# MiniProject Report *on*

**Football Tournament Database Management**

**FootWiki**

## Submitted by

**Siddarth Mohan**

**Contents**

[Abstract III](#_TOC_250005)

1. [Introduction… 1](#_TOC_250004)
2. Problem

definition… 2

1. [Tools and Technologies used… 3](#_TOC_250003)
2. Database Design… 4
3. [Database Schema… 5](#_TOC_250002)
4. [Relational Database Design Using Schema Diagram… 6](#_TOC_250001)
5. [Database Normalization till 3 NF… 7](#_TOC_250000)
6. DDL… 8
7. TRIGGER… 10
8. Screenshots of WebPages… 12
9. Conclusion… 19

## Abstract

The purpose of this project is to provide a friendly environment to maintain the details of players, team, league and every recode needed. The main purpose of this project provide different reports for different function. This project report describes the hardware and software interface requirement using ER diagrams and database queries and screenshots of the GUI of the web application.

## 1 . Introduction

1. Football tournament Database Management System is an application that will be used to maintain and store data of football tournaments. Main objective is to collect and store various details of the tournament including details of participating clubs with their managers and players and match details such as final score and goals scored by players. This data can be collected and be used for tournament analysis and to keep records of the tournaments. Initially details of participating clubs , their players and their managers are stored in the database and as each match is played, the details of the match between two teams is stored in the database.

Data collected can be used to find facts such as the top scorer , club with most goals scored

, club with least goal scored .

1. The following data will be stored and maintained in the application:
   * Various clubs and their information ( Club id , Club name , home stadium and country,user\_id)
   * Details of Managers (manager id, name , age , country,club ID,user\_id)
   * Details pf players ( player id , name , position , age, nationality , club ID , manager ID,user\_id )
   * Match details(match id , home score , away score , home ID , away ID match date,user\_id )
   * Goals scored (goal id , goal time, player ID, match ID,user\_id )

## 2. P roblemDefinition

Create an application to store various details and data of a club football tournament The application should be able to sore and maintain following details :

* Participating clubs their details . Each club has an unique club ID which is stored along with name of the club , home stadium name and their country.
* Details of Managers -. Each manager is hired by a club and has an unique manager ID which is stored along with name of the manager , age and nationality
* Details of players -. Each player plays for one of the participating club and is managed by a

manager. Each player has an unique player ID which is stored along with name of the player , playing position , age and nationality

* Match details - Each match is played between two clubs and has an unique match ID

which is stored along with home score , away score and match date.

* Goals scored - Zero or more goals are scored in a match and each goal is scored by a player

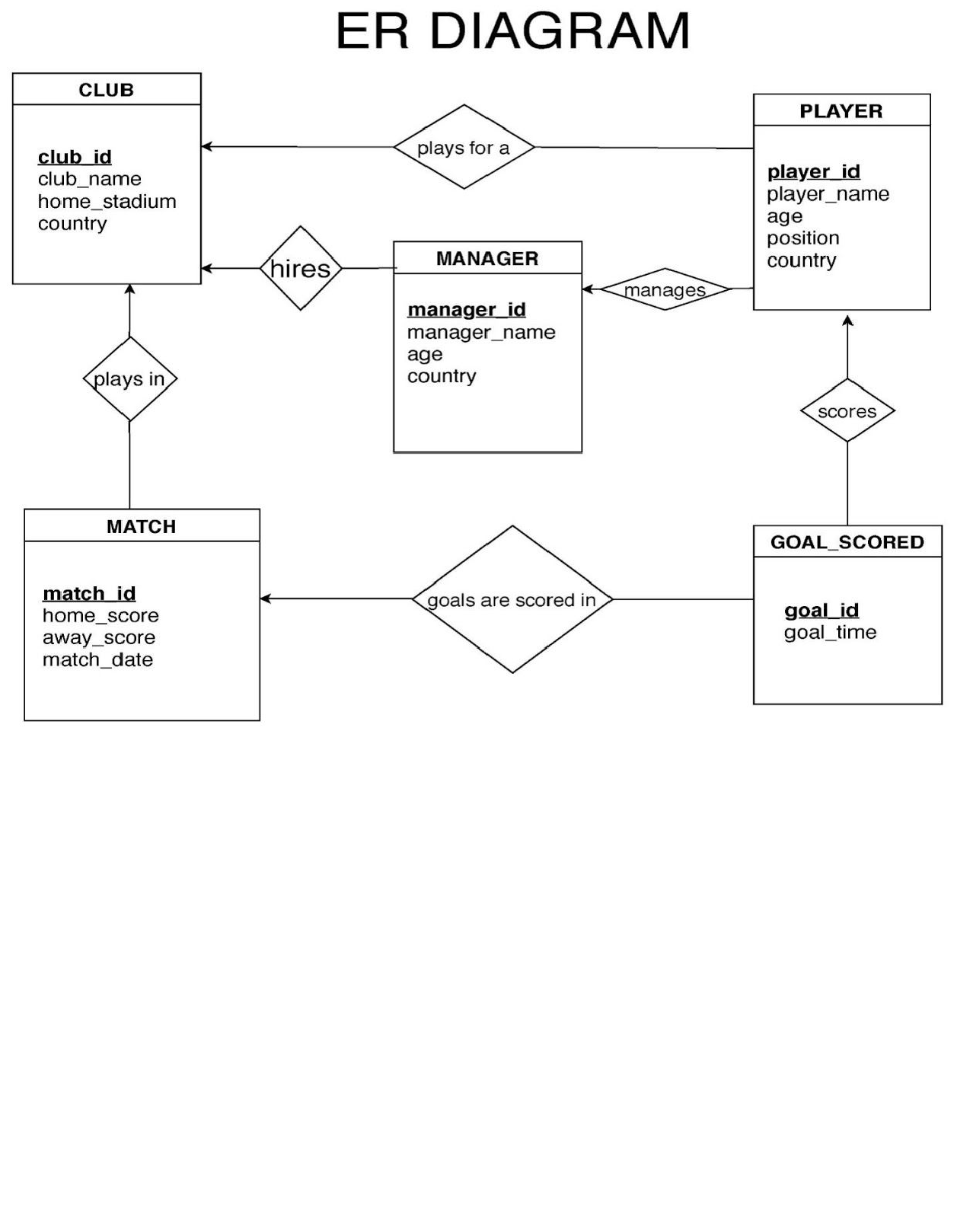
. goal details are stored with a unique goal ID and time of the scored goal.

## 3. Tools and Technologies Used

* Draw.io : for ERD and Relational Schema
* Goorm ide. : for writing the code for web application
* Html, CSS, Javascript : for front end web development
* Nodejs, express framework , MySQL database : for backend web development
* Packages used:
* "bcrypt": "^5.0.0",
* "body-parser": "^1.19.0",
* "ejs": "^3.1.5",
* "express": "^4.17.1",
* "express-fileupload": "^1.2.0",
* "express-session": "^1.17.1",
* "mysql": "^2.18.1",
* "req-flash": "0.0.3"
* Linux : Operating System used

CET,SCET, MITWPUPage 4

## Database Design (ER Diagram)



User\_id

User\_id

User\_id

User\_id

User\_id

## . Database Schema

1)Club

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **C lub\_ID** | Club\_Name | Home\_stadium | Country | user\_id |

2)Manager

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **M anager\_id** | Manager\_name | Age | Country | Club\_id | user\_id |

3)Player

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **P layer\_id** | Player\_Name | Age | Position | Country | Club\_id | Manager\_id | user\_id |

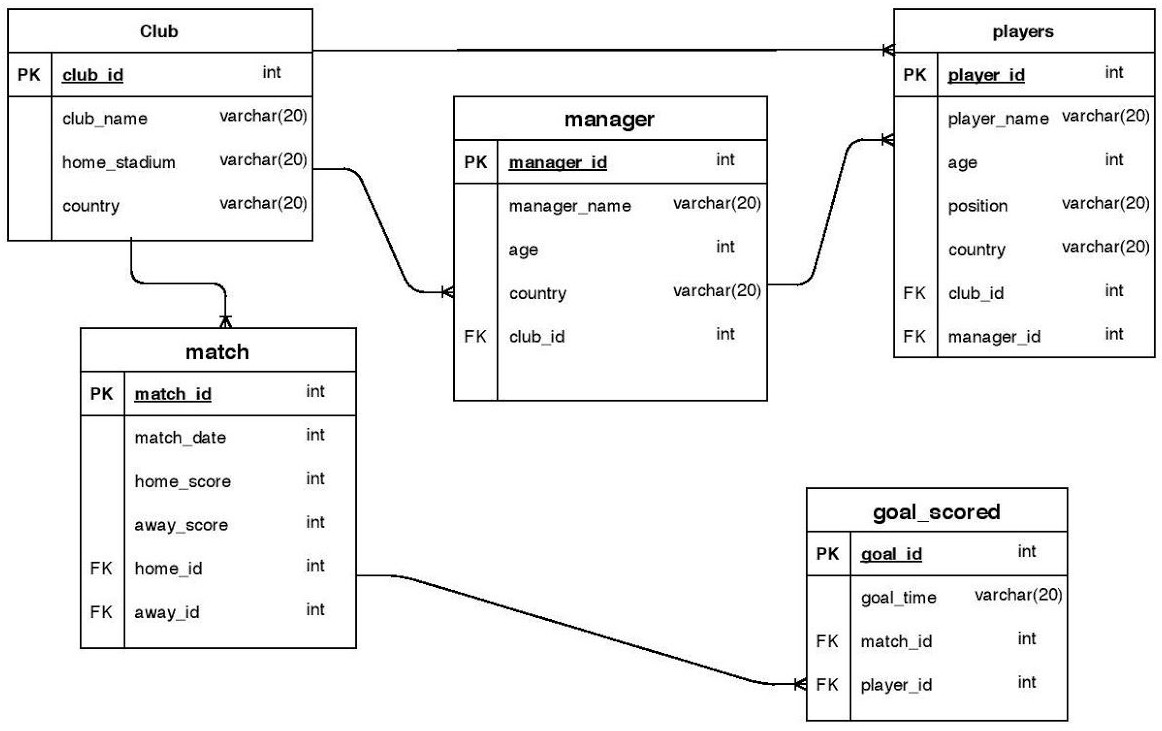
4)Match

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **M atch\_id** | Match\_date | Home\_score | Away\_score | Home\_id | Away\_id | user\_id |

5)Goal\_scored

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G oal\_id** | Goal\_time | Match\_id | Player\_id | user\_id |

## R elational Database Design using schema diagram



## . Database Normalization till 3 NF

1NF:

The given database is of 1st normal form because all rows are atomic, that is all the tables are no multivalued attributes so we don’t have to split rows to make it in the normal form.

2NF:

The given database is of 2nd normal form because there are no partial dependencies in the schema’s.

There are no columns in any table that are functionally dependent on more than one column.

This satisfies the condition that there are no partial functional dependencies. 3NF:

All transitive dependencies have been removed. So there can be only one functional dependency to determine each column. That is there is no column which is dependent on any one column which is again dependent on another column which means there are no transitive dependencies. So the database is of 3rd normal form.

**DDatabase Creation commands:**

create table clubs( club\_id int primary key,

club\_name varchar(20), home\_stadium varchar(20), country varchar(20),

image varchar(255),

user\_id int

);

create table managers(manager\_id int primary key,

manager\_name varchar(20), age int, country varchar(20), club\_id int ,

image varchar(255),

user\_id int,

foreign key(club\_id) references clubs(club\_id) on delete cascade

);

create table players(player\_id int primary key,

player\_name varchar(20), age int,

position varchar(30), country varchar(20), club\_id int, manager\_id int, image varchar(255),

foreign key(club\_id) references clubs(club\_id) on delete cascade,

foreign key(manager\_id) references managers(manager\_id) on delete

cascade,

user\_id int,

);

create table matches(match\_id int primary key,

match\_date date, home\_score int, away\_score int, home\_id int, away\_id int,

user\_id int,

foreign key(home\_id,club\_id) references clubs(club\_id) on delete cascade

);

create table goals(goal\_id int primary key,

goal\_time int, match\_id int, player\_id int,

user\_id int,

foreign key(match\_id) references matches(match\_id),

foreign key(player\_id) references players(player\_id) on delete cascade

);

create table club\_count(club\_id int,

no\_of\_players int default 0

);

**TTriggers:**

delimiter $

create trigger club\_insert after insert on clubs for each row

begin

insert into club\_count values(new.club\_id , 0); end $

create trigger club\_delete before delete on clubs for each row

begin

delete from club\_count where club\_id = old.club\_id; end$

-- insert trigger delimiter $

create trigger player\_insert after insert on players for each row

begin

update club\_count set no\_of\_players = no\_of\_players + 1 where club\_id=new.club\_id; End $

-- update trigger delimiter $

create trigger player\_update before update on players for each row

begin

update club\_count set no\_of\_players = no\_of\_players -1 where club\_id=old.club\_id; update club\_count set no\_of\_players = no\_of\_players +1 where club\_id=new.club\_id; end$

-- delete trigger delimiter $

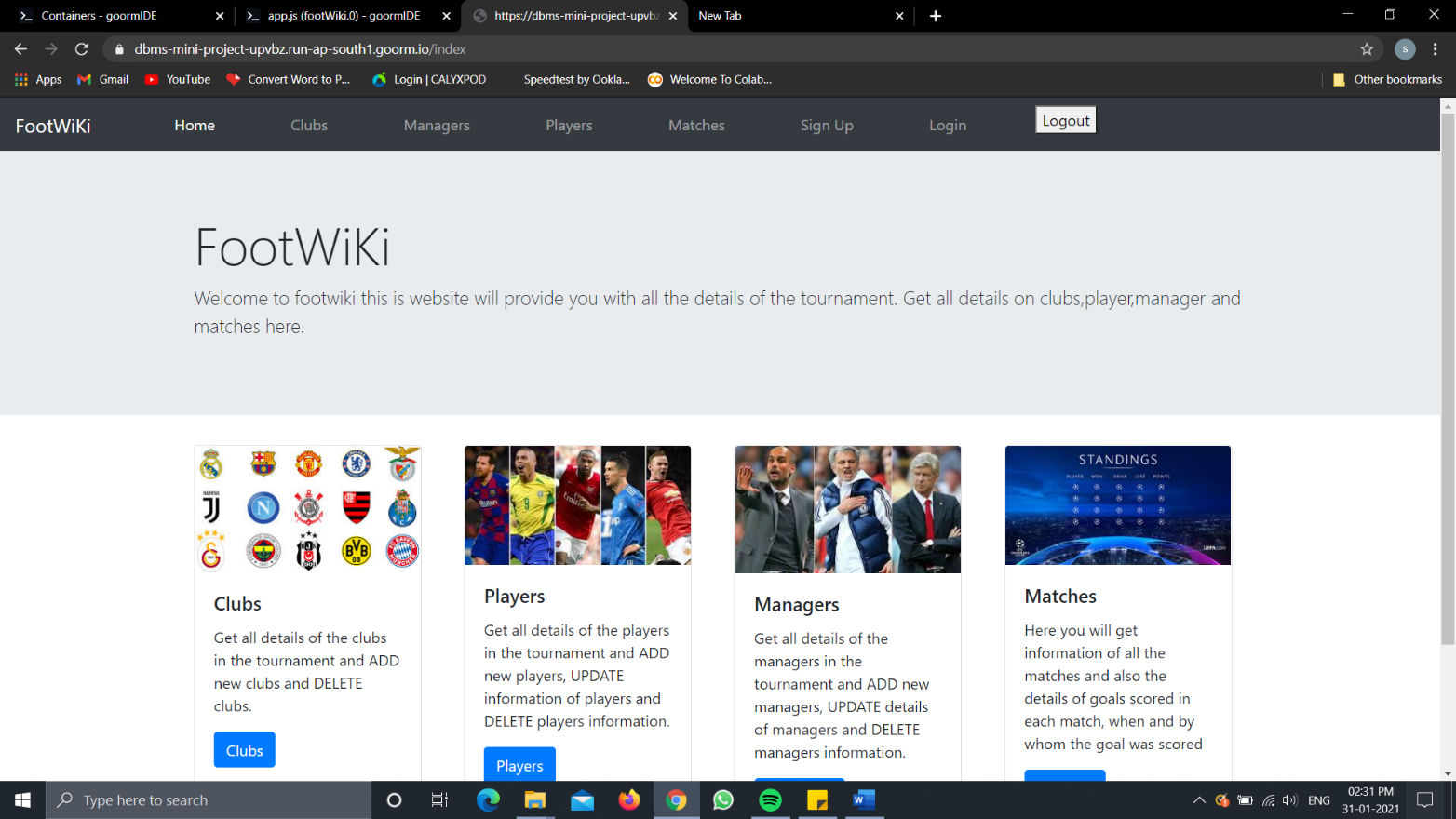
create trigger player\_delete after delete on players for each row

begin

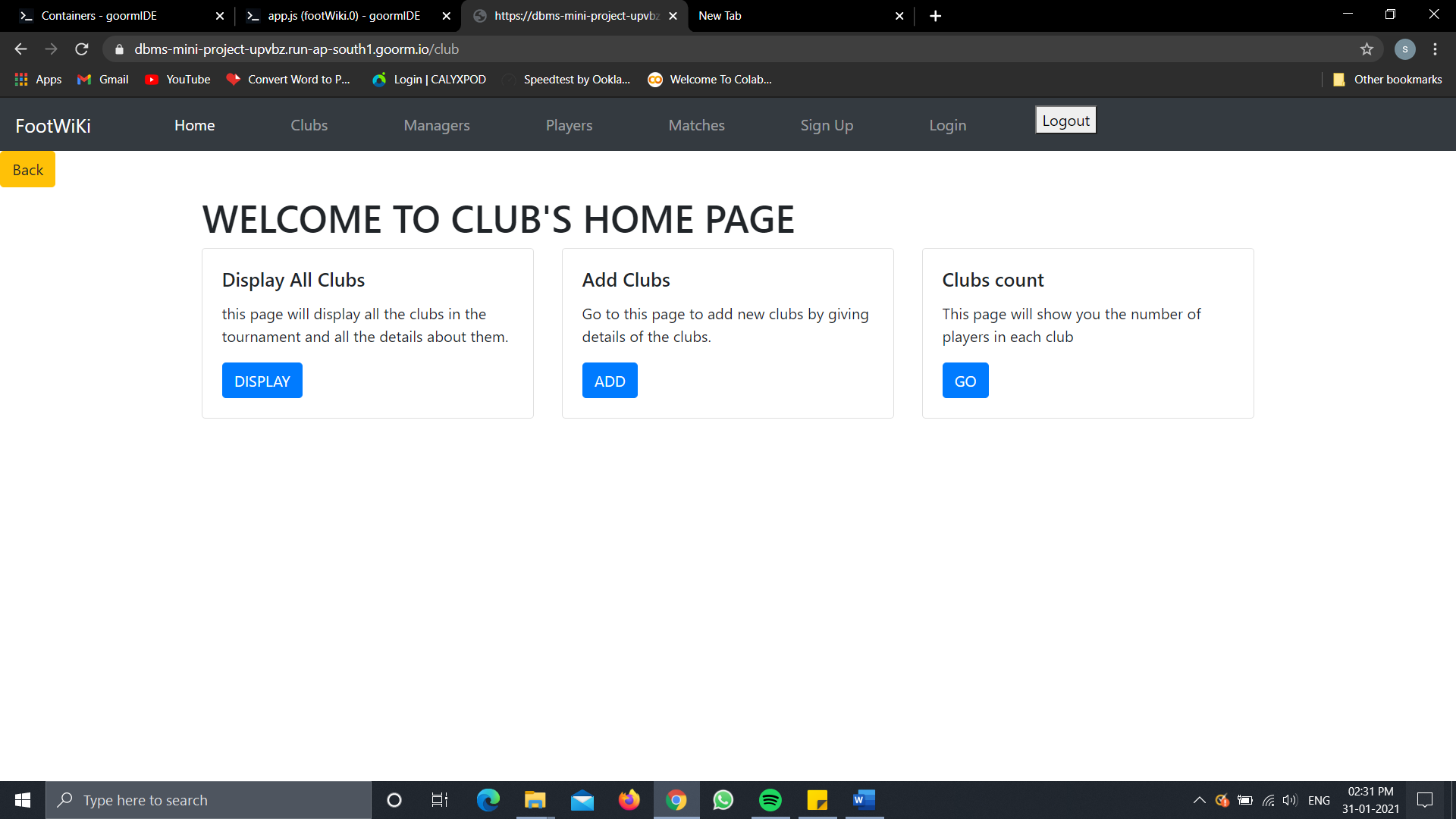
update club\_count set no\_of\_players = no\_of\_players-1 where club\_id = old.club\_id; end $

## S CREENSHOTS 1 .INDEX PAGE

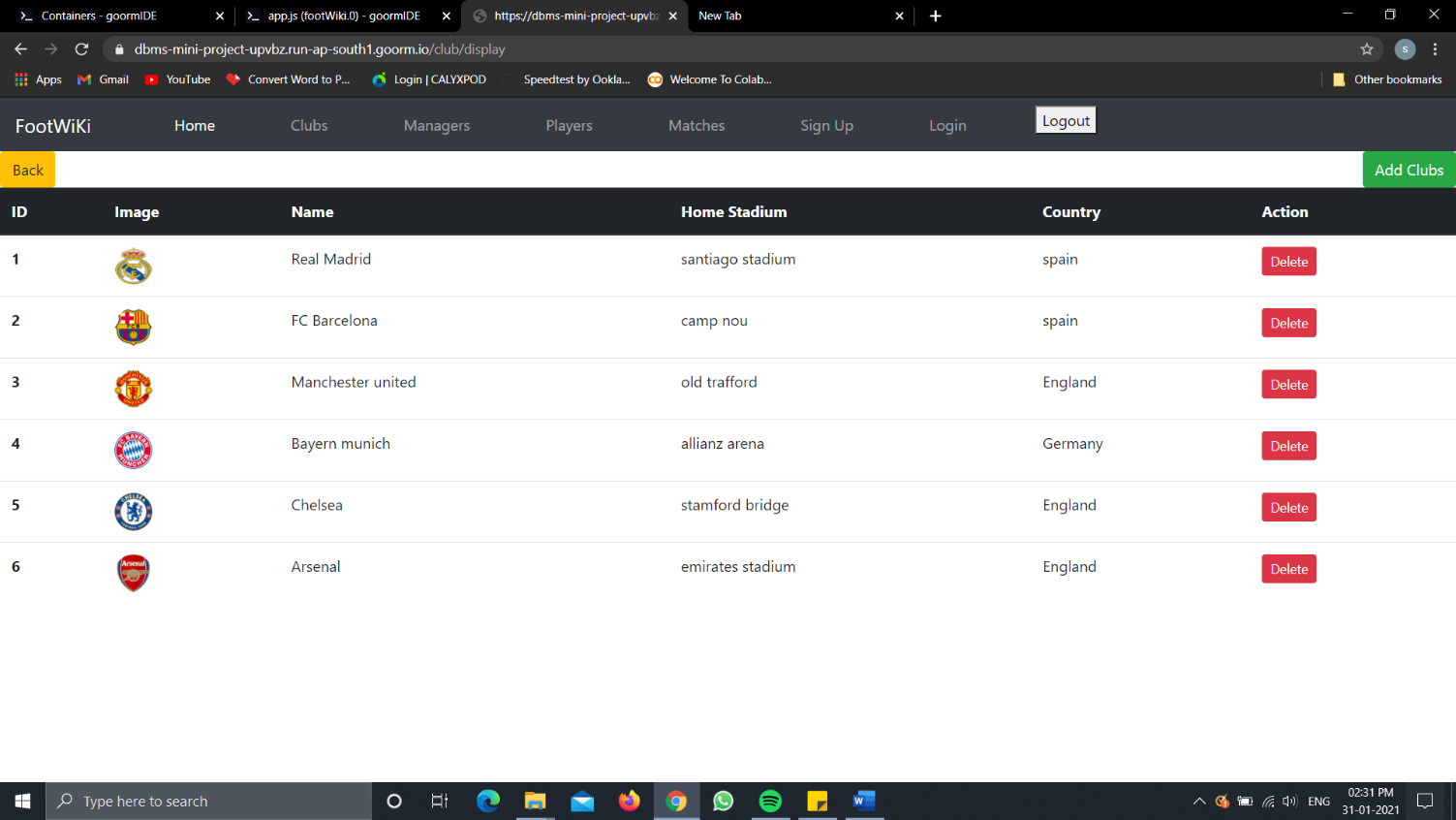
1. **HOME PAGE**



1. **CLUBS DASHBOARD**

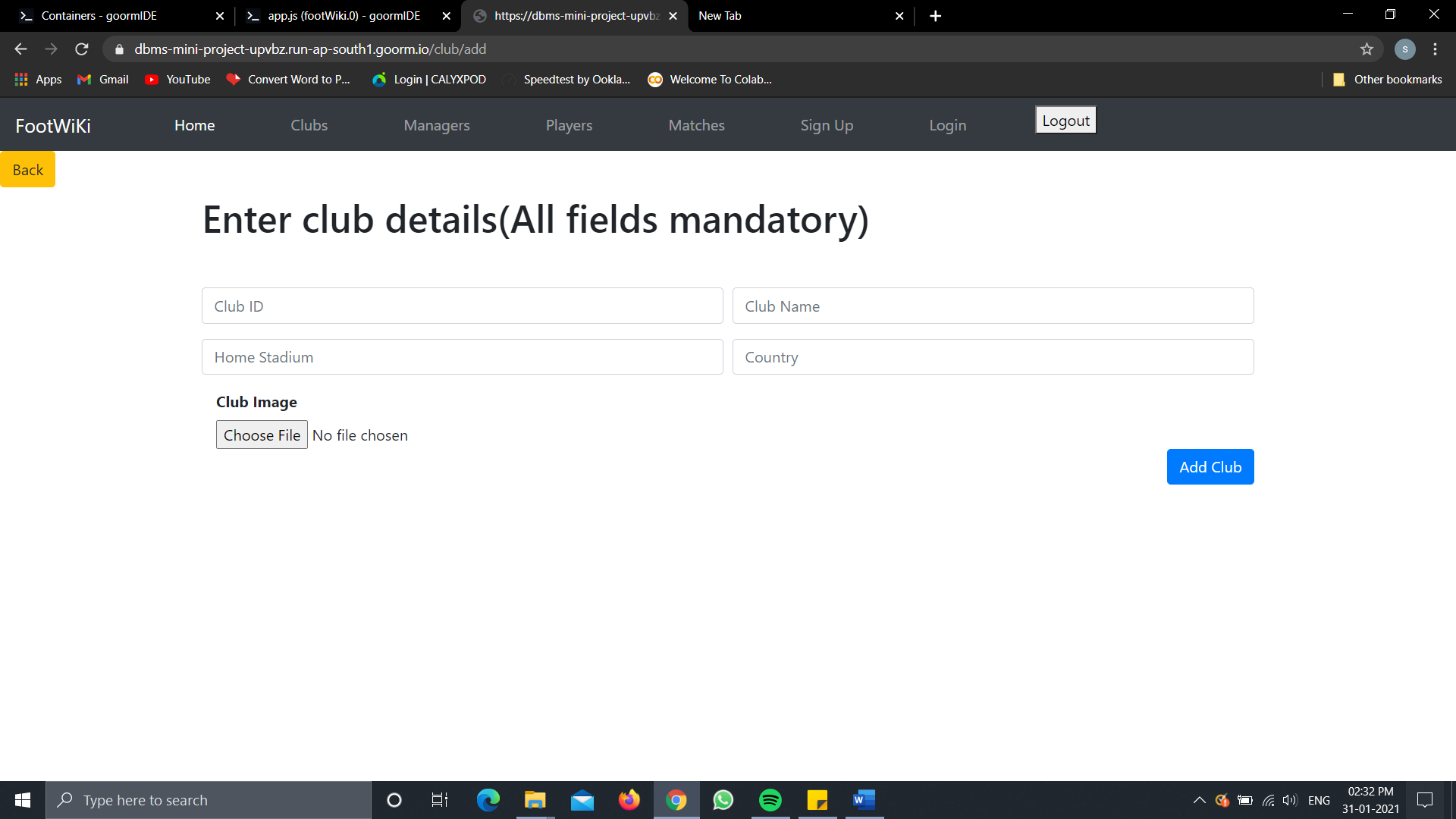


1. **DISPLAYING CLUBS**

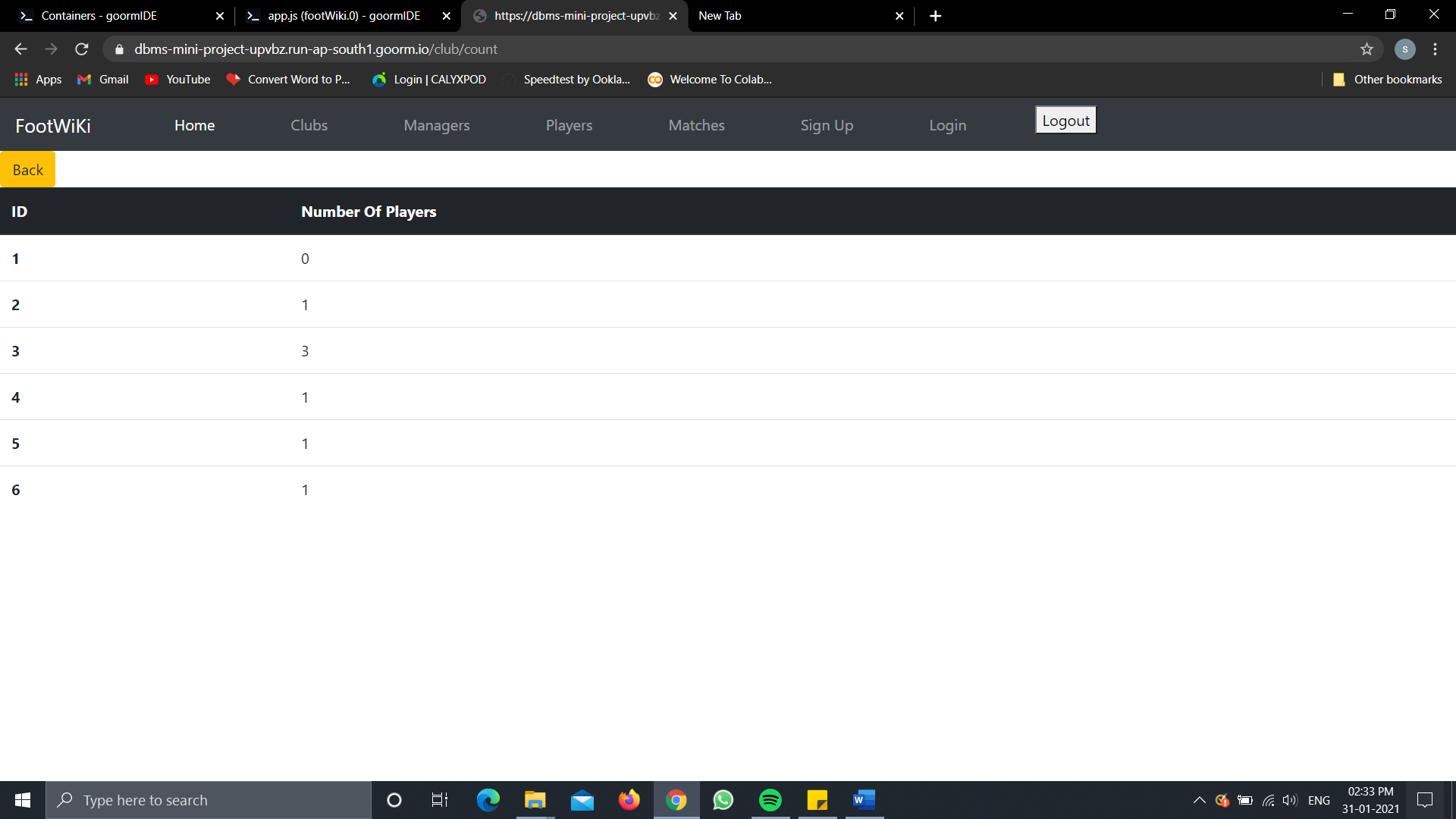


1. **ADD CLUBS (VALIDATING INPUT AS CLUB NAME WILL NOT**

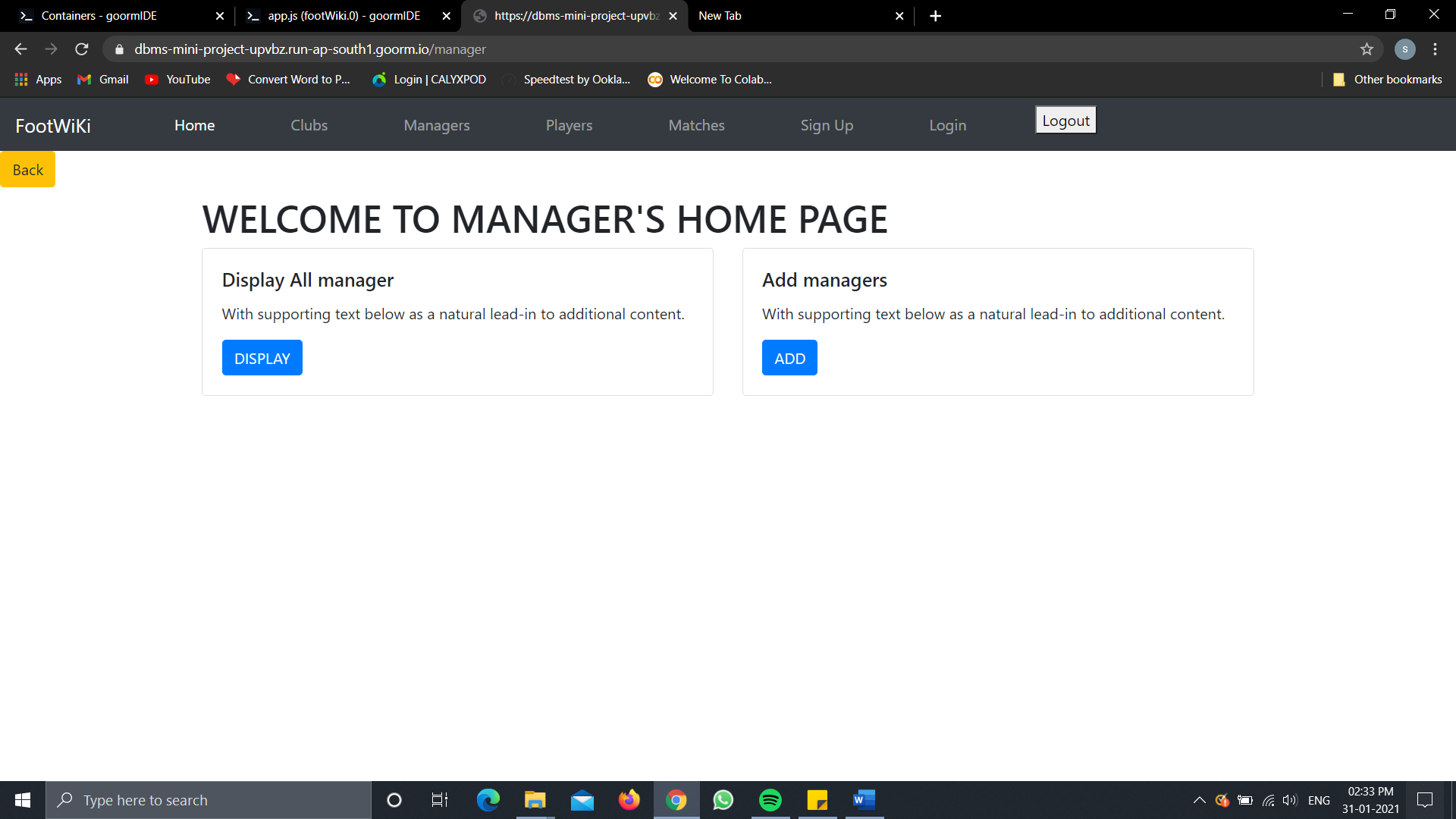
**A CCEPT NUMBERS)**



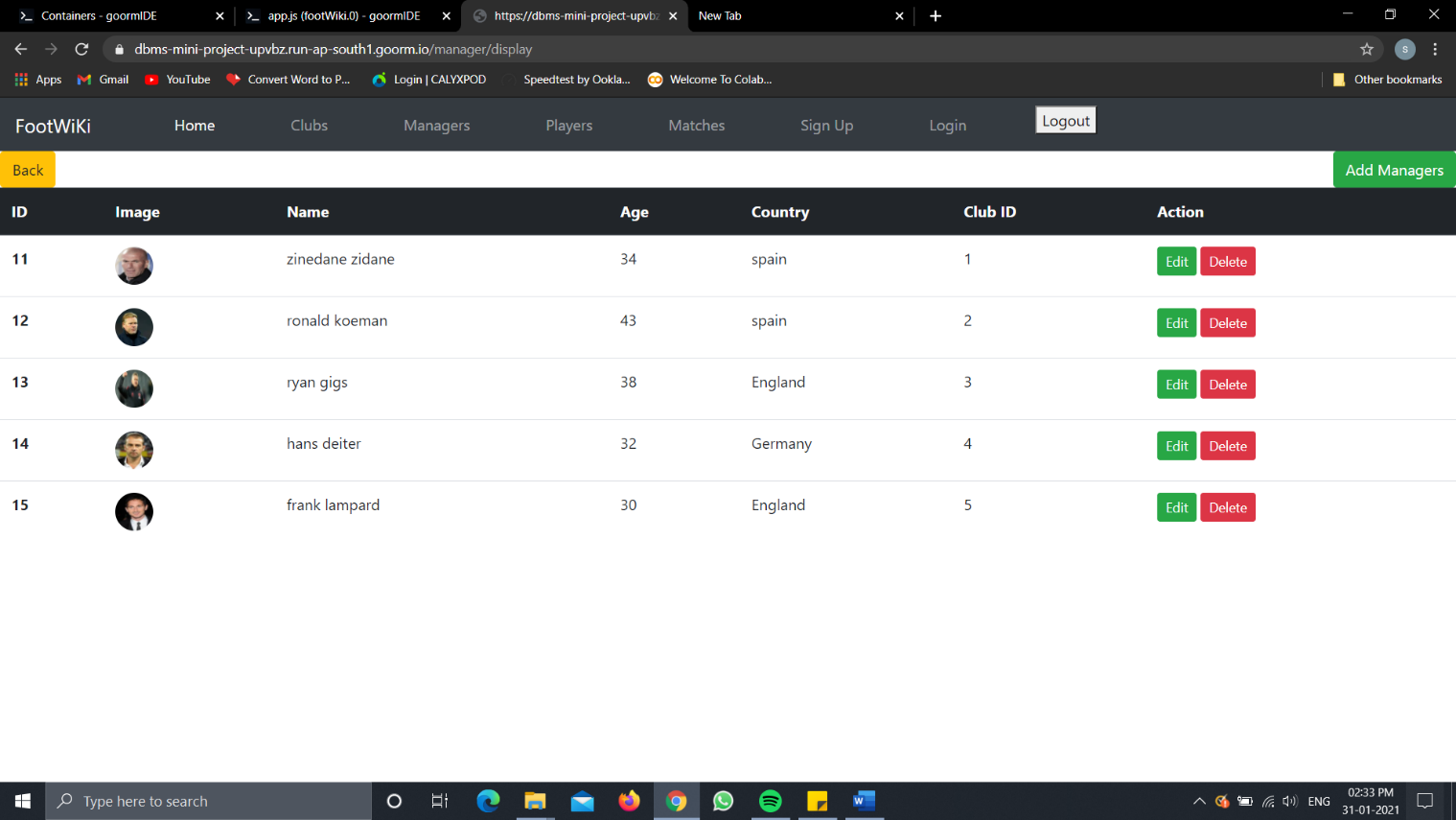
1. **DISPLAYING CLUB COUNT (IMPLEMENTED USING TRIGGERS)**



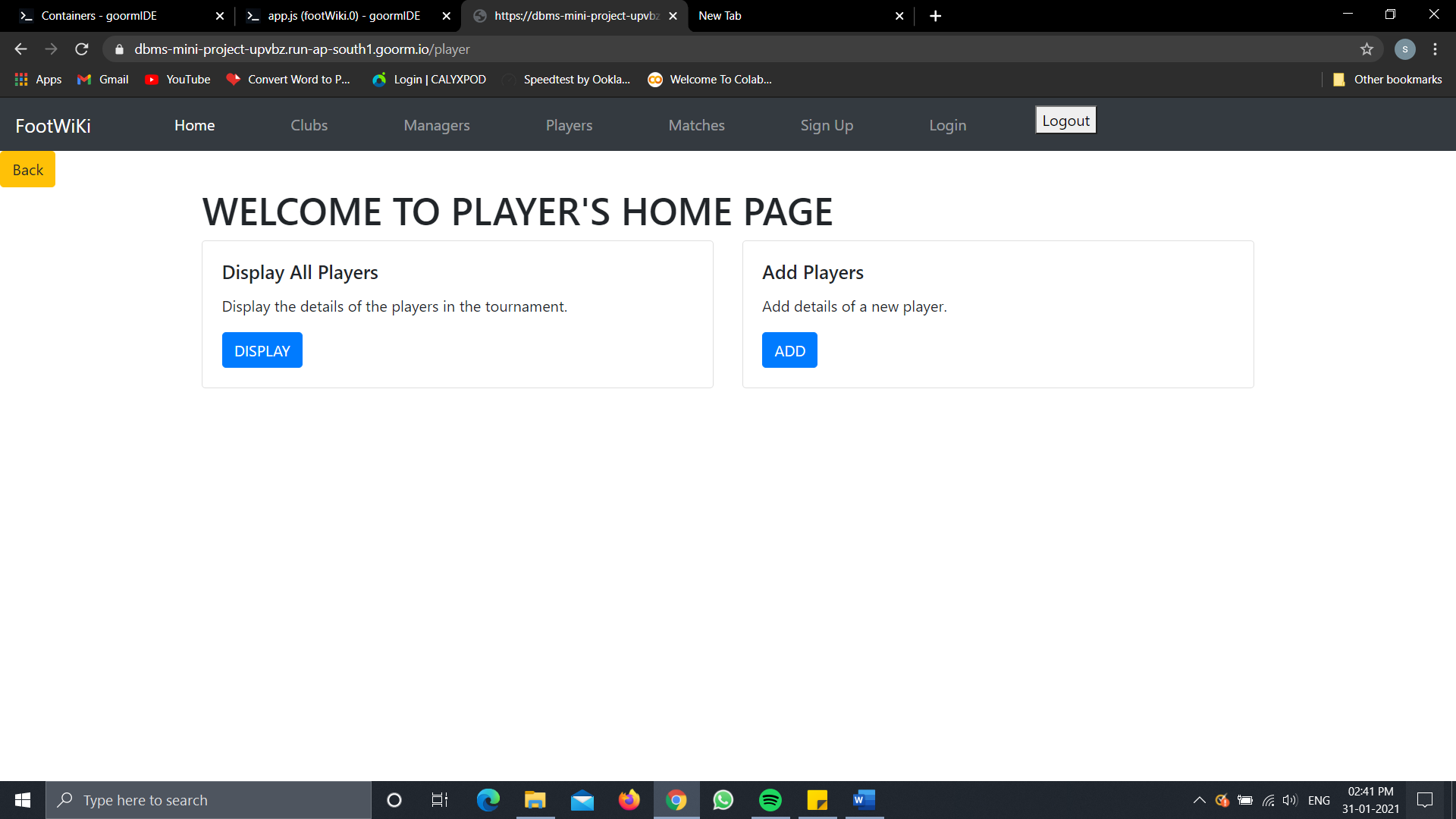
1. **MANAGER DASHBOARD**



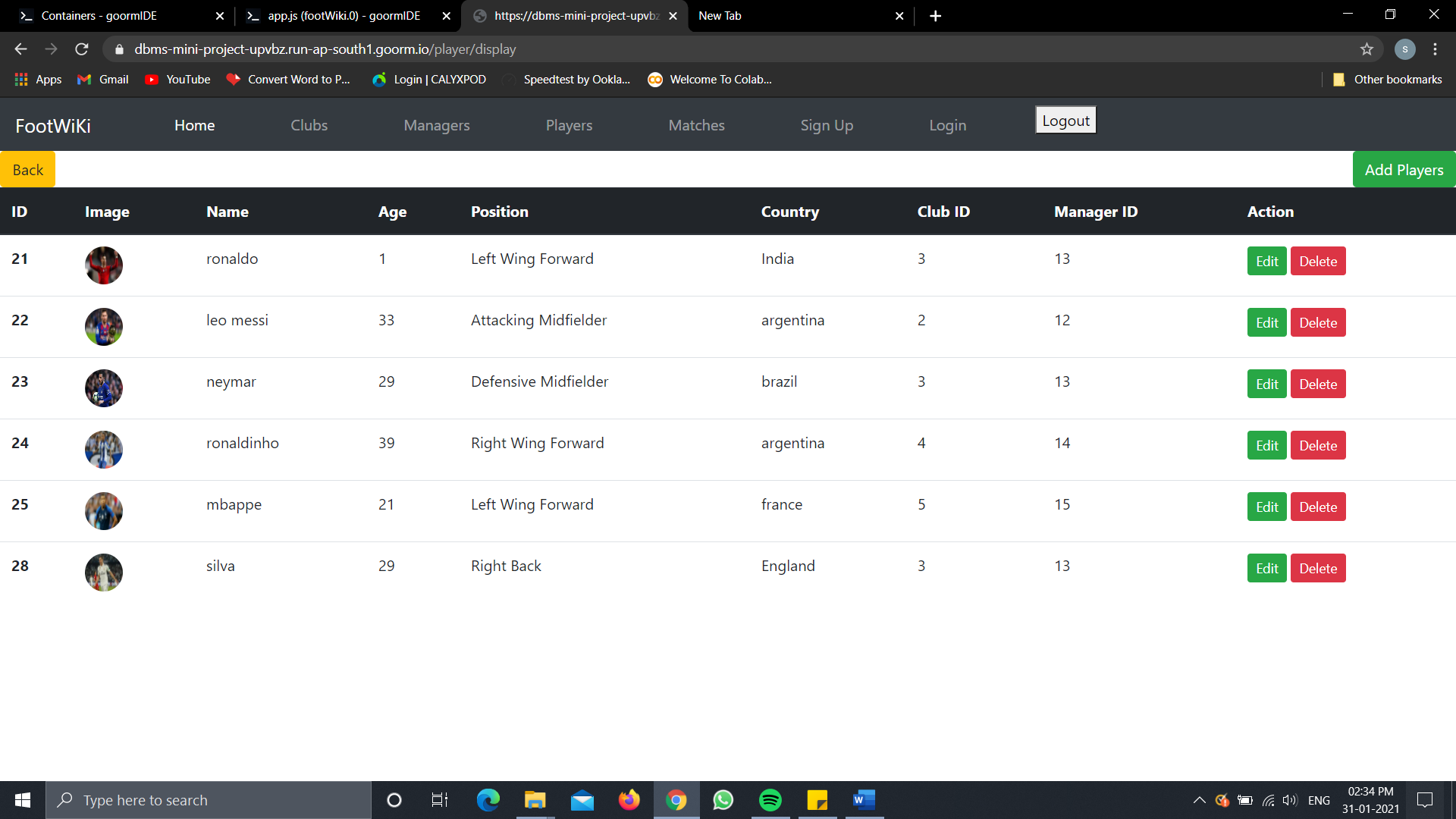
1. **Displaying MANAGER**



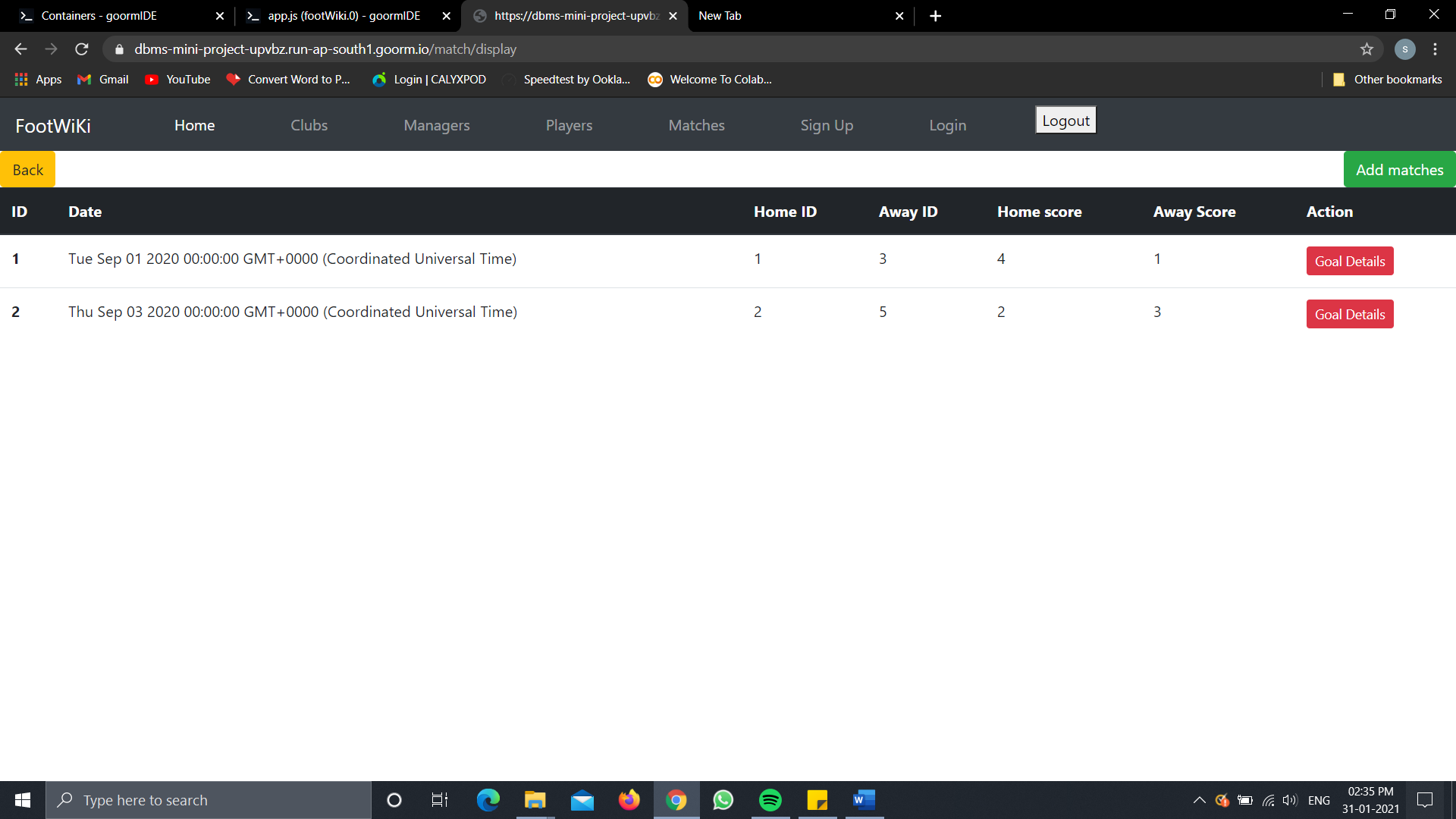
1. **PLAYER DASBOARD**



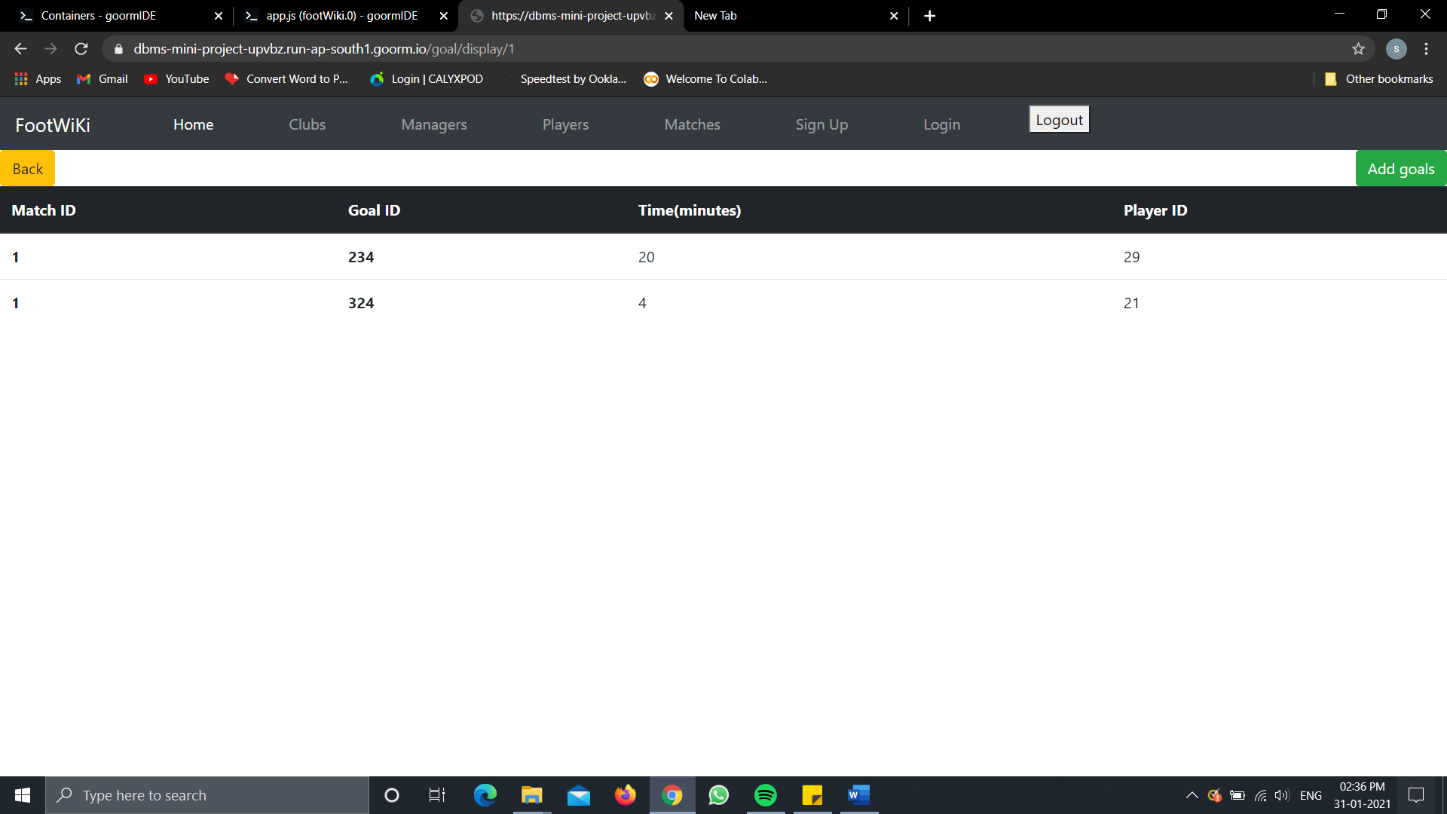
1. **DISPLAYING PLAYERS**

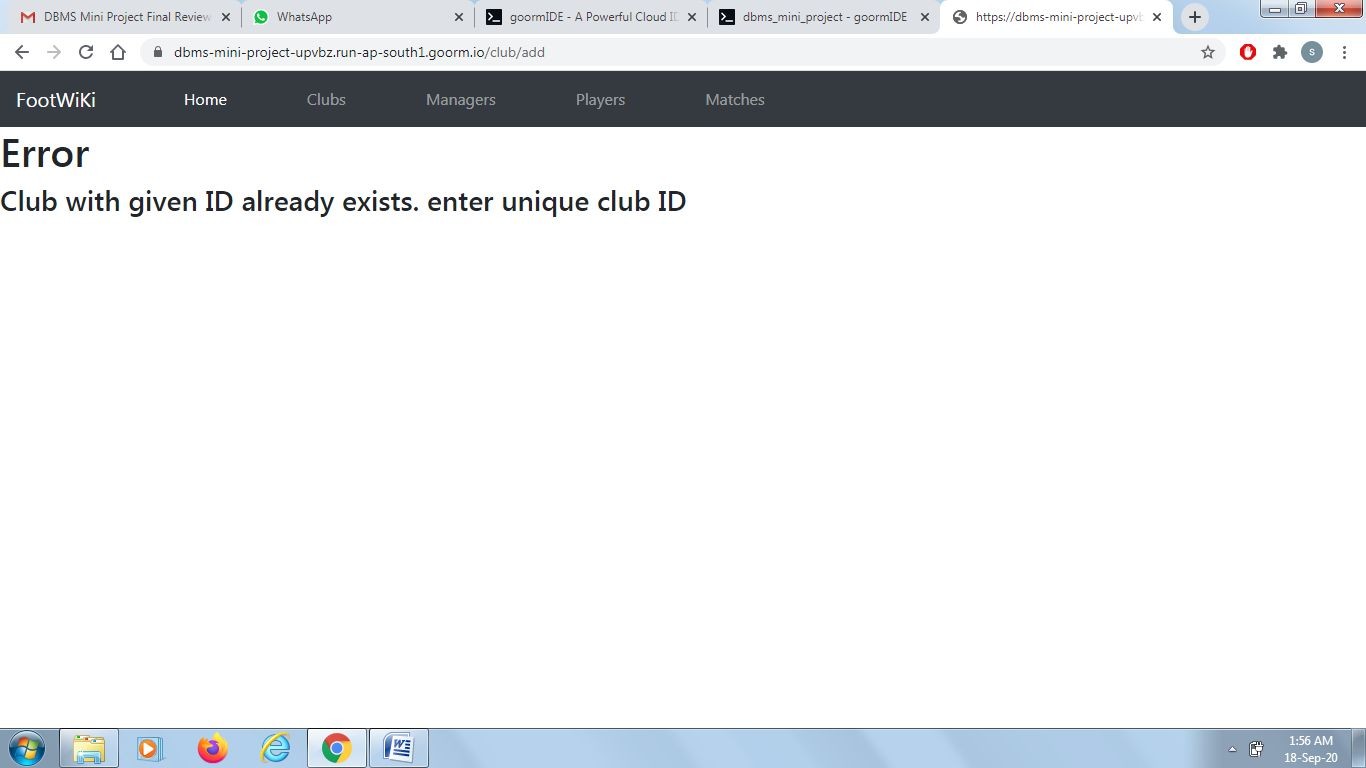
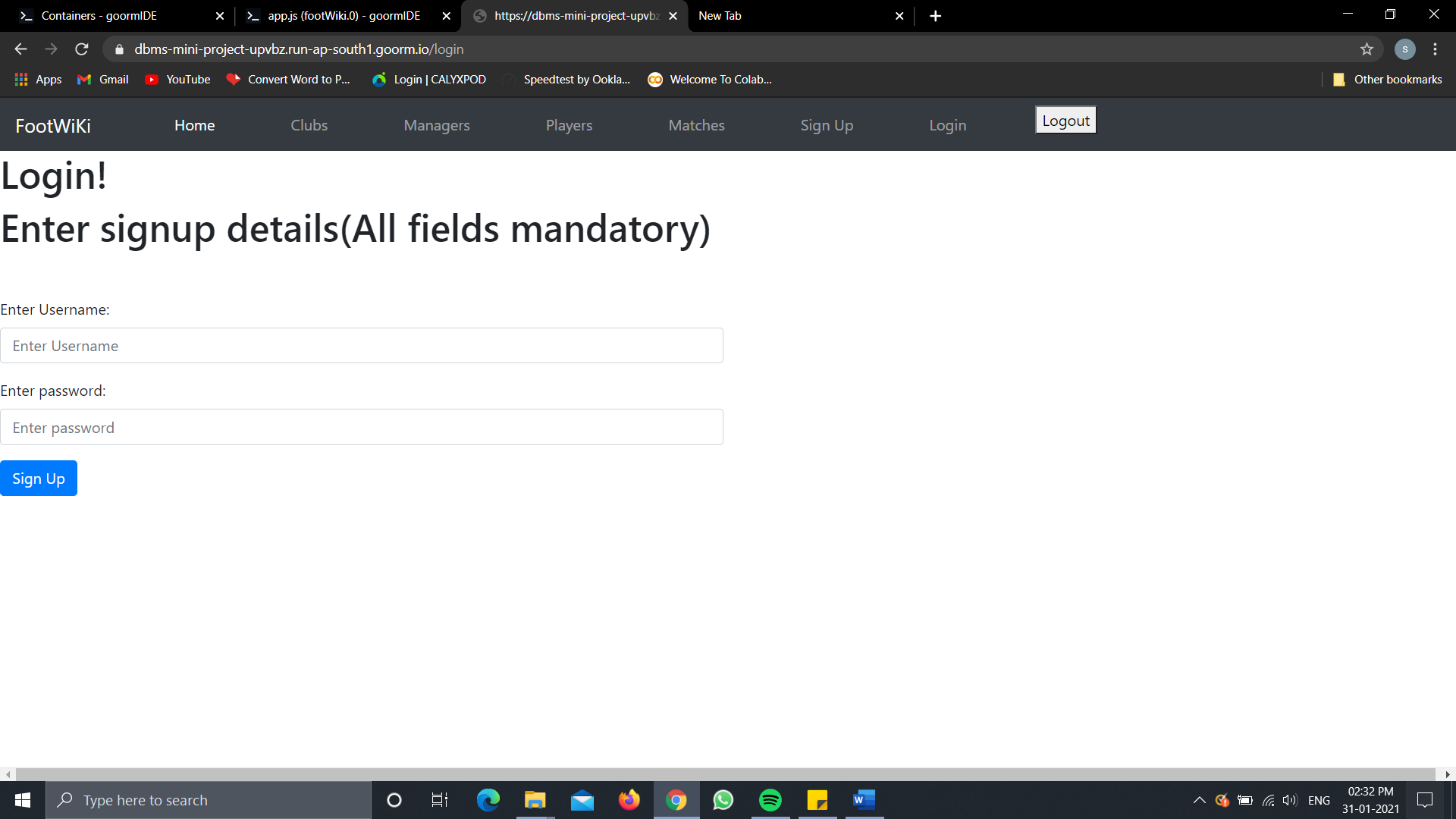


1. **DISPLAYING MATCHES**



1. **DISPLAYING GOALS SCORED IN MATCH**



1. **ERROR PAGE** 
2. **Signup/Register Page**
3. **Login Page**

**C onclusion**

This application provides a computerized version of a football player management system for a football club tournament .

Football player management system is an application which can be used to view , insert or update details of a tournament as it is being played.

It makes the work much easier for the person responsible to store and maintain various data of a tournament or for a user to just view various information about the matches , clubs , managers and players involved in the tournament.

The website has a facility for getting all the details of the participating clubs, adding new clubs and deleting existing clubs.

The website also has a facility displaying details of players and managers , adding new players and managers and deleting existing players and managers.

The user can also add a new match or display the details of matches played.

Every new data entry or updation of an existing entry directly influences the backend of the application instantly.

The website also checks the input given by the user for various errors such as if the datatype of the input matches the datatype of the field.