RoseIM

Final Report

Rose-Hulman Institute of Technology – CSSE333

Jacob Petrisko

Praneet Chakraborty

February 15, 2019

Table of Contents

1. Executive Summary

This document will provide a detailed explanation on the complications with the current management system of Intramural Leagues and how we intend to fix it. The next couple sections will outline the topic, the problem description, and the solution description. Further through, major challenges to the solution will be discussed along with an outline of the database design and the strengths and weaknesses of the design. At the end, an appendix will show the final entity relationship diagram and the relational schema diagram for the database. References, a glossary, and an index are also included for consideration.

2. Introduction

This is the final report of RoseIM, an innovative solution to replace IMLeagues at Rose-Hulman, created by Jacob Petrisko and Praneet Chakraborty. The purpose of this document is to analyze the capability of the implemented solution to the problem outlined in previous documents. The project team will share direct feedback on the effectiveness of this solution, outlining the strengths and weaknesses of the solution with topics from the Security Analysis [1] and Final Problem Statement [2].

3. Problem Description

Intramural Sports are an organized way for people of the same community to be involved in athletic competition with and against each other in a non competitive format [3]. These activities are predominant in colleges and universities around the world. Currently, an outdated and highly complained about monopoly runs 95% of intramural league systems in universities and colleges across the country [4].

IMLeagues has a monopoly on intramural management at schools and colleges around the country. Their website and application are flooded with ads after every click, are slow and unresponsive, and are too complex to navigate for simple tasks. Joining and creating a team and viewing a team's schedule, record, and roster are the essentials for a player in an intramural league. Unfortunately, these tasks confuse most on this current platform. Creating a new sport, game, and league are too difficult for the referees and directors as well.

The goal is to create a simple, ad-free, and streamlined user interface to complete the necessary vital tasks for Rose-Hulman's Intramural Leagues. Access for other colleges and university will be limited at this time.

Listed below are the features included in the working system as described by the RoseIM Final Problem Statement.

Feature #	Feature Name	Feature Description
1	Web-based application	This project will be available on the internet.
2	Create/Join teams	Players can create and join teams in different leagues.
3	Track different teams	Ability to see different teams in different leagues/sports
4	Schedule events at different times/places	Games can be held at various locations at different times.
5	View schedules, teams, rosters, records, locations	Can see user's teams' schedules, rosters, records, locations of games
6	Login with Rose-Hulman email	Login is only allowed through Rose-Hulman email
7	Limit data displayed based on user type	Players can create and join teams, view schedules and rosters, etc. Referees can create and score games

4. Solution Description

The solution designed includes a web-application using PHP and MySQL as the database to manage and display the desired functionality to manage an intramural league management system.

4a. Front-End Discussion

One of the main goals with RoseIM was to create a simple, ad-free, and streamlined user interface such that it is easy for users to access and see various information about intramural leagues and teams. This information includes everything from different leagues within sports, games between teams, team rosters and schedules, etc. In order to show information in a streamlined manner, it was decided that the UI would consist of large buttons with descriptive yet concise names, forms which showed various information depending on the page, and drop downs with different options for users to choose from. The design of the web pages are such that everything is easy to see and there is no clutter to distract users while on the website.

These goals were mostly met with the current developed platform of RoseIM. Due to a lack of experience in website development, the UI is not attractive; however it is functional with all the goals met for what is necessary for an intramural leagues management system.

4b. Back-End Discussion

Through LAMP Stack development, the RoseIM platform is built on a Linux operating system. An Apache web server on the Linux base hosts the website and all users who interact with it. The MySQL database on the Linux base is the database which holds all information pertaining to RoseIM. On top of all of these platforms, PHP acts as the communication between the client-side HTML and the server-side database and web host. PHP connects the HTML to the MySQL database and feeds the website information regarding what the user wants to see. It connects through the standard PHP mysqli library for connecting to a MySQL database.

The RoseIM platform is hosted on the CSSE Department's server on a VM. The MySQL database contains information of all users, sports, leagues, teams, games, etc. Database access is limited through the front-end through the usage of various stored procedures, functions, and views. This maintains the security of the database from SQL Injection attacks while also improving overall performance due to the high efficiency of stored procedures and functions.

5. Key Challenges

Challenge: Little experience with MySQL and PHP - full stack web development
Solution: A lot of help was asked of Dr. Sriram Mohan in order to aid the development of the project. Due to the difficulty of the problem, a lot of time was spent trying to fix basic problems.
Analysis: This approach was fine in the beginning of the project as front-end work was just starting. However, this solution started to fail as the weeks went on and the concepts for website development became more difficult. The amount of time spent fixing bugs exponentially grew as bugs became more complex and this started to impact the quality of the website. Overall, a web-dev class would have been very beneficial prior to starting an ambitious project like this.

Challenge: Only a 2 person group
Solution: There was not a real logical solution to this. The idea was that the work would be fine for the 2 teammates as it would be a good "learning experience" for the group.
Analysis: The workload was not fine for two teammates. This was a very ambitious project for a small group. With very little experience in web development, too much time was spent learning basics and debugging simple issues rather than focusing on making the project the best it could be. Though the project is mostly functional, the lack of experience forced a dip in overall efficiency which compromised features of the project.

Challenge: Limitations of PHP
Solution: There is a lot of PHP documentation in relation to how it works with MySQL. Using these documents were vital.

Analysis: Because of the staggering amount of documentation available online along with various StackOverflow posts with PHP and web development, there were always plenty of answers to problems encountered during development. This still was not enough however due to the fact that PHP has basic limitations which make development in this language more difficult. Usage of another scripting language like JavaScript would have been more applicable and easier to use compared to PHP.

6. Database Design

6a. Security Measures

RoseIM requires special user permissions when logging in. A player can only access the player pages while referees can only access the referee pages. A user cannot access pages of the website if the user is not logged in. If not logged in, then the website automatically redirects the user to the login page to enter their credentials or register a new user. Depending on the login status, a permission variable is added to the current PHP session. All pages check the session permission before allowing access to its contents. This prevents players from altering the scores of games and records of teams. Referees are prevented from joining and creating teams. Every query was executed using prepared statements with parameters passed as stored procedures or stored functions. This eliminates the chance of harmful SQL Injections by using set pre-processed code.

Since PHP sessions and post methods were used, RoseIM is vulnerable to XSS(cross-site scripting) attacks. Knowledge of preventing this was unknown during development.

RoseIM stores passwords for each user in a hashed version of the original password inputted by the user in the front-end. The hashing algorithm is the default PHP hashing algorithm, bcrypt. PHP handles the hashing of plain-text passwords and also checking the hash with the password inputs from login. This method remains future proof as the PHP default algorithm advances and becomes more security as PHP is updated.

6b. Integrity Constraints

On remove, all operations will restrict. We chose to restrict Removes because we wanted to make sure that all deletions were for the right reasons and from the right places (tables). For example, deleting an item in the Person table when the same key exists in Player should not allow a remove. This is because we want a Person to have the ability to become a Player even if they are not on any teams. If a Person has been removed, this means that the Person has likely deactivated their account or they are no longer a student at Rose-Hulman. Foreign key constraints are enforced such that the foreign key takes precedence.

On update, all operations will restrict. The reasons are the same as the reasons for remove. Updates will restrict so that updates will be made in the right areas.

a. For the Person table, each person is uniquely identified by a person_ID attribute which is a non null unique integer and is automatically created every time a new “Person” is created. Each person also has a first and last name which are not null and 20 and 25 length varchars respectively. Email is also a 50 length varchar which cannot be null and must be unique for each person. Password is a 100 length varchar which cannot be null. It is stored as a hashed version of the original password that the user inputs on the front-end so that passwords cannot be hacked easily. Lastly, each person has a sex which is a 6 length varchar and must either be “Male” or “Female”.
b. For the Player table, person_ID is a foreign key to the Person table and is also the primary key.

c. For the Referee table, person_ID is a foreign key referencing the Person table and is the primary key.
d. For the Sport table, name is a 25 length varchar which cannot be null and is the primary key. Rules is a 5000 length varchar which explains the basic rules of the game.
e. For the League table, league_ID is a non null automatically incrementing integer which is the primary key. Location is a 100 length varchar which cannot be null. A league also has a sport which is a foreign key referencing the name field of Sport.
f. For the Game table, game_ID is an automatically incrementing integer which is the primary key. Ref is a non null integer which is a foreign key referencing Referee. Facility is a non null integer which is a foreign key referencing Facility. Finally, league is a non null integer which is a foreign key referencing League. StartDateTime is a datetime field which stores the date and time for when a game will be played.
g. For the Team table, name is a non null 50 length varchar. Team_ID is a non null automatically incrementing integer which serves as the primary key. League is an integer which references league_ID in the League table. Teams also have wins and losses which are both set to 0 by default on creation of a team.
h. For the PlaysOn table, role is a 30 length non null varchar. Player is a non null integer referencing person_ID in Player. Team is a non null integer referencing team_ID in Team.
i. For the Plays table, home_Score and away_Score are integers which must be greater than or equal to 0 or null if the game hasn't been played yet. Team1 and Team2 are non null integers which are foreign keys referencing two different teams in Team. Team1 is

considered to be the home team while Team2 is considered to be the away team by default. Game is a non null integer which references game_ID in the Game table.

6c. Stored Procedures and Functions

Stored Procedure or Function Name	Procedure Purpose
Get_Games_Ref	Given the email address of a referee, this returns the teams playing, the game location, the game time, and the score of the game for reference by the referee.
Get_Permission	This returns the specific permission given a user's email address. This permission is used when navigating the website.
Get_Roster	This return the roster of a specific team, listing the player's names and their roles on the team.
Get_Schedule	Returns the schedule of a given team, showing who plays who, the location, start date/time, and the score.
Get_Teams	Given a user login email address, this shows all associated teams with their sport and league.
League_Standings	Returns team names ordered by win percentage given a specific league. Standings of the league.
Create_Facility, Update_Facility, Remove_Facility	These stored functions allow for the creation, update, and deletion of a facility.
Create_Game, Update_Game, Remove_Game	These stored functions allow for the creation, update, and deletion of a game.
Create_League, Update_League, Remove_League	These stored functions allow for the creation, update, and deletion of a league.
Create_Person, Update_Person, Remove_Person	These stored functions allow for the creation, update, and deletion of a person.
Create_Player, Update_Player, Remove_Player	These stored functions allow for the creation, update, and deletion of a player.
Create_Plays, Update_Plays, Remove_Plays	These stored functions allow for the creation, update, and deletion of a plays relation between two teams.
Create_PlaysOn, Update_PlaysOn, Remove_PlaysOn	These stored functions allow for the creation, update, and deletion of a relation where a player plays on a team..
Create_Referee, Update_Referee, Remove_Referee	These stored functions allow for the creation, update, and deletion

	of a referee.
Create_Sport, Update_Sport, Remove_Sport	These stored functions allow for the creation, update, and deletion of a sport.
Create_Team, Update_Team, Remove_Team	These stored functions allow for the creation, update, and deletion of a team.

6d. Views

View Name	View Purpose
My_Teams	Lists the teams associated by email address.
Player_Person	A join between the Player table and the Person table. This gives all the attributes of a player.
Ref_Person	A join between the Referee table and the Person table. This gives all the attributes of a referee.
Team_Win_Percentage	Displays the win percentage of all teams based on their win loss record. This is used for determining the standings of teams within a league, ordered by their win percentage.

6e. Triggers

Trigger Name	Trigger Purpose
Update_Record	After update on the Plays table, (scoring a game between two teams), the records of the teams involved are automatically update to reflect a win/loss.

7. Design Analysis

7a. Strengths

- The backend database is completely functional. Any additions to front-end functionality would require little to no provisions in the database.
- Appropriate security measures are maintained throughout the project such as password hashing, login redirecting, and avoidance of SQL Injection attacks through stored procedures/functions and prepared statements.
- Appropriate integrity constraints are in place and maintained.
- Code is clean and relatively easy to understand. Proper naming conventions were used.

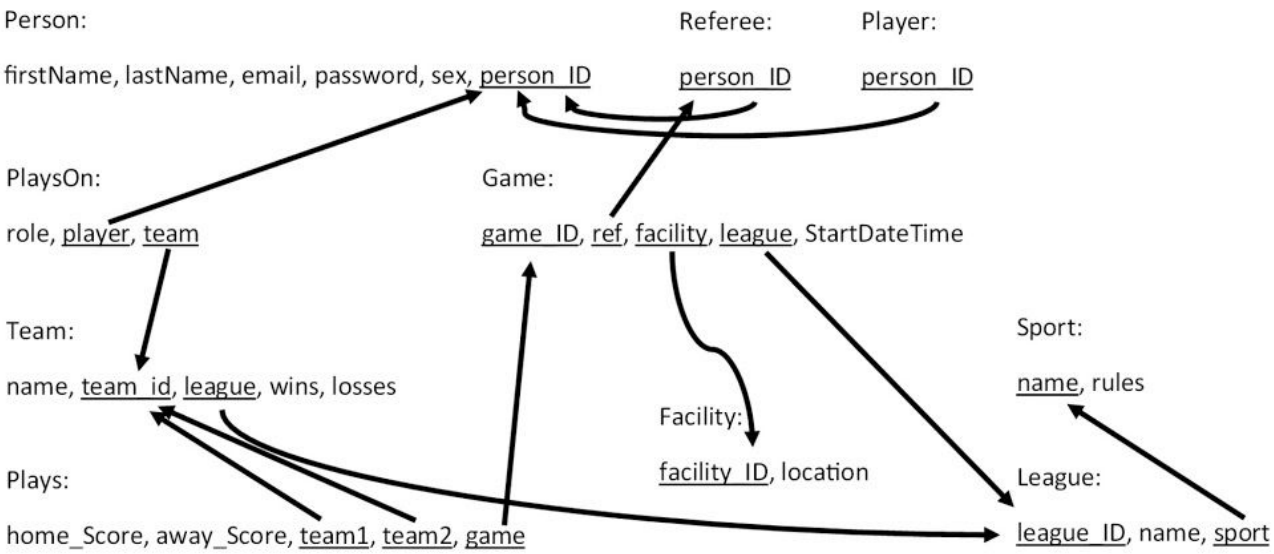
7b. Weaknesses

- Creating a game between two teams requires creating a game in a league, then creating a plays relation between the two teams and the game. The optimal solution would include a function that first creates the instance of the game, and then creates the relation between the two teams and the game.
- Functionality of the front-end does not match the functionality of the database. Not all designed functions were implemented. These extra functions take up space.
- The user interface still contains random bugs. Once in a while, login randomly stops working. Repeatedly refreshing and navigating to previous and after pages disrupts session flow, manipulating the success of functionality.
- Dummy data is handmade and weak, causing bugs to appear when accessing a duplicated version of RoseIM. Duplicating script for RoseIM requires search and replace, an efficient solution. The duplication database uses regular account connections rather than assigning new permissions.

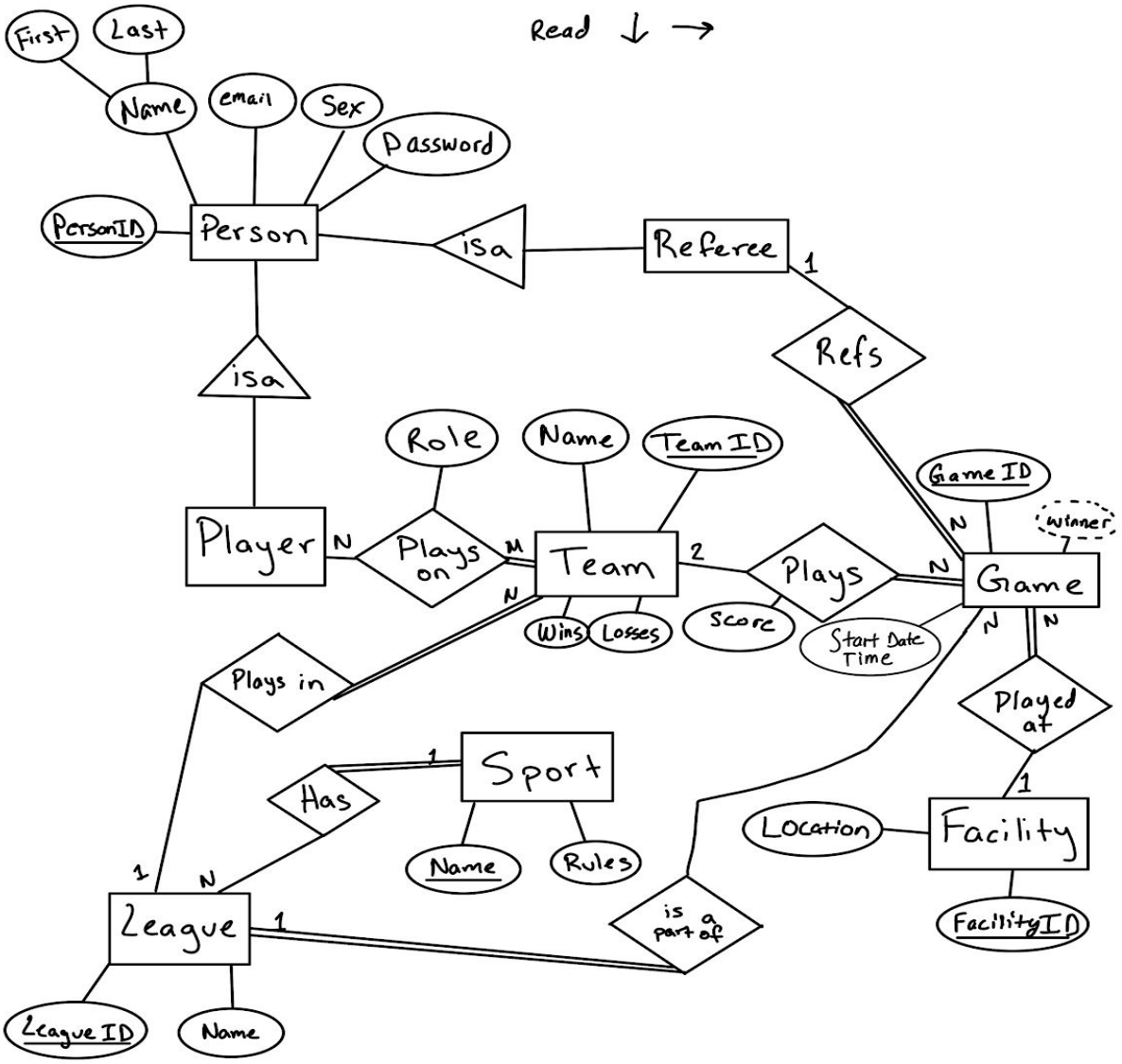
- Many queries on Person were done using the email field, placing an index on this field would have boosted overall performance since this is not the primary key.

8. Appendix

8a. Relational Schema



8b. Entity Relation (ER) Diagram



8c. Explanation of Entity Relationship Diagram

A person is the super identity of player and referee. This means a player's and referee's attributes are held in the person table given the "isa" relationship. Players play on specific teams

with a role such as “Captain” or “Player” while referees referee specific games. Sports sit at the highest level. All sports have leagues while all games are part of specific leagues. Teams play in leagues, tracking wins and losses. Games are set at a start date and time and are played at a facility.

9. Glossary

[5]

Entity relationship (ER) diagram - an abstract way of representing the layout of a database.

Intramural League (IM) - an organized way for people of the same community to be involved in athletic competition with and against each other.

Integrity Constraints - Limitations bound to a database, preserving specific relationships among data and permitting specific values to exist.

Query - a request for data from table(s) within a database.

SQL - a programming language used for managing data within a relational database management system.

MySQL - an Oracle-backed open source relational database management system based on SQL.

Stored procedures - a set of SQL statements with an assigned name, stored in a relational database management system as a group to be reused.

Trigger - procedural code used to automatically execute SQL statements on a table or view given special conditions in a database.

Views - a virtual table derived by data from one or more table columns in a SQL database.

Stored functions - a set of SQL statements that accepts only input parameters, perform actions and return result on a relational database.

PHP - Hypertext Preprocessor, a scripting language primarily used to connect a website to a database.

CSS - Cascading Style Sheet, a client-side stylesheet used to add colors, effects, and design to HTML code.

HTML - HyperText Markup Language, a client-side language which tells a web browser how to display the text, images, and other forms of multimedia on a webpage.

APACHE - an open source web server creation, deployment and management software.

Linux - Operating system on server which hosted the back-end and front-end.

SQL Injection Attack - Malicious input from front-end which can manipulate database back-end in an unintended manner

Prepare Statements - Partial SQL queries from PHP which take inputs after the query has been sent, used to stop SQL Injection attacks.

PHP Session - a simple way to store data for individual users against a unique session ID. Used to keep states of data between web pages.
LAMP - Linux, Apache, MySQL, PHP, an open source web development platform that uses a Linux operating system, an Apache server, a MySQL database, and a PHP backend to create a website.
Virtual Machine (VM) - a computer file known as an image, that behaves like an actual computer. Run a computer virtually within a computer.
Full Stack Web Development - the creation and management of a database, backend, and front end.
MySQLi - A relational database driver used in PHP to provide an interface with MySQL databases.
Dr. Sriram Mohan - A professor in the CSSE Department at Rose-Hulman Institute of Technology, for more information please visit https://www.rose-hulman.edu/~mohan/

10. References

[1] Team RoseIM documentation. Security and Data Integrity Analysis
[2] Team RoseIM documentation. Final Problem Statement.
[3] "What Are Intramural Sports??" *NOVA Athletics*, 2018, www.novaathletics.com/athletics/intramural_files/whyintramurals.
[4] "IMLeagues." *IMLeagues*, www.imleagues.com/.
[5] https://www.wikipedia.org/

11. Index