LONDON METROPOLITAN UNIVERSITY

BENG (HONS) SOFTWARE ENGINEERING (TOP UP)

APPLICATION DEVELOPMENT

CS6004ES

GROUP COURSEWORK – (2022/2023)

Team 01: Group Members

LMU ID	STUDENT'S NAME
22015391	W. Rashmi Sharon Fernando
22015392	S. Sandun Lakshan Silva
22015393	W. Samitha Ruwan Lowe

Acknowledgement

First and foremost, we would like to thank the Almighty God, who always guides us to work on the right path in our lives.

Next to him, we would like to express our gratitude to our parents, who always brought us up with their kind cooperation, love, and affection. Not only that, we would like to thank them for their encouragement, which helped us in the completion of this project successfully.

Then we are highly indebted to our lecturer (Mrs. Malsha Setunga), who gave us better knowledge to do this group project (CS6004ES) of Advanced Software Engineering. At last but not least, thanks to our dear loving group members as well.

Table of Content

Table of Table

Table of Figure

Figure 1 - E - Sport LOGO

Abstract

The auction has been one of the most exciting parts of the awards in cricket for many years. Auctions for trophies in cricket are one of the best opportunities for all talented cricketers because they can gain multi-million dollar contracts and a chance to show their talent on what can be named the grandest step in cricket. Many viewers will be fascinated by their televisions as players showcase their cricketing chops and fans in the stands chant or vocalize for their favored team, for example, the Indian Premier League (IPL). IPL is one of the biggest T20 tournaments and is an extravaganza or festival that attracts millions or billions of fans from around the world.

XYZ Pvt Ltd. E-Sport is well known for leading Cricket Player auctions in Sri Lanka. E-Sport Pvt Ltd has provided an outsourced software development project to build a web application system for the Player auction. So that, the web application which is named as E – SPORT was designed.

Through this E-SPORT web application, an auction procedure occurs for a day. Many proceed to happen behind the scenes for months efficiently. Teams complete the pre-auction analysis and perform a particular strategy to be employed during the auction. The remaining budget, the type of player or performance required, the player's international commitment, injury history, his equation with the coach, everything concerns when a franchise selects to bid for a player.

1) Introduction

Cricket is a sport that delights audiences and fans around the world. It is played on the international level and is a global sensation. According to the normal procedure, the list of verified players, who desire to play in the tournament is released before the auction (weeks before the auction). Given their skill and the attribute that causes them to stand out, the different franchises prepare and strategize to sign proper players for their team. Apart from nationwide players who represent their country (that means national players), domestic cricketers are also scouted, which can provide a break to a cricketer's career. Each team is allocated a budget, and within that budget, the franchise must bid one by one for the services of players from a particular type. Players are marked as Capped and Uncapped. A base price is set for each player who signs up for the auction and makes it to the finalized list released. Each franchise owner, coach, and analyst examines and makes an offer for the player they want.

The auctioneer announces the name list, and the teams will start bidding on them. Each cricketer will be auctioned off depending on their base price, with bidding continuing until the auctioneer announces the player sold. If no team requests or bids for a player, they will be considered under the unsold list. The auctioneer will provide teams the option of listing the unsold players they are interested in, and the bidding for those players will be renewed with the base price decreased to half of what it was originally. The players will be considered unsold in the auction if they stay unclaimed for the next time.

"It is a familiar sight for the IPL viewers to look at the team owners, managers, coaches, and at times, analysts sitting at the assigned tables during the multi-day IPL auction going live on television."

Although the auction procedure happens for a couple of days, many things proceed behind the scenes for months. Teams complete the pre-auction estimation and work on the strategy to be employed during the auction. The remaining funding or budget, the type of player required, the player's international obligation, injury record, his equation with the trainer, everything concerns when a franchise decides to bid for a player.

- **Base price**

The base price is the lowest amount, which a team must propose for a player. Before the significant auction, the player chooses a price and submits it to the corresponding company. For instance Board of Control for Cricket in India (BCCI). Along with a no-objection certification from his parent board authorizing him to contest in the tournament. High-profile global capped players frequently establish higher starting prices. The reason is they expect enormous offers. Before fixing the base price, players must evaluate their previous performances, popularity, and social media following, among other factors.

- **Advantages of such auction for a cricket player**

1) **Entertainment Medium for the Masses**

The cricket match gives an adrenaline rush in the body, and one cannot move his/her eyes from the Television. Especially, in the last two overs. Because of the selected players for a team. For example, the twenty20 leagues are short, and thus the excitement gains further. In IPL, teams are divided on the name of Indian states, which creates people to watch it. The reason is that it is a way of showing support and cheering for their state.

2) **Platform to Enhance New Cricketers**

The IPL includes all kinds of cricketers, the retired ones, the recently became cricketers, the currently-popular ones, and the presently unpopular players. Here, the latest cricketers get a chance to perform with experienced cricketers and enhance the quality of their play. They get an opportunity to learn well and shape sufficiently. Through the auction, all players which are mentioned above have a chance to perform.

3) **Adds Revenue to the Economy**

IPL league provides immense employment opportunities. Such as the job of cricket columnists, umpires, outfit designers for all teams, police guards to defend the players, etc. The money obtained by selling the tickets is also a considerable sum. Through the auction, all players can demand their base price. Team owners can bid for them, and consequently, players have a chance to obtain a significant amount which was more than they requested based on price.

4) **Brings the Cricket World Closer**

This is inclusive of cricket players from all over the world. The reason is that it makes it very diverse when players from various countries play together. There will be an amalgamation of culture, values, and learning. So it makes people more open-minded and brings different groups closer.

1.1) About the system

The system which is designed for that scenario is named E-SPORT. This is a web application that is used ASP.NET MVC with C# programming language in Visual Studio 2022. Not only that, Ajax, Bootstrap, jQuery, Entity Frmework.net core 6, and iText are used as technologies. One of the significant advantages of this system over manual procedure is an auction procedure occurs for a day. Many proceed to happen behind the scenes for months efficiently. Teams complete the pre-auction analysis and perform a particular strategy to be employed during the auction. The remaining budget, the type of player or performance required, the player's international commitment, injury history, his equation with the coach, everything concerns when a franchise selects to bid for a player. So, any players, and team leaders should provide the details, which are mentioned above for registration because it is more important to apply for any trophy. These people are registered by the administrative officer. Once registered, they obtain their username and password.

Only the admin has full authority over the entire system. Consequently, all modifications concerning the registrations, and an auction can be done through an administration officer. Only the admin has the authority to create related accounts for the players and team owners, maintain the login systems, and set and manage player-based price, bid price, player auction, and maximum price for each team. The trophy Details and participating team details can be done by the administrative officer only. And also, only the admin can generate various reports such as all players' biodata, team details, auction details, summary, and so on. Other registered people have restrictions for the system. That means players and team owners have limited access. Players can register for any available trophy with their details through the admin, but they can view the result of the auction by login into the system. That means how many team owners bid him and about the bid prices. And also, he can see whether he sold or unsold. If the

particular player is selected, he can view the detail of the team that he was selected. (Who the team owner is, and who other players are in his team.).

Any team owner can view his or her team details and the player's profile and can view to whom he or she voted and what the bid prices that the team owner recommended for each player are using their logins. However, if the team owner wants to update the mentioned bid price for a particular player, it can be done through administration only. Every team can view the status of their team during bidding.

❖ Process design highlights in E-SPORT system

- Players and team leaders are registered by the administrative officer. Once registered, they obtain their username and password. All players should give their facts for registration and applying for a trophy, for example, first name, last name, contact number, etc.
- All modifications concerning the registrations and an auction can be done through an administration officer.
- Only the admin has the authority to create related accounts and maintain the login systems.
- The trophy Details and participating team details can be done by the administrative officer only.
- Players can register for any available trophy with their details through the admin.
- Any team owner can view his or her team details and the player's profile using their logins.
- Only the admin can set and manage player-based price, bid price, player auction, and maximum price for each team.
- Any player can view the result of the auction. That means how many team owners bid him and bid prices. And also, he can see whether he sold or unsold. If the particular player is selected, he can view the detail of the selected team. (Who team owner is, and who other players are in his team.)
- The team owner can view to whom he or she voted and what are the bid prices that the team owner recommended for each player.

- If the team owner wants to update the mentioned bid price for a particular player, it can be done through administration only.
- Only the admin has full authority over the entire system.
- Others have restrictions. That means players and team owners have limited access.
- Every team should be able to view the Status of their team during bidding.
- Only the admin can generate various reports such as all players' bio data, team details, auction details, summary, and so on.

❖ Requirement Specification

Functional

- The system should create login details for only registered players and team owners.
- The system should create a username and password for the registered ones.
- The system should provide access to the administrative officer for the entire system, for example, to register players, teams, and team leaders for an auction and to do all modifications.
- The system should request personal facts for the registration, and require the details about the type of player or performance required, the player's international commitment, injury history, his equation with the coach, first name, last name, contact number so on.
- The system should restrict the access of all players and team owners. Consequently, no one can do any modifications. That means, the system should restrict access for the person according to their job role.
- The system should automatically close the bidding time as they wished or planned.
- The system should provide access only for the administrative officer to generate reports.

- The system should provide the ability for the players and team owners to check updated details. For example, any player can check whether he sold or unsold. If he sold what the team he has selected, what is the bid price, who are the team owners that voted for him, and who are the other team members? Likewise, any team player can view the updated details.

- The system should provide access only for the admin to create related accounts and maintain the login systems.

- The system should provide power only for the admin to set and manage player-based price, bid price, player auction, and maximum price for each team.

Non-functional requirements

Usability	For the user to interact with the system, the system includes a help and support menu on all interfaces. By reading the help and support materials, the user can learn how to operate the system. • Simple UI and Responsive
Security	To prevent unwanted access to the system, the system gives a login and password. The password for the Admin, player, and team owners must be longer than eight characters. Only authorized people should be able to access the system's secured page. • Computer misuse act of 1990 • ISO 17799 – information security best practices • Computer security Act of 1987
Performance	The system response time for each user-initiated command must be less than 10 seconds. When executing user input, the system should have a high-performance rate and be able to respond in a reasonable amount of time. • Normalized database

Availability	This system should be available for given time period. In the event of a serious system failure, the system should be available within 1 to 2 working days, ensuring that business processes are not disrupted.
Error handling	Errors should be reduced as much as possible, and a suitable error message should be provided to help the user through the recovery process. Validation of the user's input is critical.
User-friendliness	Given the consumers' level of understanding, a basic but high-quality user interface should be created to make it simple to grasp and require minimal training.

Table 1 - Nonfunctional requirements

❖ Assumptions

- There is only one administrative officer. He or she was registered by the HR department.

❖ Advantages of this system

1) Save time on admin tasks

This web application will vastly cut down the time that staff spends on manual administrative tasks. The software does the majority of the work and lets staff divert their time to more important tasks. Through this automated system, the staff of XYZ Pvt Ltd can make significant time savings and also, boost staff productivity and satisfaction.

2) Develop strong relationships with the players and team owners

Choosing this best property management software will likely mean an increased level of retention in players, team owners, and staff.

3) **Increase the online visibility**

This software has an important factor in developing an online presence. The staff can integrate guest-facing software.

4) **Implement an effective revenue management system**

5) **Accurate daily reports**

From this system, the administrative officer can generate reports, Such as all players' bio data, team details, auction details, summary, and so on.

6) **Prevent manual errors**

Processing an auction manually denotes a lot of work, and often it can make errors that cost money. Through this automated system, the players and team owners will be able to ensure the auctions. So, there's no necessity for cross-checking auction statuses and spreadsheets.

7) **Analyze the customer base**

❖ Future Development of this system

- To create a mobile app for this.
- To give authority for the entire system to all authorized people.
- Provide option to any players and any team owner have to be registered through the system by themselves.
- Provide access to all authorized people. Consequently, everyone can save time more than now.

❖ Work Flow - E-SPORT system

Figure 2 - Work Flow - E-SPORT system

❖ Tools and development Environment

1) ASP.NET MVC

Figure 3 - ASP.NET MVC

ASP.NET is a free web framework for building websites and web applications on .NET Framework utilizing HTML, CSS, and JavaScript. ASP.NET MVC 5 is a web framework founded on Model-View-Controller (MVC) architecture. Designers can build dynamic web applications utilizing the ASP.NET MVC framework that allows a neat separation of references, rapid development, and TDD friendly.

The web application topography is visited and varied. Microsoft's ASP.NET is built on top of a mature and strong framework. This Framework is one of the most trusted platforms in the industry. ASP.NET MVC is Microsoft's most delinquent expansion to the world of ASP.NET providing web developers with an alternative development approach that supports them build a web application with ease.

2) Ajax

Figure 4 – Ajax

AJAX = **A**synchronous **J**avaScript **A**nd **X**ML. AJAX is not a programming language. AJAX is a misleading name. AJAX applications might utilize XML to transport data, but it is equally familiar to transport data as plain text or JSON text. AJAX permits web pages to be updated asynchronously by swapping data with a web server behind the scenes. The meaning is, that is feasible to update parts of a web page without reloading the whole page.

AJAX utilizes a variety of:

- A browser built-in XMLHttpRequest thing or object (to request data from a web server)
- JavaScript and HTML DOM (to show or use the data)

The XMLHttpRequest Object

All current or modern browsers help the XMLHttpRequest object. The XMLHttpRequest object can be utilized to exchange information with a server behind the scenes. The meaning is that feasible to update parts of a web page without reloading the whole page.

How AJAX Works

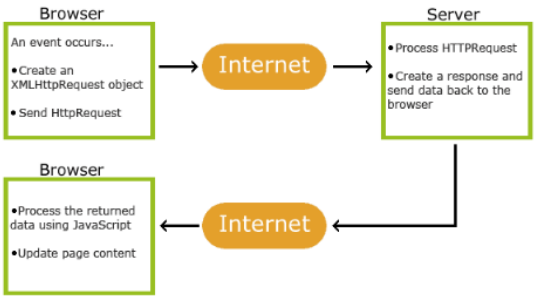


Figure 5 - How AJAX Works

3) Bootstrap

Figure 6 – Bootstrap

Bootstrap uses Sass for modular and customizable architecture. Import only the features anyone needs, enable international opportunities or options like gradients and shadows, and write the CSS with variables, maps, functions, and mixings. Get started with Bootstrap, the world's most famous framework for building responsive, mobile-first sites, with jsDelivr and a template starter page. Bootstrap is a free front-end framework for faster and more comfortable web development. Bootstrap contains HTML and CSS-based design templates for typography, forms, buttons, tables, navigation, modals, image carousels, and many others, as well as optional JavaScript plugins. Bootstrap also offers you the capability to design responsive designs efficiently. Bootstrap was developed by Mark Otto and Jacob Thornton at Twitter and released as an open-source development in August 2011 on GitHub.

Why Use Bootstrap?

- **Easy to operate or use:** Anyone with just fundamental knowledge of HTML and CSS can begin utilizing Bootstrap.
- **Responsive characteristics**: Bootstrap's responsive CSS adjusts to phones, tablets, and desktops.
- **Mobile-first method:** In Bootstrap 3, mobile-first styles are part of the core framework.
- **Browser compatibility:** Bootstrap is consistent with every modern browser (Chrome, Firefox, Internet Explorer, Edge, Safari, and Opera).

4) JQuery

Figure 7 - JQuery

JQuery is a small, quick, and feature-rich JavaScript library. It creates things like HTML document traversal and manipulation, event handling, animation, and Ajax much easier with an easy-to-use API that operates across a multitude of browsers. With a mix of versatility and extensibility, jQuery has modified the way that millions of people write JavaScript. JQuery is a "write less, do more" JavaScript library. The objective of jQuery is to create it much more comfortable to utilize JavaScript on a particular website. JQuery takes a lot of basic tasks that need multiple lines of JavaScript code to execute and wraps them into methods that anyone can call with a single line of code. JQuery also facilitates many complex things from JavaScript, like AJAX calls and DOM manipulation.

There are many other JavaScript libraries out there, but jQuery is presumably the most popular and also the most extendable. Many of the most significant enterprises on the Web use jQuery, such as Google, Microsoft, IBM, and Netflix.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

5) Entity Framework.net core 6

Before .NET 3.5, we (developers) usually utilized write ADO.NET code or Enterprise Data Access Block to keep or recover application information from the underlying database. It uses to open a connection to the database, create a DataSet to convey or submit the data to the database, transform data from the DataSet to .NET objects, or vice-versa to use business laws. That was an unmanageable and error-prone procedure. Microsoft has supplied a framework denoted "Entity Framework" to automate all these database-related activities for the application. It is an open-source ORM framework for .NET applications supported by Microsoft. It allows designers to work with data-applying objects of domain-specific classes without concentrating on the underlying database tables and columns where this data is stored. With the Entity Framework, developers can perform at a higher level of generalization when they deal with data and can build and maintain data-oriented applications with minor code or less code compared with traditional applications.

Entity Framework Core is a contemporary or modern object-database mapper. NET. It helps LINQ queries adjust tracking, updates, and schema migrations. EF Core functions with many databases, consisting of SQL Database (on-premises and Azure), SQLite, MySQL, PostgreSQL, and Azure Cosmos DB.

6) iText

iText, which is a Java PDF library using, anyone can develop Java programs that create, transform, and operate PDF documents.

Features of iText

The following are the significant characteristics of the iText library −

- **Interactive** − iText supplies classes (APIs) to develop interactive PDF documents. Utilizing these, anyone can create maps and books.
- **Adding bookmarks, page numbers, etc.** − Applying iText, a developer can add bookmarks, page numbers, and watermarks.
- **Split & Merge** − Using iText, a user can break an existing PDF into multiple PDFs and also add/attach additional pages to it.
- **Fill Forms** − Utilizing iText, a designer can fill interactive documents in a PDF document.
- **Save as Image** − Using iText, any user can save PDFs as image files, such as PNG or JPEG.
- **Canvas** − iText library provides users with a Canvas class utilizing which can be drawn various geometrical shapes on a PDF document like a circle, lines, etc.
- **Create PDFs** − Using iText, users can make a new PDF file from particular Java programs. Users can contain images and fonts too.

[Kept space intentionally]

❖ Mission and Vision of E-SPORT system

Mission

To furnish our registered players and team owners with progressive influences auction technology solutions.

Vision

Count importance to the property auction procedure and enhance the opportunities to the players; with transparent competition utilizing advanced technology, and be identified as the primary effects auction software provider internationally".

Project plan

Task	Assigned person	Starting date	Finishing date
Requirement gathering	All members of the group	10/1/2022	10/8/2022
Requirement analysis	All members of the group	10/9/2022	10/10/2022
Database design	Rashmi	10/10/2022	10/15/2022
Interface design	Samitha	10/9/2022	10/15/2022
Coding	Sandun	10/16/2022	11/7/2022
Testing	All members of the group	11/1/2022	11/11/2022
Documentation	All members of the group	10/1/2022	11/11/2022
Submission	All members of the group	11/12/2022	11/12/2022

Table 2 - Project plan

SWOT analysis.

Figure 8 - SWOT

A SWOT analysis is a strategy for considering the four aspects (Strengths, Weaknesses, Opportunities, and Threats) of the business that is a tool that can help to analyze what the companies do best now and to devise a successful method for tomorrow. SWOT can also uncover areas of the business that are holding a person back or that his or her competitors could control if he or she doesn't protect herself or himself. SWOT analysis studies both internal and external factors – that is, what's going on inside and outside the organization. So some of these characteristics will be within their control, and some will not. In either case, the wisest action they can take in reaction will become clearer once they've found, recorded, and analyzed as many factors as they can.

Why Is SWOT Analysis Important?

SWOT Analysis can support people to challenge difficult assumptions and find risky blind spots in the organization's performance. If they use it carefully and collaboratively, it can provide new insights into where their business is nowadays and help them to develop exactly the right approach for any situation. For instance, people may be well aware of some of their organizations' strengths, but until they record them alongside weaknesses and threats, they might not recognize how inconsistent those strengths are. Equally, they probably have reasonable concerns about some of the business weaknesses by going through the analysis systematically they could uncover an opportunity, previously overlooked, that could more than repay.

What is a SWOT Analysis for Auction?

SWOT is a complete and efficient framework to examine the overall business or specific projects and initiatives to support and identify the benefits while watching out for what could go wrong. If someone fails to prepare, he or she prepares to fail. A SWOT analysis is the road map to success. A SWOT analysis for their E-SPORT suction system can be utilized anytime they're considering larger investments, such as updating furnishings, installing new technologies, and modifying operational processes. Strengths, weaknesses, opportunities, and threats will differ from company to company, depending on market circumstances or conditions, the enterprise's mission, and its unique recommendation.

Strengths	What do you do well? What unique resources can you draw on? What do others see as your strengths?
Weaknesses	What could you improve? Where do you have fewer resources than others? What are others likely to see as weaknesses?
Opportunities	What opportunities are open to you? What trends could you take advantage of? How can you turn your strengths into opportunities?
Threats	What threats could harm you? What is your competition doing? What threats do your weaknesses expose to you?

Table 3 - SWOT table

Strengths:

- Anywhere access - Anytime, anywhere permit software to handle the auction remotely
- Touchless registration and apply for any trophy.
- Centralized rate management
- Manage all characteristics of the business from one dashboard to facilitate operations
- Real-time, and maximizes productivity.
- Online registration lightens the workload and is loved by visitors.

Weaknesses:

- Staff requires to be oriented away from the old way.
- Need to invest in PC
- Industrialization or automation could feel impersonal
- Some players or team leaders resistant to technology

Opportunities:

- Client database for personalized marketing
- Boost ancillary revenue opportunities.
- Surprise and delight enterprises to build loyalty and repeat bid
- Sustainability struggles or efforts.

Threats:

- Forthcoming challenger from another market's franchise.
- Government rules and regulation was unexpected.
- There is more competition nowadays.
- Customer service is lacking (wait times, etc.)
- Good staff members are leaving the company.
- Social media is a term that guides the use of complacency.
- Pandemic travel restrictions and protocols
- Data security breaches
- Competitors with better research-intensive approaches
- Economy

Scope

The goal of this project is to design an automated auction system for XYZ Pvt Ltd. The system can manage and handle the activities that are involved in XYZ Pvt Ltd in an organized, cost-effective, and reliable way. It will help the administration to do their work in a very simple manner without causing any trouble and players and team leaders to their registration. They can use any electronic device like a laptop, computer, smartphone, tablet, etc. to register and apply for any trophy. This E-SPORT system is the solution that has been provided for better management of the organization. The system's data is saved in a database that will serve as the hub for all information held by the admin. Things can be simplified and significantly accelerated as a result, making the duties of those involved easier. It maintains the present procedure but centralizes it, allowing decisions to be made more quickly and easily.

Objectives of the project

- Organize and keep records of a vast number of players and team leaders.
- To manage all information on players and team leaders who have registered for trophies that are available.
- To send a verification email that consists of login details (Username and password) to the registered ones.
- To provide access to the admin officer for registrations and manage the auction.
- To provide access only to the admin for generating reports.
- To show available trophy details to the players and team players for auction.
- To display the status to the players and team owners as well.
- To create statistics reports.
- To obtain their feedback to assist in future improvement

2) Structure of the proposed system

The design process for the project is documented in this section of the document. The application summary with UML diagrams and the hardware and software requirements for running this system are briefly addressed here.

- **Software Architecture Diagram.**

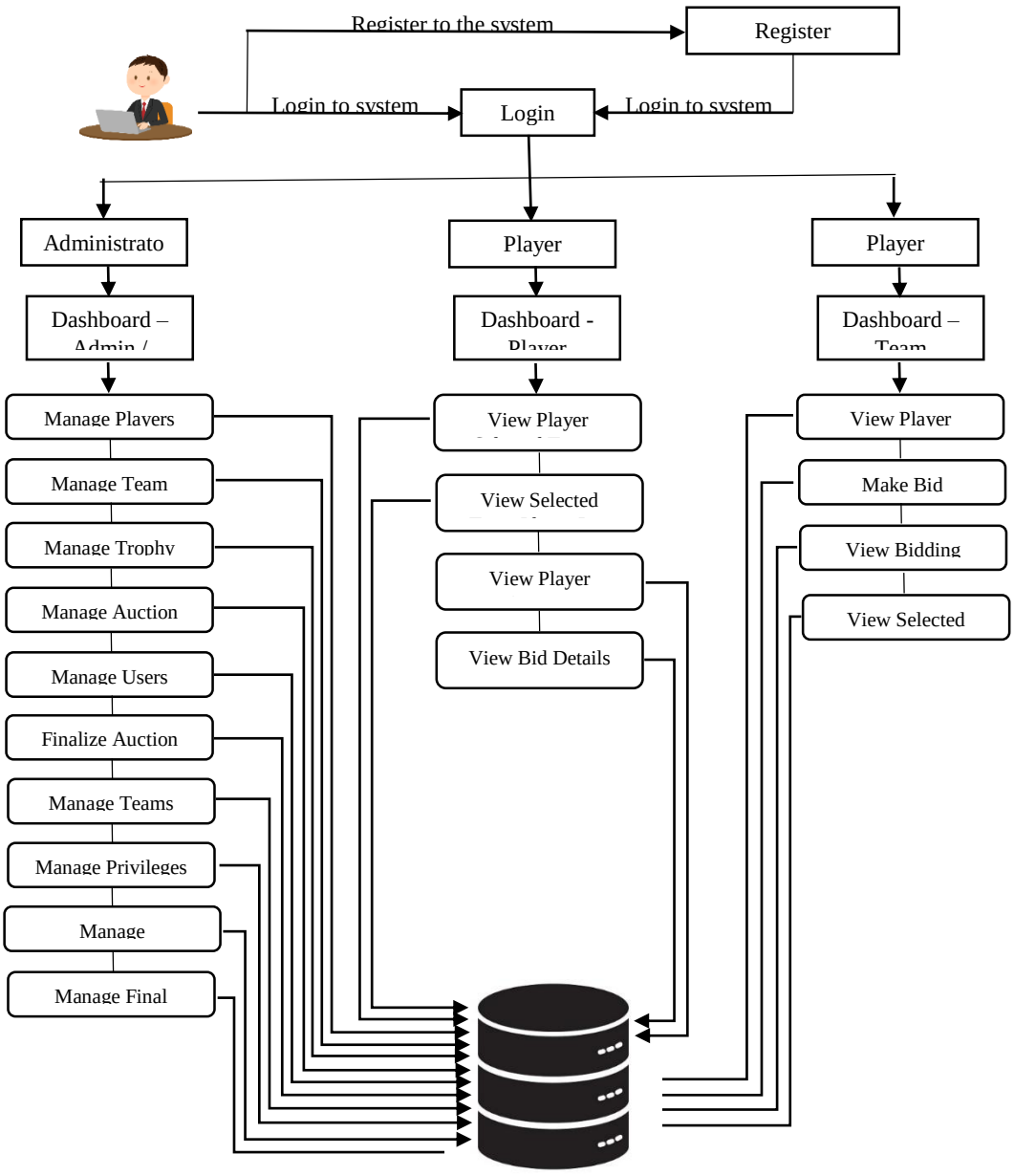


Figure 9 - Software Architecture Diagram

- **ER Model Diagram**

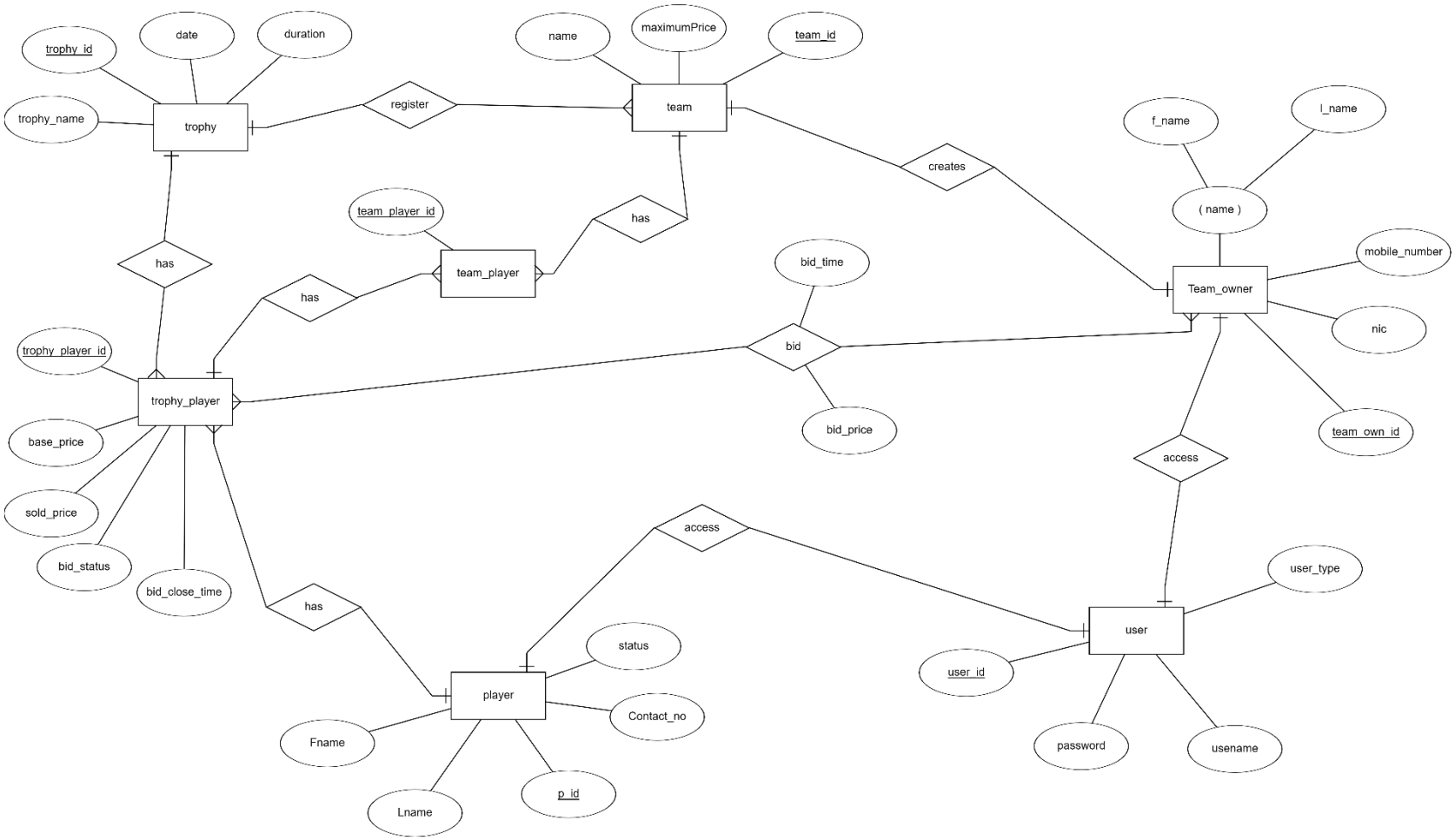


Figure 10 - Entity Relationship diagram

- **Use Case Diagram**

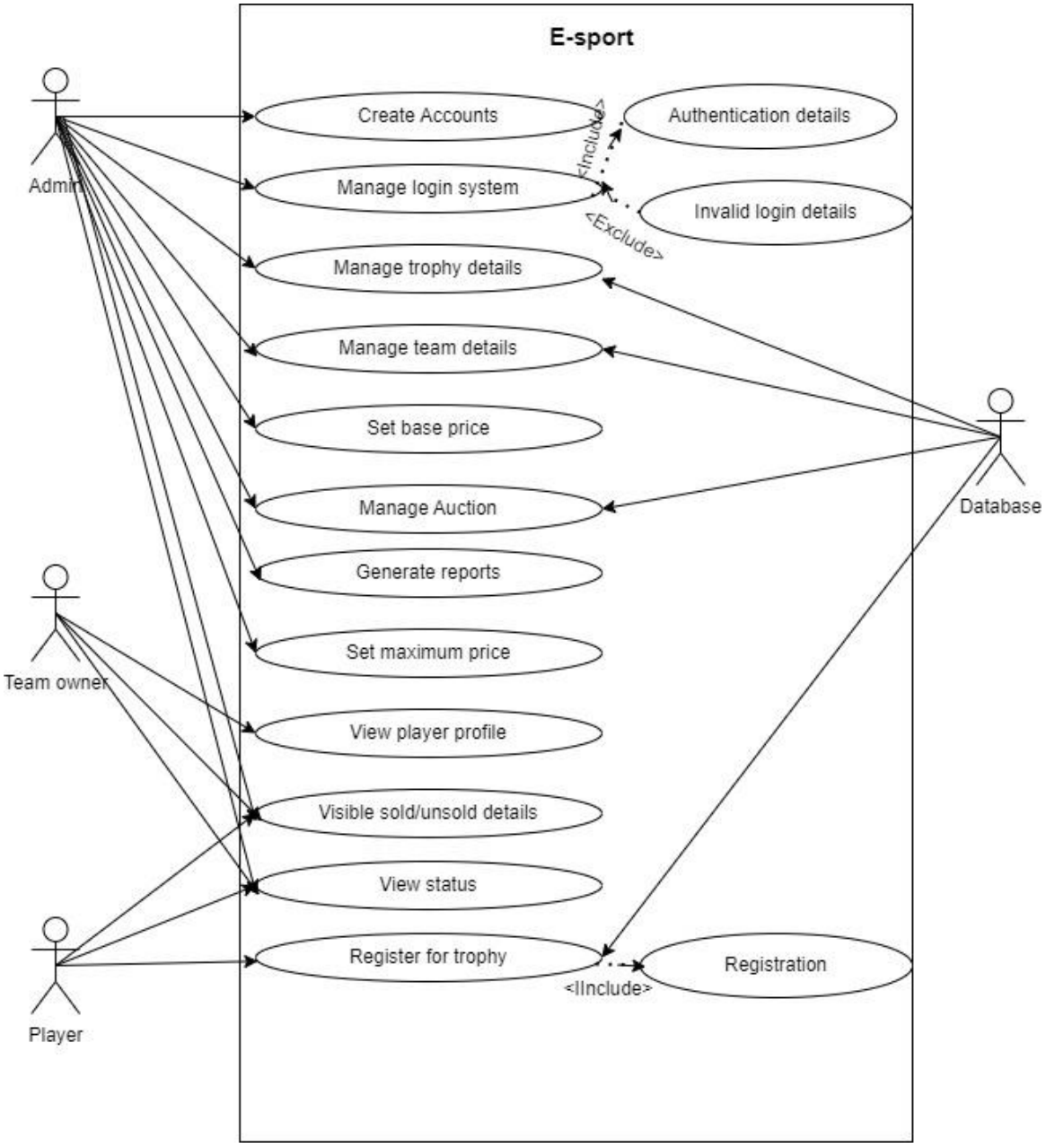


Figure 11 - Use case Diagram

- **Relational Schema**

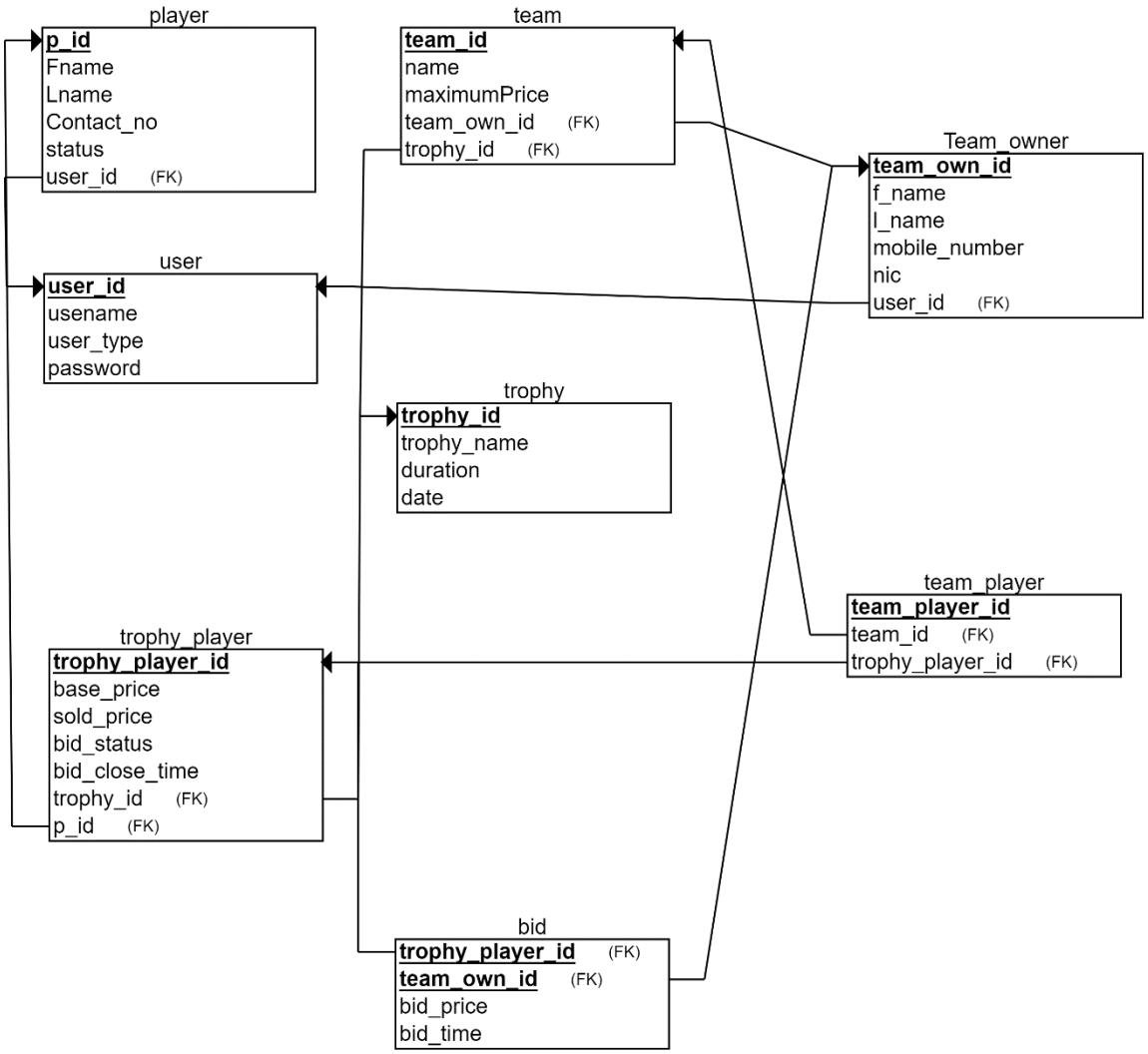


Figure 12 - Relational Schema

- **Class Diagram**

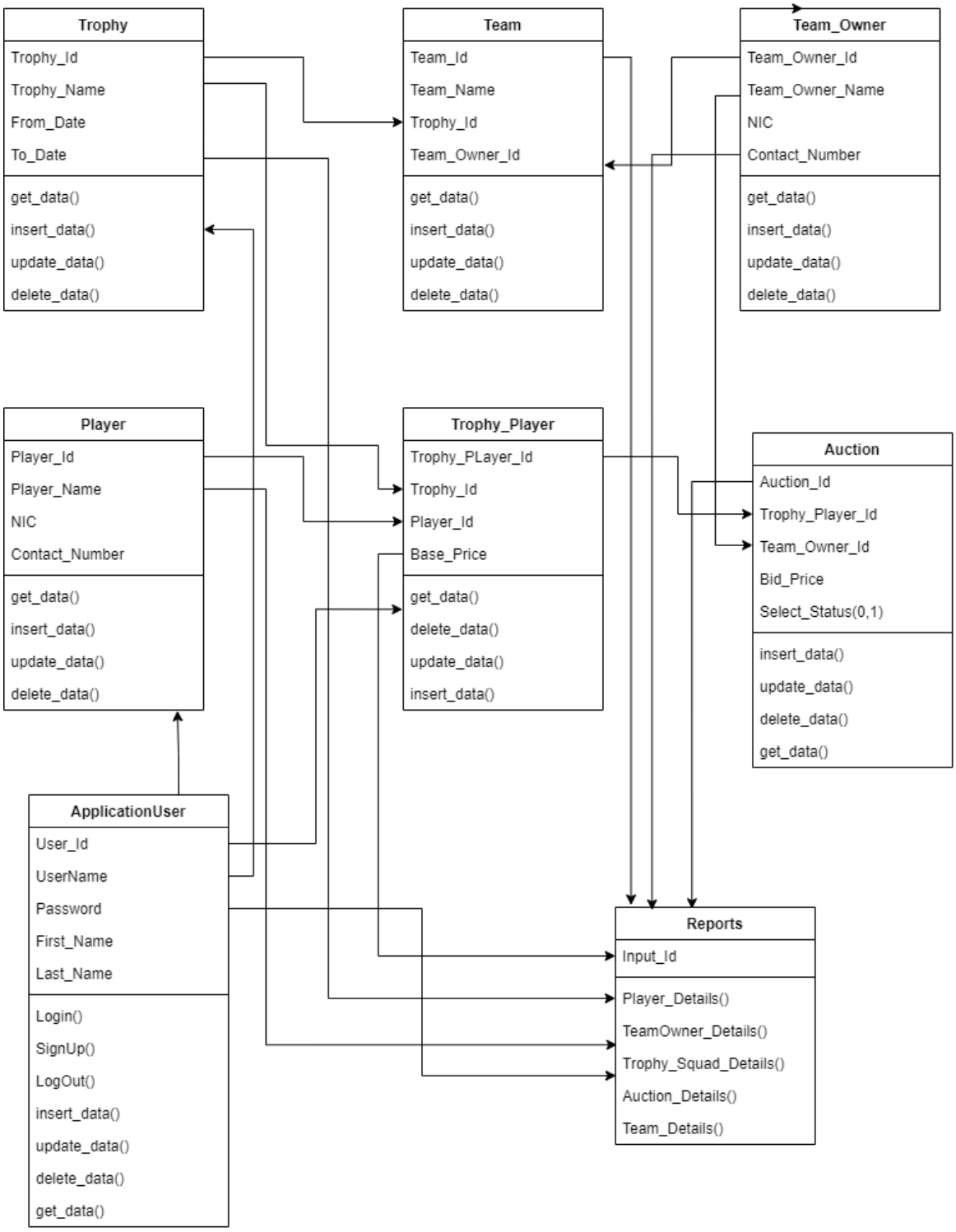


Figure 13 - Class Diagram

3) Database

Microsoft SQL Server is a relational database management system (RDBMS) that helps a wide variety of transaction processing, business intelligence, and analytics applications in corporate IT environments. It is a market-leading database technology, along with Oracle Database, and IBM's DB2. So, SQL Server Management Studio 2019 is used for this project for database creation and design. From SQL Server to Azure Database, this is an integrated platform for helping any SQL infrastructure applied in the project backend. The below tables were constructed. In SQL Server Management Studio 2019, this is how the database architecture looks:

Application User Table

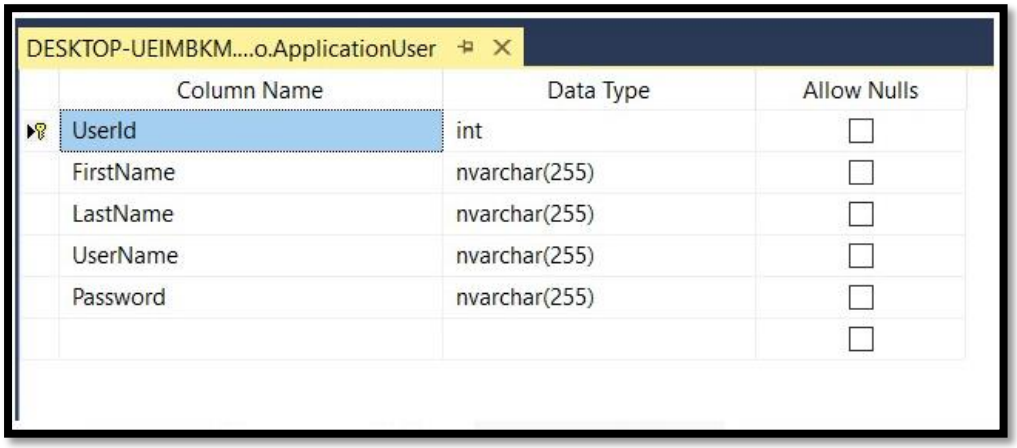


Figure 14 - Application User Table

Auction Table

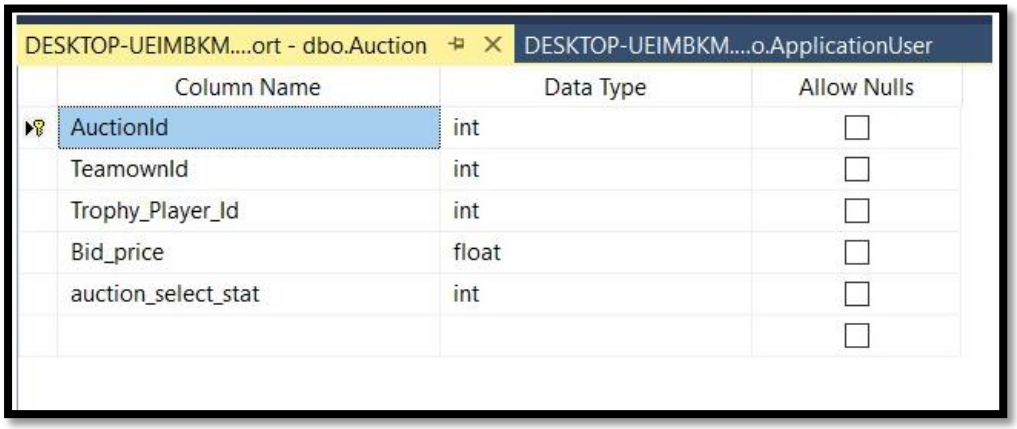


Figure 15 - Auction Table

Player Table

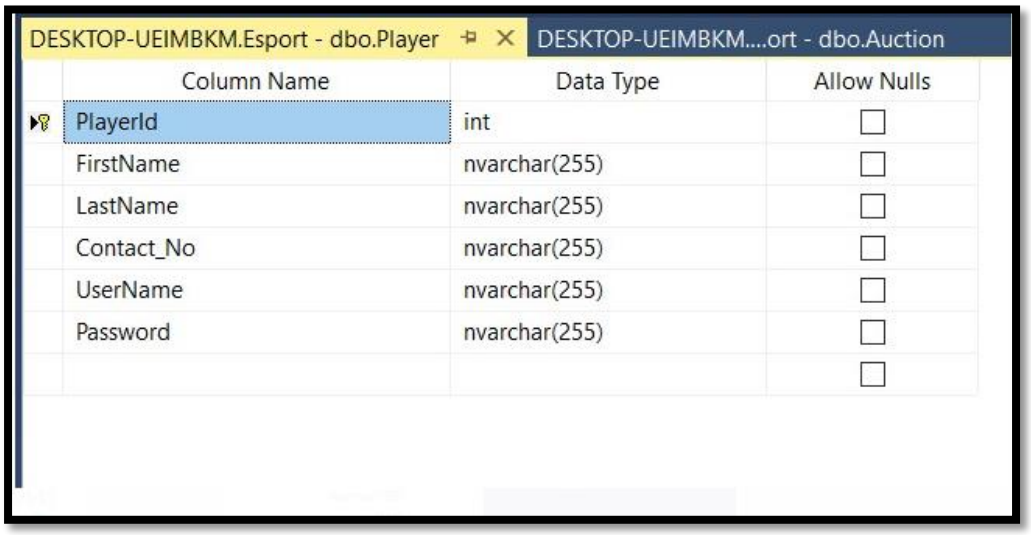


Figure 16 - Player Table

Team table

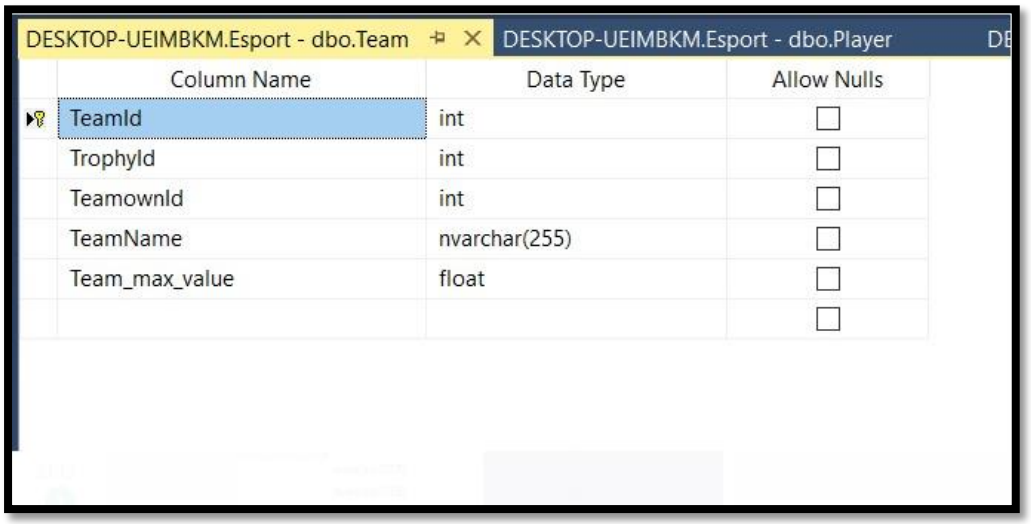


Figure 17 - Team table

Team Player Table

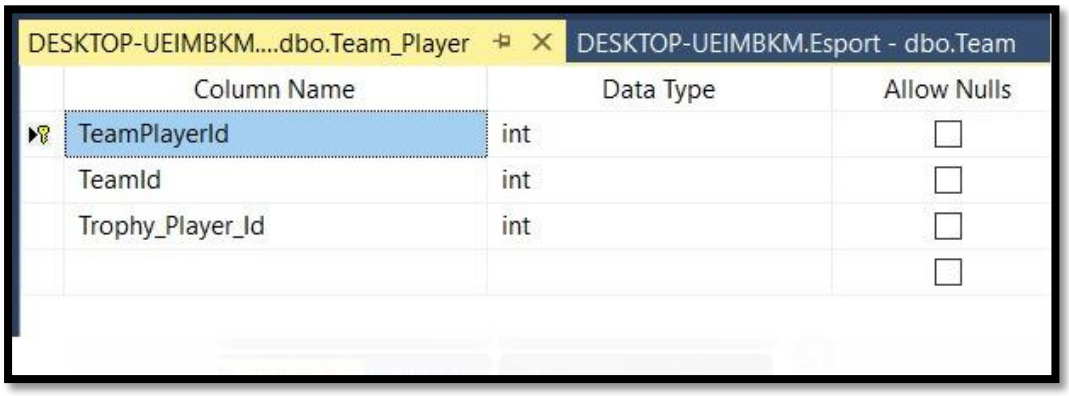


Figure 18 - Team Player Table

Team Owner Table

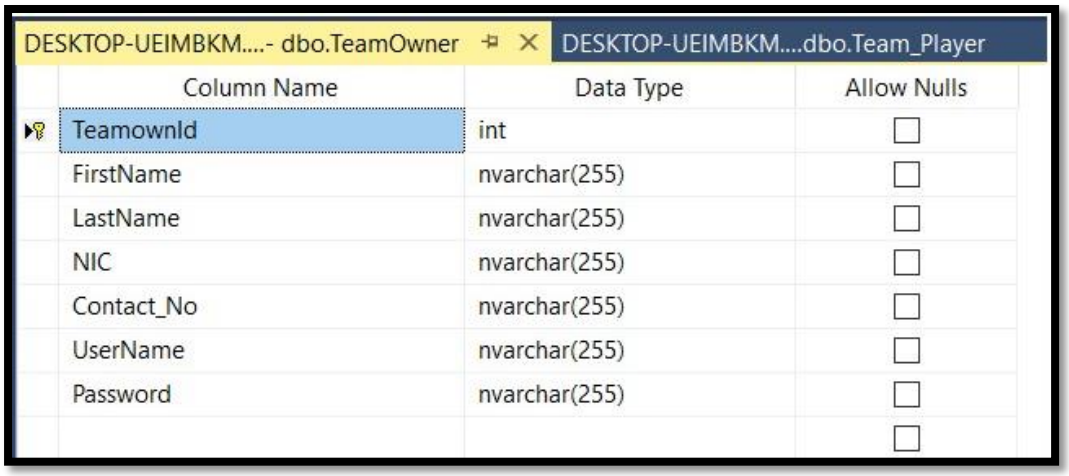


Figure 19 - Team owner table

Trophy Table

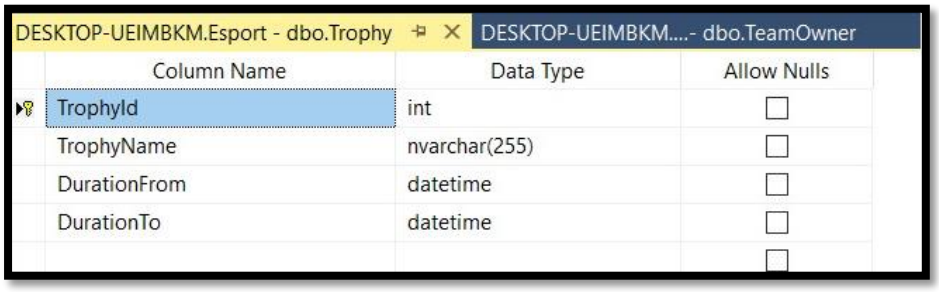


Figure 20 - Trophy Table

4) Methodology

Agile methodology

AGILE methodology is a system that promotes continuous iteration of expansion and testing during the software development lifecycle of the project. In the Agile model, both development and testing actions are simultaneous, unlike in the Waterfall model. The Agile software development methodology is one of the easiest and most efficient processes to turn a vision for a company's requirements into software clarifications. Agile is a word applied to explain software development approaches that employ regular planning, learning, improvement, team collaboration, evolutionary development, and early delivery. It supports flexible answers to change. The final value of agile development methodology is that it allows teams to deliver value faster, with greater quality and predictability, and greater ability to respond to change. Scrum and Extreme programming (XP) are two of the most widely used agile methodologies.

Reason for choosing agile methodology

Agile methodology has more powerful. Its focus is on businesses to break down obstacles and guarantee that everyone can move on the correct path to reach a win-win project within an adequate timeframe the support of affecting customers and getting continuous feedback from them, the end product can be delivered faster to the market. So with the benefit of agile methodology, the E-SPORT auction system can provide its products to the market sooner. Then XYZ Pvt Ltd.-SPORT Pvt Ltd has a high chance to compete against their competitors their product can reach more customers and, it can make considerable business profits.

[Kept space intentionally]

5) Detailed instruction with user interfaces.

- **Interfaces**

1) Admin Login

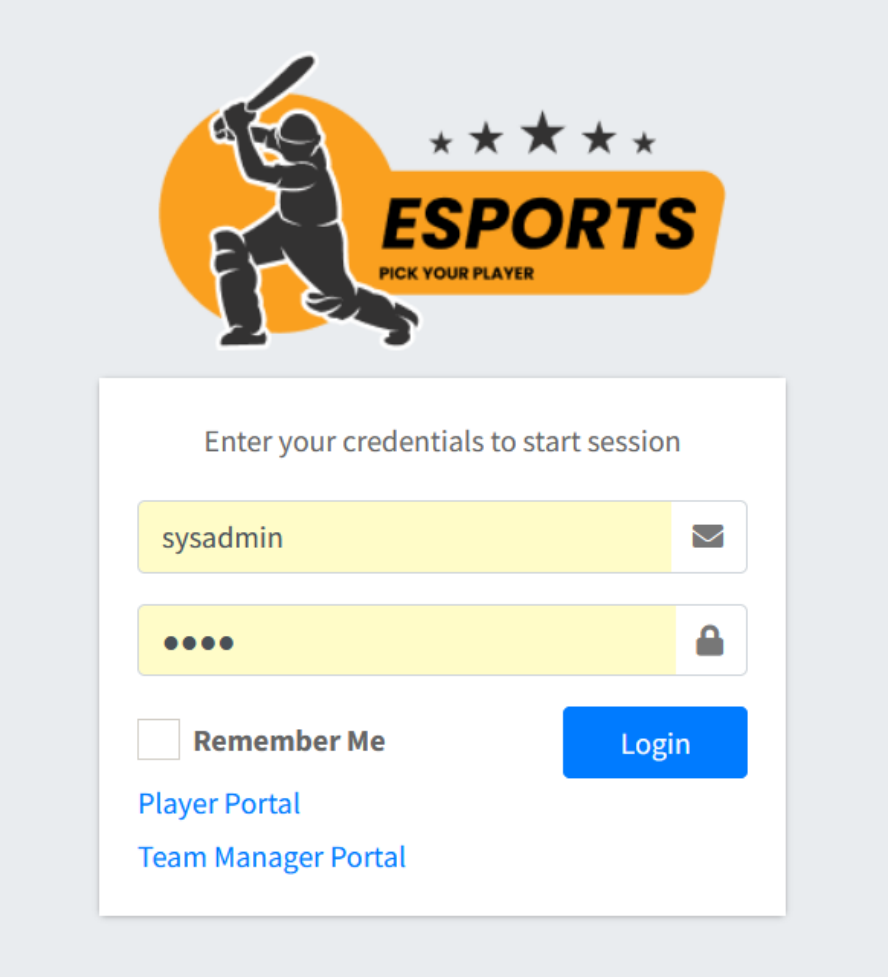


Figure 21 - Admin Login

2) Team owner login

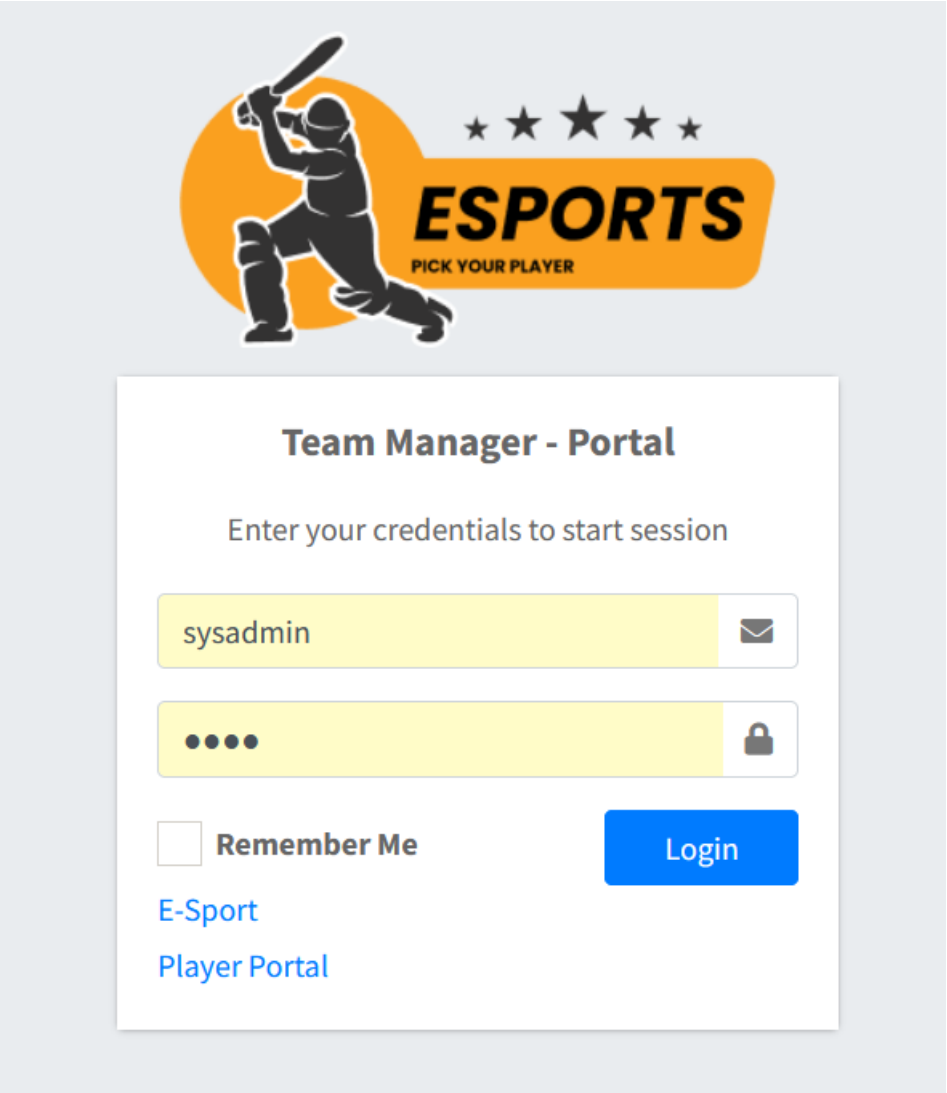


Figure 22 - Team owner login

3) Player Login

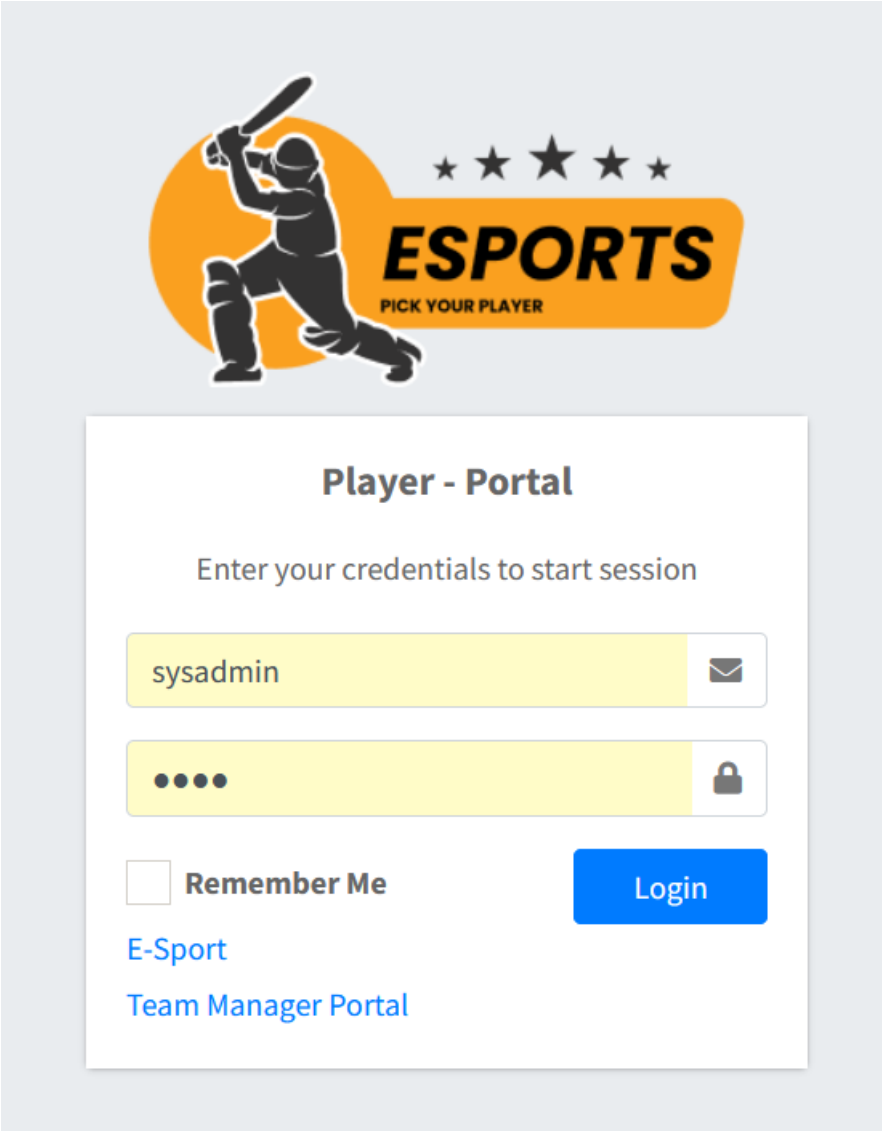


Figure 23 - Player Login

4) Auction Finalize

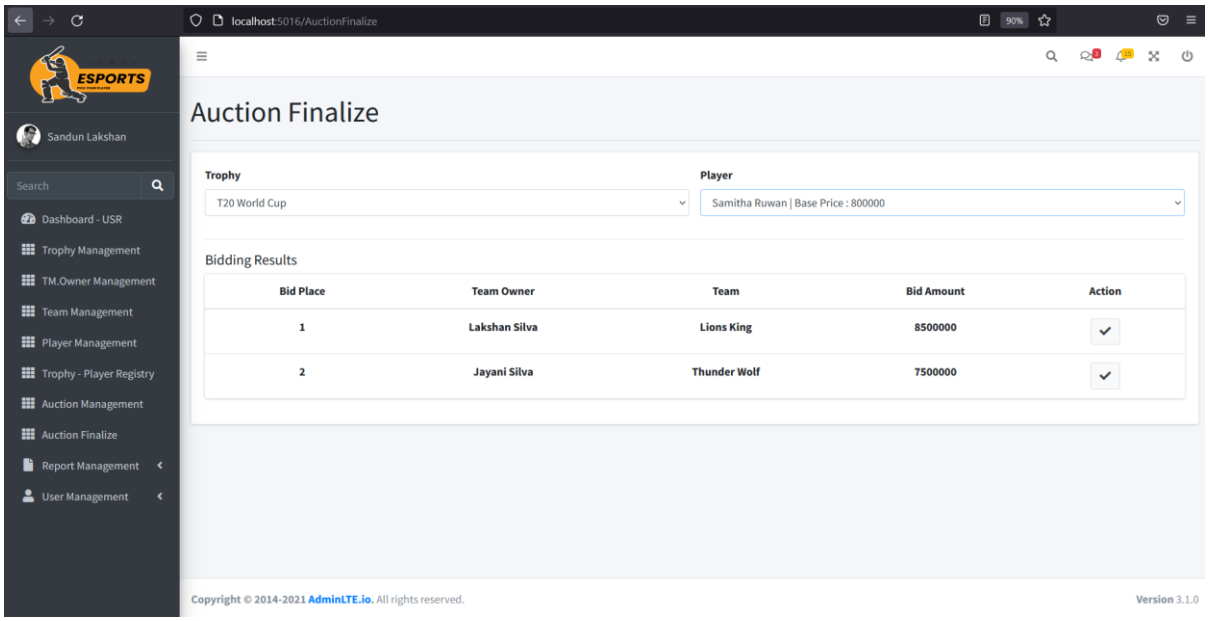


Figure 24 - Auction finalize

5) Auction Management

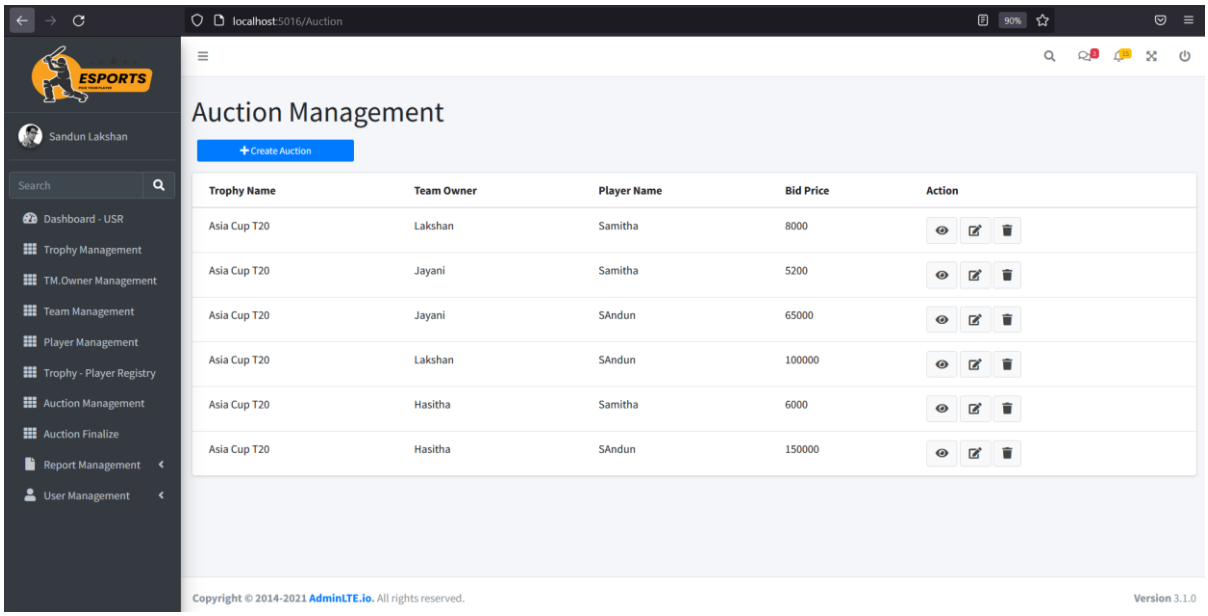


Figure 25 - Auction Management

6) Add player

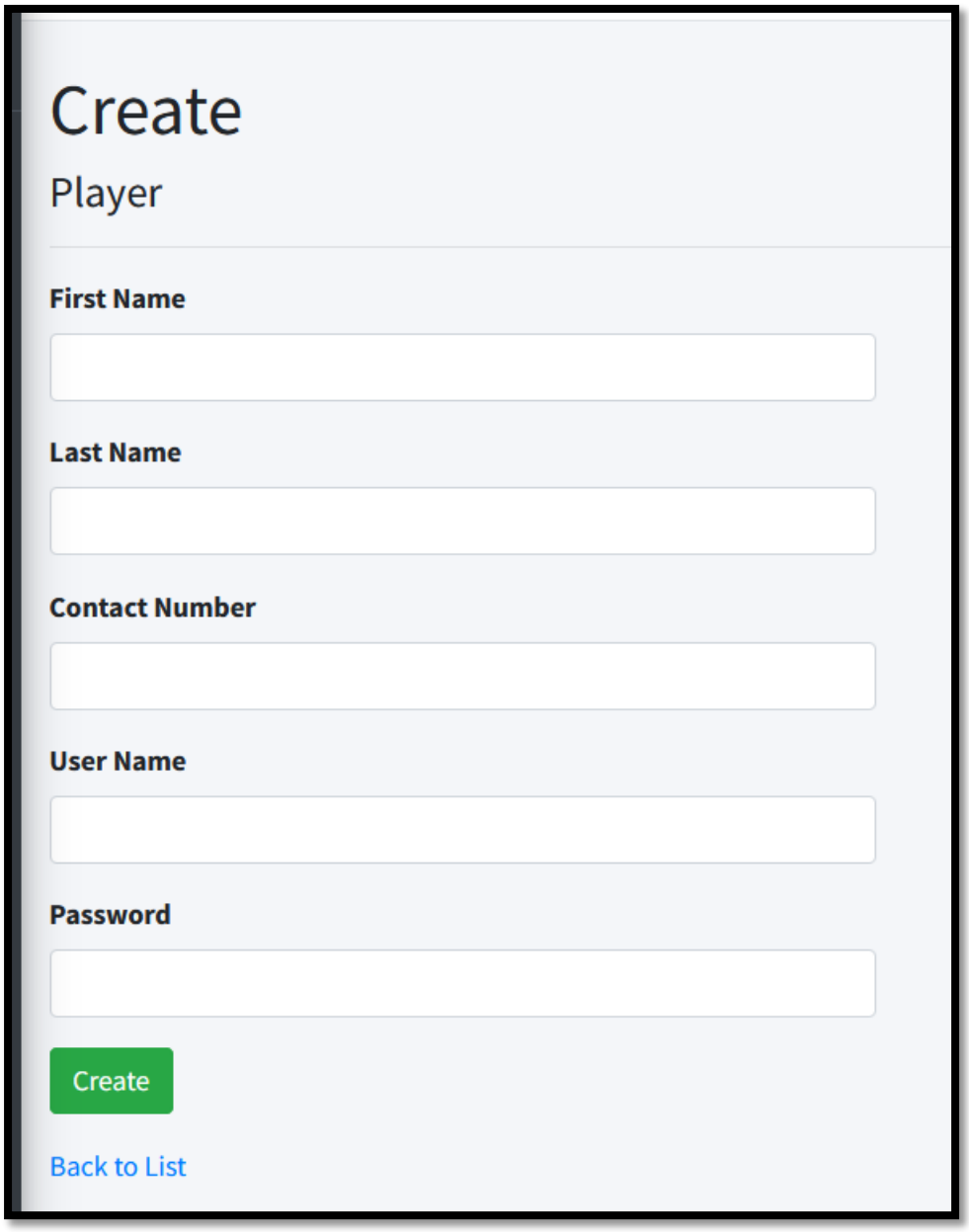


Figure 26 - Add player

7) Player Management

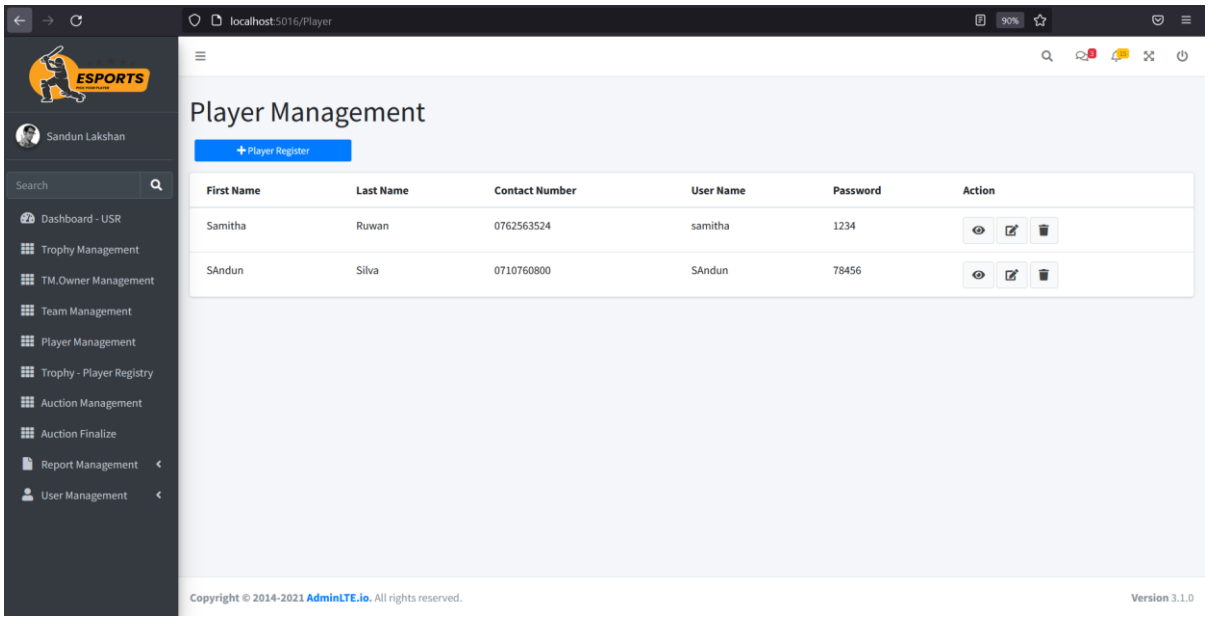


Figure 27 - Player Management

8) Team details management

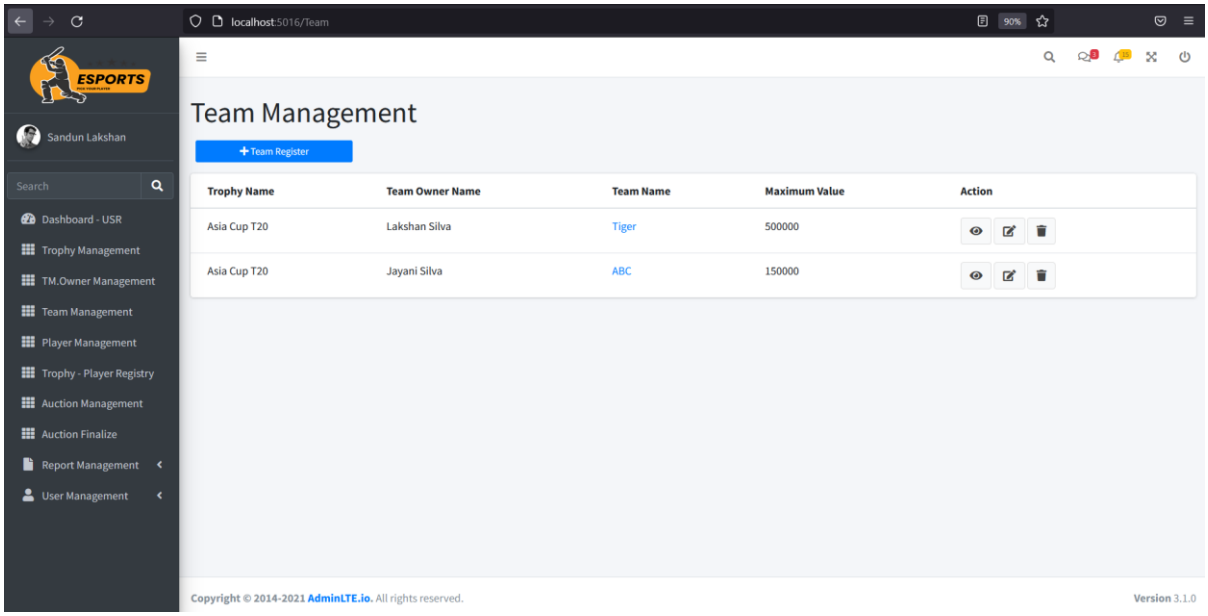


Figure 28 - Team details management

9) Team owner delete

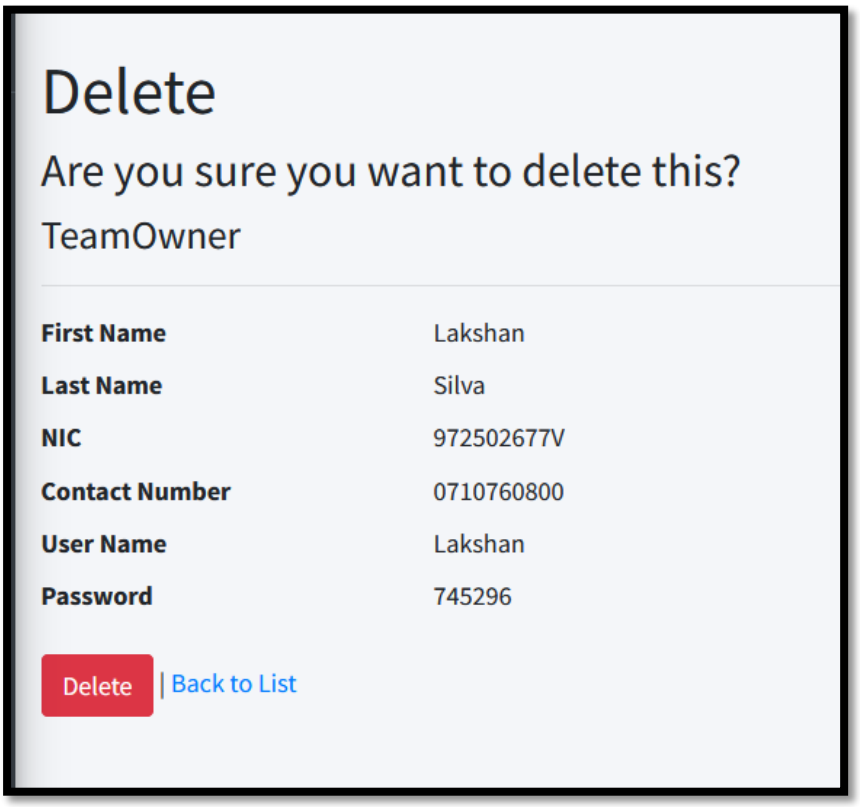


Figure 29 - Team owner delete

10) Trophy Management

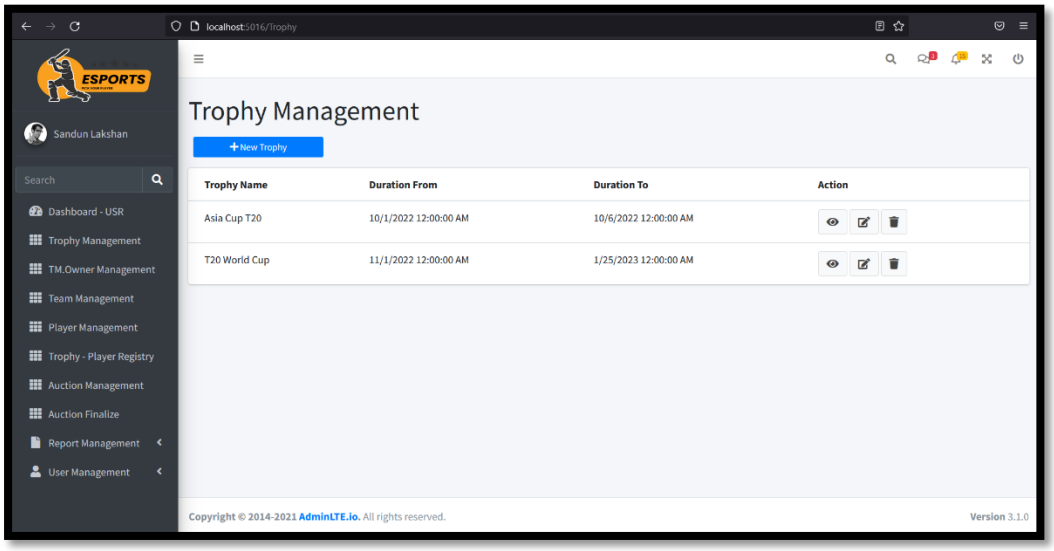


Figure 30 - Trophy Management

11) Team owner details

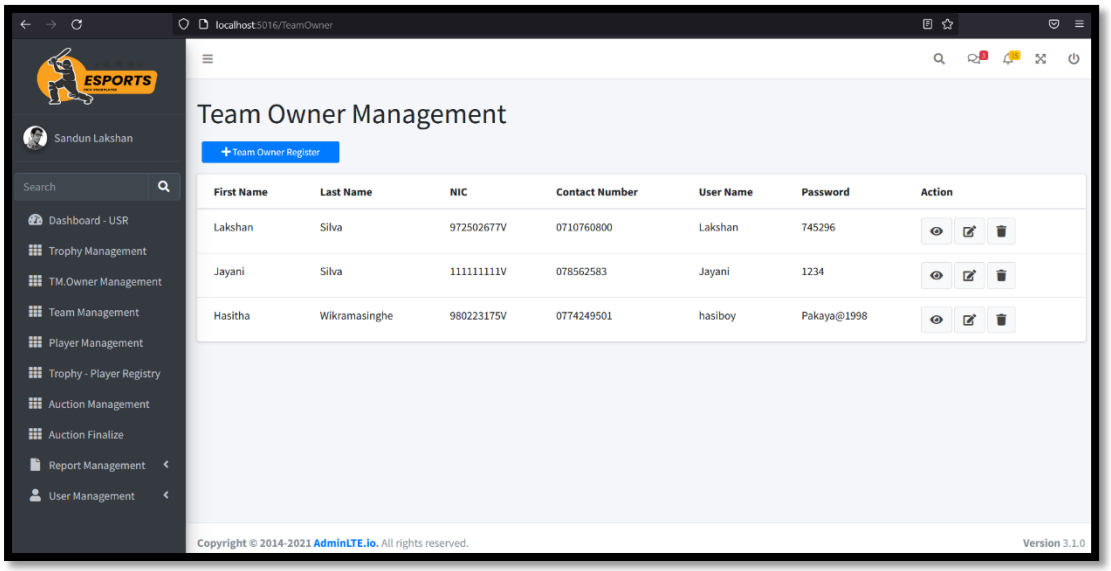


Figure 31 - Team owner details

12) Team player details

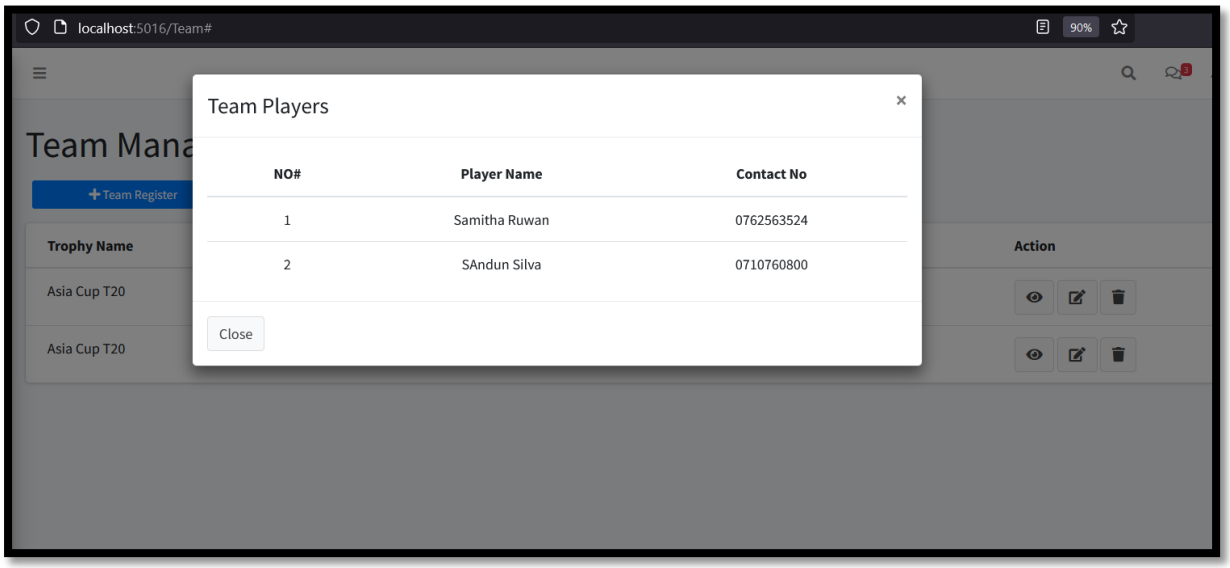


Figure 32 - Team player details

13) Team owner registration

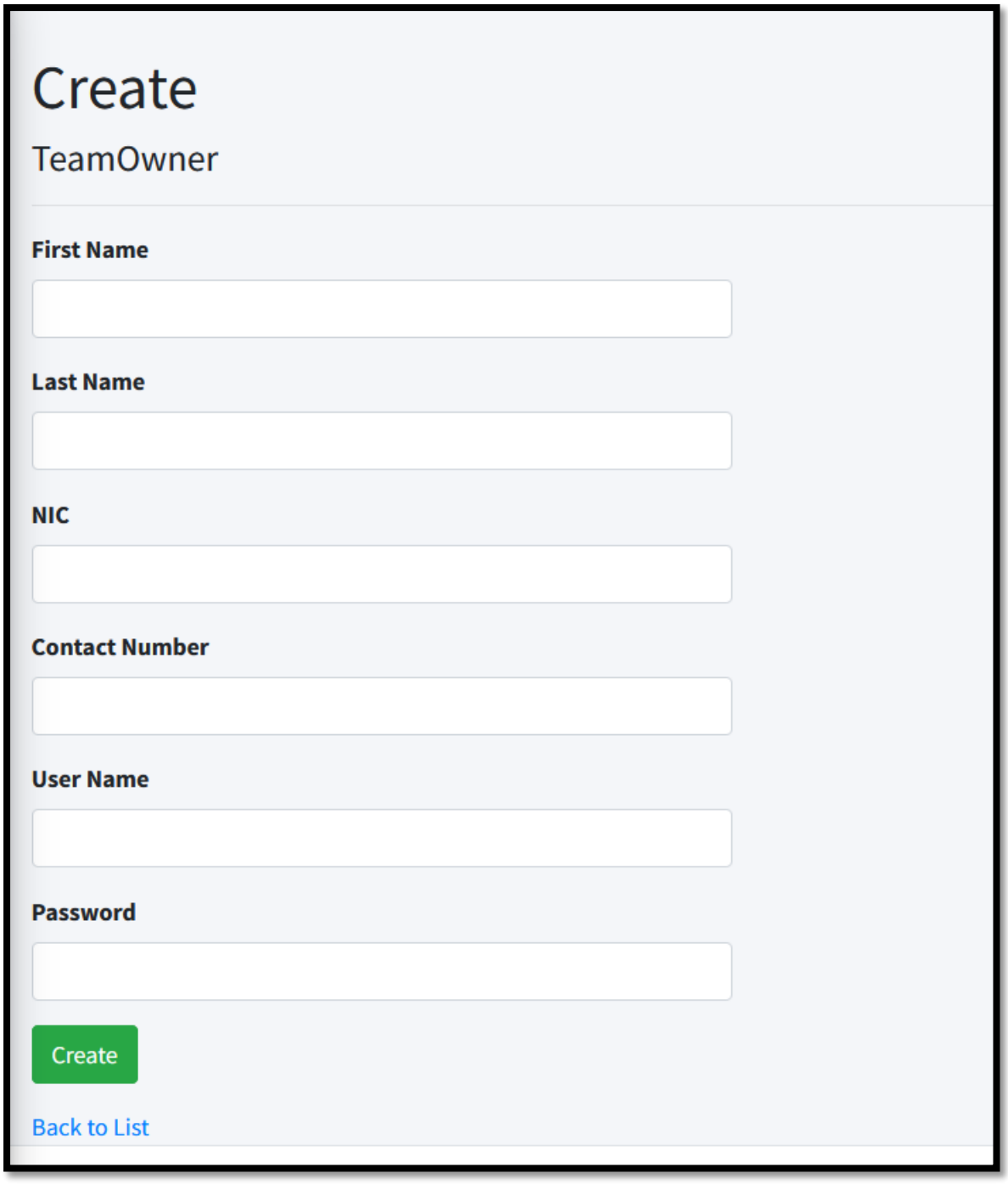


Figure 33 - Team owner registration

14) Edit team owners

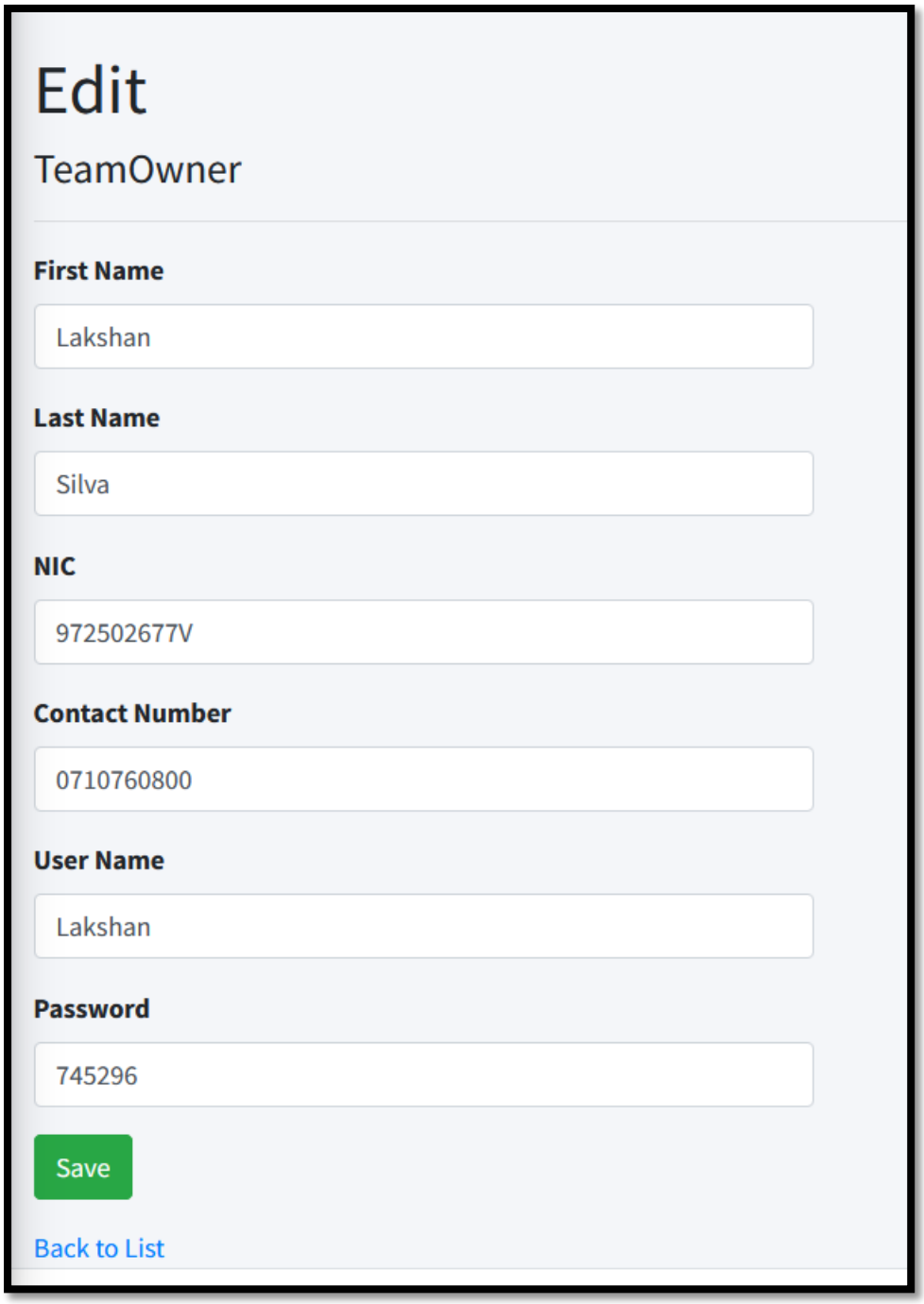


Figure 34 - Edit team owners

15) Add trophy

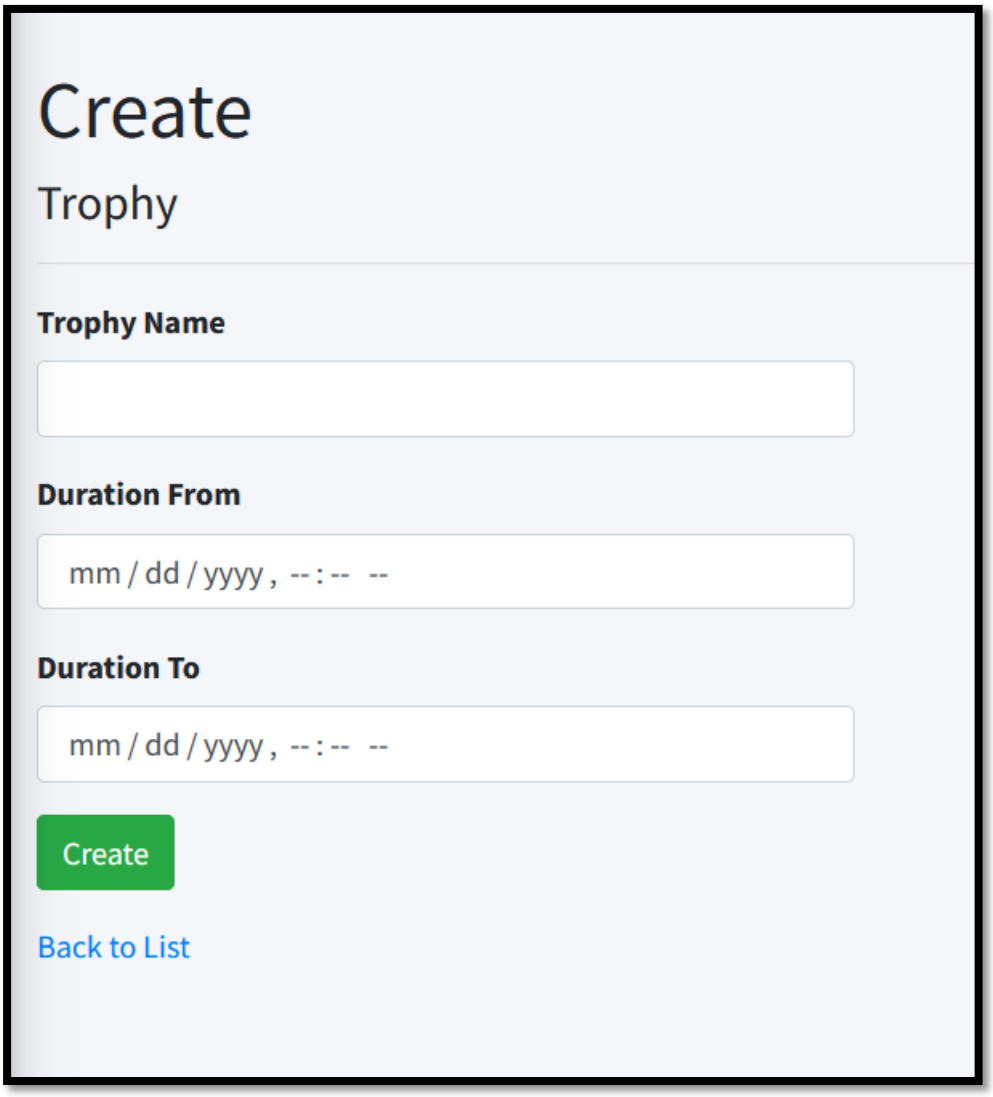


Figure 35 - Add trophy

16) Create trophy player

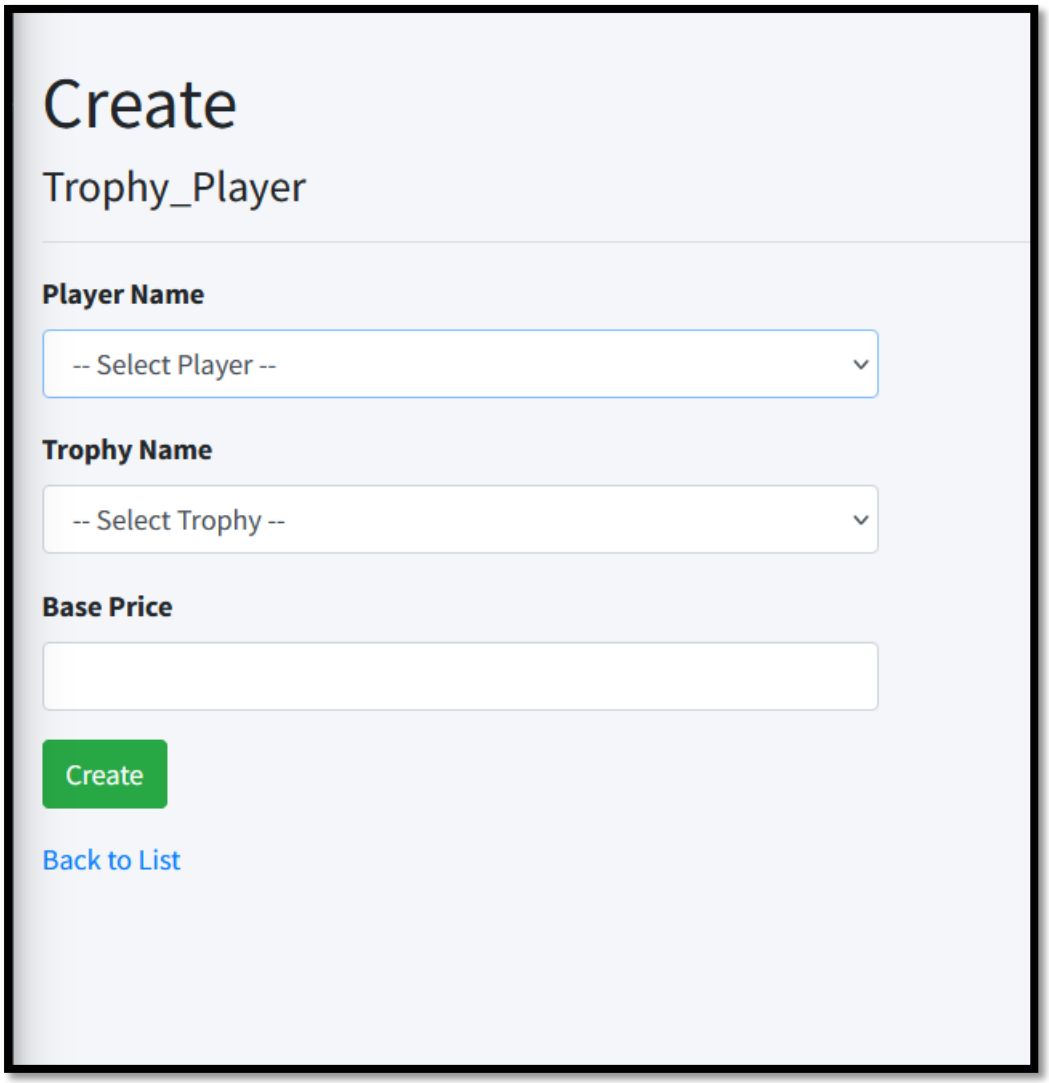


Figure 36 - Create trophy player

17) Edit trophy player

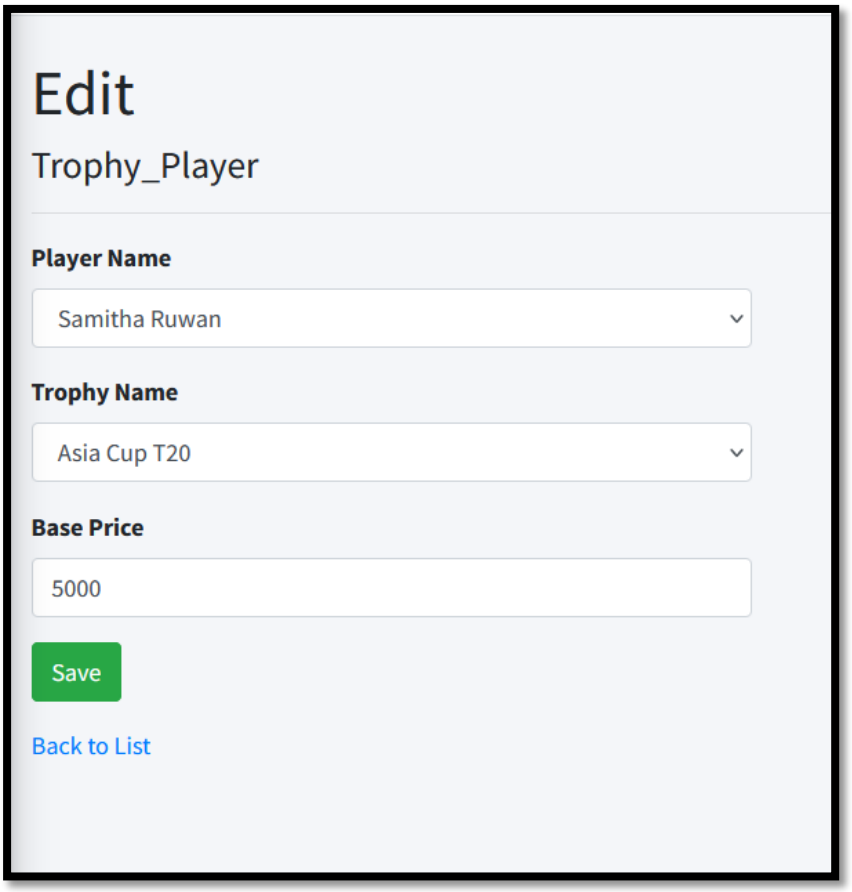


Figure 37 - Edit trophy player

18) Trophy player registration

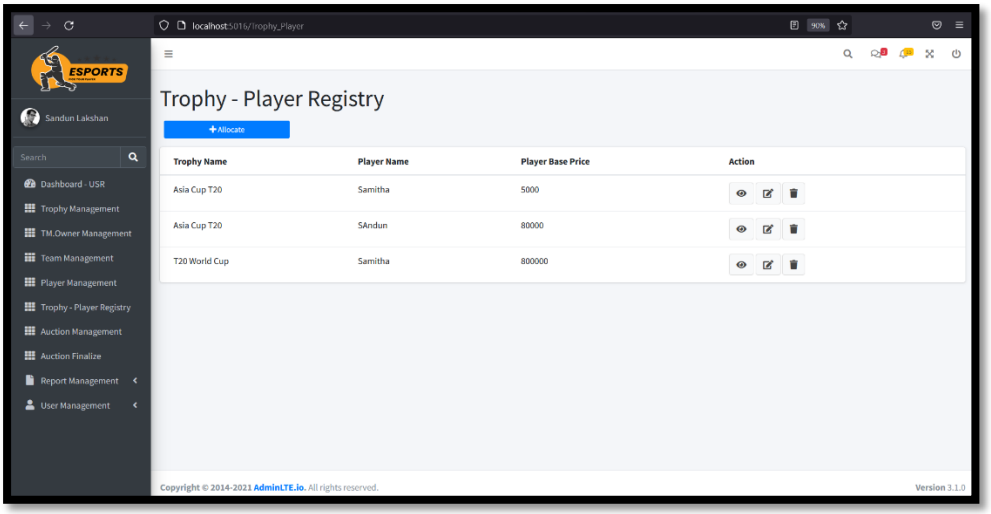


Figure 38 - Trophy player registration

- **Coding**

1) Ajax use to fill the data table

```
function get_bidding_det() {

    var trophyid = $('#comboboxtrophy option:selected').val();

    $.ajax({
        type: 'GET',
        data: {
            trophyid: trophyid,
        },
        url: '@Url.Action("LoadBidDetails","PlayerBidDetails")',
        dataType: 'json',
        success: function (response) {

            var len = response.length;

            $('#TblDataSet tbody').empty();

            for (var i = 0; i < len; i++) {

                var stat = response[i]['selectstatus'];

                var span = '';
                if (stat == 1){
                span = '<span class="right badge badge-success">SOLD</span>';
                } else {
                span = '<span class="right badge badge-danger">UN-SOLD</span>';
                }

                output = '<tr>';
                output += '<td class="text-center">' + (i + 1) + '</td>';
                output += '<td class="text-center">' + response[i]['teamOwnerFirstName'] + ' ' + response[i]['teamOwnerLastName'] + '</td>';
                output += '<td class="text-center"><a href="#" onclick="viewteamplayers(' + response[i]['teamid'] + ')">' + response[i]['teamname'] + '</a></td>';
                output += '<td class="text-center">' + response[i]['bidamount'] + '</td>';
                output += '<td class="text-center">' + span + '</td>';
                output += '</tr>';

                $('#TblDataSet tbody').append(output);

            }

        }
    });
}
```

Figure 39 - Ajax use to fill the data table

2) Application User Model

```
ApplicationUser.cs | AddDbContext.cs | Auction.cs
Project | Project.Models.ApplicationUser

using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace Project.Models
{
    22 references
    public class ApplicationUser
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        12 references
        public int UserId { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("First Name")]
        [Column(TypeName = "nvarchar(255)")]
        13 references
        public string FirstName { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("Last Name")]
        [Column(TypeName = "nvarchar(255)")]
        13 references
        public string LastName { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("User Name")]
        [Column(TypeName = "nvarchar(255)")]
        15 references
        public string UserName { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("Password")]
        [Column(TypeName = "nvarchar(255)")]
        15 references
        public string Password { get; set; }


    }
}
```

Figure 40 - Application User Model

3) Auction Model

```
Auction.cs
Project                                   Project.Models.Auction
using Microsoft.EntityFrameworkCore.Metadata.Internal;
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;

namespace Project.Models
{
    19 references
    public class Auction
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        13 references
        public int AuctionId { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("Team Owner")]
        [Column(TypeName = "int")]
        21 references
        public int TeamownId { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("Player Name")]
        [Column(TypeName = "int")]
        23 references
        public int Trophy_Player_Id { get; set; }

        [Required(ErrorMessage = "This Field is Required!")]
        [DisplayName("Bid Price")]
        [Column(TypeName = "float")]
        20 references
        public float Bid_price { get; set; }

        [Column(TypeName = "int")]
        2 references
        public int auction_select_stat { get; set; }
    }
}
```

Figure 41 - Auction Model

4) Create form code

```
Create.cshtml   _Layout.cshtml   BiddingController.cs   TrophyController.cs   Trophy_PlayerController.cs   TeamOwnerController.cs
@model Project.Models.Player

@{
    ViewData["Title"] = "Create";
}

<h1>Create</h1>

<h4>Player</h4>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Create">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="FirstName" class="control-label"></label>
                <input asp-for="FirstName" class="form-control" />
                <span asp-validation-for="FirstName" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="LastName" class="control-label"></label>
                <input asp-for="LastName" class="form-control" />
                <span asp-validation-for="LastName" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Contact_No" class="control-label"></label>
                <input asp-for="Contact_No" class="form-control" />
                <span asp-validation-for="Contact_No" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="UserName" class="control-label"></label>
                <input asp-for="UserName" class="form-control" />
                <span asp-validation-for="UserName" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-danger"></span>
            </div>
            <div class="form-group">
                <input type="submit" value="Create" class="btn btn-success" />
            </div>
        </form>
    </div>
</div>

<div>
83 %   No issues found
```

Figure 42 - Create form code

5) Data load to table

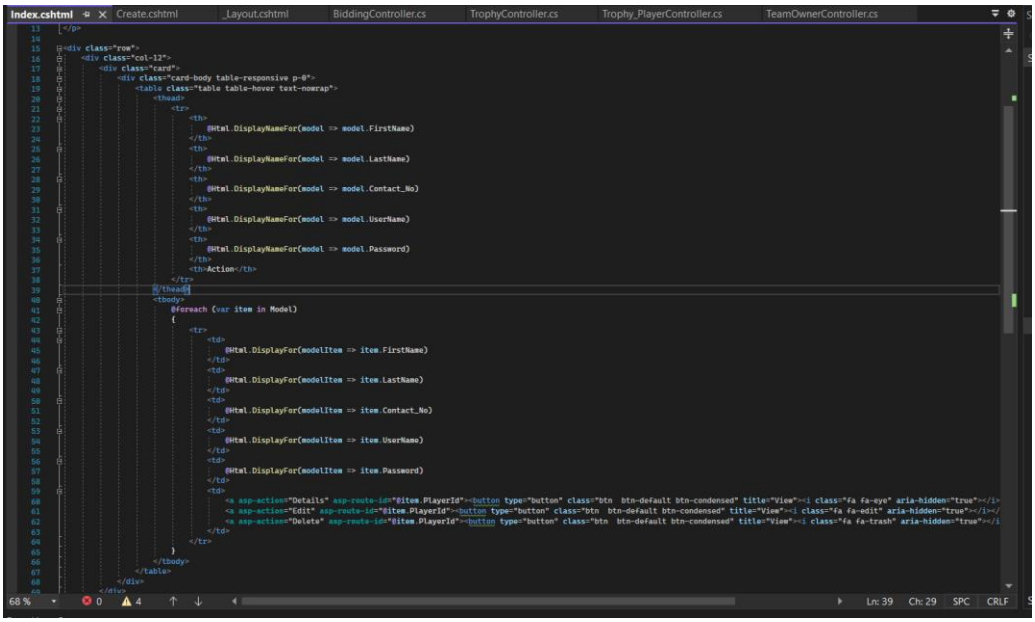


Figure 43 - Data load to table

6) Database context

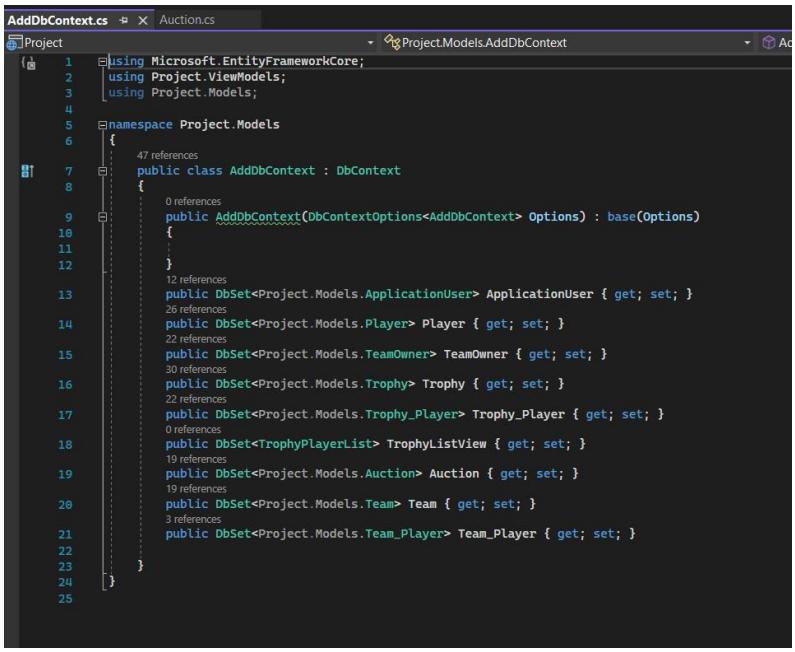


Figure 44 - Database context

7) Delete query

```
// POST: Player/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> DeleteConfirmed(int id)
{
    if (_context.Player == null)
    {
        return Problem("Entity set 'AddDbContext.Player'  is null.");
    }
    var player = await _context.Player.FindAsync(id);
    if (player != null)
    {
        _context.Player.Remove(player);
    }

    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}
```

Figure 45 - Delete query

8) Insert query

```
// POST: Player/Create
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Create([Bind("PlayerId,FirstName,LastName,Contact_No,UserName,Password")] Player player)
{
    if (ModelState.IsValid)
    {
        _context.Add(player);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(player);
}
```

Figure 46 - Insert query

9) json using data pass to view side

```
0 references
public ActionResult LoadTrophyPlayer(int trophyid, int teamownId)
{
    //List<TrophyPlayerList> trophyplayer = new List<TrophyPlayerList>();

    var trophyplayer = (from a in _context.Trophy_Player
                        join b in _context.Trophy
                        on a.TrophyId equals b.TrophyId
                        join c in _context.Player
                        on a.PlayerId equals c.PlayerId
                        where a.TrophyId == trophyid
                        where !_context.Auction.Any(e2 => e2.Trophy_Player_Id == a.Trophy_Player_Id && e2.TeamownId == teamownId)
                        select new
                        {
                            trophy_Player_Id = a.Trophy_Player_Id,
                            trophyName = b.TrophyName,
                            playerFirstName = c.FirstName,
                            playerLastName = c.LastName,
                            playerBasePrice = a.base_price
                        }
                    ).ToList();

    return Json(trophyplayer);
}
```

Figure 47 -json using data pass to view side

10) Login code

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Login(ApplicationUser appusr)
{
    var checklogin = _context.ApplicationUser.Where(x => x.UserName.Equals(appusr.UserName) && x.Password.Equals(appusr.Password)).FirstOrDefault();
    if (checklogin != null)
    {
        HttpContext.Session.SetString(SessionFName, checklogin.FirstName);
        HttpContext.Session.SetString(SessionLName, checklogin.LastName);
        HttpContext.Session.SetInt32(SessionUserId, checklogin.UserId);
        HttpContext.Session.SetString(SessionUserType, "USR");

        return RedirectToAction("Index", "Home", new { WelcomeMessage = "welcome" });
    }
    else
    {
        return RedirectToAction("Index", new { ErrorMessage = "fail" });
    }
    return View();

}
```

Figure 48 - Login code

11) Player Model

Player.cs | ApplicationUser.cs | AddDbContext.cs | Auction.cs

Project | Project.Models.Player

```
1   using Microsoft.EntityFrameworkCore.Metadata.Internal;
2   using System.ComponentModel.DataAnnotations.Schema;
3   using System.ComponentModel;
4   using System.ComponentModel.DataAnnotations;
5
6   namespace Project.Models
7   {
        22 references
8       public class Player
9       {
10          [Key]
11          [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            24 references
12          public int PlayerId { get; set; }
13
14          [Required(ErrorMessage = "This Field is Required!")]
15          [DisplayName("First Name")]
16          [Column(TypeName = "nvarchar(255)")]
            25 references
17          public string FirstName { get; set; }
18
19          [Required(ErrorMessage = "This Field is Required!")]
20          [DisplayName("Last Name")]
21          [Column(TypeName = "nvarchar(255)")]
            25 references
22          public string LastName { get; set; }
23
24          [Required(ErrorMessage = "This Field is Required!")]
25          [DisplayName("Contact Number")]
26          [Column(TypeName = "nvarchar(255)")]
            14 references
27          public string Contact_No { get; set; }
28
29          [Required(ErrorMessage = "This Field is Required!")]
30          [DisplayName("User Name")]
31          [Column(TypeName = "nvarchar(255)")]
            15 references
32          public string UserName { get; set; }
33
34          [Required(ErrorMessage = "This Field is Required!")]
35          [DisplayName("Password")]
36          [Column(TypeName = "nvarchar(255)")]
            15 references
37          public string Password { get; set; }
38      }
39  }
40
```

Figure 49 - Player Model

12) Session check code

```
namespace Project.Controllers
{
    1 reference
    public class PlayerController : Controller
    {
        private readonly AddDbContext _context;

        const string SessionUserId = "_UsrId";
        const string SessionFName = "_FName";
        const string SessionLName = "_LName";
        const string SessionUserType = "_UsrTyp";

        0 references
        public PlayerController(AddDbContext context)
        {
            _context = context;
        }

        // GET: Player
        3 references
        public async Task<IActionResult> Index()
        {
            ViewData["UsrId"] = HttpContext.Session.GetInt32(SessionUserId);
            ViewData["FName"] = HttpContext.Session.GetString(SessionFName);
            ViewData["LName"] = HttpContext.Session.GetString(SessionLName);
            ViewData["UsrTyp"] = HttpContext.Session.GetString(SessionUserType);

            if (HttpContext.Session.GetString(SessionFName) == null)
            {
                return RedirectToAction("Index", "Login");
            }
            else
            {
                return View(await _context.Player.ToListAsync());
            }

        }

        // GET: Player/Details/5
```

Figure 50 - session check code

13) Session clear code

```
[HttpGet]
0 references
public ActionResult SignOut()
{
    HttpContext.Session.Remove(SessionUserId);
    HttpContext.Session.Remove(SessionFName);
    HttpContext.Session.Remove(SessionLName);
    HttpContext.Session.Remove(SessionUserType);
    return RedirectToAction("Index", "Login");

}
```

Figure 51 - session clear code

14) Session create code

```
HttpContext.Session.SetString(SessionFName, checklogin.FirstName);
HttpContext.Session.SetString(SessionLName, checklogin.LastName);
HttpContext.Session.SetInt32(SessionUserId, checklogin.UserId);
HttpContext.Session.SetString(SessionUserType, "USR");

return RedirectToAction("Index", "Home", new { WelcomeMessage = "welcome" });
```

Figure 52 - Session create code

15) Sub query using code

```
0 references
public ActionResult LoadBiddingSelectedList(int trophyid, int teamid)
{
    var trophyplayer = (from a in _context.Team_Player
                        join b in _context.Team
                        on a.TeamId equals b.TeamId
                        join c in _context.Trophy_Player
                        on a.Trophy_Player_Id equals c.Trophy_Player_Id
                        join d in _context.Player
                        on c.PlayerId equals d.PlayerId
                        where b.TrophyId == trophyid && b.TeamId == teamid
                        select new
                        {
                            playerFirstName = d.FirstName,
                            playerLastName = d.LastName,
                            bidamount = (from h in _context.Auction where h.TeamownId == b.TeamownId && h.Trophy_Player_Id == c.Trophy_Player_Id select h.Bid_price).FirstOrDefault()

                        }
                    ).ToList();

    return Json(trophyplayer);
}
```

Figure 53 - Sub query using code

16) Team player Model

Team_Player.cs | Team.cs | Player.cs | ApplicationUser.cs | AddDbContext.cs

Project | Project.Models.Team_Player

```
1   using System.ComponentModel;
2   using System.ComponentModel.DataAnnotations;
3   using System.ComponentModel.DataAnnotations.Schema;
4
5   namespace Project.Models
6   {
        3 references
7       public class Team_Player
8       {
9           [Key]
10          [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            0 references
11          public int TeamPlayerId { get; set; }
12
13          [Column(TypeName = "int")]
            3 references
14          public int TeamId { get; set; }
15
16          [Column(TypeName = "int")]
            3 references
17          public int Trophy_Player_Id { get; set; }
18
19      }
20  }
21
```

Figure 54 - Team Player Model

17) Table join use in entity framework

```
// GET: Trophy_Player
3 references
public async Task<IActionResult> Index()
{
    ViewData["UsrId"] = HttpContext.Session.GetInt32(SessionUserId);
    ViewData["FName"] = HttpContext.Session.GetString(SessionFName);
    ViewData["LName"] = HttpContext.Session.GetString(SessionLName);
    ViewData["UsrTyp"] = HttpContext.Session.GetString(SessionUserType);

    if (HttpContext.Session.GetString(SessionFName) == null)
    {
        return RedirectToAction("Index", "Login");
    }
    else
    {
        List<TrophyPlayerList> trophyplayer = new List<TrophyPlayerList>();

        trophyplayer = (from a in _context.Trophy_Player
                        join b in _context.Trophy
                        on a.TrophyId equals b.TrophyId
                        join c in _context.Player
                        on a.PlayerId equals c.PlayerId
                        select new TrophyPlayerList
                        {
                            Trophy_Player_Id = a.Trophy_Player_Id,
                            TrophyName = b.TrophyName,
                            PlayerFirstName = c.FirstName,
                            PlayerLastName = c.LastName,
                            PlayerBasePrice = a.base_price
                        }
                        ).ToList();

        //return View(await _context.Trophy_Player.ToListAsync());

        return View(trophyplayer);
    }

}
```

Figure 55 - Table join use in entity framework

18) Team Model

Team.cs | Player.cs | ApplicationUser.cs | AddDbContext.cs | Auction.cs

Project | Project.Models.Team

```
1   using System.ComponentModel;
2   using System.ComponentModel.DataAnnotations;
3   using System.ComponentModel.DataAnnotations.Schema;
4
5   namespace Project.Models
6   {
        15 references
7       public class Team
8       {
9           [Key]
10          [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            15 references
11          public int TeamId { get; set; }
12
13          [Required(ErrorMessage = "This Field is Required!")]
14          [DisplayName("Trophy Name")]
15          [Column(TypeName = "int")]
            19 references
16          public int TrophyId { get; set; }
17
18          [Required(ErrorMessage = "This Field is Required!")]
19          [DisplayName("Team Owner Name")]
20          [Column(TypeName = "int")]
            19 references
21          public int TeamownId { get; set; }
22
23          [Required(ErrorMessage = "This Field is Required!")]
24          [DisplayName("Team Name")]
25          [Column(TypeName = "nvarchar(255)")]
            15 references
26          public string TeamName { get; set; }
27
28          [Required(ErrorMessage = "This Field is Required!")]
29          [DisplayName("Team Maximum Value")]
30          [Column(TypeName = "float")]
            12 references
31          public float Team_max_value { get; set; }
32
33      }
34  }
35
```

Figure 56 - Team Model

19) Team Owner Model

TeamOwner.cs | Team_Player.cs | Team.cs | Player.cs | ApplicationUser.cs

Project | Project.Models.TeamOwner

```
1   using Microsoft.EntityFrameworkCore.Metadata.Internal;
2   using System.ComponentModel.DataAnnotations.Schema;
3   using System.ComponentModel;
4   using System.ComponentModel.DataAnnotations;
5
6   namespace Project.Models
7   {
        22 references
8       public class TeamOwner
9       {
10          [Key]
11          [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            25 references
12          public int TeamownId { get; set; }
13
14          [Required(ErrorMessage = "This Field is Required!")]
15          [DisplayName("First Name")]
16          [Column(TypeName = "nvarchar(255)")]
            22 references
17          public string FirstName { get; set; }
18
19          [Required(ErrorMessage = "This Field is Required!")]
20          [DisplayName("Last Name")]
21          [Column(TypeName = "nvarchar(255)")]
            22 references
22          public string LastName { get; set; }
23
24          [Required(ErrorMessage = "This Field is Required!")]
25          [DisplayName("NIC")]
26          [Column(TypeName = "nvarchar(255)")]
            13 references
27          public string NIC { get; set; }
28
29          [Required(ErrorMessage = "This Field is Required!")]
30          [DisplayName("Contact Number")]
31          [Column(TypeName = "nvarchar(255)")]
            13 references
32          public string Contact_No { get; set; }
33
34          [Required(ErrorMessage = "This Field is Required!")]
35          [DisplayName("User Name")]
36          [Column(TypeName = "nvarchar(255)")]
            15 references
37          public string UserName { get; set; }
38
39          [Required(ErrorMessage = "This Field is Required!")]
40          [DisplayName("Password")]
41          [Column(TypeName = "nvarchar(255)")]
            15 references
42          public string Password { get; set; }
43
44      }
45  }
46
```

Figure 57 - Team Owner Model

20) Trophy Model

Trophy.cs | TeamOwner.cs | Team_Player.cs | Team.cs | Player.cs | ApplicationUs

Project | Project.Models.Trophy

```
1   using Microsoft.EntityFrameworkCore.Metadata.Internal;
2   using System.ComponentModel.DataAnnotations.Schema;
3   using System.ComponentModel;
4   using System.ComponentModel.DataAnnotations;
5
6   namespace Project.Models
7   {
        18 references
8       public class Trophy
9       {
10          [Key]
11          [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            22 references
12          public int TrophyId { get; set; }
13
14          [Required(ErrorMessage = "This Field is Required!")]
15          [DisplayName("Trophy Name")]
16          [Column(TypeName = "nvarchar(255)")]
            23 references
17          public string TrophyName { get; set; }
18
19          [Required(ErrorMessage = "This Field is Required!")]
20          [DisplayName("Duration From")]
21          [Column(TypeName = "datetime")]
            12 references
22          public DateTime DurationFrom { get; set; }
23
24          [Required(ErrorMessage = "This Field is Required!")]
25          [DisplayName("Duration To")]
26          [Column(TypeName = "datetime")]
            12 references
27          public DateTime DurationTo { get; set; }
28
29      }
30  }
31
```

Figure 58 - Trophy Model

21) Trophy player Model

Trophy_Player.cs | Trophy.cs | TeamOwner.cs | Team_Player.cs | Team.cs | Player.cs

Project | Project.Models.Trophy_Player

```
1   using Microsoft.EntityFrameworkCore.Metadata.Internal;
2   using System.ComponentModel.DataAnnotations.Schema;
3   using System.ComponentModel;
4   using System.ComponentModel.DataAnnotations;
5
6   namespace Project.Models
7   {
        0 references
8       public class Trophy_Player
9       {
10          [Key]
11          [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            0 references
12          public int Trophy_Player_Id { get; set; }
13
14          [Required(ErrorMessage = "This Field is Required!")]
15          [DisplayName("Player Name")]
16          [Column(TypeName = "int")]
            20 references
17          public int PlayerId { get; set; }
18
19          [Required(ErrorMessage = "This Field is Required!")]
20          [DisplayName("Trophy Name")]
21          [Column(TypeName = "int")]
            26 references
22          public int TrophyId { get; set; }
23
24          [Required(ErrorMessage = "This Field is Required!")]
25          [DisplayName("Base Price")]
26          [Column(TypeName = "float")]
            14 references
27          public float base_price { get; set; }
28
29          [Column(TypeName = "int")]
            2 references
30          public int bid_close_stat { get; set; }
31
32      }
33  }
34
```

Figure 59 - Trophy Player Model

22) Update query

```
// POST: Player/Edit/5
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Edit(int id, [Bind("PlayerId,FirstName,LastName,Contact_No,UserName,Password")] Player player)
{
    if (id != player.PlayerId)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(player);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!PlayerExists(player.PlayerId))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(player);
}
```

Figure 60 - Update query

23) User Menu Code

```
_Layout.cshtml | BiddingController.cs | TrophyController.cs | Trophy_PlayerController.cs | TeamOwnerController.cs | LoginController.cs | PlayerController.cs

                    </ul>
                </li>-->

                <li class="nav-item">
                    <a asp-area="" asp-controller="Home" asp-action="Index" class="nav-link">
                        <i class="nav-icon fas fa-tachometer-alt"></i>
                        <p>
                            Dashboard - @ViewBag.UsrTyp
                        </p>
                    </a>
                </li>

                @if (ViewBag.UsrTyp == "USR")
                {
                    <li class="nav-item">
                        <a asp-area="" asp-controller="Trophy" asp-action="Index" class="nav-link">
                            <i class="nav-icon fas fa-th"></i>
                            <p>
                                Trophy Management
                            </p>
                        </a>
                    </li>
                }

                @if (ViewBag.UsrTyp == "USR")
                {

                    <li class="nav-item">
                        <a asp-area="" asp-controller="TeamOwner" asp-action="Index" class="nav-link">
                            <i class="nav-icon fas fa-th"></i>
                            <p>
                                TM.Owner Management
                            </p>
                        </a>
                    </li>

                }

                @if (ViewBag.UsrTyp == "USR")
                {

                    <li class="nav-item">
                        <a asp-area="" asp-controller="Team" asp-action="Index" class="nav-link">
                            <i class="nav-icon fas fa-th"></i>
                            <p>
                                Team Management
                            </p>
```

Figure 61 - User Menu Code

24) View player details

```
// GET: Player/Details/5
0 references
public async Task<IActionResult> Details(int? id)
{
    ViewData["UsrId"] = HttpContext.Session.GetInt32(SessionUserId);
    ViewData["FName"] = HttpContext.Session.GetString(SessionFName);
    ViewData["LName"] = HttpContext.Session.GetString(SessionLName);
    ViewData["UsrTyp"] = HttpContext.Session.GetString(SessionUserType);

    if (HttpContext.Session.GetString(SessionFName) == null)
    {
        return RedirectToAction("Index", "Login");
    }
    else
    {
        if (id == null || _context.Player == null)
        {
            return NotFound();
        }

        var player = await _context.Player
            .FirstOrDefaultAsync(m => m.PlayerId == id);
        if (player == null)
        {
            return NotFound();
        }

        return View(player);
    }
}
```

Figure 62 - View player details

- **Coding best practices followed in this system development.**

Use of Camel Case.

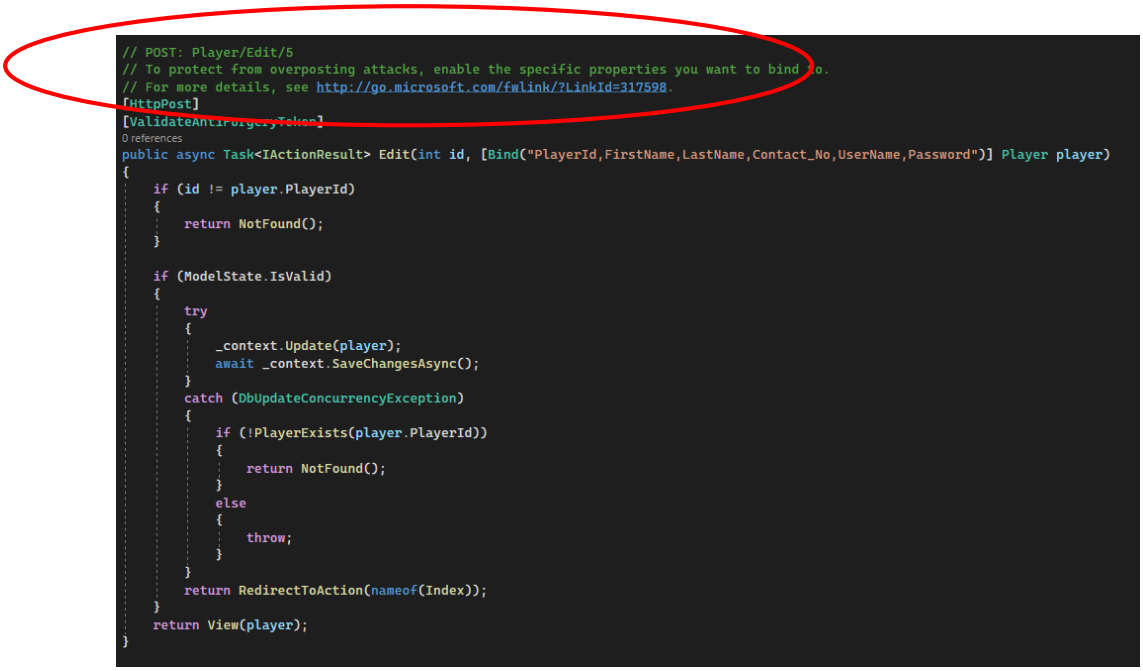
Variable naming is a significant factor in making the code readable.

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Edit(int id, [Bind("UserId,FirstName,LastName,UserName,Password")] ApplicationUser applicationUser)
{
    if (id != applicationUser.UserId)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(applicationUser);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!ApplicationUserExists(applicationUser.UserId))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(applicationUser);
}

// GET: ApplicationUsers/Delete/5
```

Figure 63 - Use of Camel Case.

Use of comments

Commenting may be the most influential form manage and segmenting code



```
// POST: Player/Edit/5
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Edit(int id, [Bind("PlayerId,FirstName,LastName,Contact_No,UserName,Password")] Player player)
{
    if (id != player.PlayerId)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(player);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!PlayerExists(player.PlayerId))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(player);
}
```

Figure 64 - Comments 1

```
// GET: Player/Details/5
0 references
public async Task<IActionResult> Details(int? id)
{
    ViewData["UsrId"] = HttpContext.Session.GetInt32(SessionUserId);
    ViewData["FName"] = HttpContext.Session.GetString(SessionFName);
    ViewData["LName"] = HttpContext.Session.GetString(SessionLName);
    ViewData["UsrTyp"] = HttpContext.Session.GetString(SessionUserType);

    if (HttpContext.Session.GetString(SessionFName) == null)
    {
        return RedirectToAction("Index", "Login");
    }
    else
    {
        if (id == null || _context.Player == null)
        {
            return NotFound();
        }

        var player = await _context.Player
            .FirstOrDefaultAsync(m => m.PlayerId == id);
        if (player == null)
        {
            return NotFound();
        }

        return View(player);
    }

}
```

Figure 65 - Comments 2

```
// GET: Trophy_Player
3 references
public async Task<IActionResult> Index()
{
    ViewData["UsrId"] = HttpContext.Session.GetInt32(SessionUserId);
    ViewData["FName"] = HttpContext.Session.GetString(SessionFName);
    ViewData["LName"] = HttpContext.Session.GetString(SessionLName);
    ViewData["UsrTyp"] = HttpContext.Session.GetString(SessionUserType);

    if (HttpContext.Session.GetString(SessionFName) == null)
    {
        return RedirectToAction("Index", "Login");
    }
    else
    {
        List<TrophyPlayerList> trophyplayer = new List<TrophyPlayerList>();

        trophyplayer = (from a in _context.Trophy_Player
                        join b in _context.Trophy
                        on a.TrophyId equals b.TrophyId
                        join c in _context.Player
                        on a.PlayerId equals c.PlayerId
                        select new TrophyPlayerList
                        {
                            Trophy_Player_Id = a.Trophy_Player_Id,
                            TrophyName = b.TrophyName,
                            PlayerFirstName = c.FirstName,
                            PlayerLastName = c.LastName,
                            PlayerBasePrice = a.base_price
                        }
                        ).ToList();

        //return View(await _context.Trophy_Player.ToListAsync());

        return View(trophyplayer);
    }

}
```

Figure 66 - Comments 3

Use of Exception Handling

Exception handling can control run tune errors that occur in the program.

```
// POST: Player/Edit/5
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Edit(int id, [Bind("PlayerId,FirstName,LastName,Contact_No,UserName,Password")] Player player)
{
    if (id != player.PlayerId)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(player);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!PlayerExists(player.PlayerId))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(player);
}
```

Figure 67 - Exception Handling 1

6) Detailed instruction with HARDWARE INTERFACE.

HARDWARE CONFIGURATION	
Processor	Intel Core i3 -1115G4 (11th Gen processor)
Ram	8 GB DDR4
Hard Disk	1 TB
LCD Monitor	24inch IPS ASUA
Keyboard	A4 tech keyboard and mouse
Mouse	PS/2 | USB
Printer	Ink-jet | Laser

Table 4 - Hardware Configuration

7) Detailed instruction with SOFTWARE INTERFACE.

SOFTWARE CONFIGURATION	
Operating System	Windows 10
Language	High Level Programming Language ASP.Net
Database	Microsoft SQL server
Other installations	Office 2019 with license version

Table 5 - Software Configuration

8) Object Oriented Design

- **Encapsulation**

Encapsulation is denoted 'as the procedure of having one or more items within a physical or logical package. Encapsulation in object-oriented programming methodology prevents access to performance details.

Abstraction and encapsulation are related characteristics in object-oriented programming. Abstraction lets for making appropriate data visible, and encapsulation allows a programmer to execute the expected level of abstraction.

Encapsulation is implemented by utilizing access specifies. C# helps the below access specifies −

- Public
- Private
- Protected
- Internal
- Protected internal

```
namespace Project.Controllers
{
    1 reference
    public class ApplicationUsersController : Controller
    {
        private readonly AddDbContext _context;

        const string SessionUserId = "_UsrId";
        const string SessionFName = "_FName";
        const string SessionLName = "_LName";
        const string SessionUserType = "_UsrTyp";

        0 references
        public ApplicationUsersController(AddDbContext context)
        {
            _context = context;
        }
```

Figure 68 – Encapsulation 01

```
namespace Project.Models
{
    47 references
    public class AddDbContext : DbContext
    {
        0 references
        public AddDbContext(DbContextOptions<AddDbContext> Options) : base(Options)
        {

        }
        12 references
        public DbSet<Project.Models.ApplicationUser> ApplicationUser { get; set; }
        26 references
```

Figure 69 - Encapsulation 2

- **Inheritance**

One of the most significant ideas in object-oriented programming is inheritance. Inheritance permits us to specify or define a class in terms of another class, which causes it easier to design and maintain an application. And also, supplies an opportunity to reuse the code functionality and runs up implementation time.

When developing a class, instead of writing completely new data members and member functions, the programmer can specify that the new class should inherit the members of a current class. This current class is named the base class, and the new class is referred to as the derived class.

```
namespace Project.Models
{
    47 references
    public class AddDbContext : DbContext
    {
        0 references
        public AddDbContext(DbContextOptions<AddDbContext> Options) : base(Options)
        {

        }
        12 references
        public DbSet<Project.Models.ApplicationUser> ApplicationUser { get; set; }
        26 references
```

Figure 70 - Database context

```
namespace Project.Models
{
    47 references
    public class AddDbContext : DbContext
    {
        0 references
        public AddDbContext(DbContextOptions<AddDbContext> Options) : base(Options)
        {

        }
        12 references
        public DbSet<Project.Models.ApplicationUser> ApplicationUser { get; set; }
        26 references
```

Figure 71 Player Model

9) Testing

System testing is explained as the procedure of determining whether or not a complete and integrated software outcome meets the needed standards. The system is put to the test by operating the procedures listed below:

- **Black Box Testing:** The testing team does black box testing, which indicates that no knowledge of the code's underlying design is essential, and each function must be tested.
- **White-box Testing:** White Box Testing is a software testing method that analyzes the product's fundamental structure, design, and code to check the input-output flow and enhance planning, usability, and security.
- **User Acceptance Testing:** User Acceptance Testing is a kind of testing in which the software system is confirmed and accepted by the end user or customer. The criteria are then designed to receive feedback on the solution, and appropriate revisions are made in response to the comments.

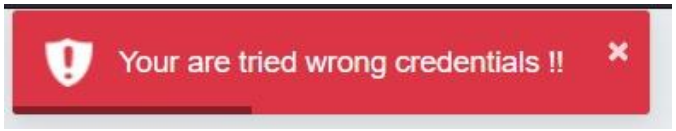
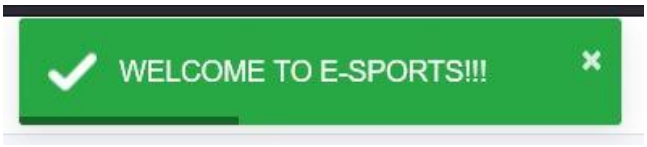
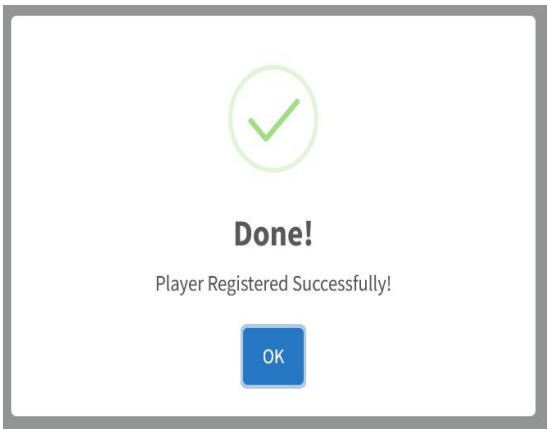
Test ID	Expected result	Actual Outcome	Result
1	Displaying error message		Pass
2	Able login to the profile		Pass
3	Displaying successful message		Pass

Table 6 - Testing 1

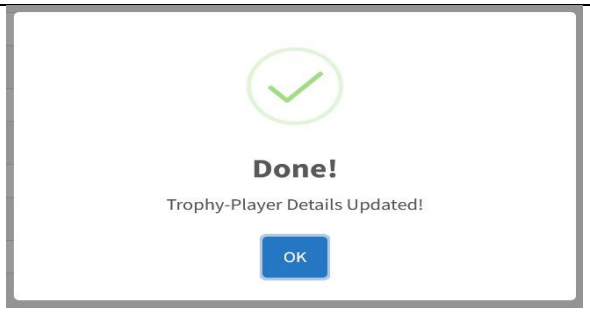
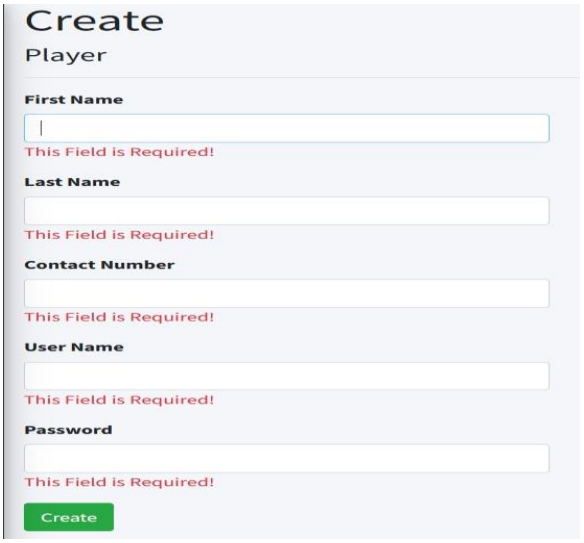
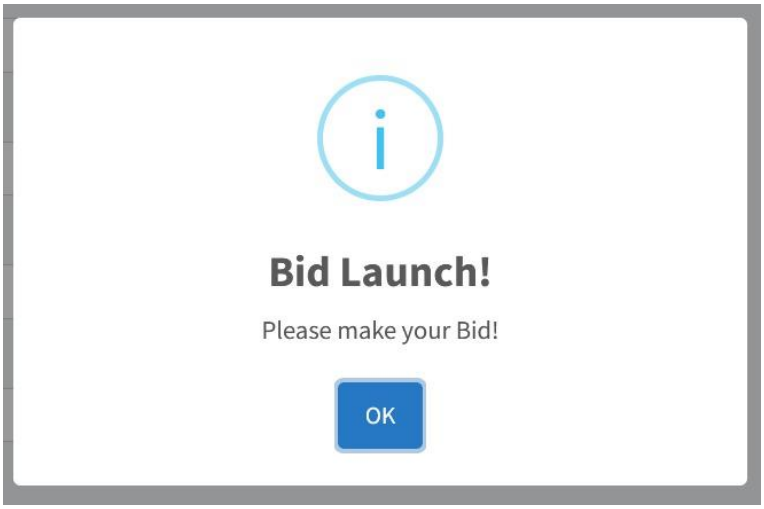
Test ID	Expected result	Actual Outcome	Result
4	Displaying successful message		Pass
5	Displaying error message		Pass
6	Team owners able bid for players		Pass

Table 7 - Testing 2

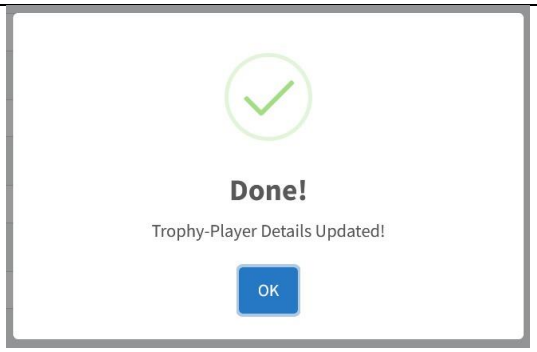
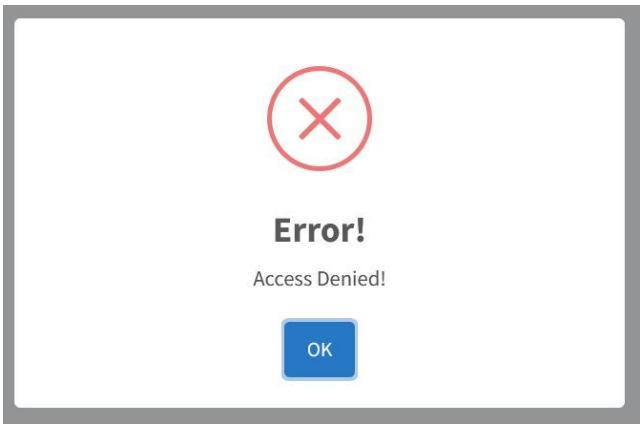
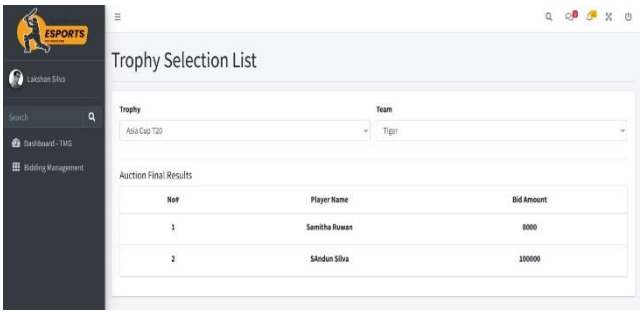
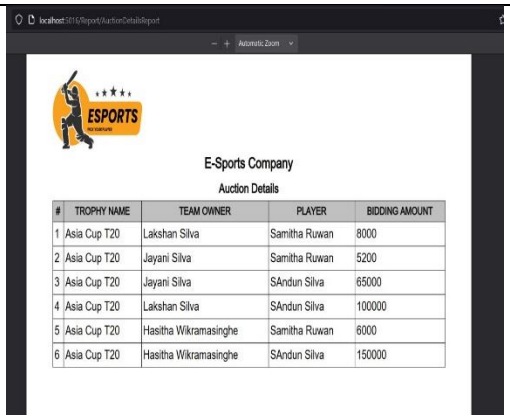
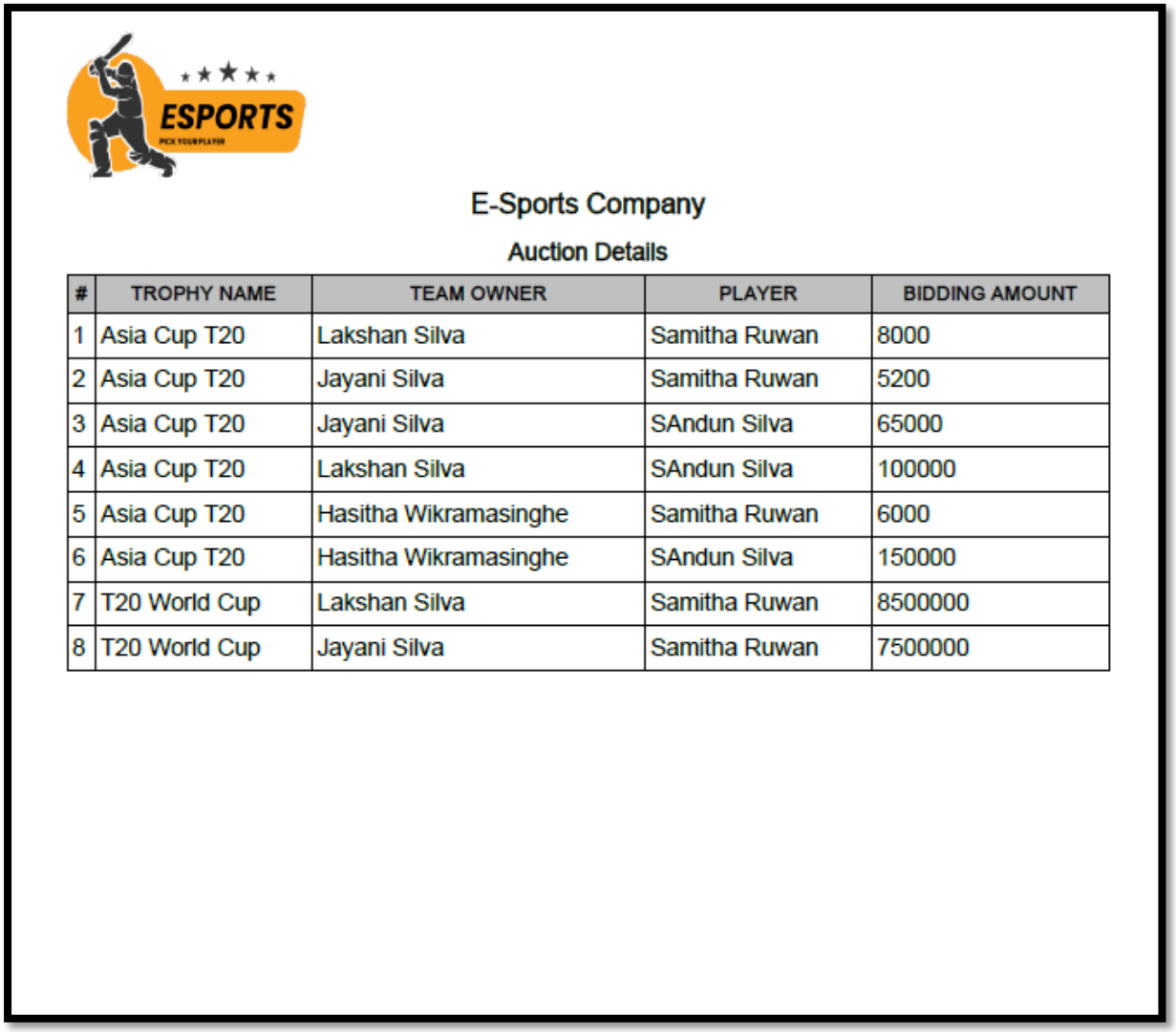
Test ID	Expected result	Actual Outcome	Result
7	Admin able to update player base price		Pass
8	Displaying error message		Pass
9	Team owners able to view team status		Pass
10	Relevant role can generate various types of reports		Pass

Table 8 - Testing 3

10) Reports

1) Auction details



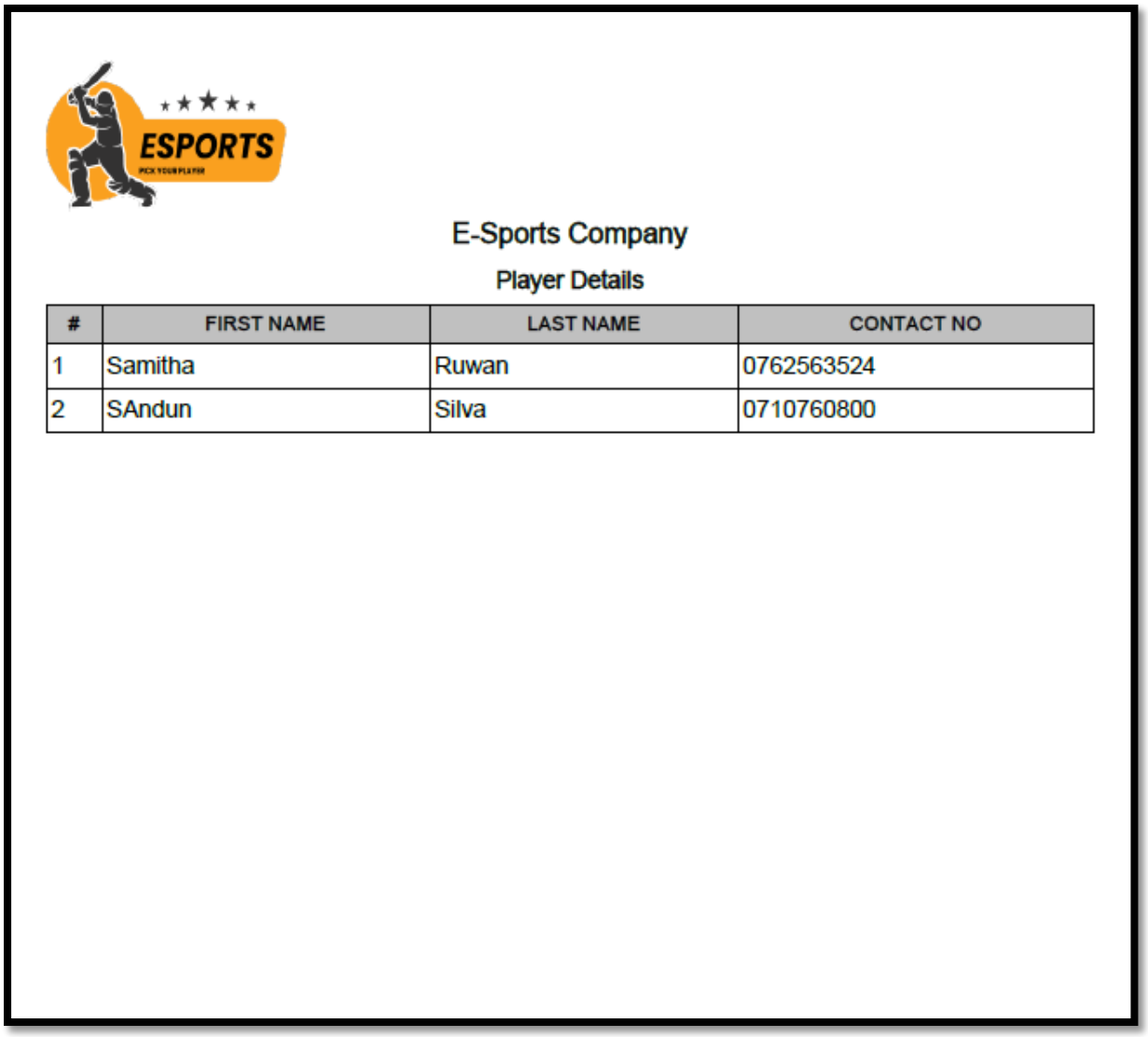
E-Sports Company

Auction Details

#	TROPHY NAME	TEAM OWNER	PLAYER	BIDDING AMOUNT
1	Asia Cup T20	Lakshan Silva	Samitha Ruwan	8000
2	Asia Cup T20	Jayani Silva	Samitha Ruwan	5200
3	Asia Cup T20	Jayani Silva	SAndun Silva	65000
4	Asia Cup T20	Lakshan Silva	SAndun Silva	100000
5	Asia Cup T20	Hasitha Wikramasinghe	Samitha Ruwan	6000
6	Asia Cup T20	Hasitha Wikramasinghe	SAndun Silva	150000
7	T20 World Cup	Lakshan Silva	Samitha Ruwan	8500000
8	T20 World Cup	Jayani Silva	Samitha Ruwan	7500000

Figure 72 - auction details

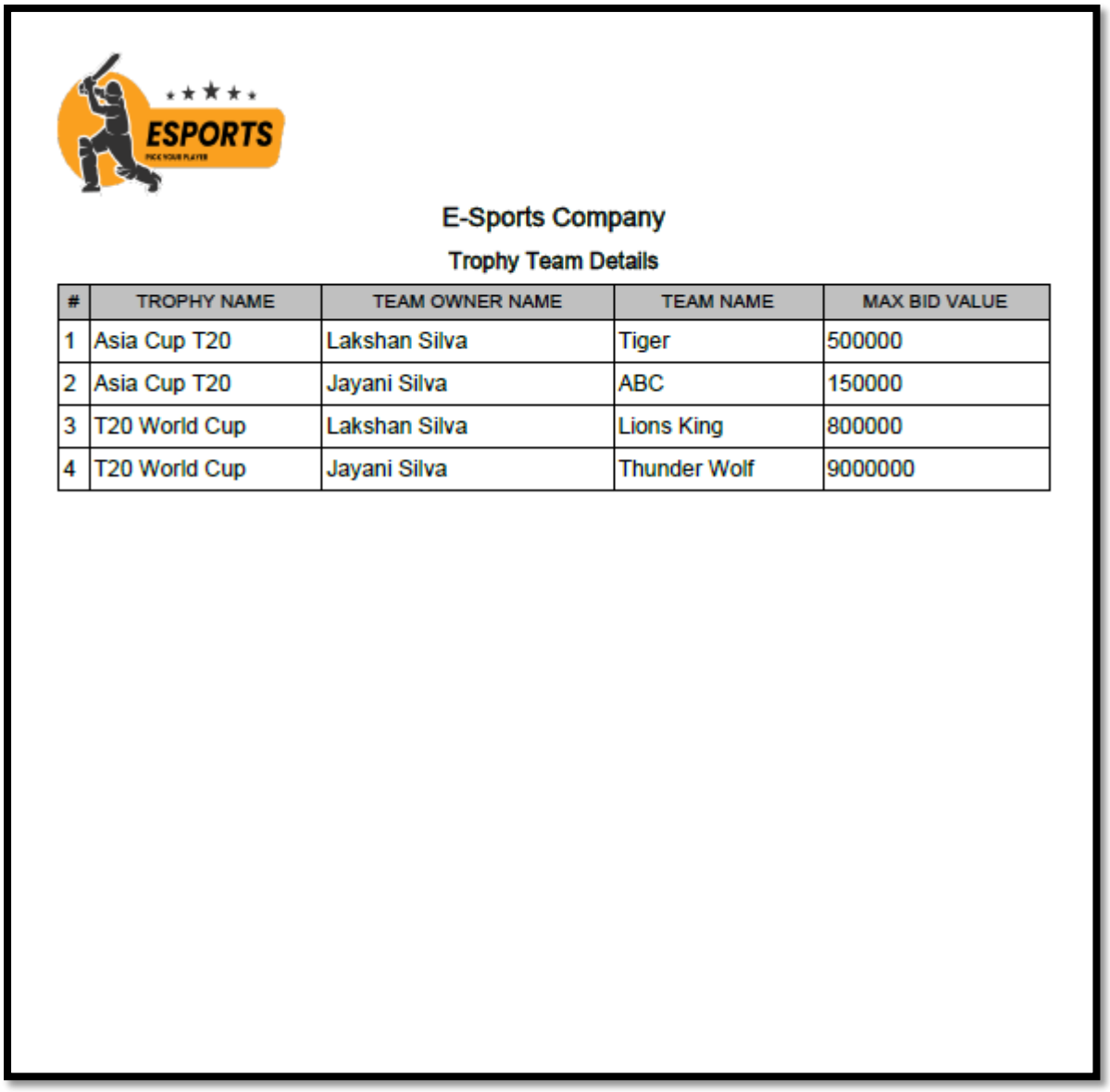
2) Player details



#	FIRST NAME	LAST NAME	CONTACT NO
1	Samitha	Ruwan	0762563524
2	SAndun	Silva	0710760800

Figure 73 - Player details

3) Team details



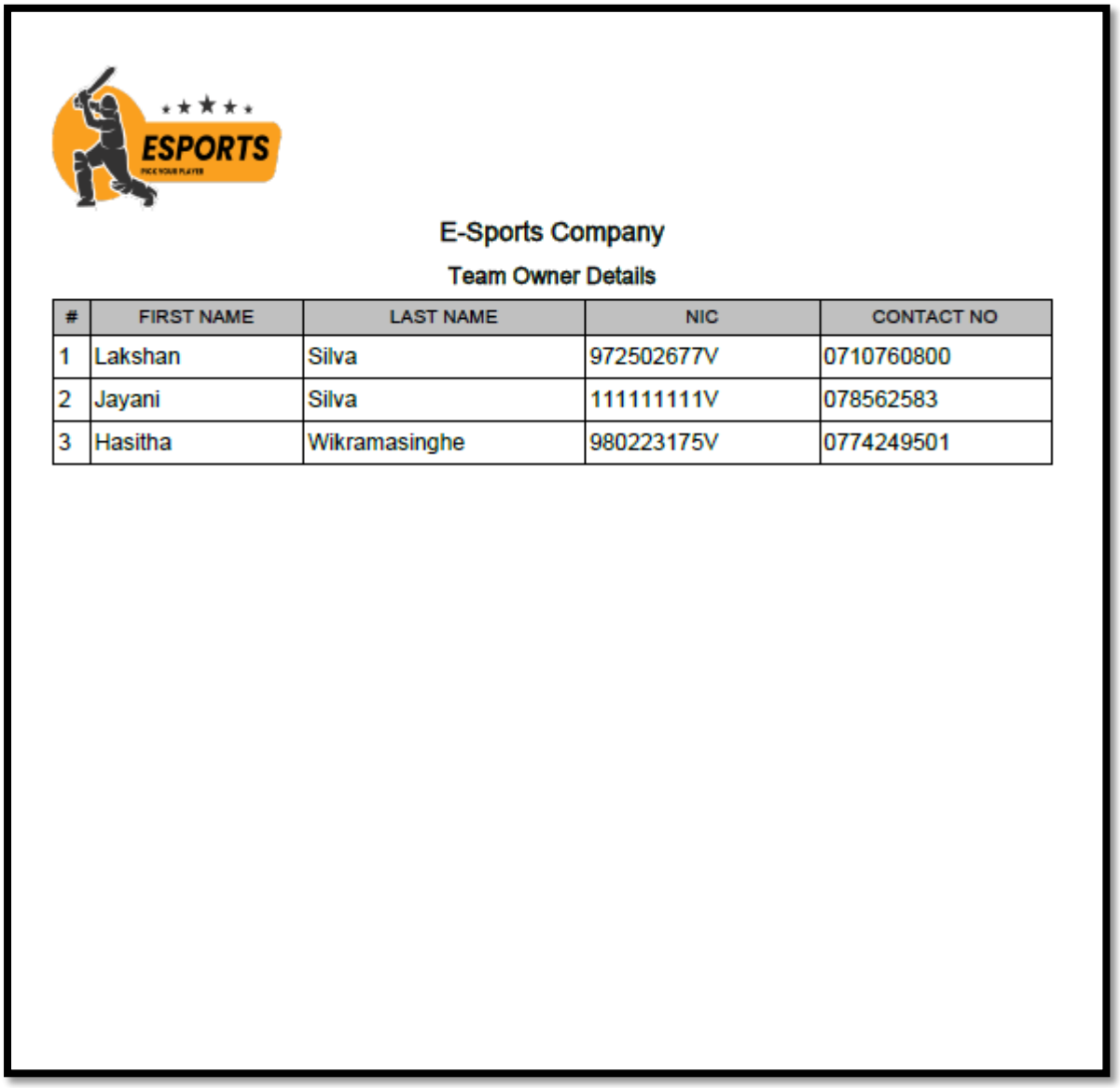
E-Sports Company

Trophy Team Details

#	TROPHY NAME	TEAM OWNER NAME	TEAM NAME	MAX BID VALUE
1	Asia Cup T20	Lakshan Silva	Tiger	500000
2	Asia Cup T20	Jayani Silva	ABC	150000
3	T20 World Cup	Lakshan Silva	Lions King	800000
4	T20 World Cup	Jayani Silva	Thunder Wolf	9000000

Figure 74 - Team details

4) Team owner details



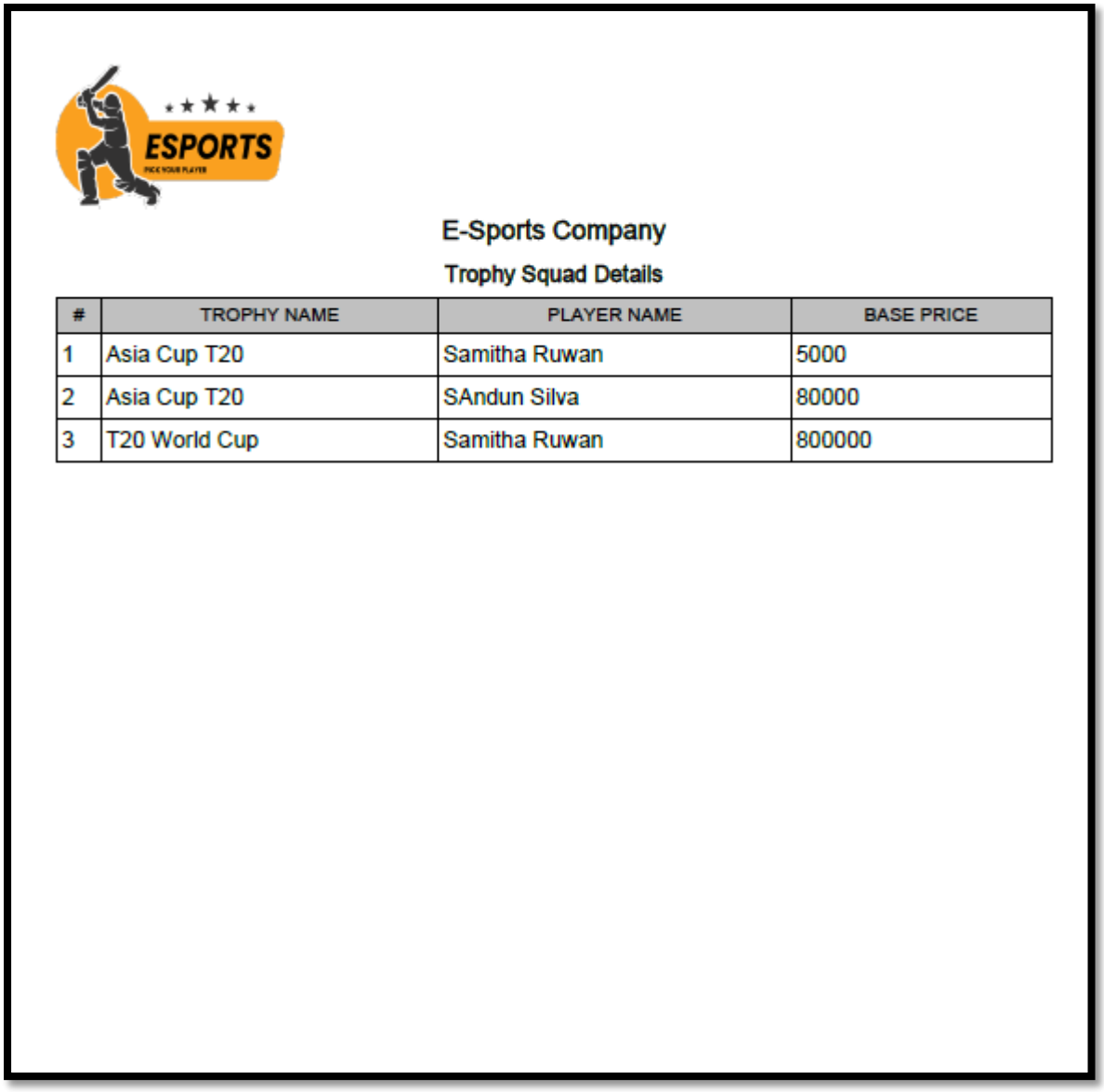
E-Sports Company

Team Owner Details

#	FIRST NAME	LAST NAME	NIC	CONTACT NO
1	Lakshan	Silva	972502677V	0710760800
2	Jayani	Silva	111111111V	078562583
3	Hasitha	Wikramasinghe	980223175V	0774249501

Figure 75 - Team owner details

5) Trophy participation (Squad) details



E-Sports Company

Trophy Squad Details

#	TROPHY NAME	PLAYER NAME	BASE PRICE
1	Asia Cup T20	Samitha Ruwan	5000
2	Asia Cup T20	SAndun Silva	80000
3	T20 World Cup	Samitha Ruwan	800000

Figure 76 - Trophy participation (Squad) details

11) Group Details

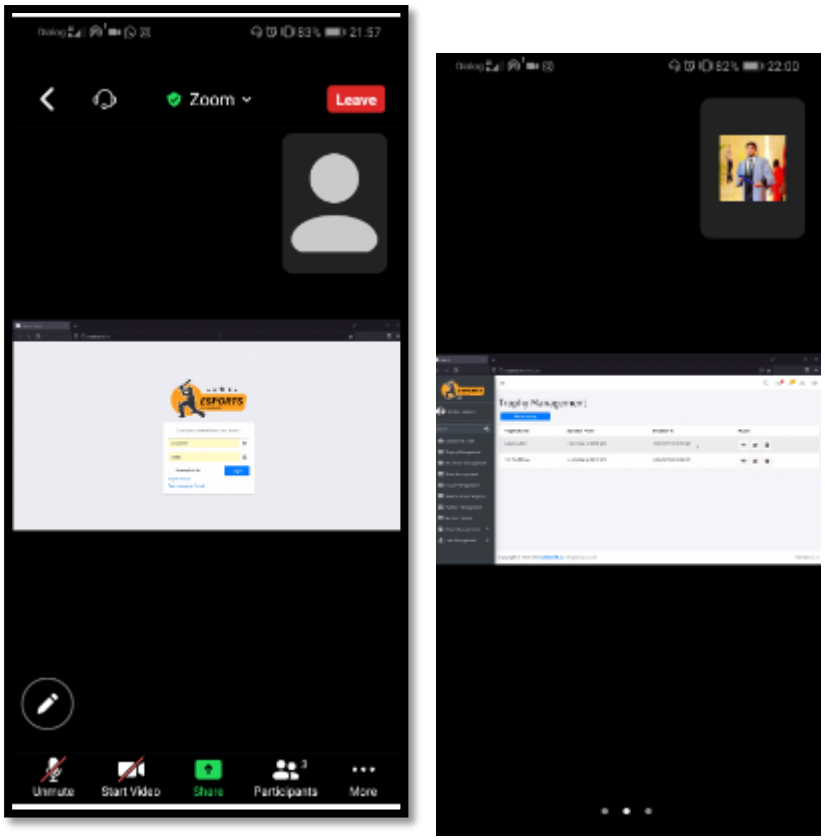


Figure 77 - Group meeting 1 and 2

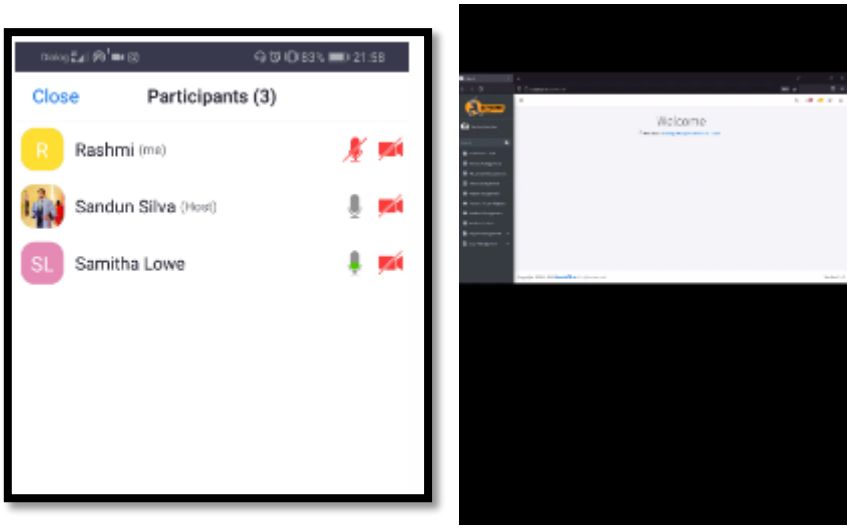


Figure 78 - Group meeting 3 and 4

12) Reflection of Own Experience

Group Member: S. Sandun Lakshan Silva / 22015392

In this assignment totally different between my previous assignment, the reason is previous assignments I have developed system using my own knowledge of coding practices but in this program, I have to learn about new framework as ASP.NET with Entity Framework. This was a bit different understand to front end coding and backend coding. However, finally I'm done this system with my friends and lectures help. And also, I get more helps on Youtube, Stack Overflow and GitHub platforms.

My own experience of this framework confuses between I learned other framework due to some of syntax are very different and it behavior also different. Furthermore, first stage in this system I have very hard experience about that language due to MVC framework. But finally, I found easy way to develop that system I also use scaffolding and raiser pages. This was very help me to done this system easily. Moreover, for more experience, I use ajax, jquery like supportive languages to implement this system. I learned on this framework has lot of OOP concepts through develop application with secured. And also, C# is the straightforward language to work with and it is easy to get the desired result quickly. Finally, I have the best confident about this framework is best way to develop business applications.

Group Member: W. Samitha Ruwan Lowe / 22015393

This project was a very successful group project to build up my knowledge. By doing this project, I've learned lots of things rather than team spirit. When we got the topic, first of all, we referred to some auction systems to get a general idea about the system. Then the main problem that we faced is the ASP.net framework. Until we start this project, I didn't have a clear idea about ASP.net and MVC architecture.

Finally, this project was very successful since we did this project with good team spirit. My knowledge of the C# language has been advantageous because it lets us finish any task without sacrificing functionality. Finally, my overall experience with this system implementation has been excellent and useful in preparing me for new challenges.

Group Member: W. Rashmi Sharon Fernando / 22015391

This is one of the great opportunities for me to enhance my knowledge doing such kind of a group project, lots of advantages could be obtained to the life. Through this system, some good qualities were highlighted, such as team spirit, time management skills, dedication, aim to reach the goal as a team without giving up any team member, understanding, and providing own maximum to the group project otherwise, it will be a bad impact to the other members. When developing this system new valuable things could be added to the knowledge because an auction system is a complicated software tool. So as a developer and a member of the development team has to cover a huge area and have to think of lots of criteria to succeed in this project. This is the first time that I worked with Model-View-Controller (MVC) in an ASP.NET application, and C# is a more specific language. It was a little bit hard to comprehend all the procedures of merging backend coding and front end. However, I worked on the database. I could design the proper database structure and could execute it properly. It suits the requirement of this auction system. I would like to mention that the overall experience with this execution, which I gained, was amazing, and unique. I learned a lot of new things that will help me in the future. I prepared a detailed document that provides a proper idea about this E-SPORT auction system, which was developed and designed by us. When anyone goes through this document, he or she can gain a better understanding of the system.

13) Individual task

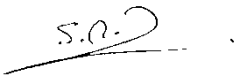
LMU ID	Student Name	Contribution for the project		Signature of the student
		Development process	**Documentation**	
22015391	Rashmi Fernando	Database	Overall documentation	
			Use case Diagram	
			Assumptions	
22015392	Sandun Lakshan	Coding	Class diagram	
			Test case	
		Report generation	Test plan	
		Login	System screenshots	
22015393	Samitha Lowe	UI Design	ER Diagram	
			Relational schema	
			Project plan	
			Gantt chart	

Table 9 - Individual task

Conclusion

"As with all products, the cheapest option is not always the best option – and the same goes for the most expensive choice." There are many cheap products that will end up costing money if people are not careful when buying an auction software system for cricket. Take the time to do some expansive research before investing in the correct system for the hotel business. Make sure to get feedback from each department and understand their specific requirements before deciding on a solution.

Auction systems for cricket are now-a-day have the benefit of modernization. Computers have done the work more efficiently. Computers is playing an essential role in management. Reports are generated by the management. This system has also a primary objective to provide facilities to customers. Software for computers makes things many times easy, these are made as user friendly to maintain checks and balances in hotel management and accounts as well. So, these things are important

Reference

Katewa, S. (2021) *How does the IPL auction work? (with Rules & Process), Cricket Mastery*. Available at: https://cricketmastery.com/how-does-the-ipl-auction-work/ (Accessed: November 8, 2022).

Why cheapest isn't always best - you get what you pay for (2021) *Build NATiVE*. Available at: https://www.buildnative.com/why-cheapest-isnt-always-best/ (Accessed: November 8, 2022).

Gantt chart

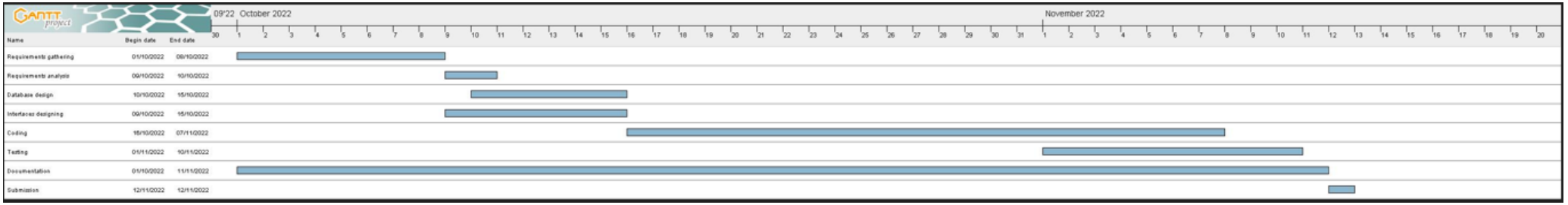


Figure 79 - Gantt chart