

PHIẾU DUYỆT ĐỒ ÁN TỐT NGHIỆP

I. Phần dành cho Sinh viên

1. Họ và tên: Lê Anh Tuấn 2. Mã Sinh viên: 102140230 3. Lớp: 14TCLC2
4. Tên đề tài: Building the web system connecting football teams
5. Số điện thoại: 0378861023 6. E-mail: leanhtuan.it14@gmail.com
7. Họ và tên GVHD: Phạm Công Thắng

II. Phần dành cho Hội đồng

STT	Nội dung đánh giá	Kết luận
1.	Trình bày báo cáo theo đúng mẫu qui định của Khoa	
2.	Không có sự sao chép nội dung báo cáo và chương trình đã có	
3.	Biên dịch mã nguồn và chạy được chương trình	
4.	Có kịch bản thực hiện với dữ liệu thử nghiệm	
5.	Kết quả thực hiện chương trình đúng theo báo cáo	
6.	Có sự đóng góp, phát triển của tác giả trong đồ án	

Ý kiến khác:

.....

Kết luận: ☐ Đạt yêu cầu ☐ Phải sửa chữa lại ☐ Không đạt yêu cầu

Đà Nẵng, ngày tháng năm

Chủ tịch Hội đồng

(Ký và ghi họ tên)

Cán bộ duyệt kiểm tra

(Ký và ghi họ tên)

[illegible]

SUMMARY

Topic title: Building the web system connecting football teams.

Student: Le Anh Tuan

Student ID: 102140230

Class: 14TCLC2

The aim of this project was to create a web system which supports users can find matches of the other teams to pair match, find the gridirons, the leagues at Da Nang city. Through the web system users can create and manage their teams, gridirons, leagues as manage member of their team, update match information of league, ...etc. The web system can help users save their time to find the matches or manage their leagues.

GRADUATION PROJECT REQUIREMENTS

Student Name: Le Anh Tuan

Student ID: 102140230

Class: 14TCLC2 Faculty: Information Technology Major: Software Engineering

1. *Topic title:*

Building the web system connecting football teams.

2. *Project topic:* ☐ *has signed intellectual property agreement for final result*

3. *Initial figure and data:*

Data is generated by myself and search on the internet.

Content of the explanations and calculations:

- Manage and show the matches.
- Manage and show the teams.
- Manage and show the gridirons.
- Manage and show the leagues.
- Pair Match with other teams.
- Register to join the leagues.

4. *Drawings, charts (specify the types and sizes of drawings):*

- User Case diagram.
- Sequence diagram.
- Activity diagram.
- Class diagram.

5. *Supervisor (s):* Pham Cong Thang Ph.D.

6. *Date of assignment:* / / 2019.

7. *Date of completion:* / / 2019.

Da Nang, / / 2019

Head of Division.....

Instructor

PREFACE

First of all, I would like to express my deepest gratitude to all teachers in the Information Technology Faculty in the past four years. I am grateful their help and transfer of precious experience. Especially, I would like to express my gratitude to Dr. Pham Cong Thang, my mentor who supports me a lot during my graduation project.

Thank you very much for your kind cooperation during my graduation project process.

Sincerely,

Le Anh Tuan

ASSURANCE

I assure:

1. The contents of this graduation project are performed by myself following the guidance of lecturer Dr. Pham Cong Thang.
2. All references used in this graduation project thesis, are quoted with the name of the author, project name, time and location to publish clearly and faithfully.
3. All invalid copies, educated statute violation or cheating will be borne the full responsibility by myself.

Student Performed

Le Anh Tuan

TABLE OF CONTENT

SUMARY

GRADUATION PROJECT REQUIREMENTS

PREFACE	i
ASSURANCE	ii
TABLE OF CONTENT	iii
LIST OF SYMBOLS AND ACRONYM	vi
INTRODUCTION	1
Chapter 1: THEORIES AND TECHNOLOGIES	3
1.1. Node JS	3
1.1.1. Introduction.....	3
1.1.2. Node JS architecture	3
1.1.3. Features.....	5
1.2. MySQL	5
1.2.1. Introduction.....	5
1.2.2. MySQL Workbench:	6
1.3. Firebase RealTime Database	7
1.3.1. Introduction.....	7
1.3.2. Key capabilities:	8
1.3.3. Benefit	9
1.4. Angular Framework	9
1.4.1. Introduction.....	9
1.4.2. Architecture overview	9
1.4.3. Main building blocks.....	10
1.4.4. Benefit	12
Chapter 2: ANALYSIS AND DESIGN	13
2.1. Analysis main features	13
2.1.1. Matches management.....	13

2.1.2. Gridirons management	13
2.1.3. Leagues management	13
2.1.4. Team management	14
2.1.5. Pairing matches	14
2.2. Use case diagram	14
2.2.1. Overall diagram	14
2.2.2. Areas management	15
2.2.3. Team careers management	16
2.2.4. Team levels management	17
2.2.5. Gridiron management.....	18
2.2.6. League management.....	19
2.2.7. Matches management.....	20
2.3. Activity Diagram	21
2.3.1. Register.....	21
2.3.2. Log in	22
2.3.3. Matches create	23
2.3.4. Matches update	23
2.3.5. Pair match	24
2.4. Interraction diagram.....	25
2.4.1. Register Account.....	25
2.4.2. Login	26
2.4.3. Create Match.....	27
2.4.4. Pair Match.....	27
2.5. Basic ERD	28
2.6. Class diagram	29
2.7. Conclusion	30

Chapter 3: IMPLEMENTATION RESULTS.....	31
3.1. Login function	31
3.2. Register function	31
3.3. Reset Password function	32
3.4. Matches management	33
3.5. Leagues management.....	35
3.6. Gridirons management	38
3.7. Teams management	41
3.8. Pairing Match.....	43
3.9. Confirm, Reject Matches	44
3.10. Register join league	45
CONCLUSION AND FUTURE WORK	48
1. Achieve results	48
2. Future works	48
REFERENCES	49

LIST OF TABLE DRAWING, DIAGRAM, PICTURE

Figure 1. 1	The Node.js architecture stack	4
Figure 1. 2	Firebase Overview	8
Figure 1. 3	Diagram of the relationship of the main building blocks	10
Figure 2. 1	Overall use case diagram	15
Figure 2. 2	Manage Areas use case diagram.	16
Figure 2. 3	Manage Team's Career use case diagram.	17
Figure 2. 4	Manage Team's Level use case diagram.	18
Figure 2. 5	Manage Gridirons use case diagram.	19
Figure 2. 6	Manage Leagues use case diagram.	19
Figure 2. 7	Manage Matches use case diagram.	Error! Bookmark not defined.
Figure 2. 8	Activity Register diagram	21
Figure 2. 9	Activity Sign In diagram	22
Figure 2. 10	Activity create new match diagram.....	23
Figure 2. 11	Activity update match diagram	24
Figure 2. 12	Activity Pair Match diagram.....	25
Figure 2. 13	Sequence diagram register	26
Figure 2. 14	The sequence diagram for login feature	26
Figure 2. 15	The sequence diagram for create matches	27
Figure 2. 16	The sequence for pair matches feature	28
Figure 2. 17	Simple entity-relationship diagram of the system.....	28
Figure 2. 18	Class diagram	29
Figure 3. 1	Login page.....	31
Figure 3. 2	Register page	32
Figure 3. 3	Reset password page.....	33
Figure 3. 4	Got email reset password.....	33
Figure 3. 5	Show list of match page.....	34
Figure 3. 6	The interface for creating new match.....	34
Figure 3. 7	The interface for updating match information	35
Figure 3. 8	Show list of leagues.....	35
Figure 3. 9	The interface for creating league.....	36
Figure 3. 10	The interface for update league information	36
Figure 3. 11	List standings of teams in the league.....	37
Figure 3. 12	List rounds of the league.....	37
Figure 3. 13	The interface update matches of the league.....	38
Figure 3. 14	Show list of gridirons	38
Figure 3. 15	The interface for creating gridiron	39
Figure 3. 16	The interface for updating gridiron	39

Figure 3. 17	The interface for creating sub-gridiron.....	40
Figure 3. 18	The interface for creating the price of gridiron.....	40
Figure 3. 19	Show list of teams	41
Figure 3. 20	The interface for creating team	41
Figure 3. 21	The interface for updating team	42
Figure 3. 22	The interface for adding new member to the team	42
Figure 3. 23	The interface for pairing match.....	43
Figure 3. 24	Show full list notification	43
Figure 3. 25	Show full list notification	43
Figure 3. 26	The interface for function confirms or reject match	44
Figure 3. 27	Show message confirmed request	44
Figure 3. 28	Show full list of notification	45
Figure 3. 29	The interface for registering to join league function.....	45
Figure 3. 30	Show list of teams registered to join league	45
Figure 3. 31	Public matches page	46
Figure 3. 32	Public gridirons page	46
Figure 3. 33	Public leagues page	47

LIST OF SYMBOLS AND ACRONYM

No.	Items	Description
1	API	Application Programming Interface
2	ERD	Entity Relationship Diagram
3	EER	Enhanced Entity Relationship
4	CRUD	Create, Read, Update, Delete

INTRODUCTION

1. Context and Purpose

Nowadays, football is king of the sports, It is popular for anyone even the children. So we have some problems:

- To connect football teams and to pair matches.
- To introduce gridirons to everyone.
- To organize a league and it's update.

About the first and second issues, Facebook can be solved but on Facebook, there are a lot of information, so it is easy to lose the user's post. For the third issue, the user can manage the league on the paper or excel.

So, I apply technology to build the web system which supports users.

This website will resolve some issues:

- Support users to search the teams at Da Nang city and to pair matches.
- Support users to manage and to show gridirons information.
- Support users to manage their leagues.
- Support users to search the gridirons information at Da Nang city.

For that reason, I decided to do project named "*Building the web system connecting football teams*".

2. Scope

Through public pages, users can use some features with do not sign in:

- Search the opening matches.
- Search the opening leagues.
- Search the gridirons at Da Nang city.
- View list of available matches.
- View list of gridirons.
- View list of available leagues.

Users have to sign in to use these features as follow.

- Manage the teams.
- Manage the matches.
- Manage the gridirons.
- Manage the leagues.

- Pair match with other teams.
- Join the leagues.

.....

Chapter 1: THEORIES AND TECHNOLOGIES

Below are theories of technologies I did use in developing my project.

1.1. Node JS

1.1.1. Introduction

Node.js can be defined as a dynamic, cross-platform and open-source JavaScript framework or runtime environment that is built on the Google Chrome JavaScript V8 engine [1]. Node.js, developed by Ryan Dahl in 2009, was initially implemented as a client-side scripting language. Nowadays, it is used to execute JavaScript code and scripts that run server-side to create dynamic web pages. The latest version of Node.js is 10.10.0.

Node.js provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

A Node.js app is run in a single process, without creating a new thread for every request [1]. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

When Node.js needs to perform an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back.

This allows Node.js to handle thousands of concurrent connections with a single server without introducing the burden of managing thread concurrency.

1.1.2. Node JS architecture

In the following section, we use Figure 1.1 [1.1] for discussion.

a, The Node runtime

Node's runtime is another term for the set executable programs that actually run Node applications, and is a combination of (middle of the stack, see Figure 1):

- *Node API*: The Node API is a set of built-in modules provided by Node.js out of the box for you to build applications [1]. Many of these modules, like the File System (fs) API, sit atop lower-level programs (the Node Core) that communicate with the underlying OS.

- *The Node core*: a set of JavaScript modules that implement the Node API [1]. (Apparently some of the modules depend on libuv and other C++ code but that's an implementation detail).
- JavaScript engine: Chrome's V8 Engine: A fast JavaScript-to-machine code compiler to load, optimize, and run your JavaScript code
 - The event loop: implemented using an event-driven, non-blocking I/O library called libuv to make it lightweight and efficient (and scalable)

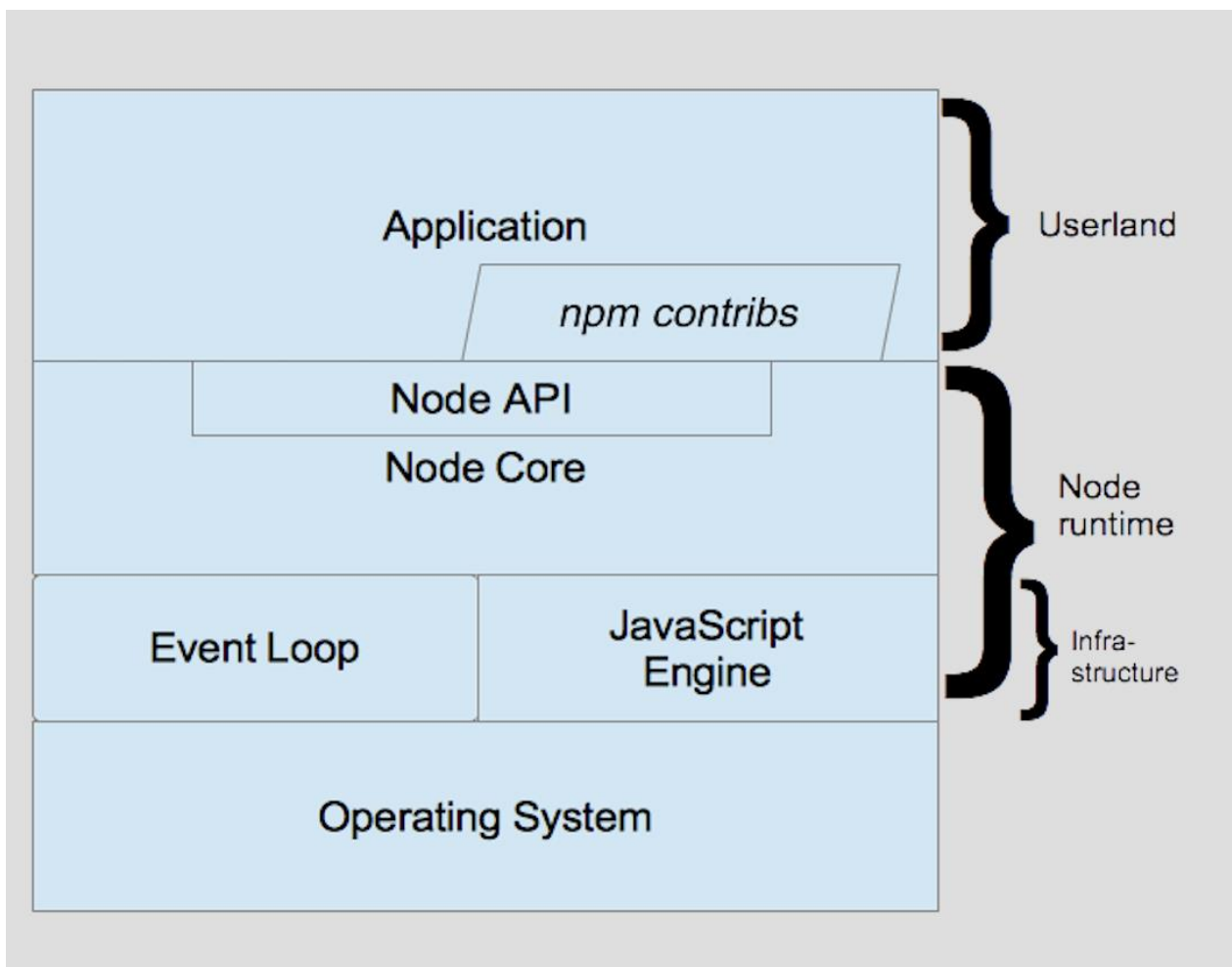


Figure 1. 1 The Node.js architecture stack

b, Infrastructure

The Node runtime's infrastructure is comprised of two major components:

- JavaScript engine: The JavaScript engine used by Node is Chrome's V8 engine, which runs all of the JavaScript code.

- Event loop: The event loop consists of various phases where callbacks are invoked:
 - Timers phase: setInterval() and setTimeout() expired timer callbacks are run
 - Poll phase: The OS is polled to see if any I/O operations are complete, and, if so, those callbacks are run
 - Check phase: setImmediate() callbacks are run

1.1.3. Features

a) Modularity

Major advantage of Node JS Platform is that it's modularity [1]. Each and every functionality is divided and implemented as a separate module or package.

b) Non-blocking or Asynchronous IO

Asynchronous event-driven IO helps concurrent request handling – This is probably the biggest selling points of Node.js [1]. This feature basically means that if a request is received by Node for some Input/Output operation, it will execute the operation in the background and continue with processing other requests.

c) Single Threaded but Highly Scalable

Node.js uses a single threaded model with event looping [1]. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

d) Better Socket API

Node JS Platform provides very good Socket Module API to develop Real-time, Multi-User Chat and Multi-Player Gaming Applications very easily. It supports Unix Socket programming like pipe().

1.2. MySQL

1.2.1. Introduction

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation [2].

- **MySQL is a database management system:** a database is a structured collection of data [2]. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network.

- **MySQL databases are relational:** a relational database stores data in separate tables rather than putting all the data in one big storeroom [2]. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment.
- **MySQL software is Open Source:** anybody can download the MySQL software from the Internet and use it without paying anything [2]. The source code can change it to suit user needs.
- **The MySQL Database Server is very fast, reliable, scalable, and easy to use:** MySQL Server can run comfortably on a desktop or laptop, alongside other applications, web servers, and so on, requiring little or no attention [2]. MySQL can also scale up to clusters of machines, networked together.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years [2]. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems:** The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different back ends [2], several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).
- **A large amount of contributed MySQL software is available.**

1.2.2. MySQL Workbench:

MySQL Workbench is a graphical tool for working with MySQL servers and databases [3].

MySQL Workbench functionality covers five main topics:

- **SQL Development:** Enables to create and manage connections to database servers. Along with enabling to configure connection parameters [3], MySQL Workbench provides the capability to execute SQL queries on the database connections using the built-in SQL Editor.

- **Data Modeling (Design):** Enables to create models of database schema graphically, reverse and forward engineer between a schema and a live database, and edit all aspects of database using the comprehensive Table Editor [3].
- **Server Administration:** Enables to administer MySQL server instances by administering users, performing backup and recovery, inspecting audit data, viewing database health, and monitoring the MySQL server performance [3].
- **Data Migration:** Allows to migrate from Microsoft SQL Server, Microsoft Access, Sybase ASE, SQLite, SQL Anywhere, PostgreSQL, and other RDBMS tables, objects, and data to MySQL [3].
- **MySQL Enterprise Support:** Support for Enterprise products such as MySQL Enterprise Backup, MySQL Firewall, and MySQL Audit [3].

1.3. Firebase Real-time Database

1.3.1. Introduction

Firebase Real-time database is a cloud-hosted database that supports multiple platforms Android, iOS and Web [4]. All the data is stored in JSON format and any changes in data reflect immediately by performing sync across all the platforms & devices.

a, How the Data is Stored – JSON Structured

Firebase real-time database is a schemaless database in which the data is stored in JSON format. Basically, the entire database is a big JSON tree with multiple nodes [4].

b, Offline Data

Firebase provides great support when comes to offline data. It automatically stores the data offline when there is no internet connection [4]. When the device connects to the internet, all the data will be pushed to the real-time database. However, enabling disk persistence stores the data offline even though app restarts. Disk persistence can be enabled by calling below one line code.

c, Security & Rules

Firebase rules provide a way to identify user role while performing read and write operations [4]. These rules will acts as a security layer on the server before perform any CRUD operation. By default the rules allow a user to perform read & write operation only after authentication.

Firebase Services can be divided into two groups, it's shown in Figure 1.2 [1.2].



Figure 1. 2 Firebase Overview

1.3.2. *Key capabilities:*

- ❖ **Realtime:** Instead of typical HTTP requests, the Firebase Realtime Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds [4]. Provide collaborative and immersive experiences without thinking about networking code.
- ❖ **Offline:** Firebase apps remain responsive even when offline because the Firebase Realtime Database SDK persists your data to disk [4]. Once connectivity is reestablished, the client device receives any changes it missed, synchronizing it with the current server state.
- ❖ **Accessible from Client Devices:** The Firebase Realtime Database can be accessed directly from a mobile device or web browser; there's no need for an application server [4]. Security and data validation are available through the Firebase Realtime Database Security Rules, expression-based rules that are executed when data is read or written.
- ❖ **Scale across multiple databases:** With Firebase Realtime Database on the Blaze pricing plan, you can support your app's data needs at scale by splitting your data across multiple database instances in the same Firebase project [4]. Streamline authentication

with Firebase Authentication on your project and authenticate users across your database instances. Control access to the data in each database with custom Firebase Realtime Database Rules for each database instance.

1.3.3. Benefit

- Real-time Database is a cloud-hosted database [4]. Data is stored as JSON and synchronized continuously to each associated client.
- Hosting is production-grade web content that facilitates the developers [4].
- Firebase Authentication gives backend services, simple to-use SDKs, and instant UI libraries to confirm clients over your application [4]. It supports authentication using passwords, email id or username.
- Firebase notification is a free service which allows targeted user notifications for mobile app developers [4]. It has the notification console GUI where you can create and send notifications to targeted users.
- This feature is used to index application in Google search results [4].
- Admob is advertising facility of the Firebase which is used to generate profits from your app [4].

1.4. Angular Framework

1.4.1. Introduction

Angular is a platform that makes it easy to build applications with the web [5]. Angular combines declarative templates, dependency injection, end to end tooling, and integrated best practices to solve development challenges. Angular empowers developers to build applications that live on the web, mobile, or the desktop.

1.4.2. Architecture overview

The basic building blocks of an Angular application are NgModules, which provide a compilation context for components [5]. NgModules collect related code into functional sets; an Angular app is defined by a set of NgModules. An app always has at least a root module that enables bootstrapping, and typically has many more feature modules.

- Components define views, which are sets of site elements that Angular can choose among and modify according to your program logic and data.
- Components use services, which provide specific functionality not directly related to views. Service providers can be injected into components as dependencies, making your code modular, reusable, and efficient.

Both components and services are simply classes, with decorators that mark their type and provide metadata that tells Angular how to use them [5].

- The metadata for a component class associates it with a template that defines a view. A template combines ordinary HTML with Angular directives and binding markup that allow Angular to modify the HTML before rendering it for display.
- The metadata for a service class provides the information Angular needs to make it available to components through dependency injection (DI).

An app's components typically define many views, arranged hierarchically [5]. Angular provides the Router service to help you define navigation paths among views. The router provides sophisticated in-browser navigational capabilities.

The diagram shown in figure 1.3 describes how these basic pieces are related.

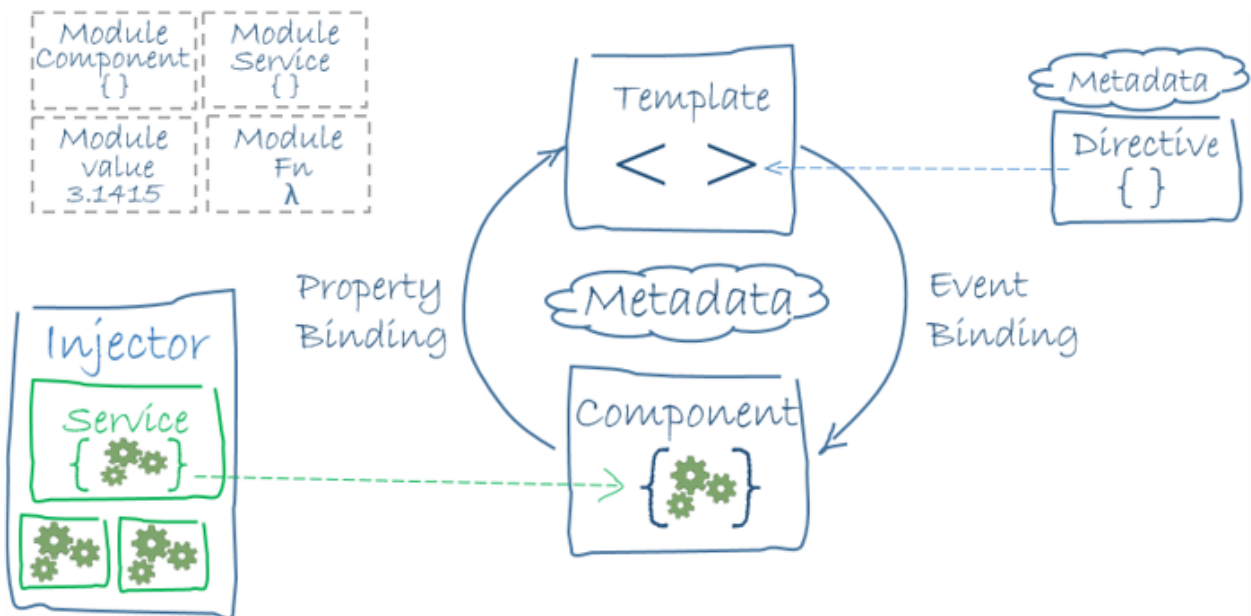


Figure 1. 3 Diagram of the relationship of the main building blocks

1.4.3. Main building blocks

a) Modules

Angular NgModules differ from and complement JavaScript modules [5]. An NgModule declares a compilation context for a set of components that is dedicated to an application domain, a workflow, or a closely related set of capabilities. An NgModule can associate its components with related code, such as services, to form functional units.

Every Angular app has a root module, conventionally named AppModule, which provides the bootstrap mechanism that launches the application [5]. An app typically contains many functional modules.

b) Components

Every Angular application has at least one component, the root component that connects a component hierarchy with the page document object model (DOM). Each component defines a class that contains application data and logic, and is associated with an HTML template that defines a view to be displayed in a target environment [5].

The @Component() decorator identifies the class immediately below it as a component, and provides the template and related component-specific metadata.

c) Templates, directives, and data binding

A template combines HTML with Angular markup that can modify HTML elements before they are displayed. Template directives provide program logic, and binding markup connects your application data and the DOM [5]. There are two types of data binding:

- Event binding lets app respond to user input in the target environment by updating application data.
- Property binding lets interpolate values that are computed from application data into the HTML.

d) Services and dependency injection

For data or logic that isn't associated with a specific view, and that you want to share across components, you create a service class [5]. A service class definition is immediately preceded by the @Injectable() decorator.

Dependency injection (DI) lets you keep your component classes lean and efficient. They don't fetch data from the server, validate user input, or log directly to the console; they delegate such tasks to services.

e) Routing

The Angular Router NgModule provides a service that let's define a navigation path among the different application states and view hierarchies in the app [5]. It is modeled on the familiar browser navigation conventions:

- Enter a URL in the address bar and the browser navigates to a corresponding page.
- Click links on the page and the browser navigates to a new page.
- Click the browser's back and forward buttons and the browser navigates backward and forward through the history of pages you've seen.

1.4.4. Benefit

- **Develop across all PlatForms:** Learn one way to build applications with Angular and reuse your code and abilities to build apps for any deployment target. For web, mobile web, native mobile and native desktop.
- **Speed and performance:** Achieve the maximum speed possible on the Web Platform today, and take it further, via Web Workers and server-side rendering. Angular puts in control over scalability. Meet huge data requirements by building data models on RxJS, Immutable.js or another push-model.
- **Incredible tooling:** Build features quickly with simple, declarative templates. Extend the template language and use a wide array of existing components. Get immediate Angular-specific help and feedback with nearly every IDE and editor.
- **Loved by millions:** From prototype through global deployment, Angular delivers the productivity and scalable infrastructure that supports Google's largest applications.

1.5. Conclusion

By studying and learning about the above technologies, we successfully applied the concepts and their mechanism operating in this project.

Some of these technologies are not new, but they are widely using and a trend for the software development industry. Therefore, understanding the concept is very important, help to apply properly technology for each project, in order to improve efficiency and usability.

We have introduced some technologies that we implemented in the project. In the next chapter, we will introduce the analysis and design section to describe more details about my system.

Chapter 2: ANALYSIS AND DESIGN

2.1. Analysis main features

To clarify this web system, we have following features.

2.1.1. Matches management

When the user login to the web system, the user can select “Manage Matches” feature on the top menu. On the site “Manage Matches” the user can click the button “Create new match” to create a new match. Once the user creates a new match, this one will show on the public home page. To update the information of a match, user can select action “Edit” on the site “Manage Matches”, user can change some fields and click “Update” button to update data.

2.1.2. Gridirons management

In this part, when the user login to the web system, the user can select “Manage Gridirons” feature on the top menu. On the site “Manage Gridirons” the user can click the button “Create new gridiron” to create a new gridiron. Once the user creates a new gridiron, this one will show on the public gridirons page. To update the information of a gridiron, user can select action “Edit” on the site “Manage Gridirons”, user can change some fields and click “Update” button to update data.

Besides that, In the detail of gridiron site, the user can add the price for each gridiron to show to everyone.

2.1.3. Leagues management

In this part, when the user login to the web system, the user can select “Manage Leagues” feature on the top menu. On the site “Manage Leagues” the user can click the button “Create new league” to create a new league. Once the user creates a new league, this one will show on the public leagues page. To update the information of a league, user can select action “Edit” on the site “Manage Leagues”, user can change some fields and click “Update” button to update data.

Besides that, when the user creates a new league, the system will generate schedule for the league. The user can update detail of every match such as score, time of competition, description of a match. Once the user updates the detail of the match, the system will auto update point, number of matches played, number of matches won, ...etc. and show the league's standings.

2.1.4. Team management

In this part, when the user login to the web system, the user can select “Manage Teams” feature on the top menu. On the site “Manage Teams” the user can click the button “Create new team” to create a new team. To update the information of a team, user can select action “Edit” on the site “Manage Teams”, user can change some fields and click “Update” button to update data.

Besides that, In the detail of team site, the user can manage the team members such as add, remove, ...etc.

2.1.5. Pairing matches

Once the other users create a new match, this one will show on the public home page. The user can search and view detail of those matches. The user can Pair Match with any matches, which has status is “New”. If the user wants to pair matches, the user has to have at least a team.

After the user gets the message “Pair match successfully”, the system generates a message to notifies the match creators. The match creators can “Confirm” or “Reject” the pairing request.

2.2. Use case diagram

2.2.1. Overall diagram

The web system includes these features:

- Manage and show areas (team, gridiron).
- Manage and show team’s career.
- Manage and show team’s level
- Manage and show the gridirons.
- Manage and show the teams.
- Manage and show the leagues.
- Manage and show the matches.
- Search matches, gridirons, leagues.

With these features above, the web system has three actors (the users, the guest and the admin) interacting with the system. The overall use-case is shown in the figure 2.1.

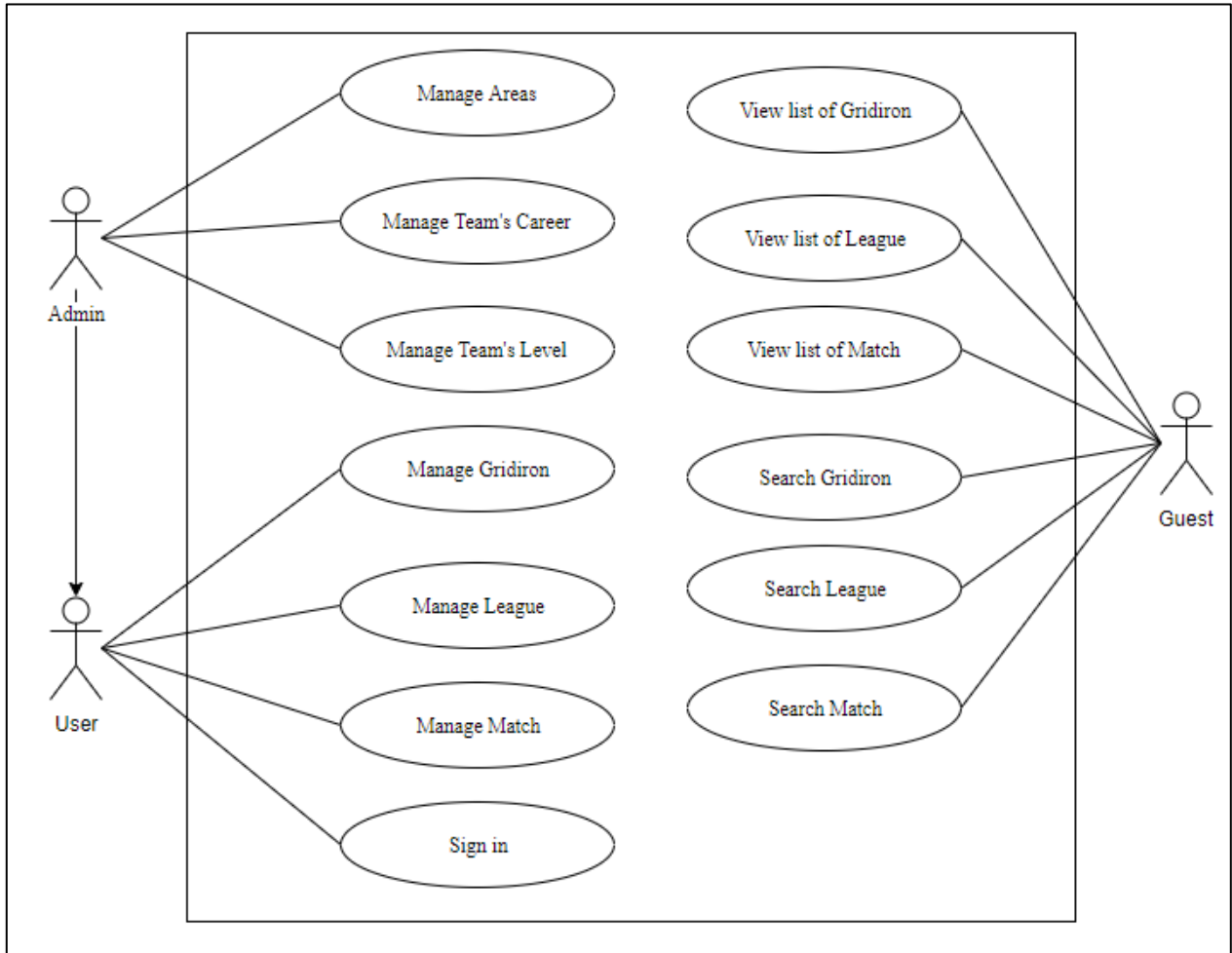


Figure 2. 1 Overall use case diagram

2.2.2. Areas management

“Manage Areas” has following these features: View list, View detail, Create, Update, Delete.

If the user is admin then user can select “Manage Areas” feature on dropdown menu. In the detail of this feature, user can view the list, search, create, update and delete the area. The use-case for this feature is shown in the figure 2.2.

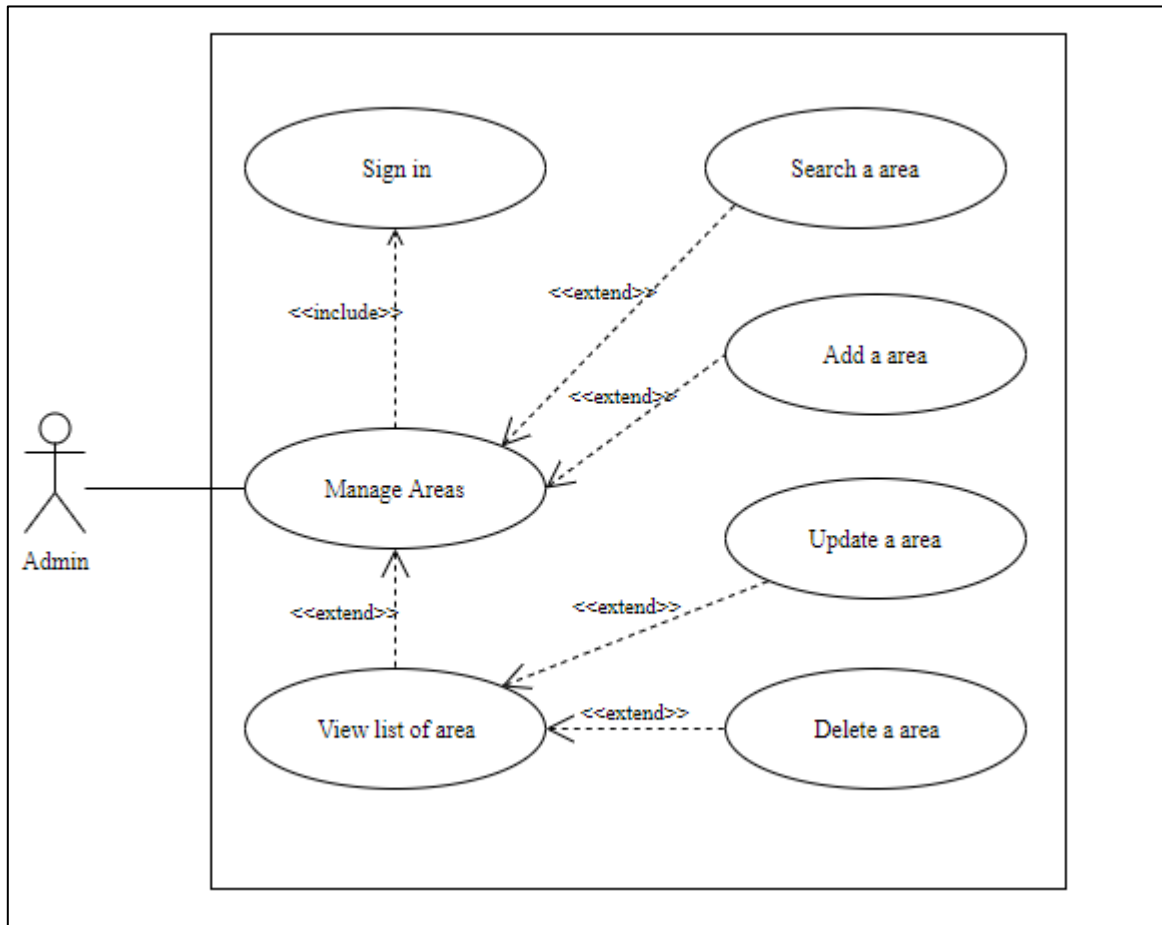


Figure 2. 2 Manage Areas use case diagram.

2.2.3. Team careers management

“Manage Team’s Career” has following these features: View, Add, View Detail, Update, Delete.

If user is admin then the user can select “Manage Team’s Career” feature on dropdown menu. In the detail of this feature, user can view the list, search, create, update and delete the career. The use-case for this feature is shown in the figure 2.3.

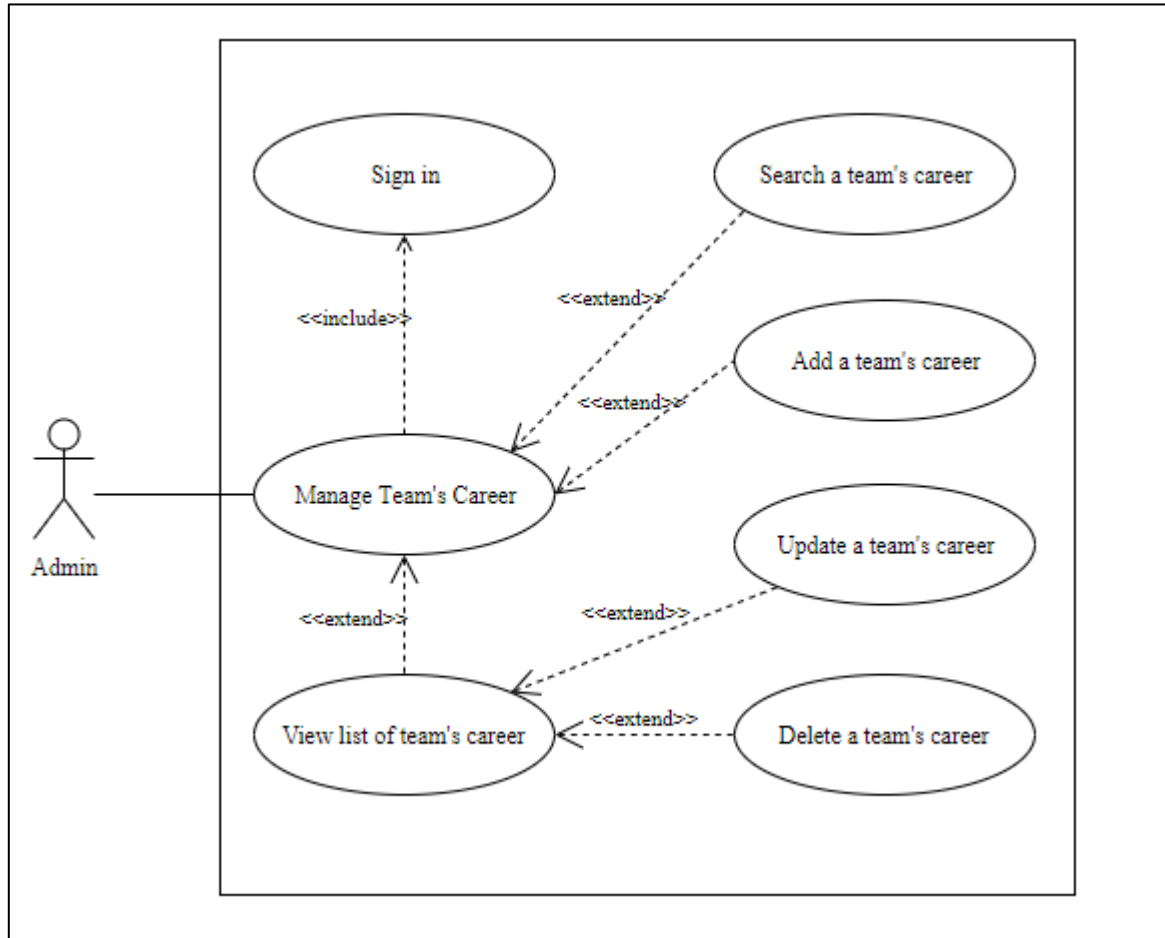


Figure 2. 3 Manage Team's Career use case diagram.

2.2.4. Team levels management

“Manage Team's Level” has following these features: View, Add, View Detail, Update, Delete.

If user is admin then the user can select “Manage Team's Level” feature on dropdown menu. In the detail of this feature, user can view the list, search, create, update and delete the team's level. The use-case for this feature is shown in the figure 2.4.

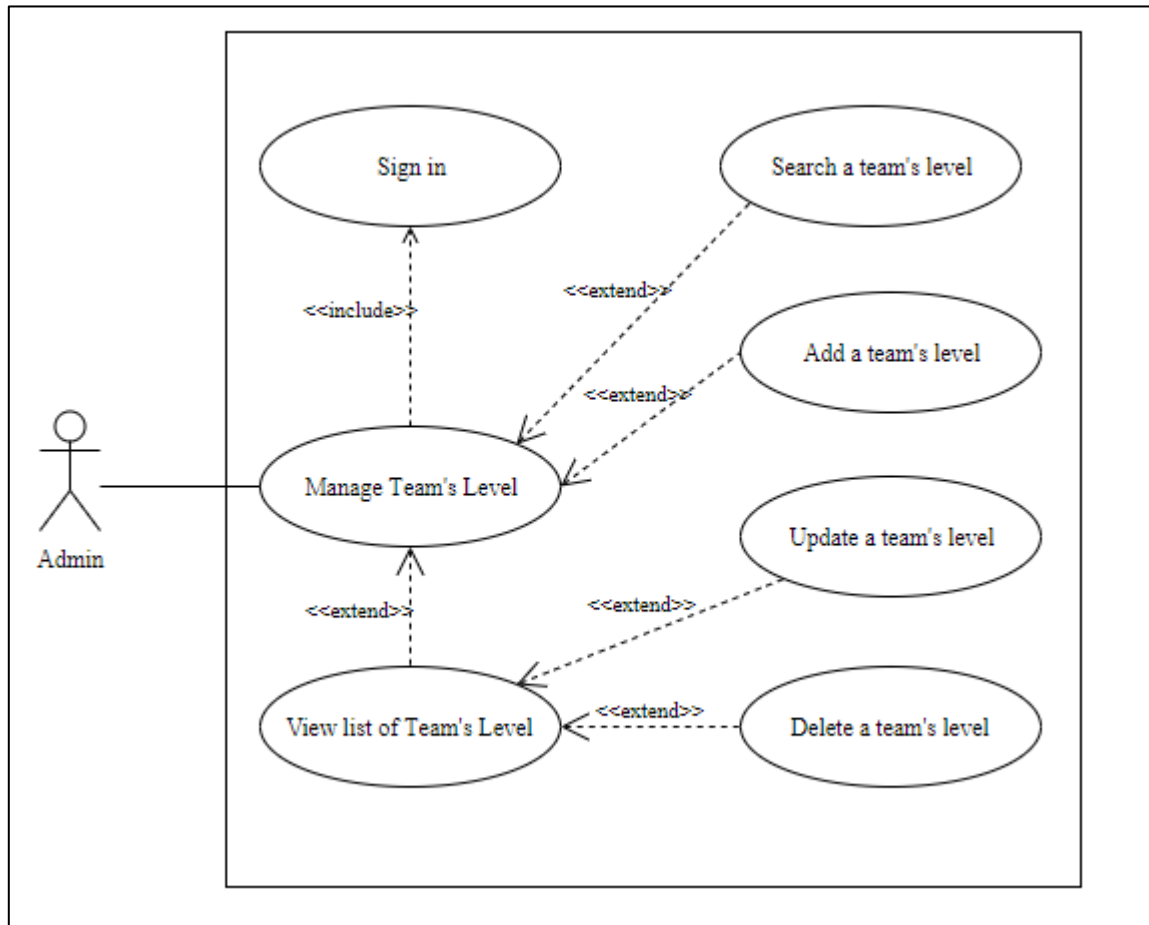


Figure 2. 4 Manage Team's Level use case diagram.

2.2.5. Gridiron management

“Manage Gridiron” has following these features: View, Add, View Detail, Update, Delete, manage the price of each gridiron.

The user can select “Manage Gridiron” feature on dropdown menu. In the detail of this feature, user can view the list, search, create, update and delete the gridiron.

Besides that, the user can create new sub-gridiron and the price of each gridiron. The use-case for this feature is shown in the figure 2.5.

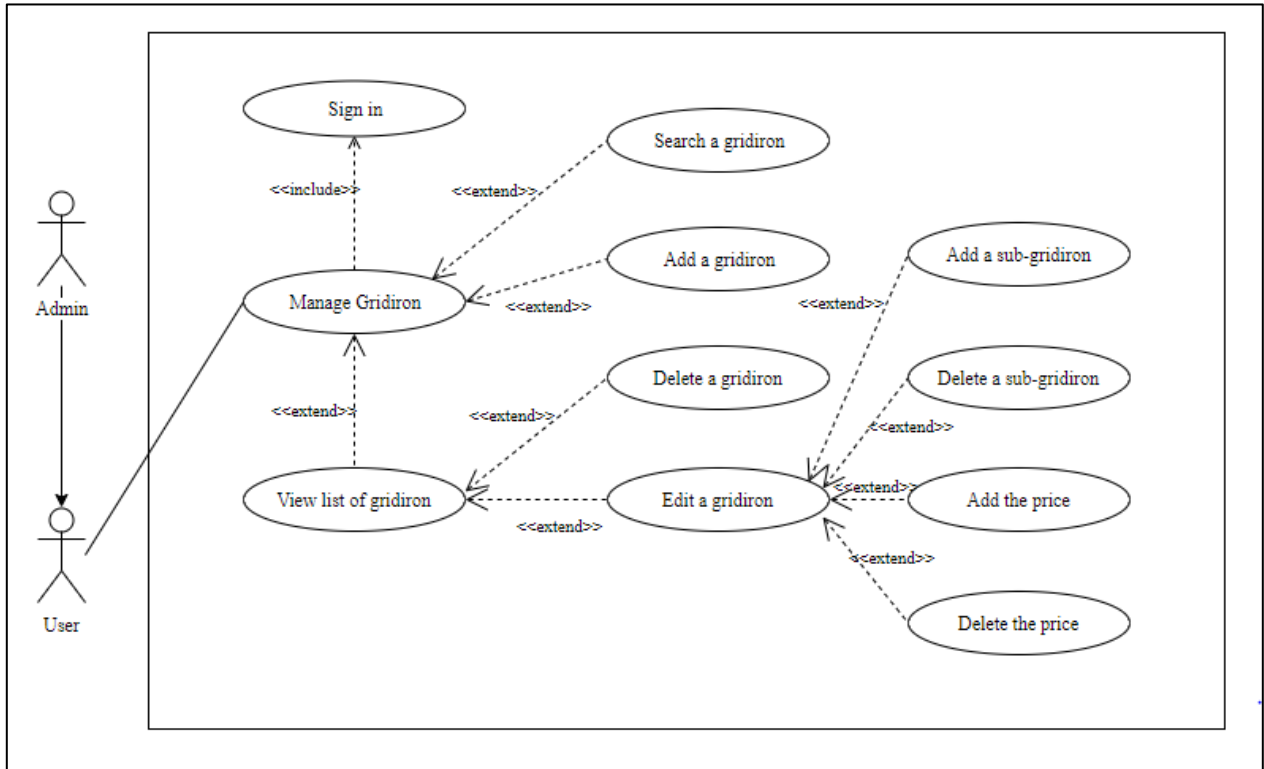


Figure 2. 5 Manage Gridirons use case diagram.

2.2.6. League management

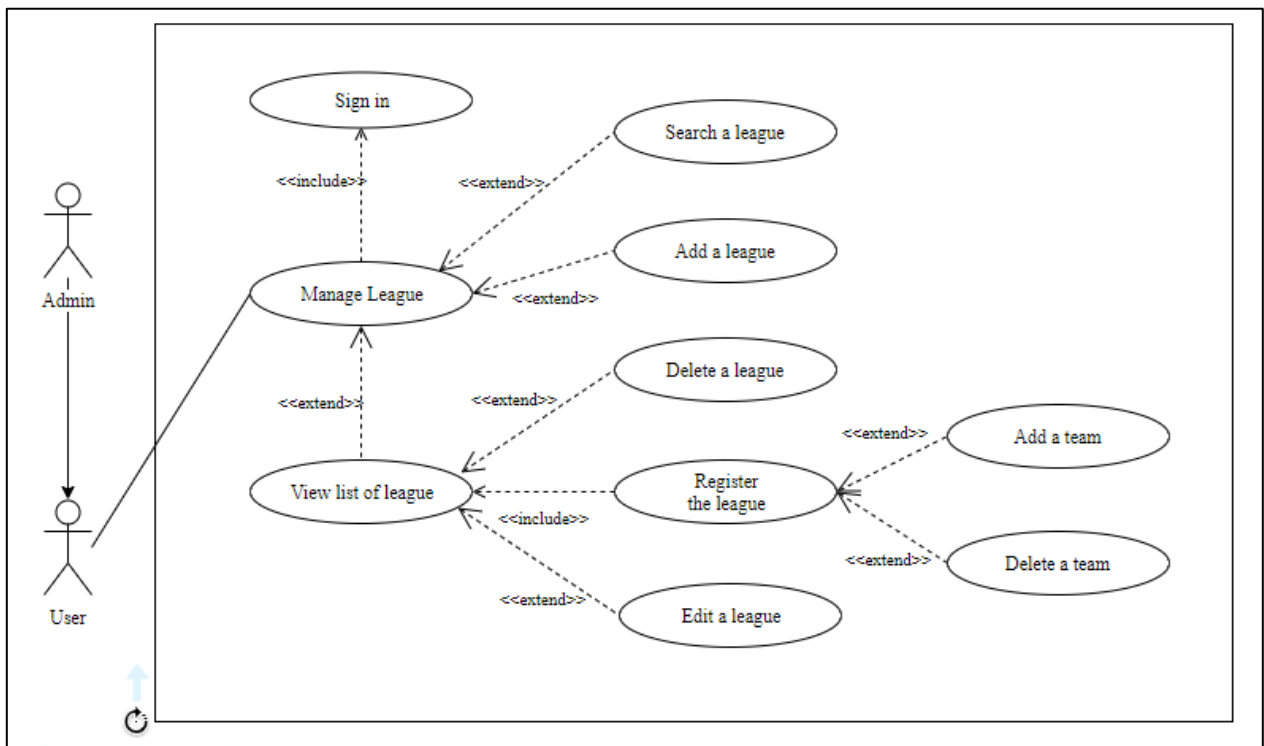


Figure 2. 6 Manage Leagues use case diagram.

The diagram shown in figure 2.6 describes the process “Manage Leagues”. “Manage Leagues” has following these features: View, Add, View Detail, Update, Manage the teams registered, Update the matches information of league.

The user can select “Manage Leagues” on the dropdown menu. In the detail of this feature, the user can view the list, search, create, update, manage teams registered and update the match’s information of the leagues.

2.2.7. Matches management

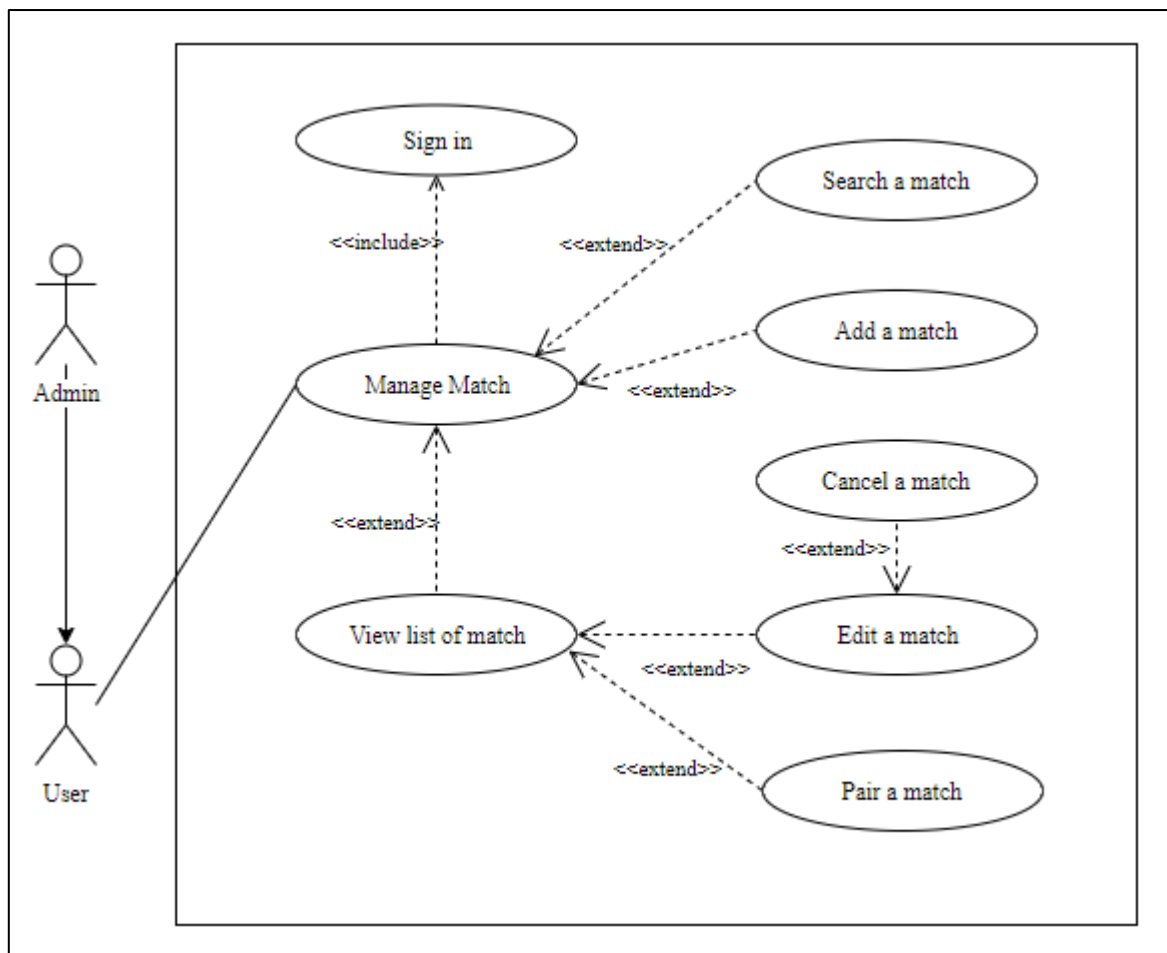


Figure 2. 7 Manage Matches use case diagram.

The diagram shown in figure 2.7 describes the process “Manage Matches”. “Manage Matches” has following these features: View, Add, View Detail, Update, Confirm or Reject request Pair Match of other teams.

The user can select “Manage Matches” feature on the dropdown menu. In the detail of this feature, user can view the list, search history matches, create, update, confirm or reject the request “pair match” of the other team.

2.3. Activity Diagram

2.3.1. Register

The diagram shown in figure 2.8 describes the process in with user register a new account. User has to fill user’ information into the registration form. After that, the system checks automatically the user’s information. If the user’s information is valid then the system saves that information to the database and notifies successful message to the user.

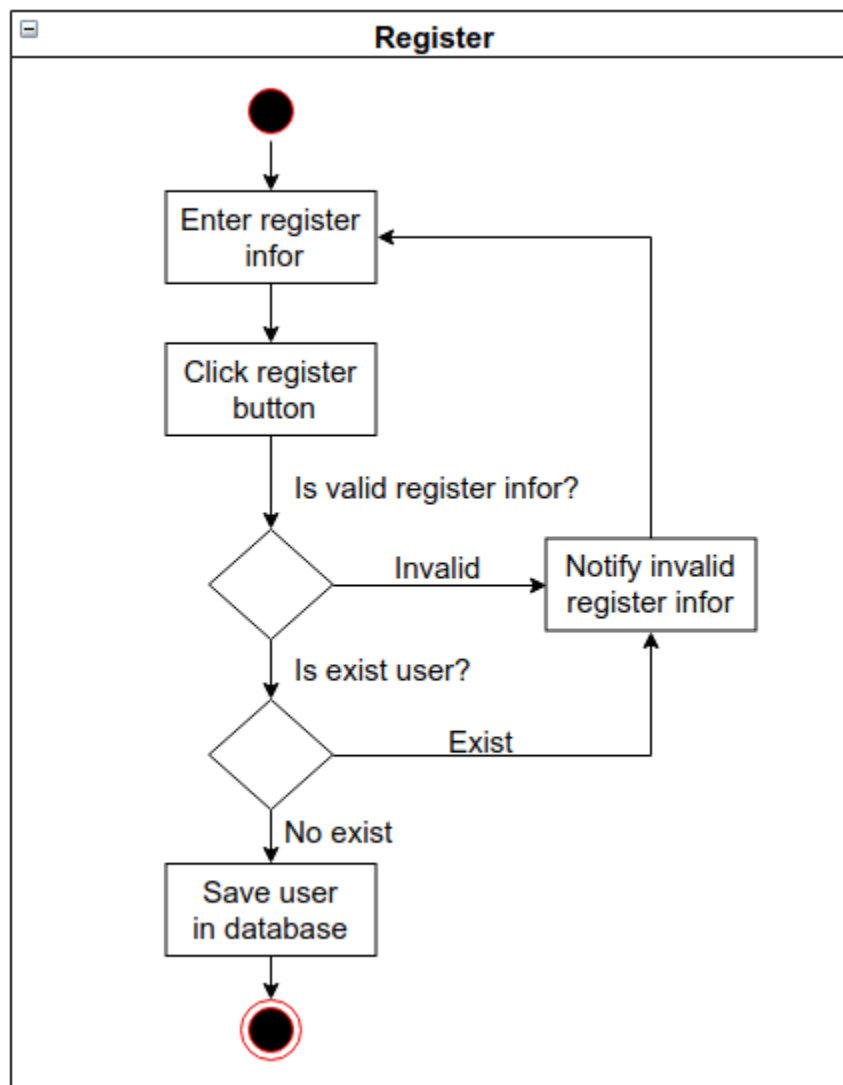


Figure 2. 8 Activity Register diagram

2.3.2. Log in

The diagram shown in figure 2.9 describes the process in with user sign in to the system. User has to fill user's information into the login form. After that, the system calls API to check automatically the user's information. If the user's information is valid then the system saves that information to the Local Storage and navigates to home page.

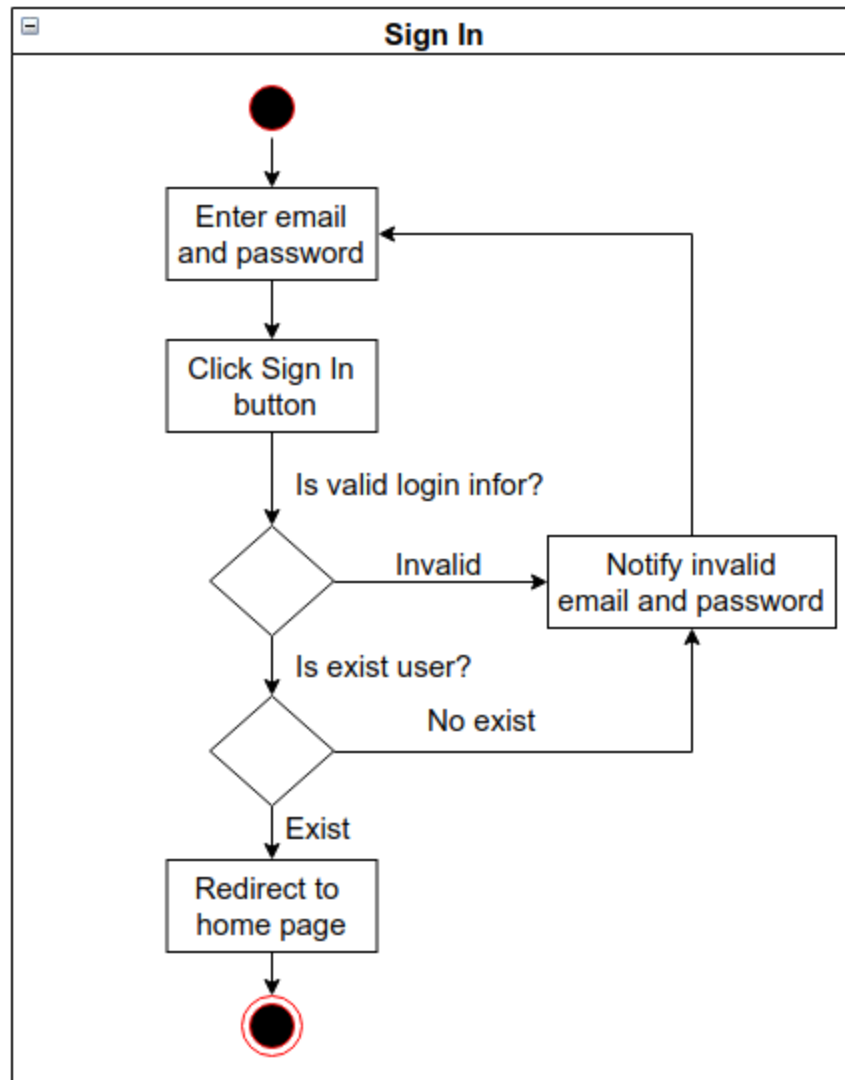


Figure 2. 9 Activity Sign In diagram

2.3.3. Matches create

The diagram shown in figure 2.10 describes the process in with user create new match. User has to fill match's information into the "Create match" form. After that, the system calls API to check automatically the match's information. If the match's information is valid then the system saves that information to the database and notifies successful message to the user.

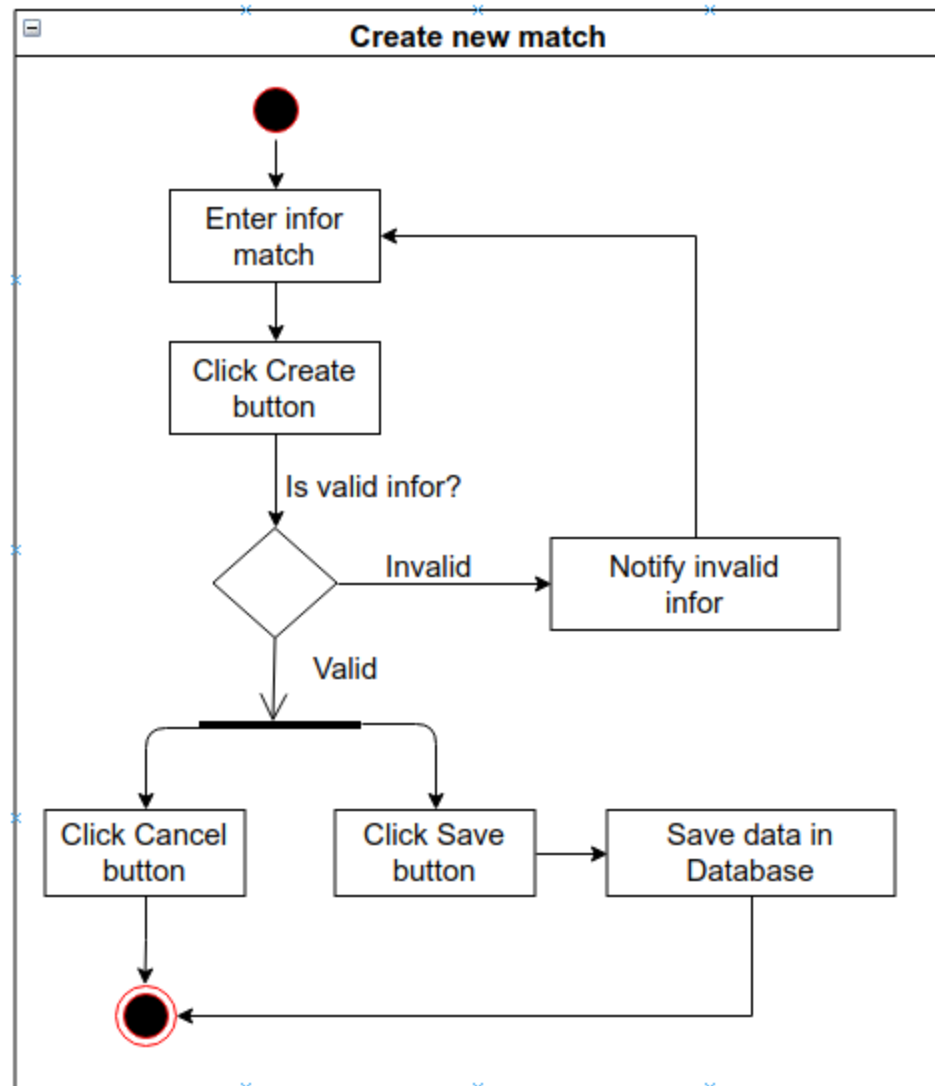


Figure 2. 10 Activity create new match diagram

2.3.4. Matches update

The diagram shown in figure 2.11 describes the process in with user update match. User change some information in the "Update match" form. After that, the system calls API

to check automatically the match's information. If the match's information is valid then the system saves that information to the database and notifies successful message to the user.

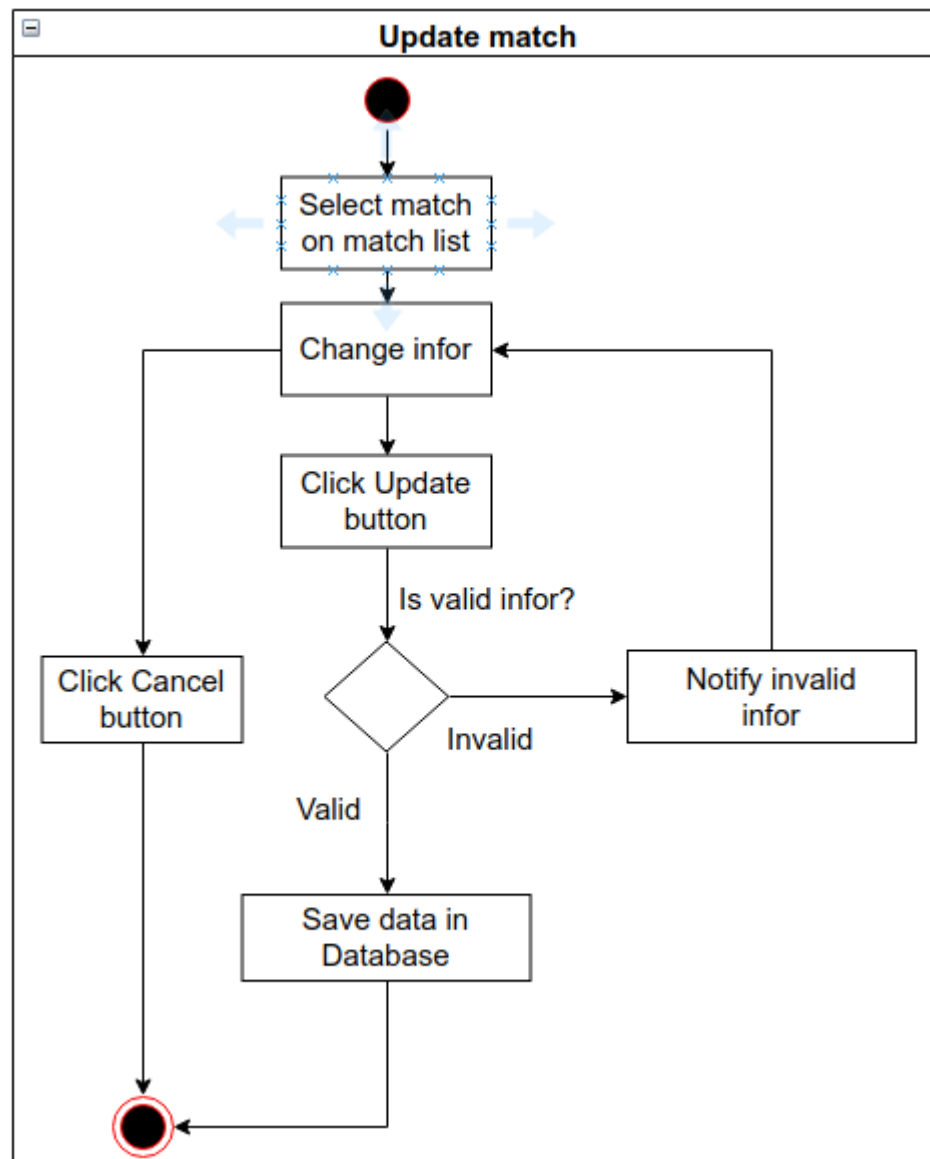


Figure 2. 11 Activity update match diagram

2.3.5. Pair match

The diagram shown in figure 2.12 describes the process in with user pair match with another team. The user has to select the team to pair match then click "Pair" button. After that, the system calls API to check automatically the team's information. If the team's information is valid then the system updates the match's information to the database and notifies successful message to the user.

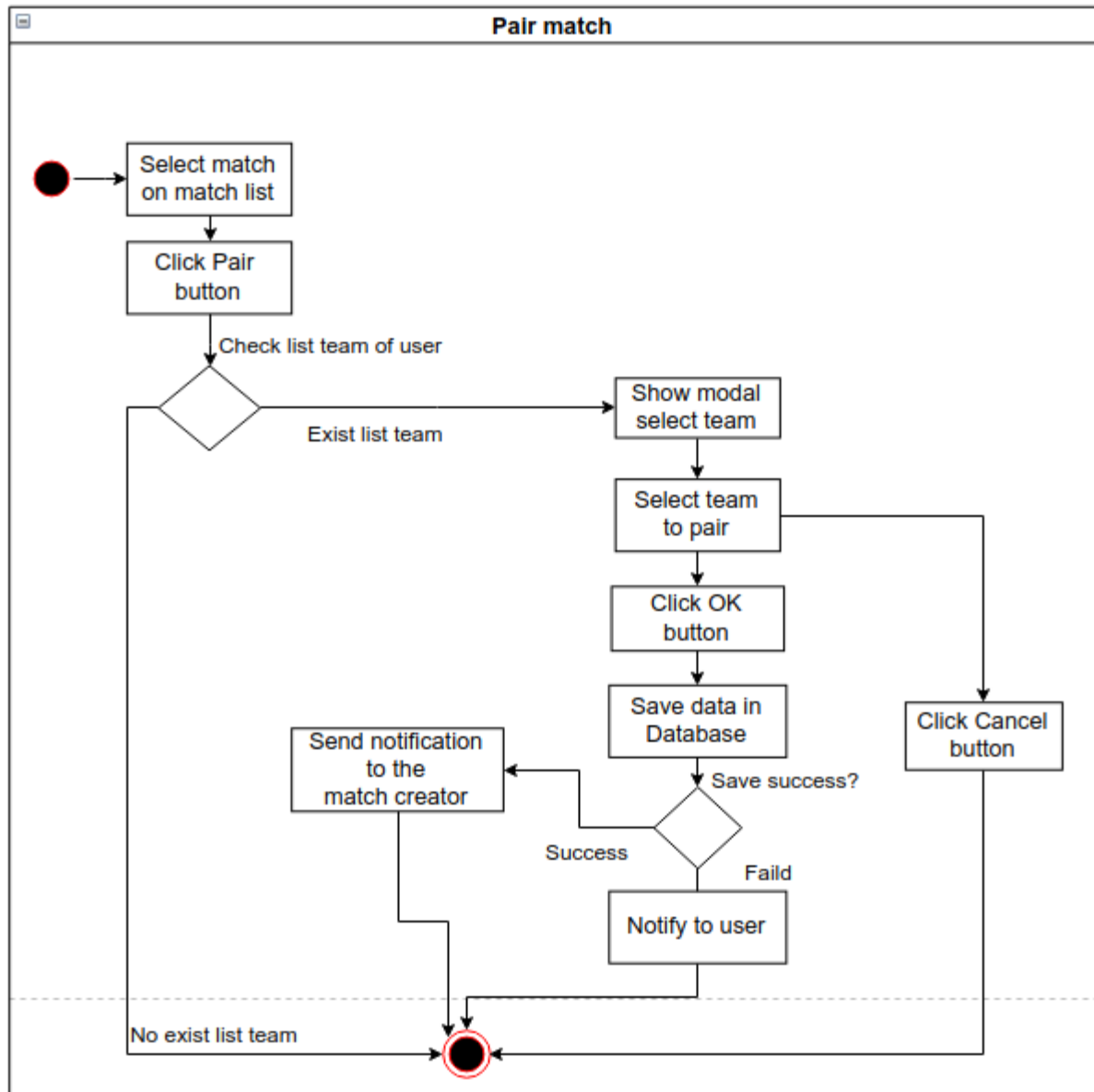


Figure 2. 12 Activity Pair Match diagram

2.4. Interraction diagram

2.4.1. Register Account

The user has to fill account's information then the system will check to validate request and call restful API to controller, after that, Modal connects to MYSQL and saves data in the database then notifies to user. The sequence diagram for this feature is shown in the figure 2.13.

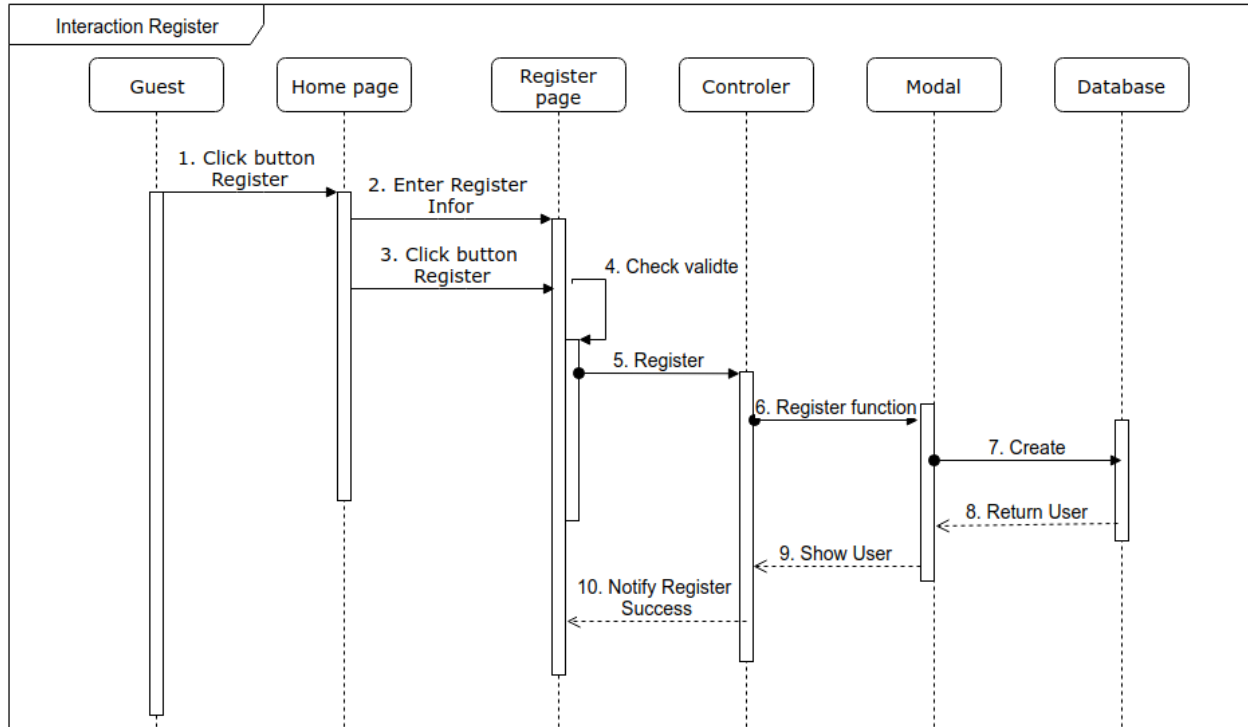


Figure 2. 13 Sequence diagram register

2.4.2. Login

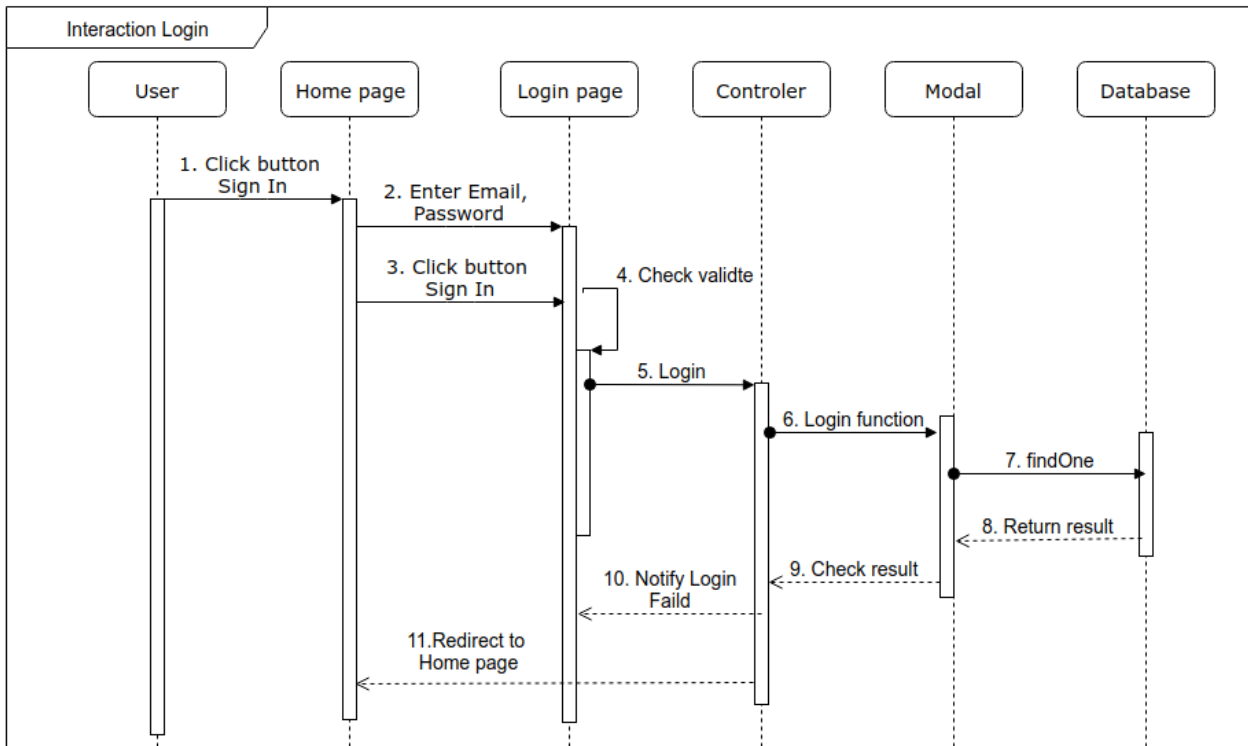


Figure 2. 14 The sequence diagram for login feature

The diagram shown in figure 2.14 describes the sequence diagram for login feature. The user has to fill account's information then the system will check to validate request and call restful API to controller, after that, Modal connects to MYSQL and checks data in the database then notifies to user.

2.4.3. Create Match

The user has to fill match's information then the system will check to validate request and call restful API to Controller, after that, Modal connects to MYSQL and save data in the database then notifies to user. The sequence diagram for this feature is shown in the figure 2.15.

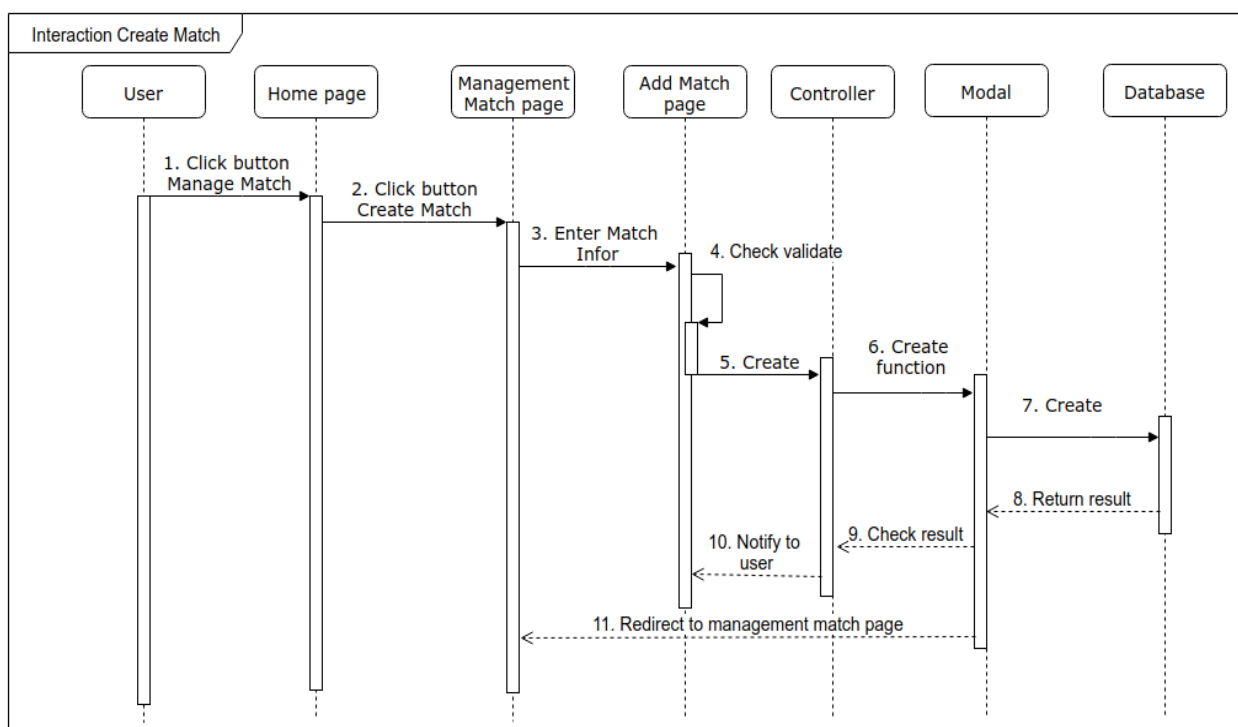


Figure 2. 15 The sequence diagram for create matches

2.4.4. Pair Match

The user has to select a team to pair a suitable football match, then the system will call restful API to Controller. After that, Modal connect to MYSQL and updates match's information in the database. Once data was saved to the database, the system generates a new notification and save to database. After that, the system sends to the match creators and notifies to the user. The sequence diagram for this feature is shown in figure 2.16.

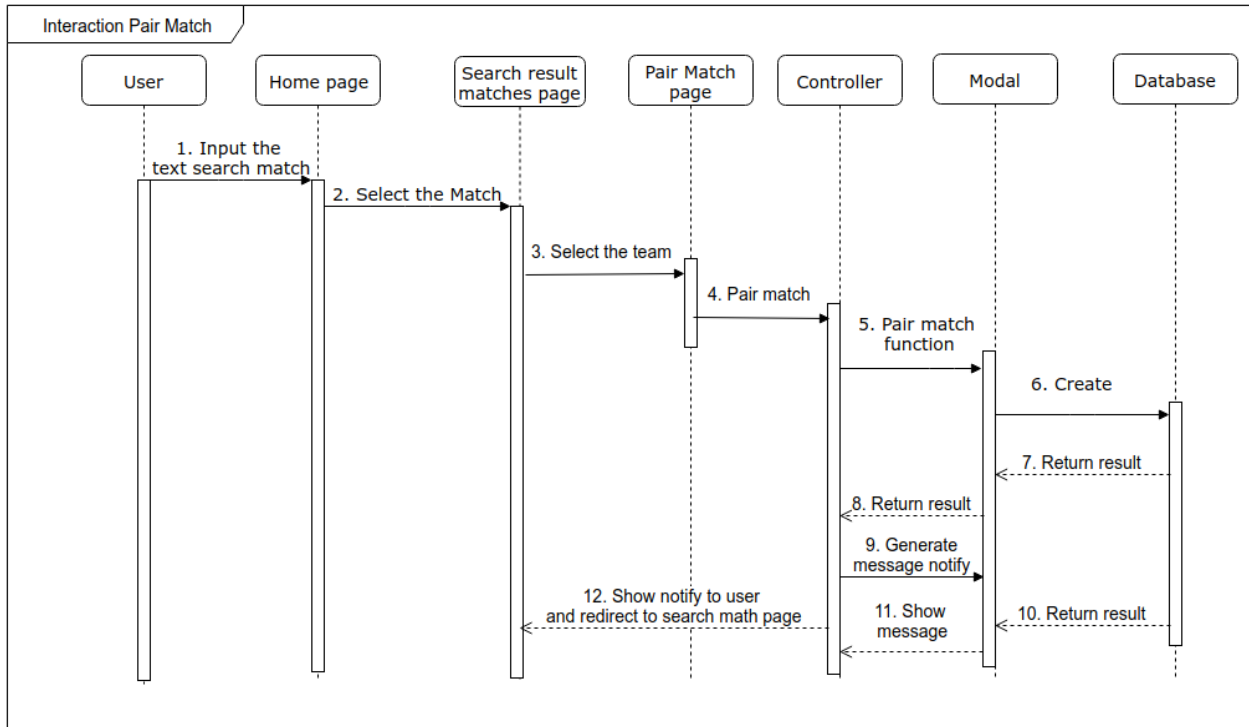


Figure 2. 16 The sequence for pair matches feature

2.5. Basic ERD

Simple entity-relationship diagram of the system shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. The basic ERD is shown in figure 2.17.

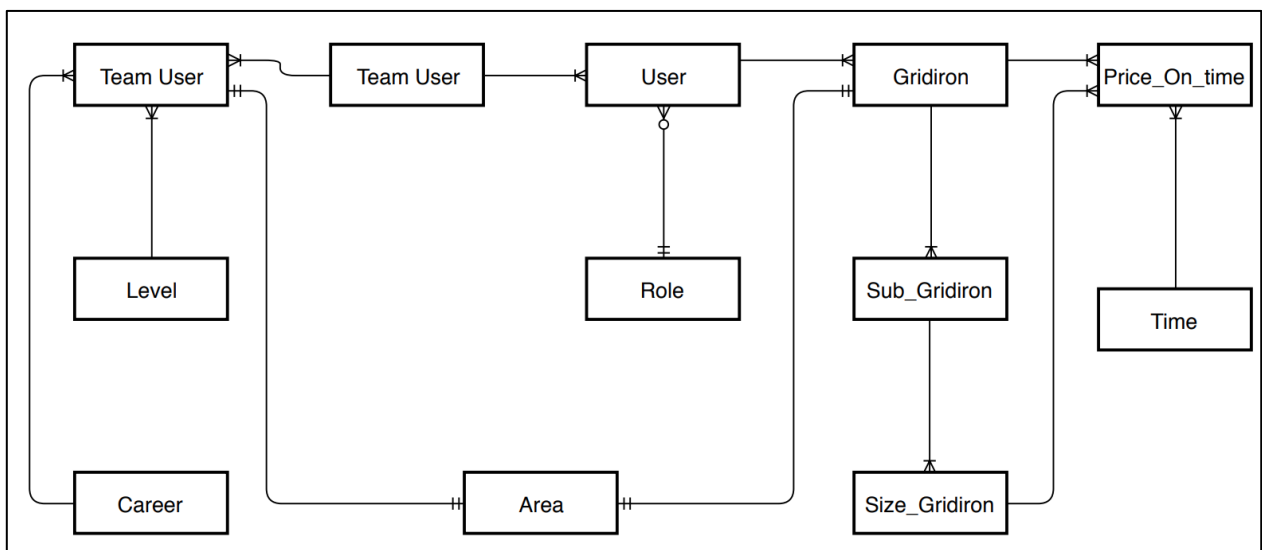


Figure 2. 17 Simple entity-relationship diagram of the system.

2.6. Class diagram

The diagram shown in figure 2.18 describes the detail information of the entities in the application. It includes details fields of each entity and relationship.

The role of the user, admin permissions by view-page. Admin has permission setting type of user and all view-page this one can access.

In short, the diagram describes the basic class and the relationship of the system, we will talk about this more detail in the design section.

The class diagram is shown in figure 2.18.

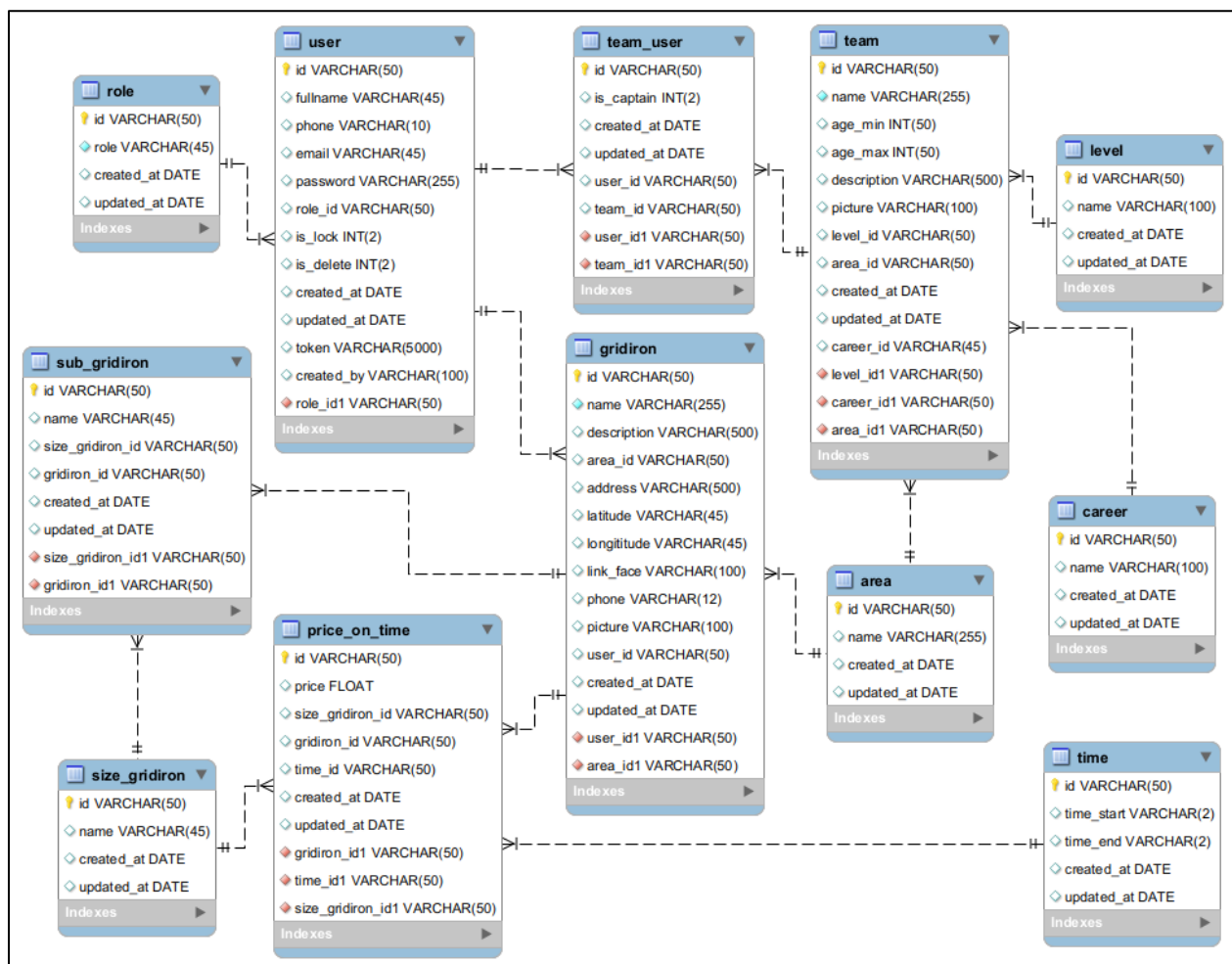


Figure 2. 18 Class diagram

2.7. Conclusion

This chapter introduced the analysis and design of the system supporting football teams. The system includes functions like as Manage the teams, Manage the matches, Manage the leagues, Manage the gridirons, ... In the next chapter, we introduce performance of the designed system.

Chapter 3: IMPLEMENTATION RESULTS

This chapter, we will talk about the implementation results.

3.1. Login function

Normally, users who do not need to log in to the system will still be able to search for gridirons information, matches, and tournaments.

To use functions such as managing matches, gridirons, users need to log into the system. The interface of the login function is shown in the figure 3.1.

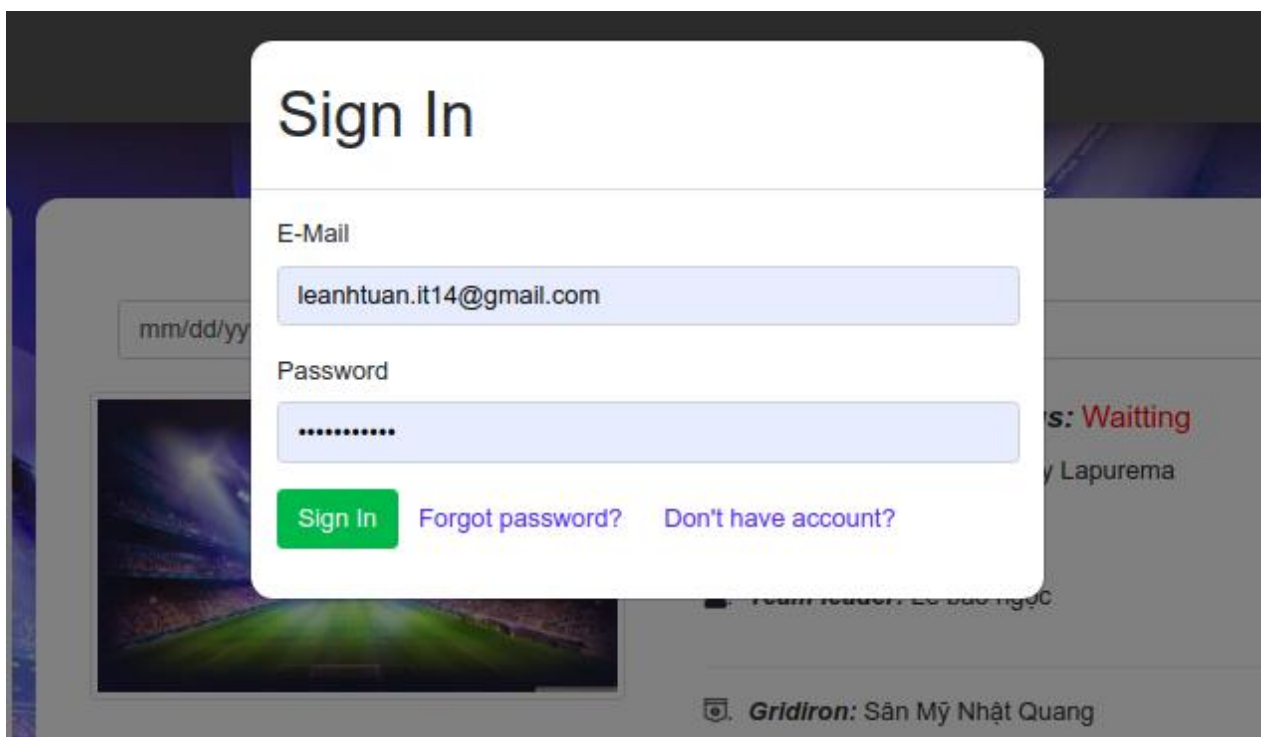
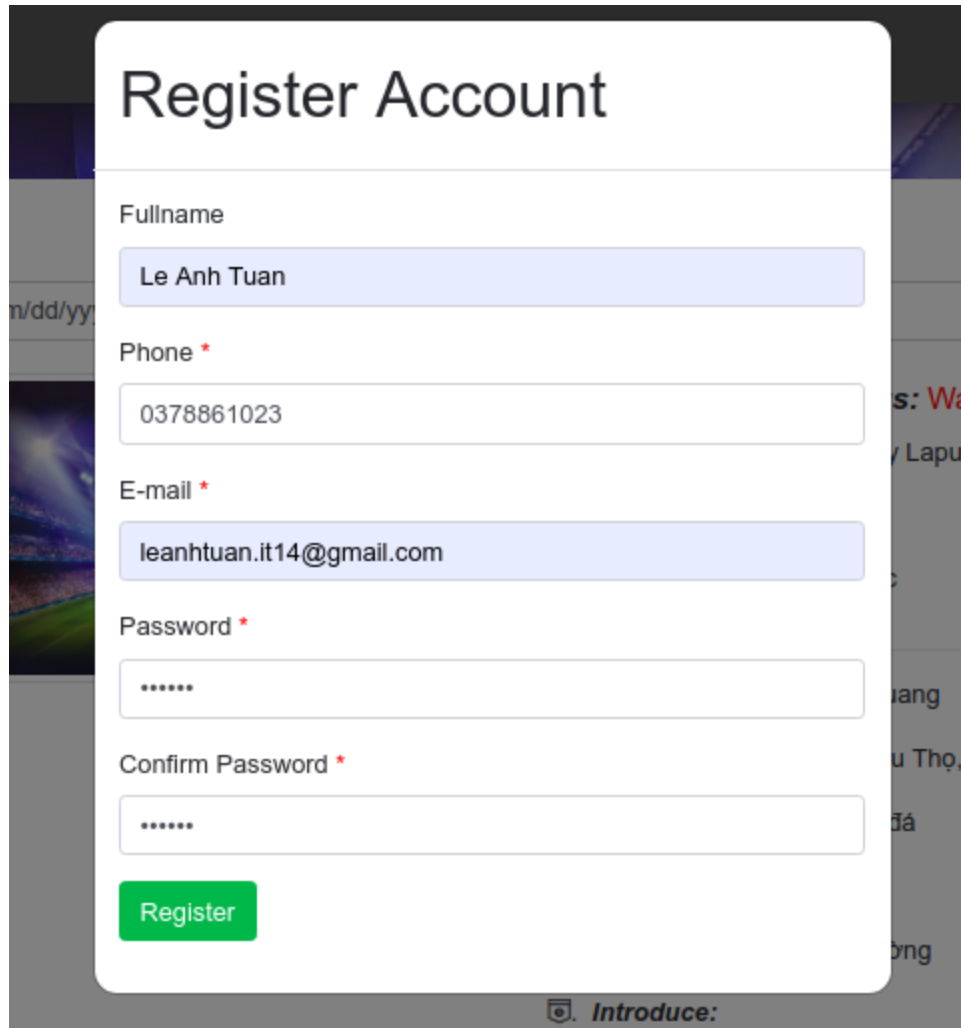


Figure 3. 1 Login page

3.2. Register function

If users do not login to the system, users just use some features such as search the gridirons, the matches, the leagues.

User can register new account to login to the system. The interface of the register function is shown in the figure 3.2.

A screenshot of a web application's registration page. The page has a dark background with a blurred image of a football stadium. In the center, there is a white rounded rectangle containing the registration form. The form is titled 'Register Account' in a large, bold, dark font. Below the title, there are five input fields, each with a label and a red asterisk indicating a required field. The labels are 'Fullname', 'Phone', 'E-mail', 'Password', and 'Confirm Password'. The input fields contain the following text: 'Le Anh Tuan', '0378861023', 'leanhtuan.it14@gmail.com', and two instances of '*****' for the password fields. At the bottom of the form is a green button with the text 'Register' in white. Below the button, there is a small icon of a camera and the text 'Introduce:'.

Register Account

Fullname

Le Anh Tuan

Phone *

0378861023

E-mail *

leanhtuan.it14@gmail.com

Password *

Confirm Password *

Register

Introduce:

Figure 3. 2 Register page

3.3. Reset Password function

If users lost their password, users can reset it. Users type the email then the system will reset and send a new password to the user's email. The interface of the reset password function is shown in the figures (3.3, 3.4).

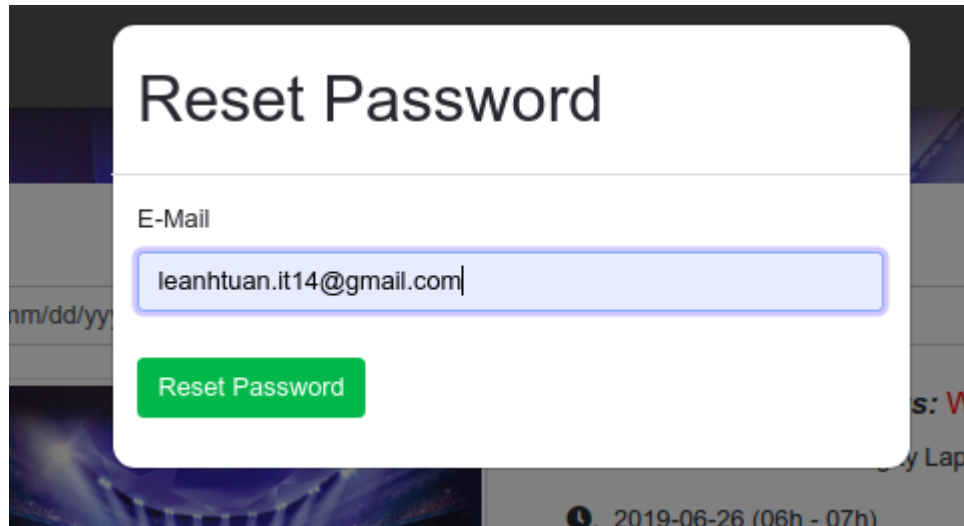


Figure 3. 3 Reset password page

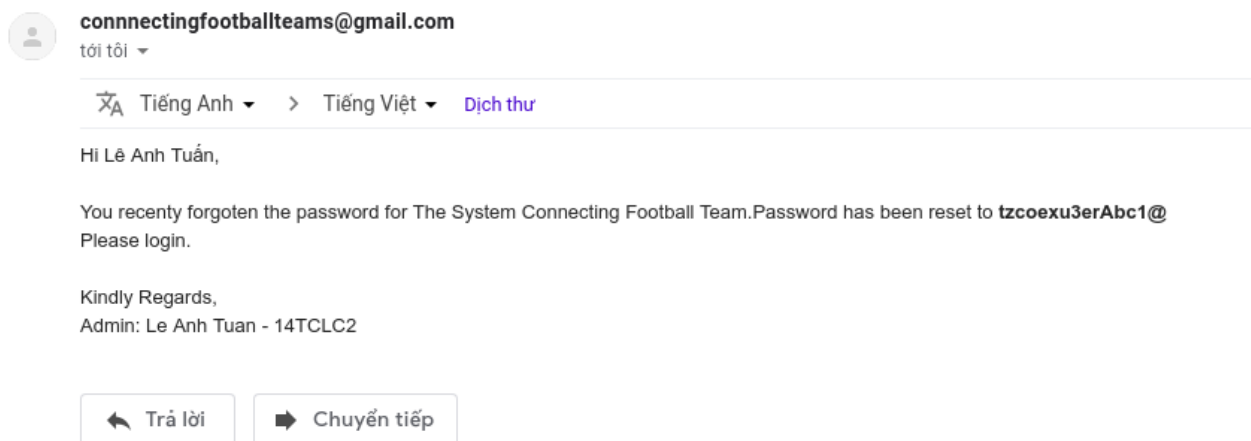


Figure 3. 4 Got email reset password

3.4. Matches management

“Manage Matches” feature requires users to log in to the system. The list of history matches, the interface for update information, “Create new match” are shown in the figures (3.5, 3.6, 3.7).

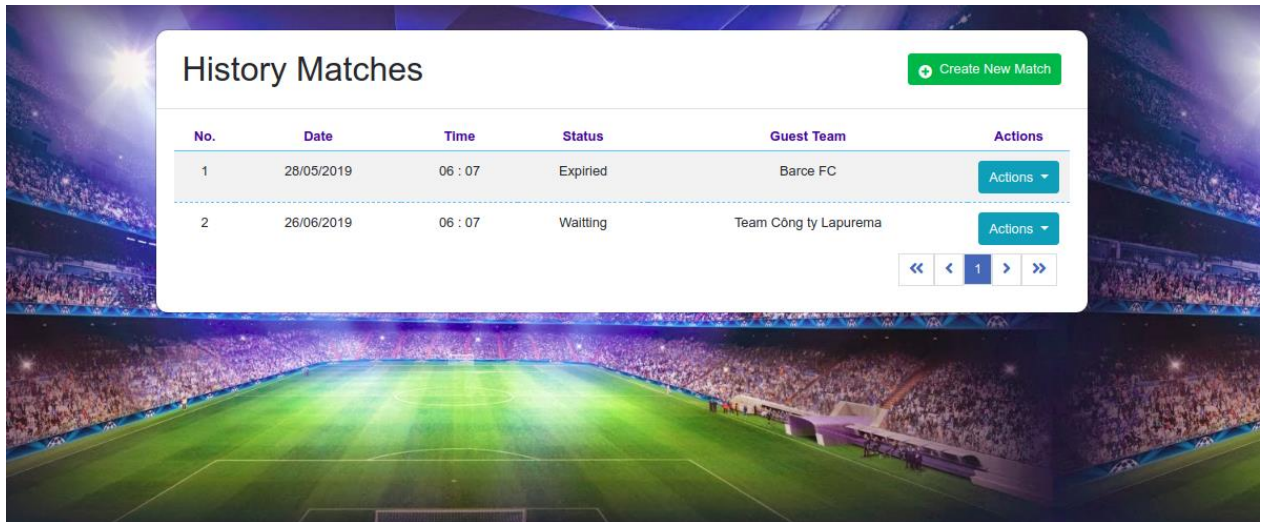
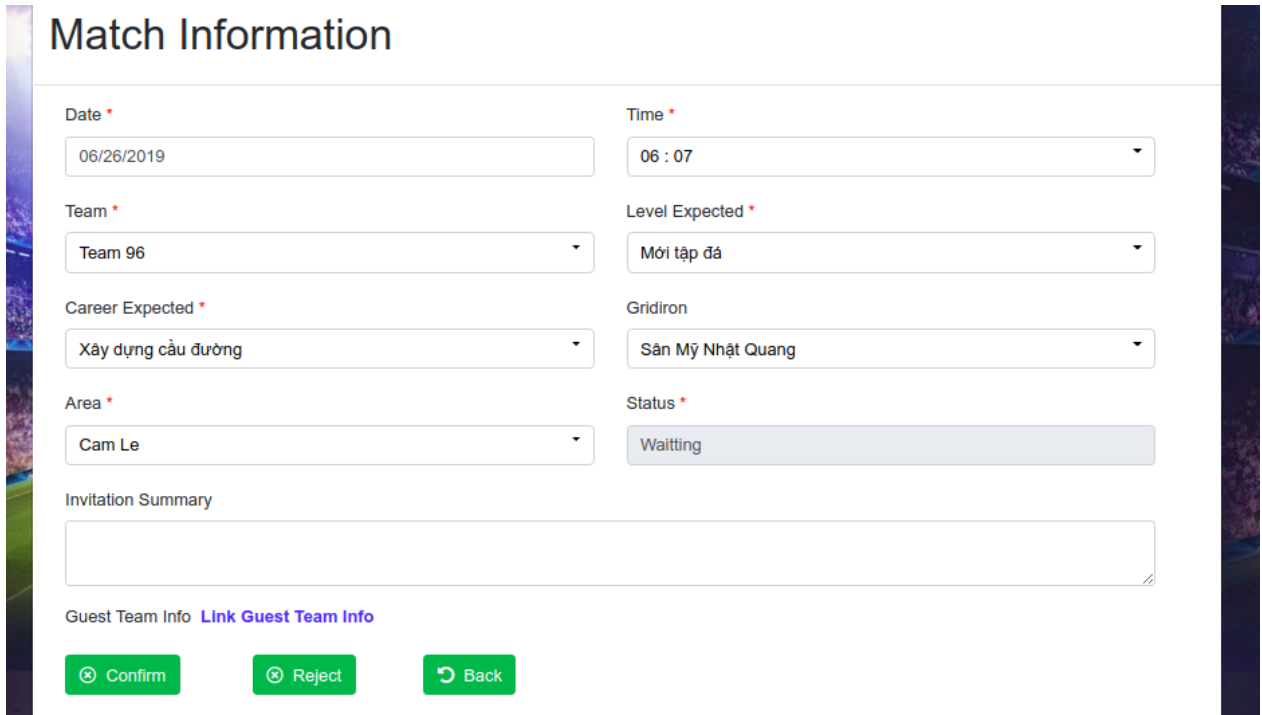


Figure 3. 5. Show list of match page

The screenshot shows a web interface titled "Create New Match". It contains several form fields with red asterisks indicating required fields: "Date" (with a placeholder "mm/dd/yyyy"), "Time" (with a dropdown menu labeled "Select Time"), "Team" (with a dropdown menu labeled "Select Team"), "Level" (with a dropdown menu labeled "Level"), "Career" (with a dropdown menu labeled "Career"), "Gridiron" (with a dropdown menu labeled "Select gridiron"), and "Area" (with a dropdown menu labeled "Area"). Below these fields is a text area labeled "Invitation Summary". At the bottom of the form are two buttons: a green "Create" button and a grey "Cancel" button.

Figure 3. 6. The interface for creating new match



The image shows a web form titled "Match Information". It contains several input fields and dropdown menus for updating match details. The fields are arranged in a grid-like fashion. At the bottom, there are three green buttons: "Confirm", "Reject", and "Back".

Field	Value
Date *	06/26/2019
Time *	06 : 07
Team *	Team 96
Level Expected *	Mới tập đá
Career Expected *	Xây dựng cầu đường
Gridiron	Sân Mỹ Nhật Quang
Area *	Cam Le
Status *	Waiting

Invitation Summary

Guest Team Info [Link Guest Team Info](#)

[Confirm](#) [Reject](#) [Back](#)

Figure 3. 7. The interface for updating match information

3.5. Leagues management

“Manage Leagues” feature requires users to log in to the system. “Manage Leagues” feature has some following interfaces.

The user’s leagues shown in the figure 3.8.



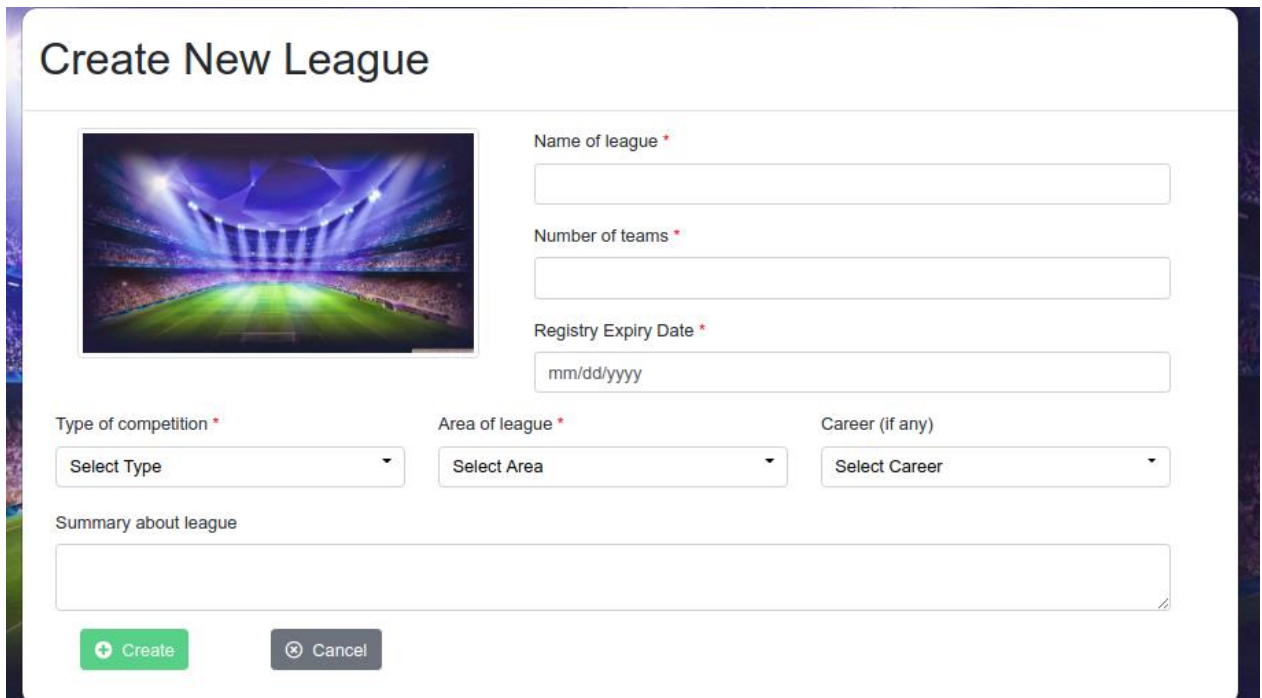
The image shows a web interface titled "List of your league". It features a table with columns for No., Name Of League, Date Expiry Register, Number Of Team Register, Status, and Actions. There are two rows of data. At the top right, there is a green button "Create New League". At the bottom right, there is a pagination control showing "1" and navigation arrows.

No.	Name Of League	Date Expiry Register	Number Of Team Register	Status	Actions
1	Giải TCS mở rộng	26/05/2019	5/9	Inprogress	Actions
2	Giải Lapurema mở rộng	23/05/2019	5/8	Inprogress	Actions

Navigation: << < 1 > >>

Figure 3. 8. Show list of leagues

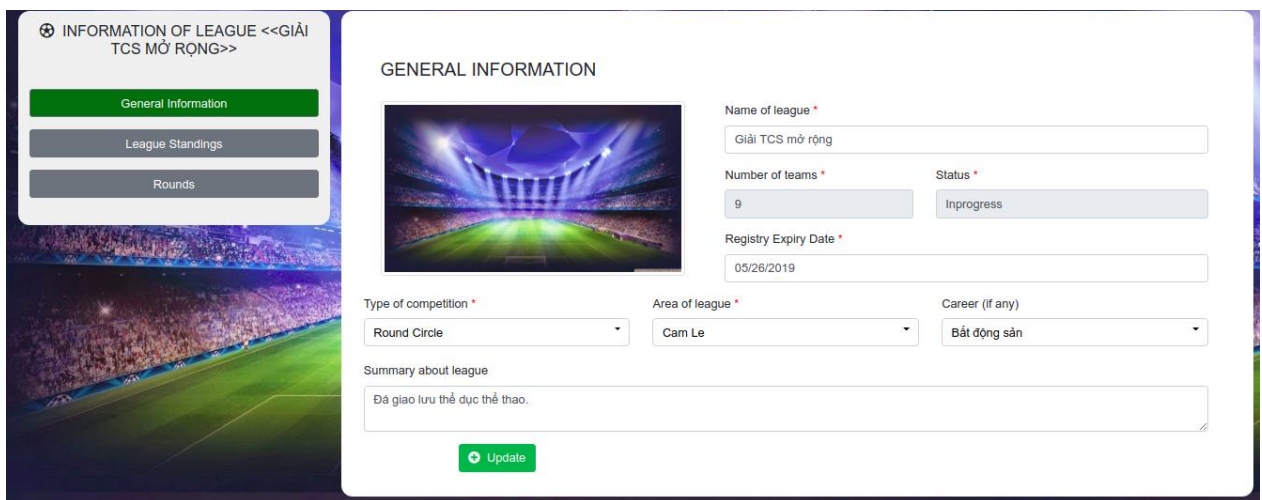
The interface for creating new league function is shown in the figure 3.9.



The screenshot shows a web form titled "Create New League". On the left, there is a placeholder image of a football stadium. The form fields include: "Name of league *" (text input), "Number of teams *" (text input), "Registry Expiry Date *" (text input with a date format hint "mm/dd/yyyy"), "Type of competition *" (dropdown menu with "Select Type"), "Area of league *" (dropdown menu with "Select Area"), and "Career (if any)" (dropdown menu with "Select Career"). Below these is a "Summary about league" text area. At the bottom, there are two buttons: a green "+ Create" button and a grey "⊗ Cancel" button.

Figure 3. 9. The interface for creating league

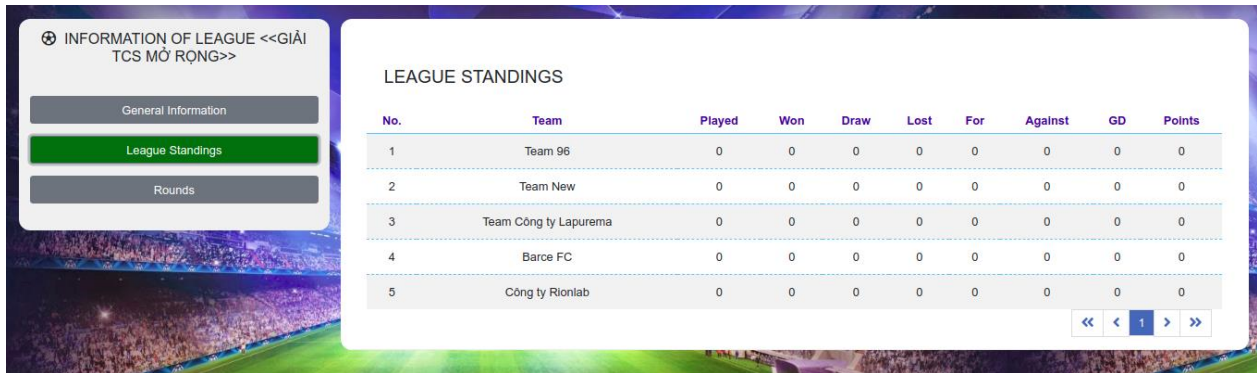
The interface for updating league information function is shown in the figure 3.10.



The screenshot shows a web form titled "GENERAL INFORMATION" for updating league information. On the left, there is a sidebar with a menu: "INFORMATION OF LEAGUE <<GIẢI TCS MỞ RỘNG>>", "General Information" (highlighted in green), "League Standings", and "Rounds". The main form area includes: "Name of league *" (text input with value "Giải TCS mở rộng"), "Number of teams *" (text input with value "9"), "Status *" (text input with value "Inprogress"), "Registry Expiry Date *" (text input with value "05/26/2019"), "Type of competition *" (dropdown menu with value "Round Circle"), "Area of league *" (dropdown menu with value "Cam Le"), and "Career (if any)" (dropdown menu with value "Bất động sản"). Below these is a "Summary about league" text area with the value "Đã giao lưu thể dục thể thao.". At the bottom, there is a green "+ Update" button.

Figure 3. 10. The interface for update league information

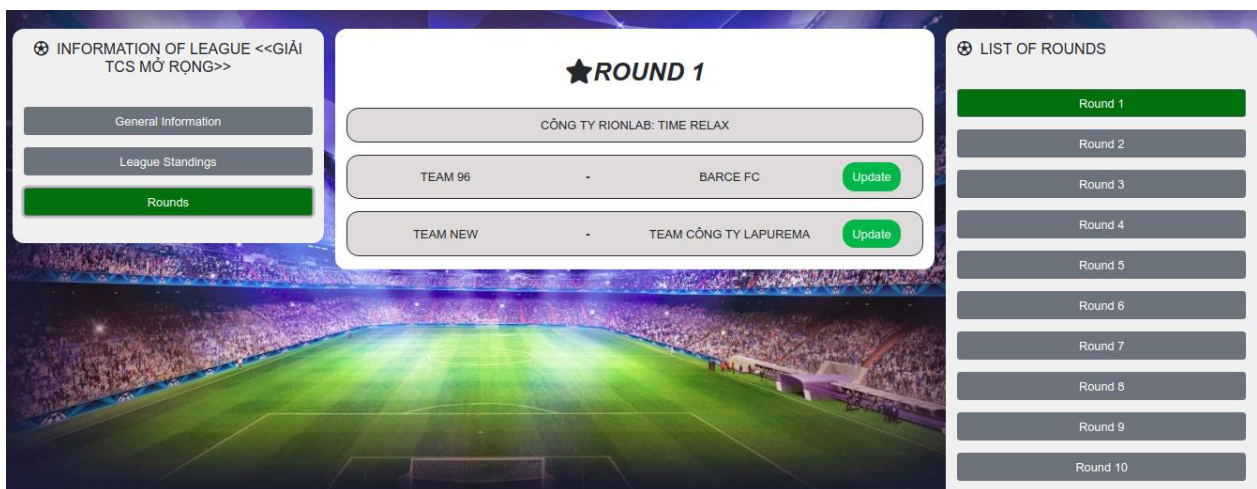
The list standings of teams shown in the figure 3.11.



No.	Team	Played	Won	Draw	Lost	For	Against	GD	Points
1	Team 96	0	0	0	0	0	0	0	0
2	Team New	0	0	0	0	0	0	0	0
3	Team Công ty Lapurema	0	0	0	0	0	0	0	0
4	Barce FC	0	0	0	0	0	0	0	0
5	Công ty Rionlab	0	0	0	0	0	0	0	0

Figure 3. 11. List standings of teams in the league

Rounds of the league is shown in the figure 3.12.



★ROUND 1	
CÔNG TY RIONLAB: TIME RELAX	
TEAM 96	BARCE FC <button>Update</button>
TEAM NEW	TEAM CÔNG TY LAPUREMA <button>Update</button>

LIST OF ROUNDS

Round 1

Round 2

Round 3

Round 4

Round 5

Round 6

Round 7

Round 8

Round 9

Round 10

Figure 3. 12. List rounds of the league

The interface for updating matches information of the league is shown in the figure 3.13.

The screenshot shows a web application interface for updating match information. The main title is 'Match Information' with a green 'Update Information Match' button. The form includes fields for 'TEAM 96' (with a dropdown showing '6'), '3', and 'BARCE FC'. There are also fields for 'Time' (07:08), 'Gridiron' (San Bach Khoa), and 'Date' (06/01/2019). A 'Summary of Match' section is at the bottom. The background features a football stadium image. On the left, there's a sidebar with 'C.F. Team' logo and navigation links: 'INFORMATION OF LEAGUE <<GIẢI TCS MỞ RỘNG>>', 'General Information', 'League Standings', and 'Rounds'. On the right, there's a 'LIST OF ROUNDS' section with buttons for Round 1 through Round 9.

Figure 3. 13. The interface update matches of the league

3.6. Gridirons management

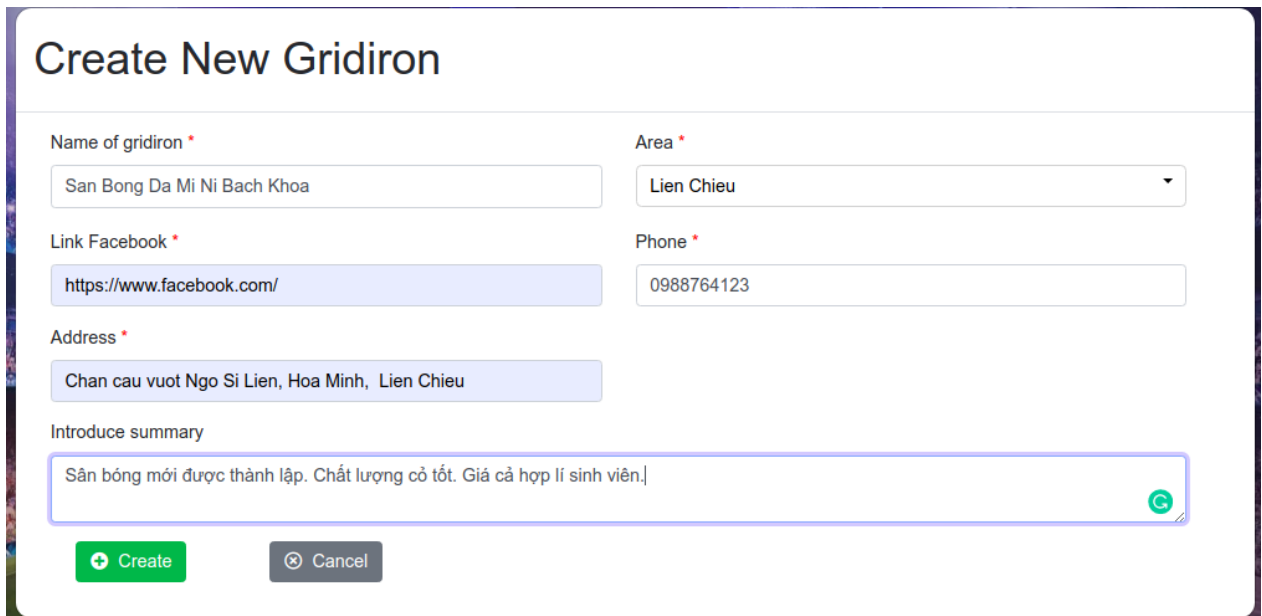
“Manage Gridirons” feature requires users to log in to the system. “Manage Gridirons” feature has some following interfaces. The user’s list of gridirons shown in figure 3.14.

The screenshot shows a web application interface titled 'List Gridiron' with a green '+ Create New Gridiron' button. Below the title is a table with the following data:

No.	Name of gridiron	Area	Actions
1	Sân Quang Nguyễn	Hai Chau	Actions
2	Sân Nam Cao	Lien Chieu	Actions
3	Sân Nguyễn Chánh	Lien Chieu	Actions
4	San Bach Khoa	Lien Chieu	Actions
5	Sân Chuyên Việt	Hai Chau	Actions
6	Sân Trung Vương	Hai Chau	Actions
7	Sân Mỹ Nhật Quang	Cam Le	Actions

Figure 3. 14. Show list of gridirons

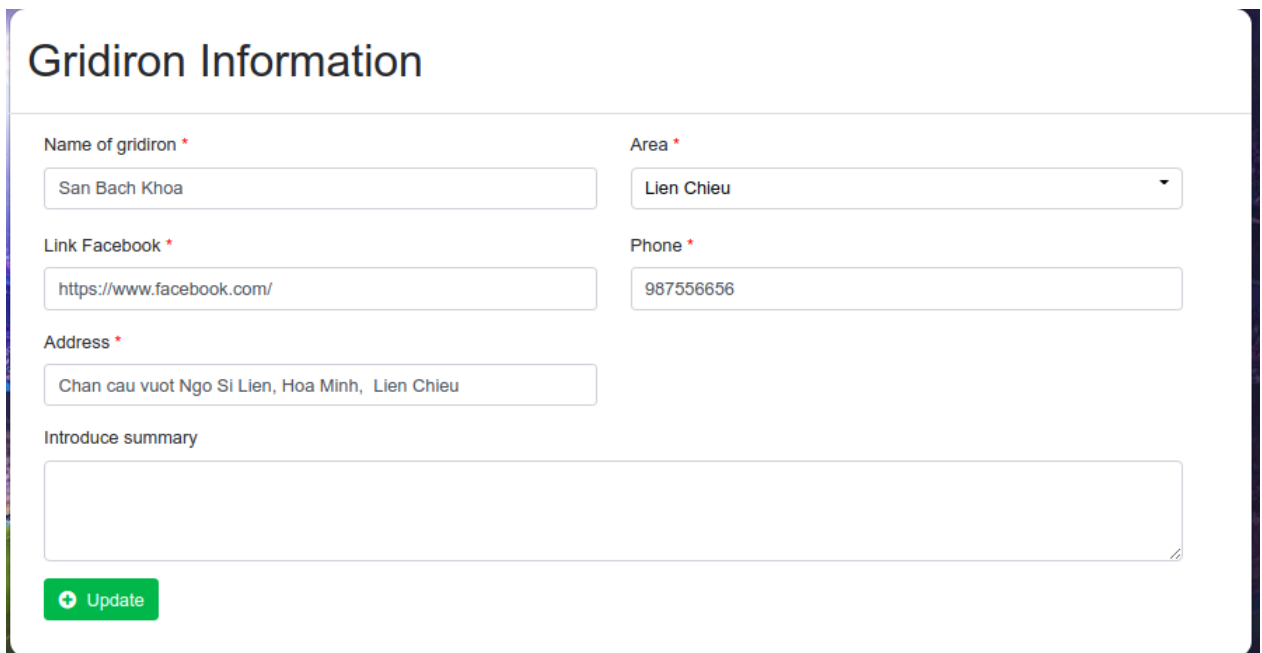
The interface for creating gridirons is shown in the figure 3.15.



The screenshot shows a web form titled "Create New Gridiron". It contains several input fields: "Name of gridiron" with the value "San Bong Da Mi Ni Bach Khoa", "Area" with a dropdown menu showing "Lien Chieu", "Link Facebook" with the value "https://www.facebook.com/", "Phone" with the value "0988764123", and "Address" with the value "Chan cau vuot Ngo Si Lien, Hoa Minh, Lien Chieu". There is also a "Introduce summary" text area containing the text "Sân bóng mới được thành lập. Chất lượng cỏ tốt. Giá cả hợp lý sinh viên." and a green circular icon with a plus sign. At the bottom, there are two buttons: a green "Create" button and a grey "Cancel" button.

Figure 3. 15. The interface for creating gridiron

The interface for updating gridirons is shown in the figure 3.16.



The screenshot shows a web form titled "Gridiron Information". It contains several input fields: "Name of gridiron" with the value "San Bach Khoa", "Area" with a dropdown menu showing "Lien Chieu", "Link Facebook" with the value "https://www.facebook.com/", "Phone" with the value "987556656", and "Address" with the value "Chan cau vuot Ngo Si Lien, Hoa Minh, Lien Chieu". There is also an empty "Introduce summary" text area. At the bottom, there is a green "Update" button.

Figure 3. 16. The interface for updating gridiron

The interface for creating sub-gridirons is shown in the figure 3.17.

Sub gridiron

Type of gridiron * Number of gridiron *

Number of sub gridiron *

☒ Create

Search ...

No.	Name	Type of Gridiron	Actions
1	San N1	Sân 5 người	Actions ▾
2	San N2	Sân 5 người	Actions ▾

Figure 3. 17. The interface for creating sub-gridiron

The interface for creating the price of gridirons is shown in the figure 3.18.

Price by the time

Type of gridiron * Time * Price (VNĐ) *

☒ Create

Search ...

No.	Time	Type of gridiron	Price	Actions
1	16 : 17	Sân 5 người	200000	Actions ▾
2	17 : 18	Sân 5 người	250000	Actions ▾
3	18 : 19	Sân 5 người	350000	Actions ▾
4	19 : 20	Sân 5 người	300000	Actions ▾

Figure 3. 18. The interface for creating the price of gridiron

3.7. Teams management

“Manage Teams” feature requires users to log in to the system. “Manage Teams” feature has some following interfaces.

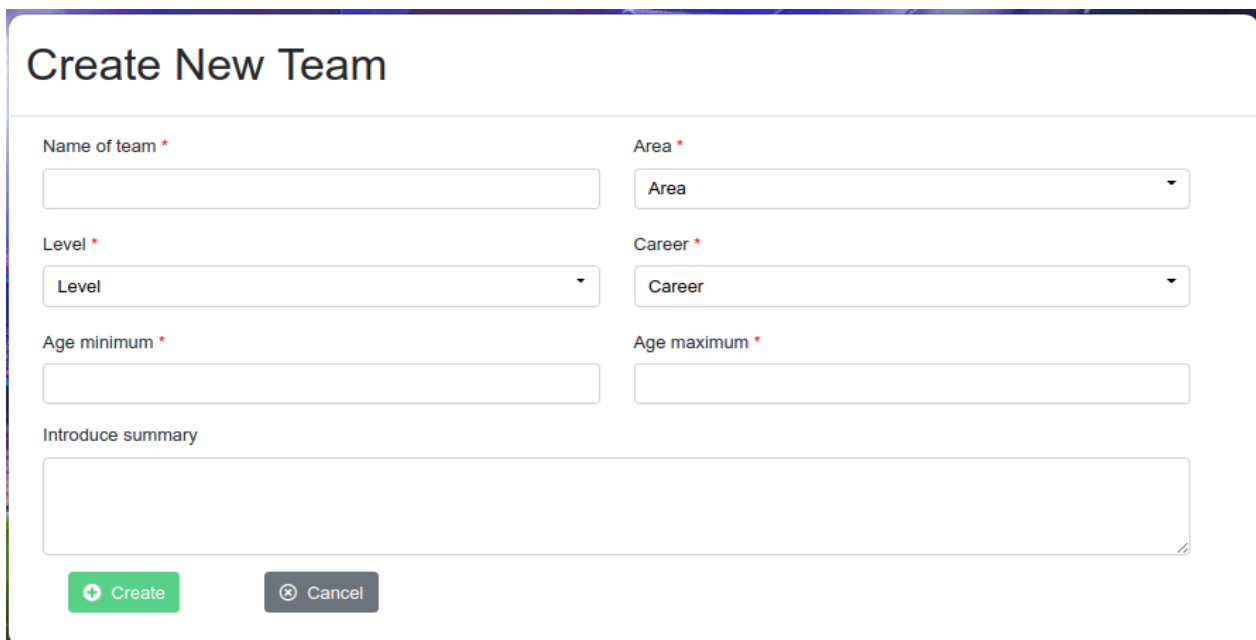
The user’s list of teams shown in the figure 3.19.



No.	Name of team	Actions
1	MU FC	Actions ▾
2	Real FC	Actions ▾
3	Barce FC	Actions ▾
4	Team 5	Actions ▾
5	Team Công ty Lapurema	Actions ▾
6	Công ty Rionlab	Actions ▾
7	Team New	Actions ▾
8	Team 96	Actions ▾

Figure 3. 19. Show list of teams

The interface for creating team is shown in the figure 3.20.



Create New Team

Name of team *

Area *

Area ▾

Level *

Level ▾

Career *

Career ▾

Age minimum *

Age maximum *

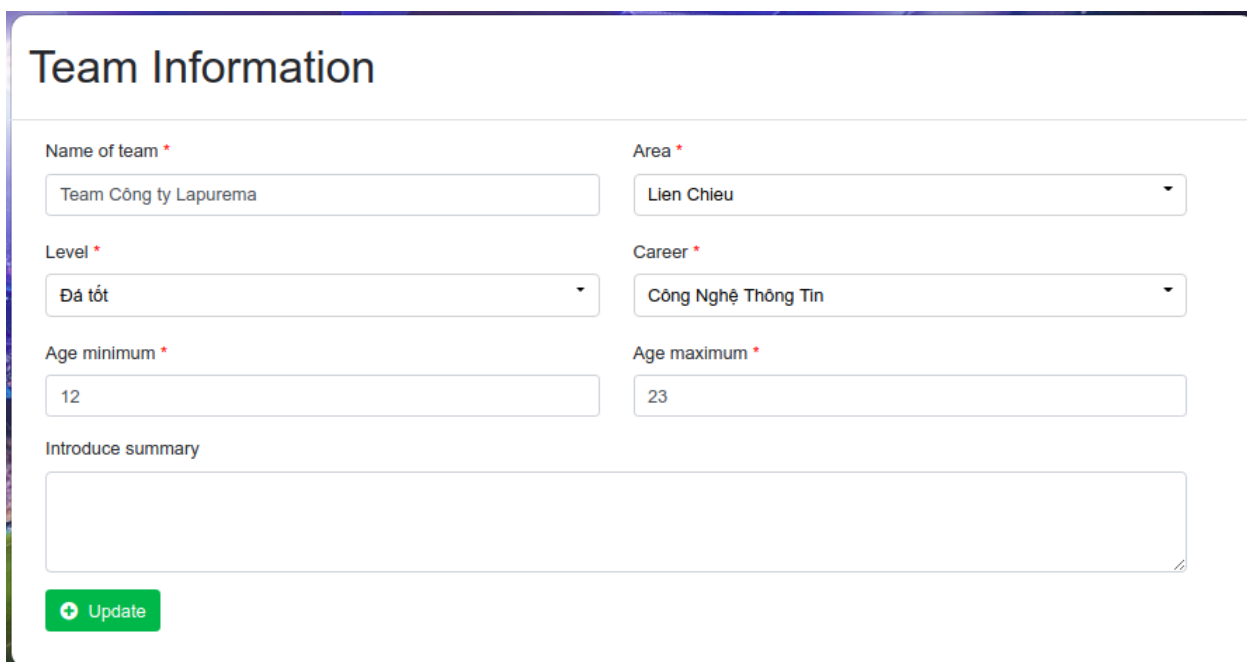
Introduce summary

Create

Cancel

Figure 3. 20. The interface for creating team

The interface for updating team is shown in the figure 3.21.



The 'Team Information' form contains several input fields and a summary section. It includes fields for 'Name of team' (text), 'Area' (dropdown), 'Level' (dropdown), 'Career' (dropdown), 'Age minimum' (text), and 'Age maximum' (text). Below these is a large text area for 'Introduce summary' and a green 'Update' button.

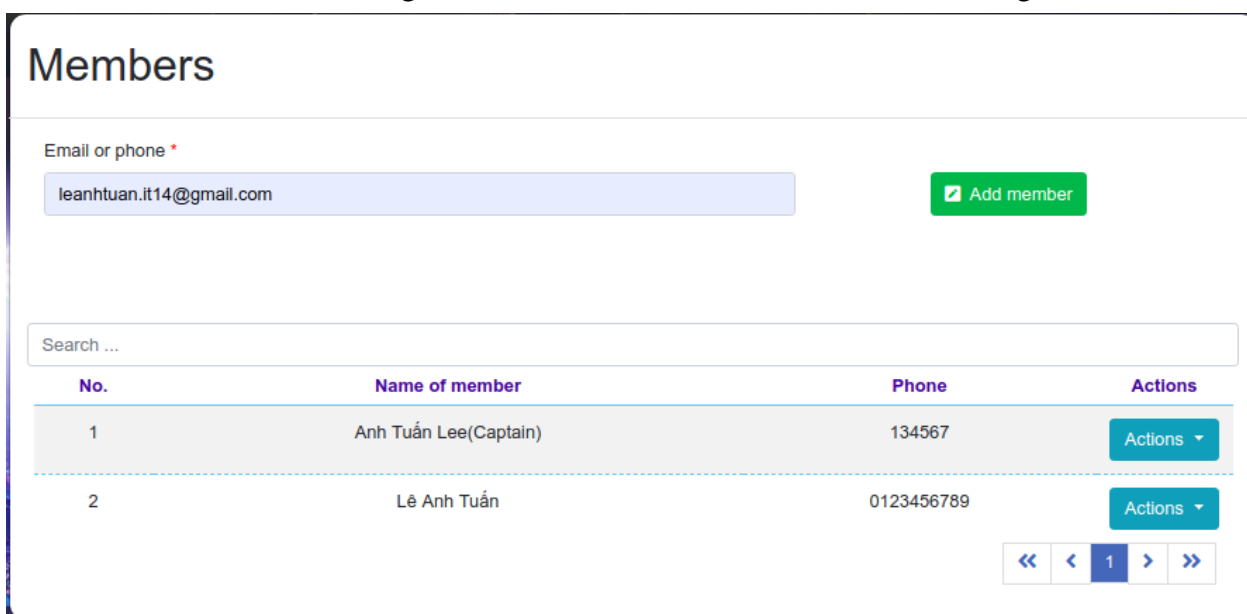
Field	Value
Name of team *	Team Công ty Lapurema
Area *	Lien Chieu
Level *	Đá tốt
Career *	Công Nghệ Thông Tin
Age minimum *	12
Age maximum *	23

Introduce summary

Update

Figure 3. 21. The interface for updating team

The interface for adding new member to the team is shown in the figure 3.22.



The 'Members' interface features a form to add a new member and a table of existing members. The form includes an 'Email or phone' field and an 'Add member' button. The table lists members with columns for 'No.', 'Name of member', 'Phone', and 'Actions'. A search bar is located above the table, and pagination controls are at the bottom right.

Email or phone *

leanhtuan.it14@gmail.com

Add member

Search ...

No.	Name of member	Phone	Actions
1	Anh Tuấn Lee(Captain)	134567	Actions
2	Lê Anh Tuấn	0123456789	Actions

« < 1 > »

Figure 3. 22. The interface for adding new member to the team

3.8. Pairing Match

On the “Public home page” site users can select “Pair” then select team to pair the match (figure 3.23).

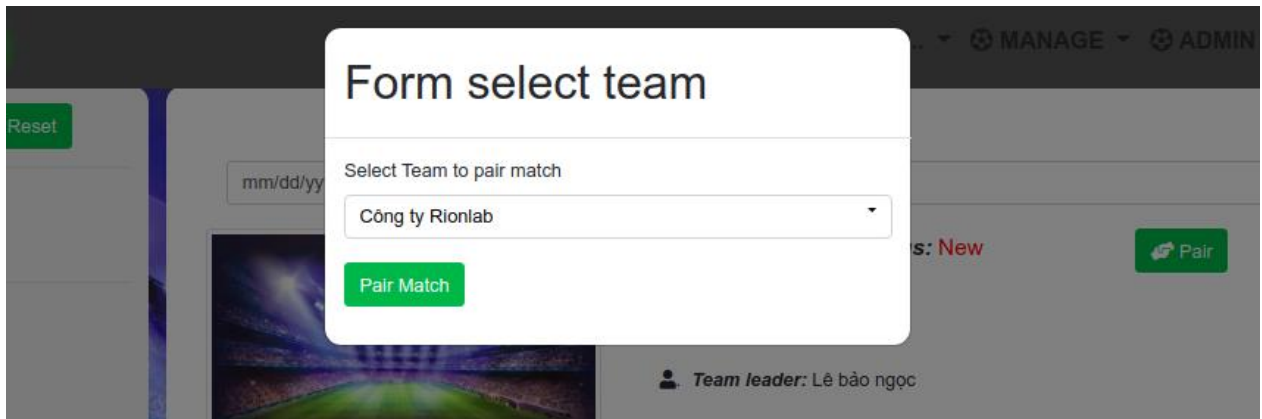


Figure 3. 23. The interface for pairing match

Once the user pair match successfully, the system generates a notification and send to the match creator. this feature is shown in the figures 3.24.

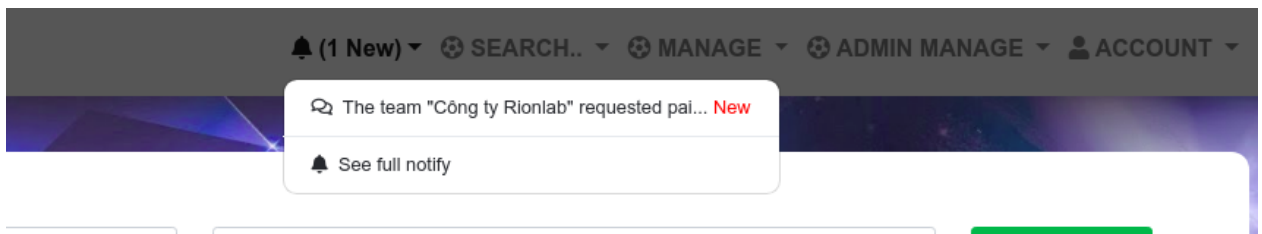


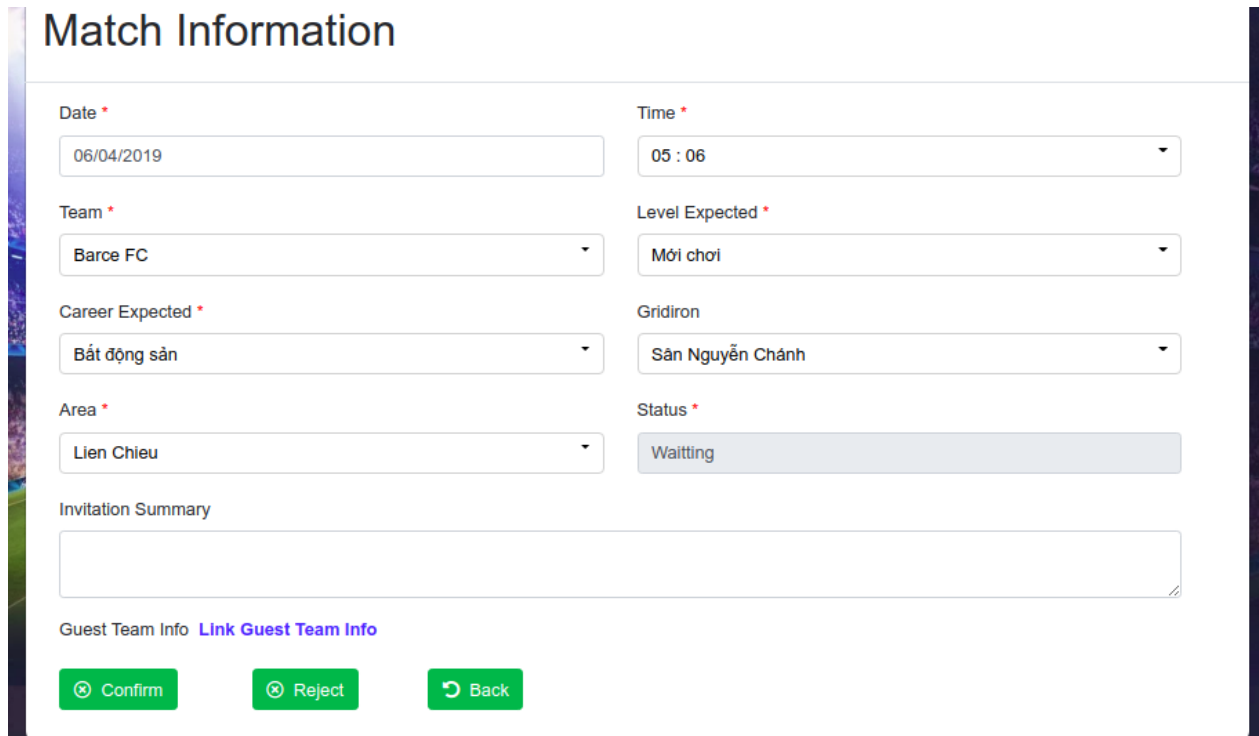
Figure 3. 24 Show full list notification



Figure 3. 25. Show full list notification

3.9. Confirm, Reject Matches

On the detail of site “Manage Matches”, the user can confirm or reject the requested pair match. this feature is shown in the figure 3.25.



The screenshot shows a web interface titled "Match Information". It contains several form fields:

- Date ***: A text input field containing "06/04/2019".
- Time ***: A dropdown menu showing "05 : 06".
- Team ***: A dropdown menu showing "Barce FC".
- Level Expected ***: A dropdown menu showing "Mới chơi".
- Career Expected ***: A dropdown menu showing "Bắt động sân".
- Gridiron**: A dropdown menu showing "Sân Nguyễn Chánh".
- Area ***: A dropdown menu showing "Lien Chieu".
- Status ***: A button labeled "Waiting".
- Invitation Summary**: A large text area for notes.
- Guest Team Info**: A link labeled "Link Guest Team Info".
- Buttons**: Three buttons at the bottom: "Confirm" (green), "Reject" (green), and "Back" (green).

Figure 3. 26. The interface for function confirms or reject match

Once confirm or reject the requested pair match, the system will auto generate a notification and send to the requester. this feature is shown in the figures 3.26.

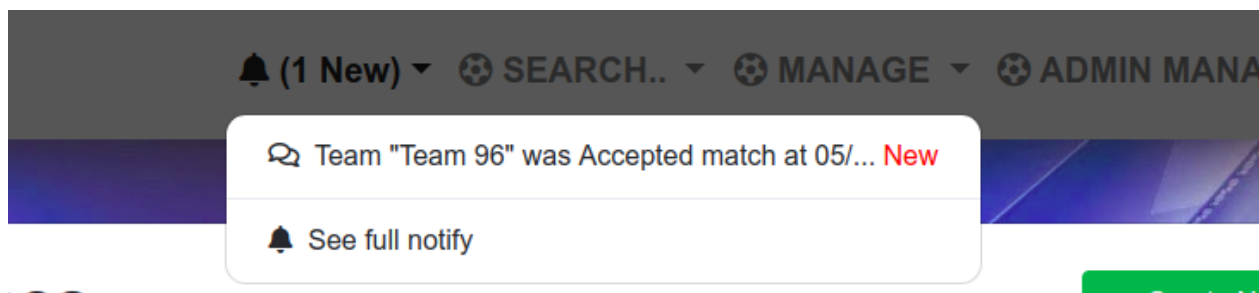


Figure 3. 27 Show message confirmed request



Figure 3. 28. Show full list of notification

3.10. Register join league

On the “Public league page” site users can select “Register” then select team to register to join the league. This feature is shown in the figure 3.27.

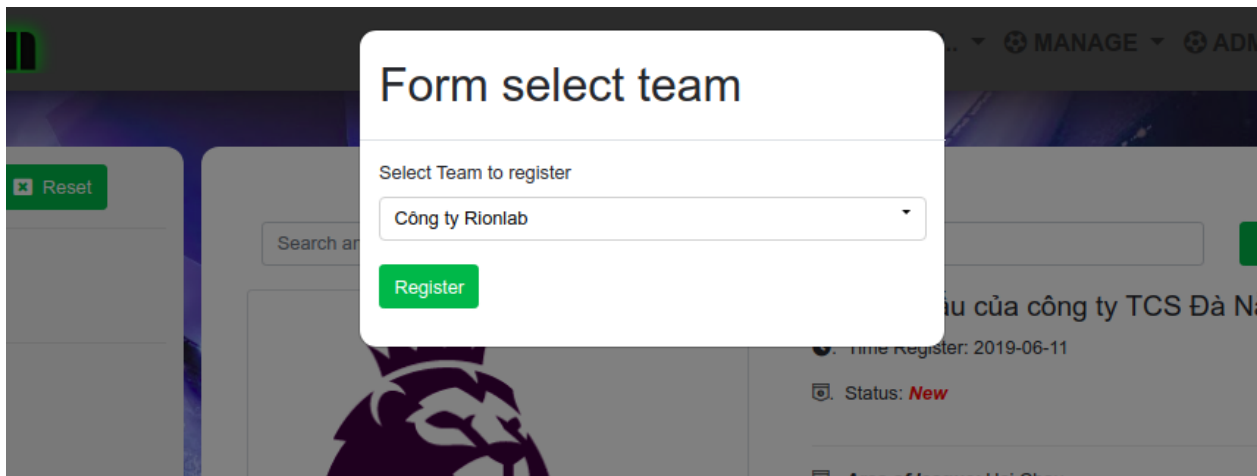


Figure 3. 29. The interface for registering to join league function

Once register successfully, the user can check in the detail of league as the figure 3.28.

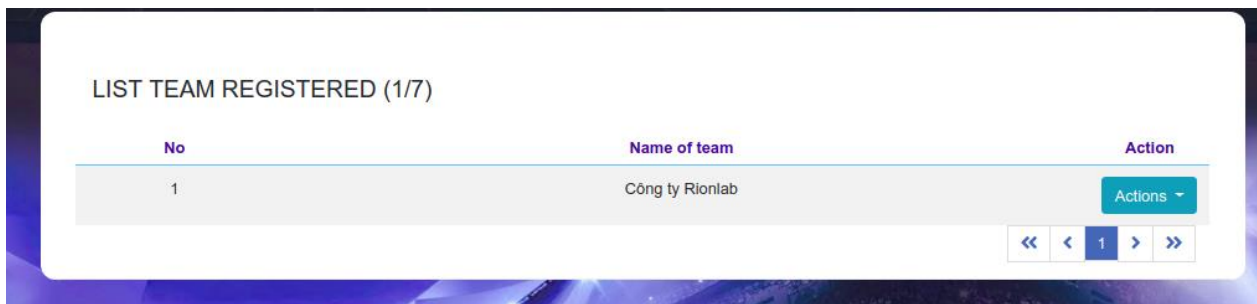


Figure 3. 30. Show list of teams registered to join league

3.11. Home page

The figure 3.31 shown “Public matches page” of the system which the user can search the matches and pair the matches.

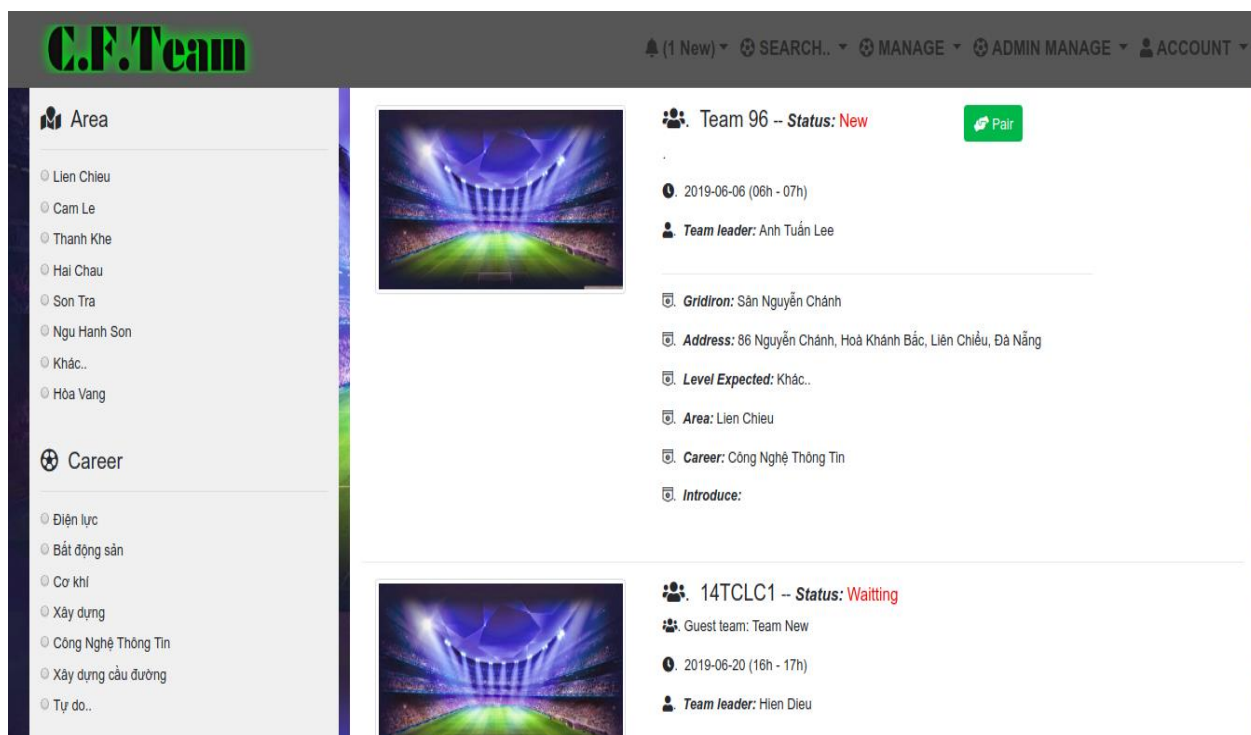


Figure 3. 31 Public matches page

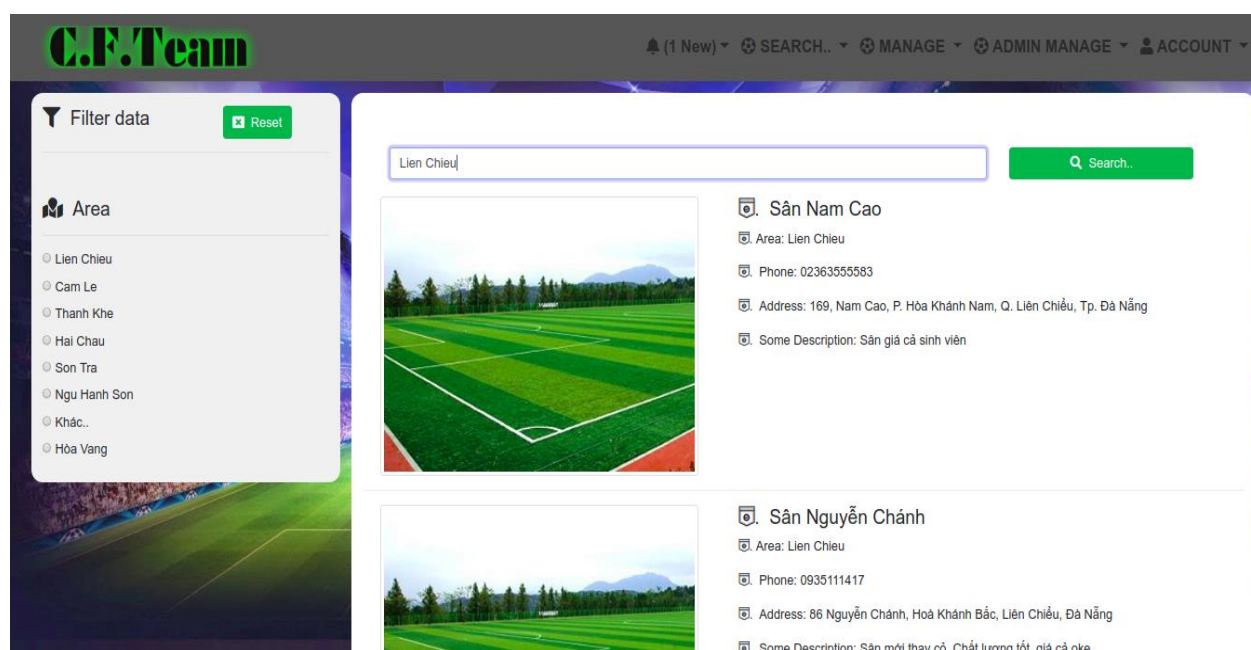


Figure 3. 32 Public gridirons page

Figure 3.32 above shown “Public gridirons page” of the system which the user can search then view detail of the gridiron.

Figure 3.33 shown “Public leagues page” of the system which the user can search then view detail or register to join.

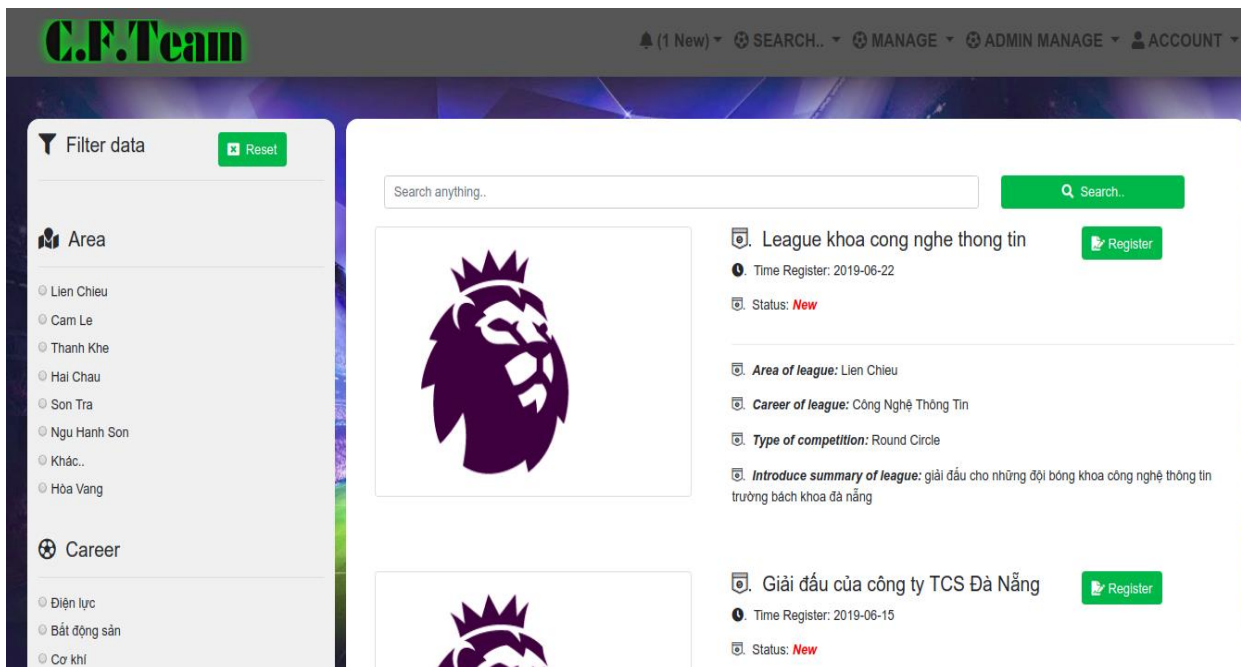


Figure 3. 33 Public leagues page

CONCLUSION AND FUTURE WORK

1. Achieve results

In this project, we have been learning JavaScript language and can understand how to work with Node JS Platform. We use JavaScript to program asynchronously with Node JS Platform to build the API server. We use the Angular framework to build the application. We use the MYSQL and Firebase database to store our database. We also improved researching skill, technical skill, presentation skill, English skill and other soft skills when we build this project.

With gained bits of knowledge, we built the graduation project satisfying initial requirements with following achievements:

- The system supports user to manage the gridirons, the teams, the leagues, the team's level, the team's career, the matches and the areas.
- The system can support user to generate the schedule of leagues, pair match with other teams, notify to the user when the request pairing match is confirmed or rejected.

However, with advantages, this web system still has following foibles:

- The user just can search the gridirons at Da Nang city but cannot track which the gridirons are empty.
- The system cannot support for the user to book the gridirons on the internet.
- The system cannot support to send a message to the user's phone when the system receives the request to pair match.

2. Future works

With the disadvantages which are mentioned above. In the future we will develop some following features:

- Supporting the user to book the gridirons on the internet.
- To send the message to the user's phone to notify if the system receives the request to book gridirons or to pair matches.
- Supporting the user to track the gridirons which are available or not.

REFERENCES

- [1] Nodejs documentation. Available on website: <https://nodejs.org/>
- [1.1] The image is available on website: <https://developer.ibm.com/tutorials/learn-nodejs-tour-node/>
- [1.2] The image is available on website: <https://www.slideshare.net/danielsju6/iosdevcamp-firebase-overview>
- [2] MySQL Server documentation available on website: <https://dev.mysql.com/doc/refman/8.0/en/>
- [3] MySQL Workbench documentation available on website: <https://dev.mysql.com/doc/workbench/en/>
- [4] Firebase database documentation available on website: <https://firebase.google.com/docs/database>
- [5] Angular framework documentation available on website: <https://angular.io/docs>
- [6] React-native framework documentation available on website: <https://facebook.github.io/react-native/docs/getting-started.html>
- [7] React-native architecture documentation available on website: <https://www.reactnative.guide/3-react-native-internals/3.1-react-native-internals.html>