TournaSmash

Super Smash Bros. Tournament Manager

Jason Nguyen   Martin Funmaker

Michael Lopez

January 19, 2022

Abstract

Organizing tournaments can prove to be challenging for the individuals who are responsible for the events and tasks that must be completed. From assigning players to their matches to updating brackets, their lead role tends to be difficult to manage. With the current systems available, they are limited in capabilities, so our goal is to simplify the process for the organizer while increasing player engagement such as by providing statistics and ranking reports.

In this proposal, we outline a system that will allow players to pre-register for events, check-in individually, and input match outcomes which would update the bracket to reflect real-time progress. Organizers will be able to create tournaments with minimal effort and provide a list of players without having to manually add each participant one by one.

This project will deliver a system that is easy to learn and pick up quickly. The web interface will be the central system while the kiosk and Discord bot will increase functionality and ease of access to allow for more time to play and decreasing the time for administrative tasks.

Department of Computer Science and Engineering

University of Arkansas – Fort Smith

Contents

1 Introduction

1.1 Problem Statement

Super Smash Bros. is a popular Nintendo fighting game which boasts a large player base involving tournament organizers and players from all over the world. Often players for these tournaments do not have the ability to conveniently manage and track their participation when competing in a tournament. The organizers must do all the tasks of entering the participants in the bracket and updating the results of each match. This can become tedious and relies on an individual to go to the projecting computer to make the necessary tournament updates. Participants also do not get notified when their next match is ready to begin, leading to unnecessary waiting and constant checking of the bracket to be updated by the organizer.

These tournaments are highly competitive and as such many of the players take them seriously but maybe they do not know what areas of the game they can improve on. Perhaps they want to know what their win-loss percentage is, or if they perform better as a specific character, or where they rank on average with other players in the tournament. As it stands now, this is data the players would have to keep track of themselves when it is something that could easily be tracked for them in an event system. In addition to these things which are already being tracked on certain websites, there is not an "all-in-one" inclusive package on any site which tracks things like Elo, or ranking, and season stats, if the site even offers a season tracker.

Participating in a tournament such as this should be an enjoyable experience for all in attendance. Having to rely on an individual to manually update and oversee these types of events not only leads to the potential of human error but also does not allow for everyone to enjoy the event in a smoother fashion. It becomes a tedious task to oversee as both participants and event organizers are having to see each other and communicate processes such as the match outcomes, tournament placement, future matches, and player statistics.

1.2 Objective

The final objective of this project is to improve the current flow of the tournament system and create a more enjoyable and easier to organize tournament event. By automating brackets, tracking player statistics, allowing self-check-in, and minimizing tournament organizers' work to be done, we intend to help streamline these events to run in a much smoother manner. Keeping this system's primary goal in mind, we intend to create a stress-free environment for both players and event organizers.

2 Background

2.1 Overview

Key concepts to the project design include a web interface, authentication and authorization, a Discord bot, and kiosk system for users to interact with the system effortlessly. The Challonge API will be utilized to track ongoing tournaments and by integrating the Discord API and a custom-made Discord bot as well, tournament flow will be able to improve immensely.

2.2 Technology

Node.js Node.js is an open-source server environment that uses JavaScript. Node.js uses asynchronous programming which eliminates the waiting and simply continues with the next request to the server. HTTP is a first-class citizen in Node.js, designed with streaming and low latency in mind. This makes Node.js well suited for the foundation of a web library or framework. This project will use Node.js to create a web server and utilize web frameworks such as Express and Bootstrap. [1] [2]

Express Express is the most widely used Node.js web framework and is the underlying library for many other popular Node web frameworks. Express provides mechanisms to write handlers for requests with different HTTP verbs at different URL paths (routes), integrate with ”view” rendering engines in order to generate responses by inserting data into templates, set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response, and add additional request processing ”middleware” at any point within the request handling pipeline. [3]

Bootstrap Bootstrap is a very popular framework for building fast and responsive sites. Bootstrap is compatible with our Node.js foundation and will allow us to create a consistent and visually appealing web design. This project will use Bootstrap for styling HTML elements and page layouts to fit our expected look and feel to the web interface. [4]

MariaDB MariaDB is an open-source relational database management system (RDBMS) forked from the MySQL RDBMS. Due to its development by some of the original developers of MySQL, it maintains high-compatibility with MySQL which allows it to support drop-in replacement capability. MariaDB has been chosen over MongoDB (non-relational database) because our database schema seems to be rigid and not requiring changes in the future. We intend for MariaDB to interact with our system by sending SQL calls through Node.js. [5]

Discord API Discord is a proprietary freeware VoIP communication application, allowing for both text and voice communication. Discord is supportive of community members taking their API and creating whatever they can imagine with it, from bots to applications to games, they love to see their users take full advantage of all the possibilities. There is a FSM Smash at UAFS Discord server that players join to chat and share all things involving the Smash community and gaming. [6]

Challonge API The Challonge API expands tournament creation and control to the programmatic level. You can create tournaments on the fly and report scores directly from your application. This allows you to define score reporting permissions that fit your user model, and provide a more seamless tournament experience for your users. [7] (The Challonge API will be integrated into our web interface and Discord bot to be able to implement mobile tournament check-in and live notifications and activity postings.)

2.3 Related Work

Challonge, Smash.gg Challonge and Smash.gg are both websites used to organize tournaments and ESports events. Challonge however, does not have a feature to find tournaments that are near your area or any seasons tracking feature. On Smash.gg there is no Elo rating for its users. Our application will be able to find local tournaments by searching a user's location, organize tournaments by seasons, and provide an Elo rating affected by all of their matches. This rating will also be tracked to show them any changes based on the outcome from their latest matches.

TournaBot, SSBToni TournaBot and SSBToni are Discord bots that integrate Smash.gg to improve user quality of life through specialized commands and display useful information. Tournabot has several features such as tournament reminders, announcements, and a search by game. SSBToni is mainly focused towards the Super Smash Bros. community by providing a player lookup from Smash.gg and character move details via Ultimate Frame Data. This project aims to fill the gaps with our own custom bot which allows users to pre-register for an event, check-in, and drop out of a tournament.

3 Design

3.1 Project Requirements

Users: Create account, sign up/check in for tournaments, disqualify self from match/tournament, view stats (tournament placings, match history, Elo), reset password, find tournaments in their area, use Discord bot to check-in and see relevant activity posting for tournament, use kiosk to check-in and input match outcomes

Admins: Add/Edit/Remove tournaments, Add/Edit/Remove users, Add/Edit/Remove seasons, enter notes about a player that is only visible to admins.

Database: Store player, tournament, match, and season data.

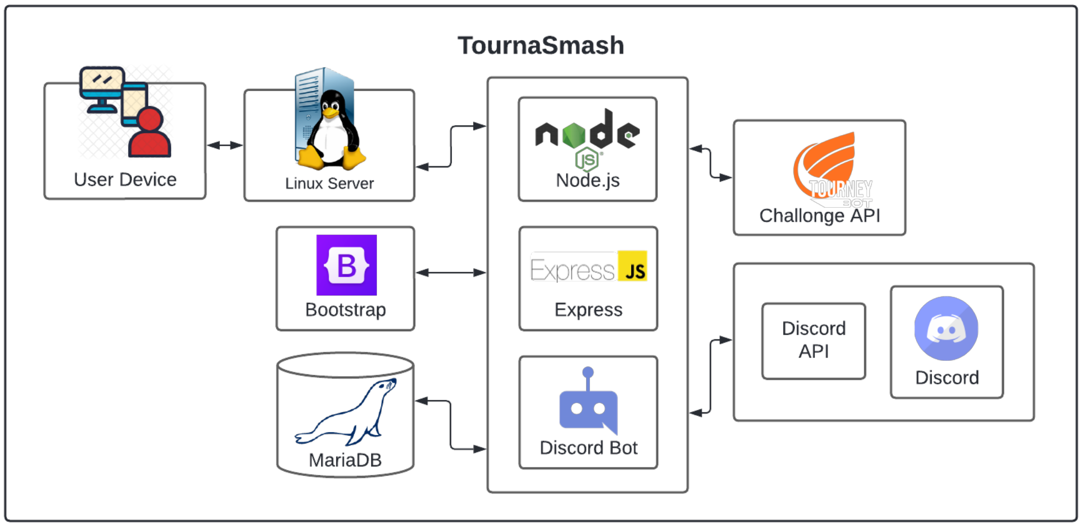
Web Interface: Create/edit profiles, create/edit/delete tournaments, view tournament details, check-in players, view stats, find local tournaments.

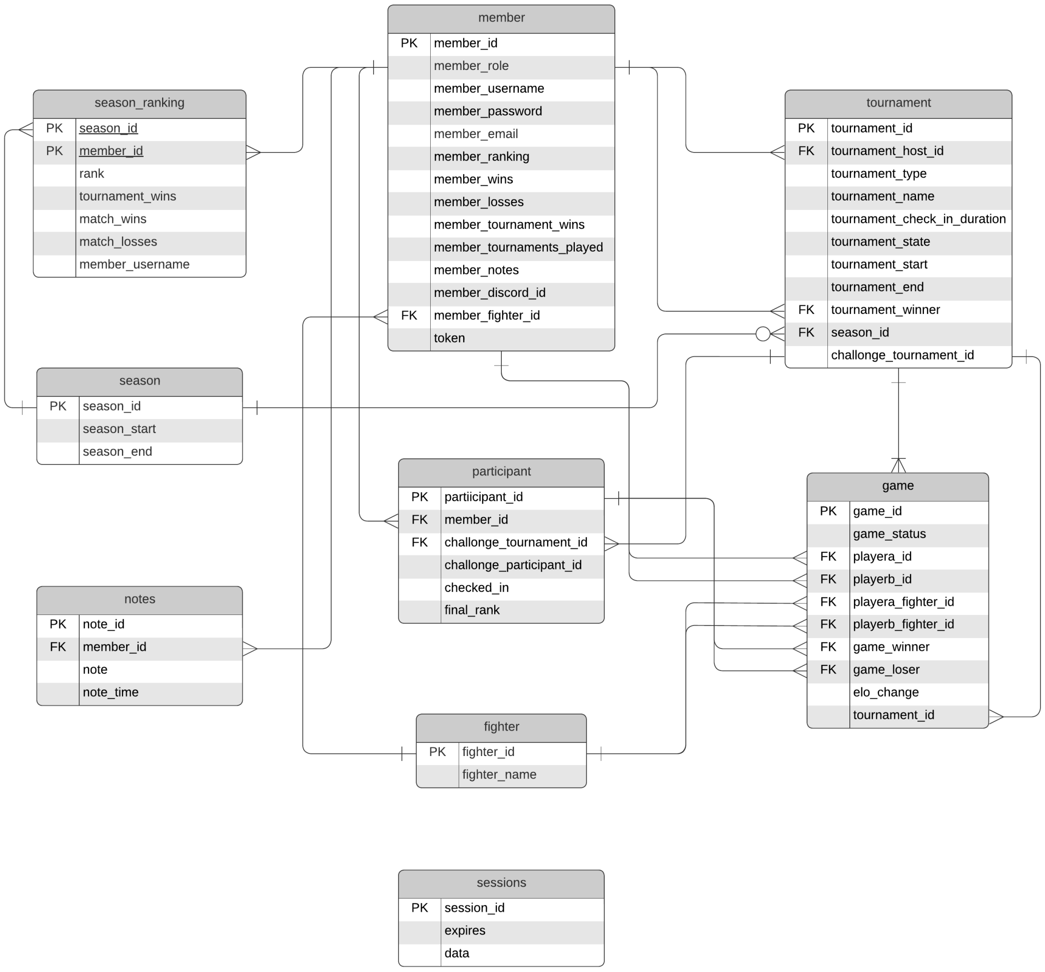
Kiosk: Check-in players, input match outcomes, view next match.

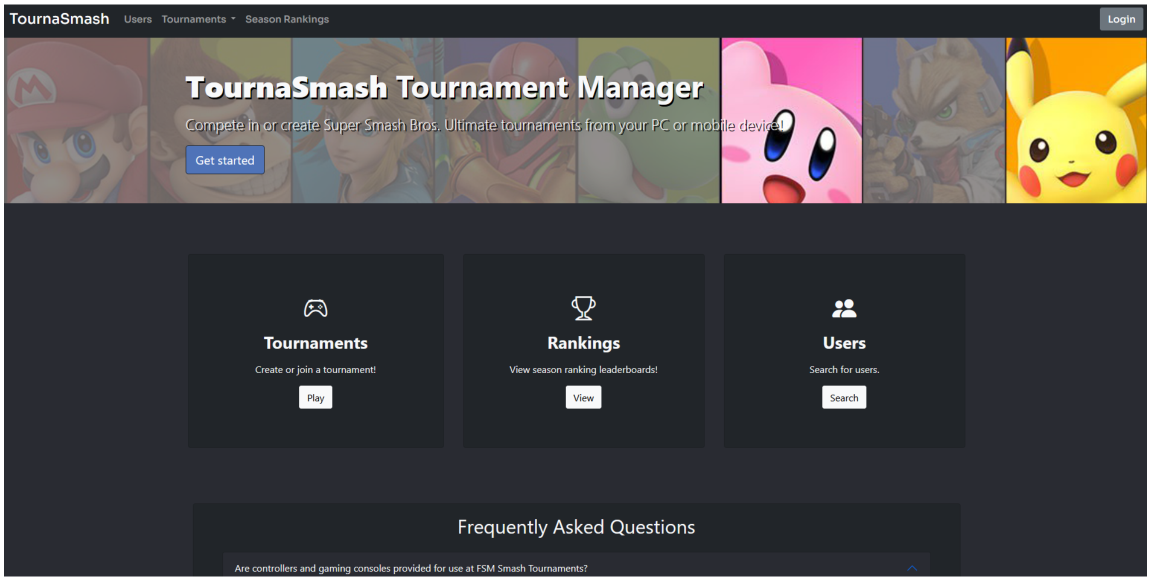
Discord Bot: Check-in players, view upcoming matches, view previous match results.

3.2 Architecture

The base of our system will be a web interface that retrieves data from the Challonge API to update and add to the MariaDB database. Player logins can be connected with a given Discord account to be able to utilize the Discord server bot for check-ins and activity postings. A kiosk system interface will give players the ability to check-in and input the outcome of their matches to the bracket and see any upcoming matches.


TournaSmash
User Device
Linux Server
Node.js
Bootstrap
Express
MariaDB
Discord Bot
Challonge API
Discord API
Discord


member
PK member_id
member_role
member_username
member_password
member_email
member_ranking
member_wins
member_losses
member_tournament_wins
member_tournaments_played
member_notes
member_discord_id
FK member_fighter_id
token
season_ranking
PK season_id
PK member_id
rank
tournament_wins
match_wins
match_losses
member_username
tournament
PK tournament_id
FK tournament_host_id
tournament_type
tournament_name
tournament_check_in_duration
tournament_state
tournament_start
tournament_end
FK tournament_winner
FK season_id
challonge_tournament_id
season
PK season_id
season_start
season_end
participant
PK partiicipant_id
FK member_id
FK challonge_tournament_id
challonge_participant_id
checked_in
final_rank
game
PK game_id
game_status
FK playera_id
FK playerb_id
FK playera_fighter_id
FK playerb_fighter_id
FK game_winner
FK game_loser
elo_change
tournament_id
notes
PK note_id
FK member_id
note
note_time
fighter
PK fighter_id
fighter_name
sessions
PK session_id
expires
data



Compete in or create Super Smash Bros. Ultimate tournaments from your PC or mobile device!

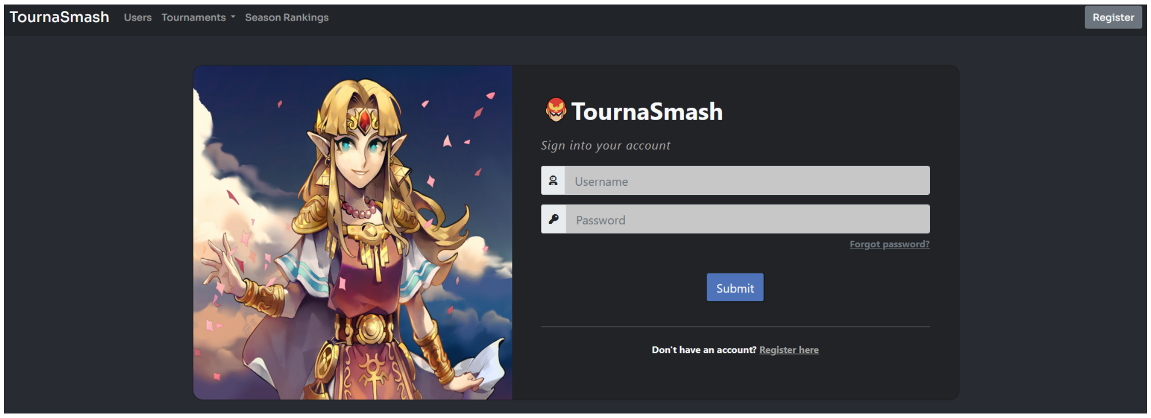
Create or join a tournament!

View season ranking leaderboards!

Search for users.

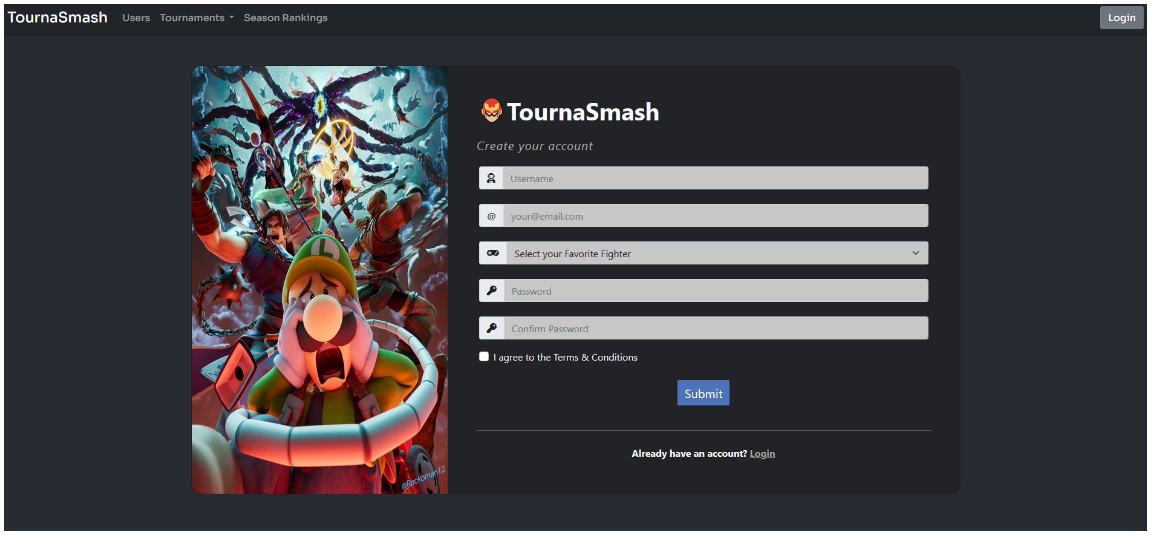
Frequently Asked Questions

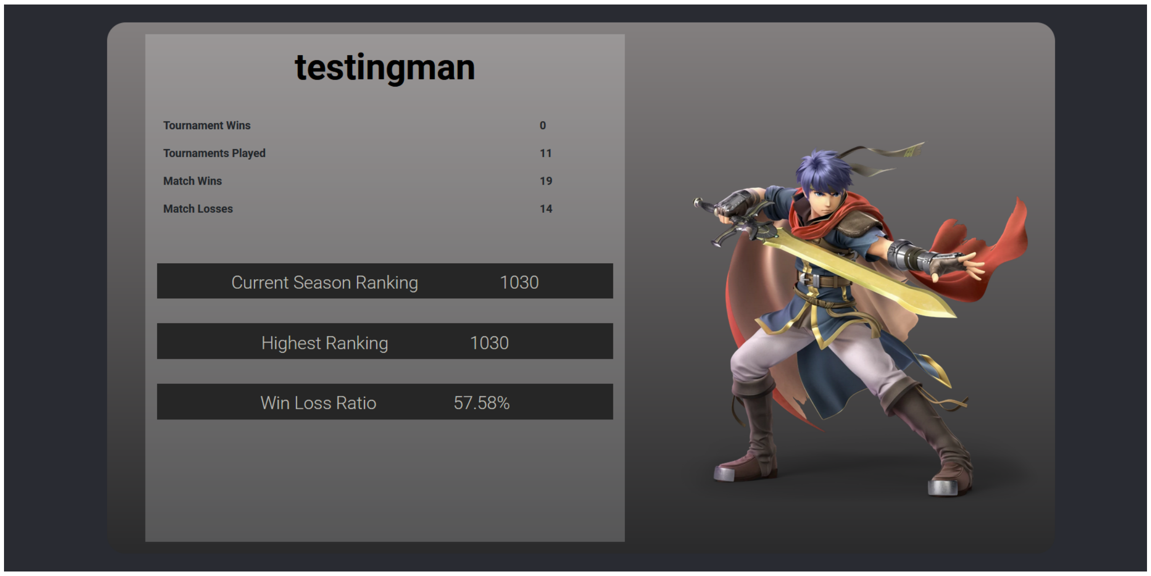
Are controllers and gaming consoles provided for use at FSM Smash Tournaments?



Sign into your account

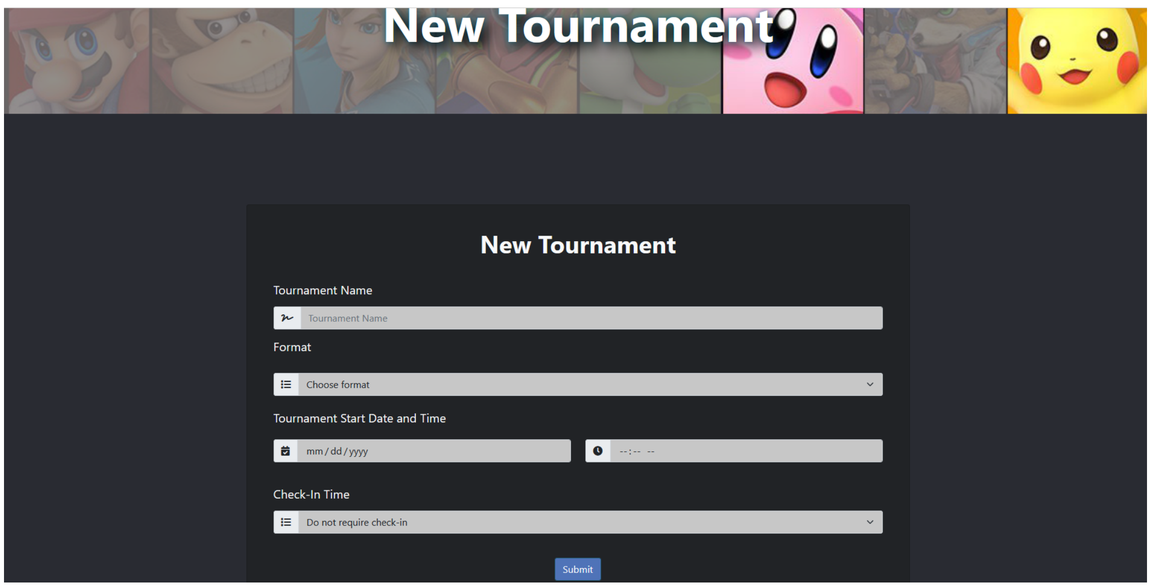
Don't have an account? Register here

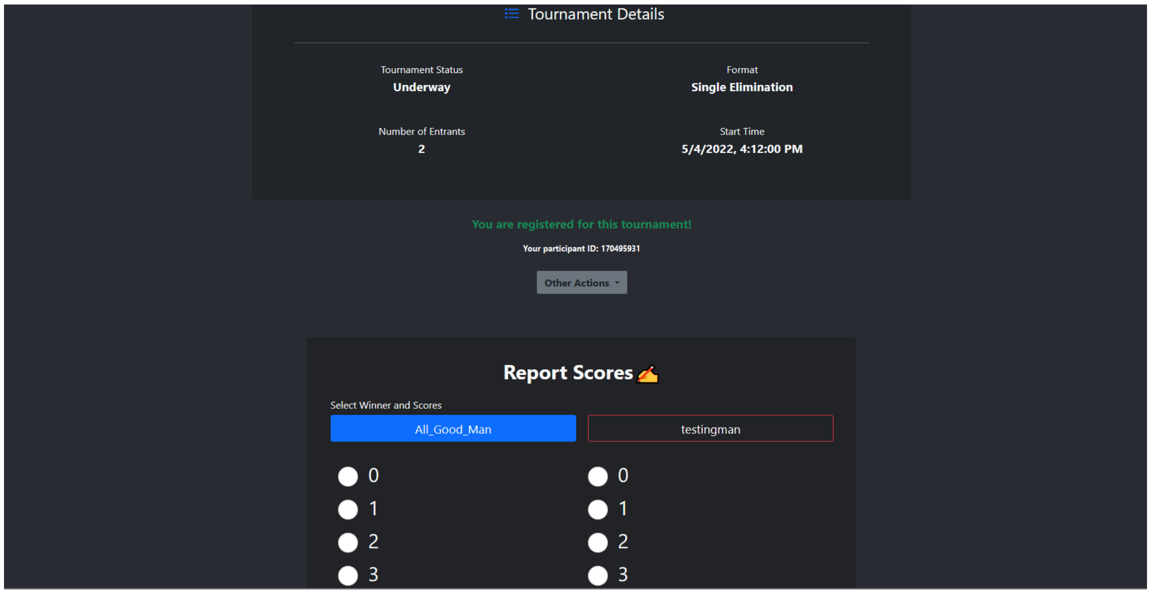


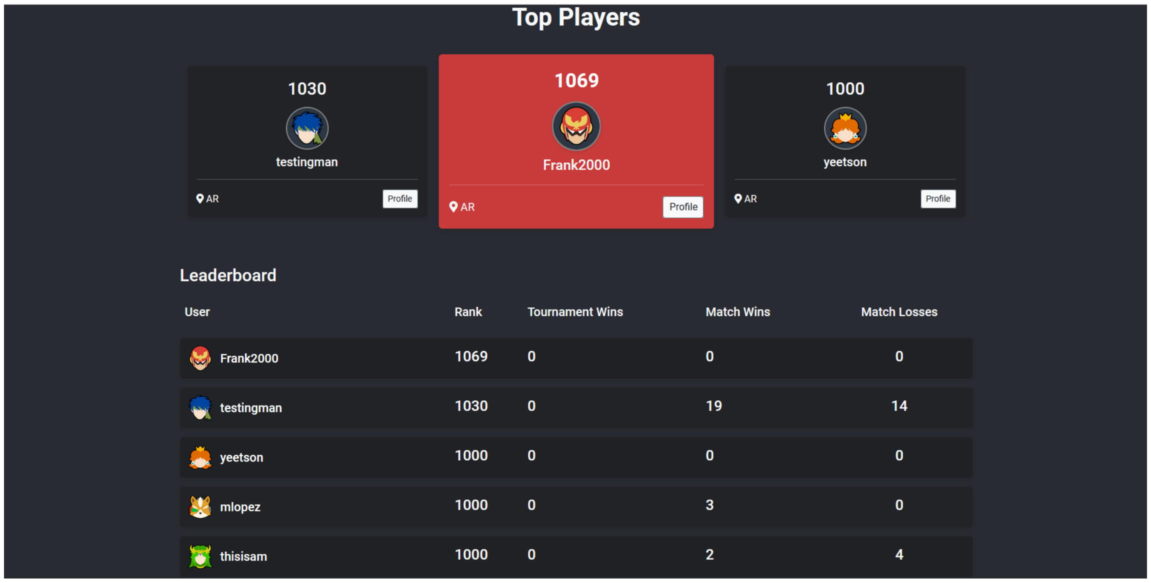


testingman

Tournament Wins	0
Tournaments Played	11
Match Wins	19
Match Losses	14
Current Season Ranking	1030
Highest Ranking	1030
Win Loss Ratio	57.58%


New Tournament
New Tournament
Tournament Name
Tournament Name
Format
Choose format
Tournament Start Date and Time
mm / dd / yyyy
--:-- --
Check-In Time
Do not require check-in
Submit


Tournament Details
Tournament Status
Underway
Format
Single Elimination
Number of Entrants
2
Start Time
5/4/2022, 4:12:00 PM
You are registered for this tournament!
Your participant ID: 170495931
Other Actions
Report Scores
Select Winner and Scores
All_Good_Man
testingman
0
1
2
3
0
1
2
3



Top Players

1030	1069	1000
testingman	Frank2000	yeetson
AR	AR	AR

Leaderboard

User	Rank	Tournament Wins	Match Wins	Match Losses
Frank2000	1069	0	0	0
testingman	1030	0	19	14
yeetson	1000	0	0	0
mlopez	1000	0	3	0
thisisam	1000	0	2	4

3.3 Risks

These are the potential risks we determined could affect our project and how we handled mitigating these risks from compromising our project :

1. **Website Security** – Use of SSL certificates to ensure a secure connection between browser and server to verify the user data will be secure, and that our application is trustworthy.
2. **Data Theft** – To prevent the theft and use of personal data we encrypted all personal information which we deemed a potential factor or theft such as user passwords.
3. **SQL Injection** – To protect from SQL injections, we implemented input sanitation and validation and SQL parameters.
4. **API Unresponsive** – In order to handle the event of an API failing, we will ensure that our system is able to operate freely of both Challonge and that all primary functionality will still be available on the web interface. Match and Tournament Data will still be stored on the available DBMS, and any data necessary to outside APIs will be available for manual updating once API is responsive again.

3.4 Tasks

3.4.1 Completed

The following is the list of initially proposed completed tasks as well as which team member completed the task:

Task	**Date Completed**	**Completed By**
Create server.	Feb. 2	Michael
Setup necessary server packages, network configuration.	Feb. 7	Jason
Create web interface skeleton.	Feb. 2	Michael, Martin, and Jason
Design player/public web interface pages (landing page, login, player, tournament view, password reset).	Feb. 14	Michael, Martin, and Jason
Connect to MariaDB database.	Feb. 17	Jason
Implement database schema (define user roles here).	Feb. 17	Jason
Connect web interface to server host and database platform.	Feb. 20	Jason
Implement data encryption and obfuscation for password and email addresses.	Mar. 4	Jason
Connect to Challonge API in order to create a wider reach for player base and tournament management purposes.	Mar. 07	Jason
Connect Discord bot to database.	Mar. 11	Michael
Implement Challonge API to incorporate bracket system/interface.	Mar. 14	Michael
Create rankings table (based on Elo system).	Mar. 24	Jason
Connect to Discord API and begin bot creation (start with checking in, build out features from there).	Mar. 27	Michael
Implement Challonge API for tournament start and finalize.	Mar. 28	Martin
Implement Challonge API to process player check-in/check-out.	Mar. 30	Jason
Implement Challonge API for tournament ”scoreboard” (match/player history, results, and ranking/Elo adjustments). Partially Complete - Tournament results page shows winner and embedded Challonge bracket displays match history, ranking/Elo adjustments are not available.	Apr. 2	Jason
Design admin web interface pages (player views, tournament view/edit, tournament administration)	Apr. 5	Martin
Admins should be able to enter notes about a player that is only visible to admins.	Apr. 5	Martin

Implement Challonge API for tournament creation, deletion, and editing. Partially Complete - Tournament creation and deletion are available, editing is not.	Apr. 5	Michael, Jason
Create check-out system to accommodate players leaving before the end of tournament (allow players, admin, or organizers to accomplish this goal). Partially Complete - Players may drop-out from their tournaments, admins and organizers cannot manually perform this action for players.	Apr. 10	Jason
Create pre-tournament check-in validation (Display all registered users who have checked in, and all remaining players yet to be checked in). Partially Complete - Kiosk lists players yet to be checked-in, does not display checked-in players.	Apr. 17	Michael
Create kiosk interface.	Apr. 18	Michael
Connect kiosk to database.	Apr. 18	Michael
Add/Edit/Remove/List users added to admin privileges.	Apr. 24	Martin
Password administration.	Apr. 25	Martin
Implement user ability to edit account information (reset password or update any account information). Partially Complete - Reset password is available, updating account information is not.	Apr. 25	Martin
Revisit Discord bot functionality (add any additional necessary support for bot such as retrieving tournament or player information).	Apr. 27	Martin
Create and implement Elo calculations.	Apr. 29	Martin

Tasks Schedule

3.4.2 Incomplete

The following is the list of initially proposed incomplete tasks which were not addressed in any form across our project:

Task
Create function to Add/Remove users to a tournament as competitors at will (Admin only. To be used if registration fails or players is a "walk on").
Create email confirmation system for user sign-up.
Create Seasons management system. (Give admin ability to add/edit).
Implement Elo history page for players.
Create blacklist system. (If players consistently dodge/leave/no-show after signing up, deny player until admin approves them).
Allow tournament organizer (and subsequently admin) to record matches (participant, character played) and sets (best of 1, best of 3, best of 5, etc.).
Add function to support to retroactively add a tournament to a season (admin only).
Allow players permissions to customize and build their profiles.

3.5 Deliverables

1. Project Proposal
2. User Interface Design Drawings and Flowcharts
3. Database Schema and Data (in `mysqldump` text format)
4. Application Development Code
5. Final Report

4 Project Members

4.1 Team Members

Jason Nguyen (4023) Jason Nguyen is an Information Technology major with concentration in Programming in the department of Computer Science and Engineering at the University of Arkansas - Fort Smith. He has completed relevant coursework for the proposed project by completing CS 2043 – Database Systems II, CS 2033 – Web Systems, CS 3013 – Human Computer Interaction, CS 3003 – Distributed Systems, CS 4003 – Software Engineering, and CS 4903 – Spec Topic Web Framework. His responsibilities will be working on front-end and back-end design and connections as well as API connections.

Martin Funmaker (4023) Martin Funmaker is an Information Technology major with a concentration in programming in the department of Computer Science and Engineering at the University of Arkansas – Fort Smith. He has completed relevant coursework for the proposed project by completing CS 2043 – Database Systems II, CS 2033 – Web Systems, CS 3003 – Distributed Systems, CS 4003 – Software Engineering, CS 4903 – Spec Topic Web Frameworks, CS 4363—Internet Of Things Development. His responsibilities for this project will be to help develop the front-end and back-end design, develop the interface design, as well as developing and integrating the discord bot.

Michael Lopez (4023) Michael Lopez is an Information Technology major with concentration in programming in the department of Computer Science and Engineering at the University of Arkansas – Fort Smith. He has completed relevant coursework for the proposed project by completing CS 3003 – Distributed Systems, 2033 – Web Systems, 4903 – Special Topic Web Framework, 3503 – IT Security, 2043 Database Systems II. He also obtained relevant experience through his employment at Walmart Global Tech as a Server Operations Technician where he was responsible for resolving opportunities with application servers. His responsibilities will include contributing to the design model and implementation of technologies used to develop the system proposed. He will work on the database initialization and connections to the system.

4.2 Departmental Advisors

Our project was supervised by Professor Cuevas and Professor Mackey in the department of Computer Science and Engineering at the University of Arkansas – Fort Smith.

Each project must have one or more faculty sponsors. If a project does not receive prior approval, a faculty sponsor will be assigned to the project.

5 References

[1] Node.js, “About node.js®.” https://nodejs.org/en/about/.

[2] W3Schools, “Node.js Introduction.” https://www.w3schools.com/nodejs/nodejs$_intro.asp$.

[3] Mozilla, “Express/Node introduction - Learn web development | MDN.” https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express$_Nodejs/Introduction$.

[4] J. Thornton and M. Otto, “Bootstrap.” https://getbootstrap.com/.

[5] MariaDB, “Mariadb/server.” https://github.com/MariaDB/server. Accessed: 18-Jan-2022.

[6] Discord, “API Docs for Bots and Developers.” https://discord.com/developers/docs/intro.

[7] Challonge, “Challonge API V1 Documentation.” https://api.challonge.com/v1.