root> lb = new List<Root>();
51         for (int i = 0; i < itmLength; i++)
52         {
53             //CultureInfo provider = new CultureInfo("en_US");
54             DateTime date = DateTime.ParseExact((dt.Rows[i].ItemArray
55                 [3]).ToString(), "EEEE, dd MMMM h:mm a", null); //provider);
56             Root r = new Root();
57             r.id = Convert.ToInt32((dt.Rows[i].ItemArray[0]).ToString());
58             r.team_h = Convert.ToInt32((dt.Rows[i].ItemArray[1]).ToString());
59             r.team_a = Convert.ToInt32((dt.Rows[i].ItemArray[2]).ToString());
60             lb.Add(r);
61         }
62         return lb;
63     }
64 }
65
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL
8 {
9     public class Bet
10    {
11        public string gameId { get; set; }
12        public string winner { get; set; }
13        public string score { get; set; }
14        public string scorer { get; set; }
15        public bool didClaim { get; set; }
16        public Bet(string gameId, string winner, string score, string scorer, int ?
            didClaim)
17        {
18            this.gameId = gameId;
19            this.winner = winner;
20            this.score = score;
21            this.scorer = scorer;
22            if (didClaim == 0)
23                this.didClaim = false;
24            else
25                this.didClaim = true;
26        }
27    }
28 }
29 }
30 }
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.IO;
4 using System.Linq;
5 using System.Net;
6 using System.Text;
7 using System.Threading.Tasks;
8 using Newtonsoft.Json;
9 using DAL.apiClases;
10
11 namespace DAL
12 {
13     public class APICall
14     {
15         public static List<Root> GetCall()
16         {
17             ServicePointManager.Expect100Continue = true;
18             ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
19             WebRequest request = HttpWebRequest.Create(@"https://
                fantasy.premierleague.com/api/fixtures/");
20             WebResponse response = request.GetResponse();
21             StreamReader reader = new StreamReader(response.GetResponseStream());
22             string football_Jason = reader.ReadToEnd();
23
24             var call = JsonConvert.DeserializeObject<List<Root>>(football_Jason);
25             return call;
26         }
27         public static Dictionary<string, Element> getListOfStats(List<string>
            ids)
28         {
29
30             ServicePointManager.Expect100Continue = true;
31             ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
32             WebRequest request = HttpWebRequest.Create(@"https://
                fantasy.premierleague.com/api/bootstrap-static/");
33             WebResponse response = request.GetResponse();
34             StreamReader reader = new StreamReader(response.GetResponseStream());
35             string football_Jason = reader.ReadToEnd();
36
37             Dictionary<string, Element> dic = new Dictionary<string, Element>();
38
39             var call = JsonConvert.DeserializeObject<BooStat>(football_Jason);
40             foreach (Element eve in call.elements)
41             {
42                 foreach (string id in ids)
43                 {
44                     if (eve.id == Convert.ToInt32(id))
45                     {
46                         dic.Add(id, eve);
47                     }
48                 }
49             }
50         }
51     }
52 }
```

```

50         return dic;
51     }
52     public static int getCurrentGameweek(int offset)
53     {
54         ServicePointManager.Expect100Continue = true;
55         ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
56         WebRequest request = HttpWebRequest.Create(@"https://fantasy.premierleague.com/api/bootstrap-static/");
57         WebResponse response = request.GetResponse();
58         StreamReader reader = new StreamReader(response.GetResponseStream());
59         string football_Jason = reader.ReadToEnd();
60
61         var call = JsonConvert.DeserializeObject<BooStat>(football_Jason);
62         foreach (Event eve in call.events)
63         {
64             if (eve.finished == false)
65             {
66                 if (eve.id is int)
67                 {
68                     return eve.id + offset?? 1;
69                 }
70             }
71         }
72     }
73     return 1;
74 }
75 public static List<Root> sortByNextGameWeek(List<Root> lr, int currGW)
76 {
77     List<Root> ret = new List<Root>();
78     foreach (Root fixture in lr)
79     {
80         if (fixture.@event == currGW)
81         {
82             ret.Add(fixture);
83         }
84     }
85     return ret;
86 }
87 private static bool DateInsideOneWeek(DateTime date1, DateTime date2)
88 {
89     DayOfWeek firstDayOfWeek =
90         System.Globalization.CultureInfo.CurrentCulture.DateTimeFormat.Firs
91         tDayOfWeek;
92     DateTime startDateOfWeek = date1.Date;
93     while (startDateOfWeek.DayOfWeek != firstDayOfWeek)
94     { startDateOfWeek = startDateOfWeek.AddDays(-1d); }
95     DateTime endDateOfWeek = startDateOfWeek.AddDays(6d);
96     return date2 >= startDateOfWeek && date2 <= endDateOfWeek;
97 }
98 public static void addGamesToDB()
99 {
100     #region connecttoDb

```

```

...User\Desktop\footballcards\footballtrading\DAL\APICall.cs 3
99     ServicePointManager.Expect100Continue = true;
100     ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
101     WebRequest request = HttpWebRequest.Create(@"https://fantasy.premierleague.com/api/fixtures/");
102     WebResponse response = request.GetResponse();
103     StreamReader reader = new StreamReader(response.GetResponseStream());
104     string football_Jason = reader.ReadToEnd();
105
106     List<Root> call = JsonConvert.DeserializeObject<List<Root>>(football_Jason);
107     #endregion
108
109     foreach (Root game in call)
110     {
111         string com = "UPDATE game SET [date] = '" + (game.kickoff_time ??
112             DateTime.Now.AddYears(-1000)).ToString("dddd, dd MMMM h:mm tt")
113             + "', homes = '" + game.team_h_score + "', ascore
114             = '" + game.team_a_score + "' Where gameID = " + game.id;
115         OleDbHelper.Execute(com);
116     }
117 }
118 public static void addALLGamesToDB()
119 {
120     //done once to add all games to database
121     #region connecttoDb
122     ServicePointManager.Expect100Continue = true;
123     ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
124     WebRequest request = HttpWebRequest.Create(@"https://fantasy.premierleague.com/api/fixtures/");
125     WebResponse response = request.GetResponse();
126     StreamReader reader = new StreamReader(response.GetResponseStream());
127     string football_Jason = reader.ReadToEnd();
128
129     List<Root> call = JsonConvert.DeserializeObject<List<Root>>(football_Jason);
130     #endregion
131
132     foreach (Root game in call)
133     {
134         string com = "INSERT INTO game (gameID,gw,hteam,ateam,
135             [date],homes,ascore) VALUES('"+game.id+"','"+game.@event
136             + "','"+game.team_h+"','"+game.team_a+"','"+
137             (game.kickoff_time ?? DateTime.Now.AddYears(-1000)).ToString(
138             "dddd, dd MMMM h:mm tt") + "','"+game.team_h_score
139             + "','"+game.team_a_score+"')";
140         OleDbHelper.Execute(com);
141     }
142 }
143 public static int getCurrentGw()
144 {
145     ServicePointManager.Expect100Continue = true;
146     ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;

```

```
139      WebRequest request = HttpWebRequest.Create(@"https://  
        fantasy.premierleague.com/api/bootstrap-static/");  
140      WebResponse response = request.GetResponse();  
141      StreamReader reader = new StreamReader(response.GetResponseStream());  
142      string football_Jason = reader.ReadToEnd();  
143  
144      var call = JsonConvert.DeserializeObject<Root2>(football_Jason);  
145  
146  
147      foreach (Event2 gw in call.events)  
148      {  
149          if ( DateTime.Now.CompareTo(gw.deadline_time) < 0)  
150          {  
151              return gw.id;  
152          }  
153      }  
154      return 1;  
155  }  
156  }  
157 }  
158
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using System.Security.Cryptography;
7
8 namespace DAL
9 {
10     public class AesCryp
11     {
12         public static string IV = @"Xp2s5v8y5BvDlGcK";// 16 chars = 128 bits
13         public static string Key = @"u7xsArDoF6JaNdRgUkXp2s5v8f2B1EaH"; // 32 chars = 256 bits
14
15         public static string encrypt(string decrypted)
16         {
17             AesCryptoServiceProvider aes = new AesCryptoServiceProvider();
18             aes.BlockSize = 128;
19             aes.KeySize = 256;
20             aes.IV = Encoding.UTF8.GetBytes(IV);
21             aes.Key = Encoding.UTF8.GetBytes(Key);
22             aes.Mode = CipherMode.CBC;
23             aes.Padding = PaddingMode.PKCS7;
24
25             // Convert string to byte array
26             byte[] src = Encoding.Unicode.GetBytes(decrypted);
27
28             // encryption
29             using (ICryptoTransform encrypt = aes.CreateEncryptor())
30             {
31                 byte[] dest = encrypt.TransformFinalBlock(src, 0, src.Length);
32
33                 // Convert byte array to Base64 strings
34                 return Convert.ToBase64String(dest);
35             }
36         }
37         public static string Decrypt(string encrypted)
38         {
39             AesCryptoServiceProvider aes = new AesCryptoServiceProvider();
40             aes.BlockSize = 128;
41             aes.KeySize = 256;
42             aes.IV = Encoding.UTF8.GetBytes(IV);
43             aes.Key = Encoding.UTF8.GetBytes(Key);
44             aes.Mode = CipherMode.CBC;
45             aes.Padding = PaddingMode.PKCS7;
46
47             // Convert Base64 strings to byte array
48             byte[] src = System.Convert.FromBase64String(encrypted);
49
50             // decryption
51             using (ICryptoTransform decrypt = aes.CreateDecryptor())
```



```
52         {
53             byte[] dest = decrypt.TransformFinalBlock(src, 0, src.Length);
54             return Encoding.Unicode.GetString(dest);
55         }
56     }
57 }
58 }
59
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL.apiClasses
8 {
9     public class A
10    {
11        public int value { get; set; }
12        public int element { get; set; }
13    }
14
15    public class H
16    {
17        public int value { get; set; }
18        public int element { get; set; }
19    }
20
21    public class Stat
22    {
23        public string identifier { get; set; }
24        public IList<A> a { get; set; }
25        public IList<H> h { get; set; }
26    }
27
28    public class Root
29    {
30        public int code { get; set; }
31        public int? @event { get; set; }
32        public bool finished { get; set; }
33        public bool finished_provisional { get; set; }
34        public int id { get; set; }
35        public DateTime? kickoff_time { get; set; }
36        public int minutes { get; set; }
37        public bool provisional_start_time { get; set; }
38        public bool? started { get; set; }
39        public int team_a { get; set; }
40        public int? team_a_score { get; set; }
41        public int team_h { get; set; }
42        public int? team_h_score { get; set; }
43        public IList<Stat> stats { get; set; }
44        public int team_h_difficulty { get; set; }
45        public int team_a_difficulty { get; set; }
46        public int pulse_id { get; set; }
47    }
48    public class Root2
49    {
50        public List<Event2> events { get; set; }
51    }
52    public class Event2
```

```
53     {
54         public int id { get; set; }
55         public string name { get; set; }
56         public DateTime deadline_time { get; set; }
57         public int average_entry_score { get; set; }
58         public bool finished { get; set; }
59         public bool data_checked { get; set; }
60         public int? highest_scoring_entry { get; set; }
61         public int deadline_time_epoch { get; set; }
62         public int deadline_time_game_offset { get; set; }
63         public int? highest_score { get; set; }
64         public bool is_previous { get; set; }
65         public bool is_current { get; set; }
66         public bool is_next { get; set; }
67         public int? most_selected { get; set; }
68         public int? most_transferred_in { get; set; }
69         public int? top_element { get; set; }
70         public int transfers_made { get; set; }
71         public int? most_captained { get; set; }
72         public int? most_vice_captained { get; set; }
73     }
74 }
75
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace DAL.apiClasses
8 {
9     public class Event
10    {
11        public int? id { get; set; }
12        public string name { get; set; }
13        public DateTime? deadline_time { get; set; }
14        public int? average_entry_score { get; set; }
15        public bool? finished { get; set; }
16        public bool? data_checked { get; set; }
17        public object highest_scoring_entry { get; set; }
18        public int? deadline_time_epoch { get; set; }
19        public int? deadline_time_game_offset { get; set; }
20        public object highest_score { get; set; }
21        public bool? is_previous { get; set; }
22        public bool? is_current { get; set; }
23        public bool? is_next { get; set; }
24        public List<object> chip_plays { get; set; }
25        public object most_selected { get; set; }
26        public object most_transferred_in { get; set; }
27        public object top_element { get; set; }
28        public object top_element_info { get; set; }
29        public int? transfers_made { get; set; }
30        public object most_captained { get; set; }
31        public object most_vice_captained { get; set; }
32    }
33
34    public class GameSettings
35    {
36    }
37
38    public class BooStat
39    {
40        public List<Event> events { get; set; }
41        public GameSettings game_settings { get; set; }
42        public List<object> phases { get; set; }
43        public List<object> teams { get; set; }
44        public int total_players { get; set; }
45        public List<Element> elements { get; set; }
46        public List<object> element_stats { get; set; }
47        public List<object> element_types { get; set; }
48    }
49    public class Element
50    {
51        public int id { get; set; }
52        public int clean_sheets { get; set; }
```

```
53     public int assists { get; set; }
54     public int goals_scored { get; set; }
55     public int minutes { get; set; }
56     public int saves { get; set; }
57     public int red_cards { get; set; }
58     public int yellow_cards { get; set; }
59
60 }
61
62
63 }
64
```