

Spring 2023

Tournament Management Software for UA Badminton Club

Comprehensive Final Report CSC 536

Junfeng Xu, Enfa George, Rupal Jain, Urvika Gola

Table of Contents

1. Introduction	3
2. Customer Need	8
3. Project Goals	12
4. System Description	18
5. Final Status	29
6. Project Management	41
7. Team	48
8. Constraints and Risks	53
9. Reflection	54

1. Introduction

Our Project

We are implementing a tournament management system for the University of Arizona Badminton Club (UABC) that they can use to organize badminton tournaments.

Our Motivations

The UABC hosts two main tournaments every year and there are normally over one hundred participants competing in each of those tournaments. One of our team members is part of the UABC and he always hears complaints from the badminton club committees that they always have to manage a tournament manually including collecting event entries manually with an excel spreadsheet, assigning seeding and making draw manually, and running the tournament manually on tournament day. The only software that is available to them to manage the tournament has a long learning curve for new users, complex user interfaces, and a high-cost annual license. We as a team decided to support our in-house sports club at the University of Arizona and help them with their struggles on tournament management.

The Novelty of Our Project

The tournament management system we built in this project has a clean and straight-forward user interface that is easy-to-use and with a short learning curve. The system also includes many automated features which can eliminate their manual efforts mentioned in the previous paragraph.

Challenges

There are 3 main challenges for our team. The first one is the lack of domain knowledge. The majority of the team members are not familiar with badminton tournament formats or badminton as a sport. To build this tournament management software requires the developer to have some basic knowledge on badminton tournaments. To address this challenge, one team member with badminton expertise will give a Badminton 101 training session for the whole team. The second challenge is to communicate with the customer in a timely manner. We have a real customer, the UABC. The representative of the club is a student in her senior year with a full packed schedule. It will be really hard to schedule meetings with her. To address this challenge, we will try to get in touch with her early and propose all the meeting times we need to meet with her so that she can block those times out early. Also we can find a secondary contact from the club in case of the absence of the primary contact. The third challenge is that the tournament management software requires front end user interfaces but none of us has much front end development experience. To address this, we will do some research to learn as a team and consult with subject matter experts.

The web development framework we preliminarily agree to use is called Flask. It is known to one team member but new to others. The team preliminarily decided to use Python for web development that is known to all team members (some are more familiar with it).

Highlights

Team accomplishment iteration 1:

- Each team member has set up the project environment on their machine and successfully ran the test program.
- Project has been set up on all the project management tools including Jira, atlassian, and Github. We have been using it for the whole iteration.

- Database design has completed.
- Design for the front-end user interfaces and back-end controller for our project are completed.
- The user interface and back-end controller portion of the implementation for the user sign-in and sign-up functions are done
- The implementation for the player dashboard and admin dashboard front-end user interfaces are completed.
- We met with the customer at the end of the iteration and collected feedback. The feedback items have been added to the backlog.

Team accomplishment iteration 2:

- We completed the backend support and integrated it with the front end UI for the user login/signup functions. Test has been performed to verify the user login and signup functions.
- We completed the database setup and database table creations. Test has been performed to verify that we can retrieve from the data tables and store data into the data tables.
- We completed most of the player dashboard functionalities including the player dashboard UI web pages, the backend support for player dashboard, retrieving and storing player information from and to the player database table. The event registration portion of the player dashboard remains to be completed. Test has been performed to verify that a player user account can use it to display their current profile and to modify the profile.
- Functional test has been performed (refer to the bullet points above). Unit testing has been performed during the implementation of each function. User testing has been performed during the customer review meeting.
- Customer review meeting was conducted for a showcase on 4/2/2023

- We completed the front end support for all the public view UI web pages. Unit testing has been performed accordingly.

Team accomplishment iteration 3:

- We completed the backend control support for admin dashboard home page, tournament creation page, event management page, and match management page.
- We completed the left-over database creation and deployment items and verified the database was fully functional.
- We completed the event registration function on player dashboard
- We completed the tournament creation function, save seeding function, make draw function, match management function.
- We completed the backend controller support for all public pages including home page, tournament detail page, player page, event page, and match page.

Changes

We have made a few changes since the proposal, including a front-end user interface design change after team review, the customer desired overall experience change since our first meeting with them, and database design changes.

Below is a table for the date, motivation, description, and implications of each change:

Date	Description	Motivation	Implication
03/02/2023	UI Design Change	To make the admin dashboard more user-friendly and easier to update.	Updated UI design to have an admin dashboard in a single screen instead of multiple screens. This will make it easy for the Admin to make updates.
03/07/2023	Database redesign	To fix errors with primitive data types in the initial design.	Reworked structure of database and created a new ER diagram.
03/12/2023	Customer desired overall experience change	To satisfy the customer's updated requirements that we have learned from the customer's feedback.	Given our current design for the project, it is still able to address customer's new surfaced problems.

2. Customer Need

Customer Need

Our primary customer is the UA Badminton Club officers. They are the primary users of our software. Our secondary stakeholders are the badminton club members or any tournament participants. Our customers want a software that they can host and manage their badminton tournaments with. The software includes a client side and a host side. On the client side, a tournament participant is able to register their entries with the events (Men's Singles, Men's Doubles, Women's Singles, Women's Doubles, Mixed Doubles) they are competing in and input their partner's name for doubles events. On the host side, the tournament organizer is able to create/set up a tournament and publish it to the public. They are also able to manage the participant list so that they can fix any incorrect entries, delete invalid entries. They can seed players and make bracket draws for each event. On the tournament day, the host side can update the match scores and advance the winners down the brackets. The client side can display the match results in real time for the participants to watch. The club officers are currently doing everything manually or using a software that is really complex to learn how to operate and with a high cost. They come to us with the request to design a new software for them that is with intuitive interfaces and can automate all their manual efforts.

Customer Need Changes

After meeting with customers for a sprint review, their desired overall experience has changed. They used to run software that they could use to organize tournaments. That software is really hard to use and has a high annual license fee. They came up with the

original desired overall experience during our initial meeting with them. Now they have lost access to that software for an undisclosed reason. This year, they are organizing the tournament manually. Includes manually creating the events tables from the player registration list, manually making draw brackets, manually managing matches and progressing event brackets. That is a massive amount of manual effort. As a result, their desired overall experience is changed to have a tournament management software that can help them eliminate all the manual efforts that mentioned above and automate everything. The original desired overall experience from the customer is still valid and is taken into consideration for our product design but the emphasis can be changed to satisfy their latest desired overall experience.

User Requirements

High priority stories

- As a tournament organizer, I want a secure way to sign into the admin page, so that I can have all the admin authorities to the tournament over regular users.
- As a tournament organizer, I want to assign points to tournament participants, so that I can use points to seed the top players.
- As a tournament organizer, I want to be able to make draws for all the events in a tournament, so that the tournament bracket for each event will be generated.
- As a tournament participant, I want to be able to create a user profile and edit it after creation, so that the tournament organizers can have my information that is required for the tournament registration.
- As a tournament participant, I want to sign up for the tournament with a straightforward user interface, so that I can indicate what events I want to participate in and who my partner is for a doubles event.

- As a tournament participant, I want to see the match results and tournament bracket update in real time on the tournament website during the tournament day, so that I can know who my next round opponents are and what round of the tournament I am currently in.

Low priority stories

- As a tournament organizer, I want to have an easy-to-use user interface to record scores for each match, mark the winner, and advance the winner during the tournament day, so that I can easily manage the tournament brackets on the overwhelming day.
- As a tournament organizer, I want to be able to make the tournament in ABCD Drop Flight format, so that each participant will be guaranteed to play at least 3 games which will attract more players to our tournaments.

Acceptance tests for high priority stories

- Given a club officer's credential is presented, when a user signs in, then the admin authorities are given.
- Given a tournament participant's credential is presented, when a user signs in, then only general user authority is given.
- Given the tournament registration is closed and the participating player roster is set and an admin logs in, when the assign point function is performed, then points can be assigned to each player based on their skill levels.
- Given the tournament registration is closed and an admin logs in, when the make draw function is performed, then the tournament bracket will be generated.

- Given the tournament participant sign up page, when a profile creation is requested and the profile fields have been filled in by the player, a player profile will be created and stored.
- Given the player signs in with their regular user credential and they are in their profile page, when the edit profile function is requested with new profile information, the player profile will be updated.
- Given the tournament participant sign up page, when a player wants to enter any events (MD, MS, WD, WS, XD) to compete in, then the player's name is listed under the event roster of which they try to enter.
- Given the tournament participant sign up page, when a player wants to enter any doubles events (MD, WD, XD) to compete in and they enter their partner name, then the player's partner name will be tagged to their entry under the doubles event roster of which they try to enter.
- Given the matches are in progress on the tournament day, when the player logs in, then they can view the brackets for all events getting updated in real time.

3. Project Goals

Customer Problem

We have chosen the customer problem of lacking an easy-to-use and cost effective tournament management software.

Customer Problem Update

In the sprint review meeting with the customer, we learned that the customer's current experience on hosting a tournament has changed. Now they have lost access to their current software so that they have to organize and run their tournaments all manually. Their problem has shifted to needing a tournament management software that can automate their manual effort. Our current design of the project can still address their new problem and satisfy their requirement.

User benefits

The system provides trivial web interfaces for both the tournament organizer (badminton club officers in our case) and the tournament participants (i.e. the players). The tournament organizer interface allows the organizer to manage player profiles, make seedings, make draws, and progress each event draw on the tournament day. The tournament participant interface allows the players to create and edit their profile, enter in the events they want to compete in, and view the tournament's progress on the tournament day. We also provide easy-to-follow user manuals. The system is built on a free of charge or low cost platform. The trivial web interfaces and the easy-to-follow user manuals can make any badminton club officer take on the software admin role with a short learning curve. The low cost on the software can help the club reduce

spending on organizing tournaments. We verified how those benefits support the customer's overall experience with the customer (UA Badminton Club officers) during sprint review meetings.

User Benefits Update

The project provides tournament organizers (our customer) with the ability to automate the tournament event tables creation, to automate the tournament event drawing making process, and to automate the match management on the tournament date. Those benefits directly solve customer's current problems.

Uses Cases

Use Case 1 (For primary customer):

Title: Tournament Event Management and Draw Creation

User Goal: The tournament organizer is able to manage tournament events (MS, MD, WS, etc.) and create tournament draws through a simple admin dashboard interface with minimal manual effort.

Basic Flow:

1. The tournament organizer logs in as an admin to an active tournament.
2. The system displays the admin dashboard Home page.
3. The tournament organizer clicks "Events" on the control panel.
4. The system brings up the event management page and the entries for each event are displayed in different event tables.
5. The tournament organizer enters seeding information for each event in the 'seed' input column of the event tables.
6. The tournament organizer clicks 'save' for each table.

7. The system will update the display with the most current seed information.
8. The tournament organizer clicks 'Make Draw'.
9. The system will bring up a new page to display the draw for each tournament event.

Alternative Flow:

Title: Invalid seeding.

Description: when invalid seeding is entered, the system will prompt error and ask the user to correct the seeding.

Connection with basic flow: this alternative flow may be invoked after action 6 and the basic flow resumes at action 7.

Use Case 2 (For primary customer):

Title: Tournament Match Management

User Goal: The tournament organizer is able to manage matches on the tournament day through a simple admin dashboard interface with minimal manual effort.

Basic Flow:

1. The tournament organizer logs in as an admin to an active tournament.
2. The system displays the admin dashboard Home page.
3. The tournament organizer clicks "Matches" on the control panel.
4. The system brings up the match management page and all the matches that are currently available to play are displayed in the Up-coming Match table, In-progress Match table and Finished Match table.

5. The tournament organizer changes the status of a match from the Up-coming Match table to “in progress” and picks a “court number”.
6. The tournament organizer clicks ‘save’.
7. The system moves the match to the In-progress Match table.
8. The tournament organizer changes the status of a match in the In-progress Match table to “finished” and enters the match score.
9. The tournament organizer clicks ‘save’.
10. The system will move the match to the Finished Match table.
11. The tournament organizer picks the winner of a match in the Finished Match table.
12. The tournament organizer clicks ‘save’.
13. The final status of a match will be saved and displayed.

Alternative Flow:

Title: Court is occupied.

Description: when a court has already been assigned to a match and the user tries to assign it to another match, the system will prompt error and cancel the action. The user can re-enter the court number.

Connection with basic flow: this alternative flow may be invoked after action 5 and the basic flow resumes at action 7.

Use Case 3 (For secondary customer):

Title: Player account sign up

User Goal: The tournament participant is able to sign up an account for the system and log into the system and see the player dashboard interface.

Basic Flow:

1. The tournament participant clicks “Signup” on the login screen.
2. The system brings up a Sign Up page.
3. The tournament participant fills in the sign up form and submit.
4. The system prompts “sign up successfully”
5. The system goes back to the login screen
6. The tournament participant logs in as a player to the system.
7. The system displays the player dashboard.

Alternative Flow:

Title: Account exists

Description: when the user tries to create an account that already exists, the system will prompt error and return to the login screen.

Connection with basic flow: this alternative flow may be invoked after action 3 and the basic flow resumes at action 5.

Use Case 4 (For secondary customer):

User Goal: The tournament participant is able to modify their profile information and register for a tournament by entering the desired events (MS, MD, WS, etc.) through a simple player dashboard interface with minimal manual effort.

Basic Flow:

1. The tournament participant logs in as a player to the system.
2. The system displays the player dashboard.
3. The tournament participant modifies the profile form on the player dashboard.

4. The tournament participant checks the desired events they want to enter and input their partners' name for doubles events.
5. The tournament participant click "save"
6. The system saves the updated profile information and enters the participant into the events selected by the participant.
7. The system displays the latest information for the participant.
8. The tournament participant logs out the system.

Alternative Flow:

Title: Gender Restriction

Description: when the player tries to enter an event and they don't meet the competing gender requirement, the system will prompt error and will not enter the event for the player.

Connection with basic flow: this alternative flow may be invoked after action 5 and the basic flow resumes at action 7.

Measures of Success

We have a real customer and the idea is from a real world problem our customer is experiencing (For more information about our real outside customer, refer to section 1 Introduction.). We schedule meetings with our customers to review the user stories and acceptance tests we initiated. If they are not satisfied, we can revise our user stories based on their feedback. Our customer-centric measure of success is to deliver all the high priority user stories and verify with the customer in periodic meetings that they meet their expectations.

4. System Description

System Overview

The system design uses an MVC architecture ie Model - View - Controller architecture. The MVC architecture separates the application into three main components: the Model, the View, and the Controller.

Model: The Model represents the business logic and data of the system. It will be responsible for storing and retrieving data from the database, as well as the processing and manipulating data. The Model will interact with the database layer to store and retrieve data.

View: The View represents the user interface and presentation of the system. It will be responsible for displaying data to the user and receiving input from the user. This is a web interface that the user can visit at a given URL. This is independent of any implementation logic.

Controller: The Controller acts as an intermediary between the Model and the View. It will receive input from the View and pass it to the Model for processing, and it will receive data from the Model and pass it to the View for presentation.

External Services:

Database: This stores all relevant data needed for the application to perform well. The database is hosted on Amazon RDS and Amazon EC2 instance. The database interacts with the Model only.

The boundaries of the system are clearly defined, with the Model representing the business logic and data, the View representing the user interface and presentation, and the Controller acting as an intermediary between the Model and the View.

System Challenges

1. Complexity of the model and controller:

The system has to handle the complex logic of creating matches within the rules and requirements of badminton, show in live the match results, automatically move players and create draws in the ABCD format. There are many rules and requirements to handle. Managing such complexity might pose challenges to data consistency and scalability if care is not taken in every step of the way.

2. Database Management and Security:

The database collects Personal Identifiable Information from players, hence contains sensitive info. The database as well as the system need to be secure and comply with local and federal regulations.

3. Interaction with the cloud service:

Since the database is hosted in AWS, this external interaction might cause latency. System must be designed in such a way that there are minimum db calls, otherwise the system would be too slow for the user and ruin the desired experience of the users.

4. Integration, Testing and Maintenance:

Since the view, controller and model are separate components and built

separately, routine integration testing needs to be done frequently to avoid big roadblocks round the road. The system must be tested and maintained frequently since its internal logic determines how the badminton match progresses and who plays when. Bugs in the system stemming from incorrect logic implementation can cost the users the game.

Rationale for the Design

The MVC architecture was very attractive to us due to the following reasons.

1. **Separation of concerns.**

We wanted to ensure that the functionalities implemented, changes made or needed at any level were independent of/ had minimal dependency on other components, inshort separation of concerns.

2. **Easily modifiable.**

The focus of the project, especially given the time frame, was to have working software that can meet the customer's needs and requirements. In future, they may want to say give the website a new look to match their branding, or want to introduce new subevents. We wanted to make sure that such requirements could be met with minimal overhead.

3. **Faster Development Process**

The separation of views, from models and that from controller ensured that we are able to build separately and faster.

Context Diagram and External Interactions

Find below the context diagram for the system. The Model represents the business logic and data, the View represents the user interface and presentation, and the Controller acts as an intermediary between the Model and the View.

Boundaries: The boundaries of the system are clearly defined. The thick black border marks the boundary of the system.

External interaction: The system interacts externally only with the database.

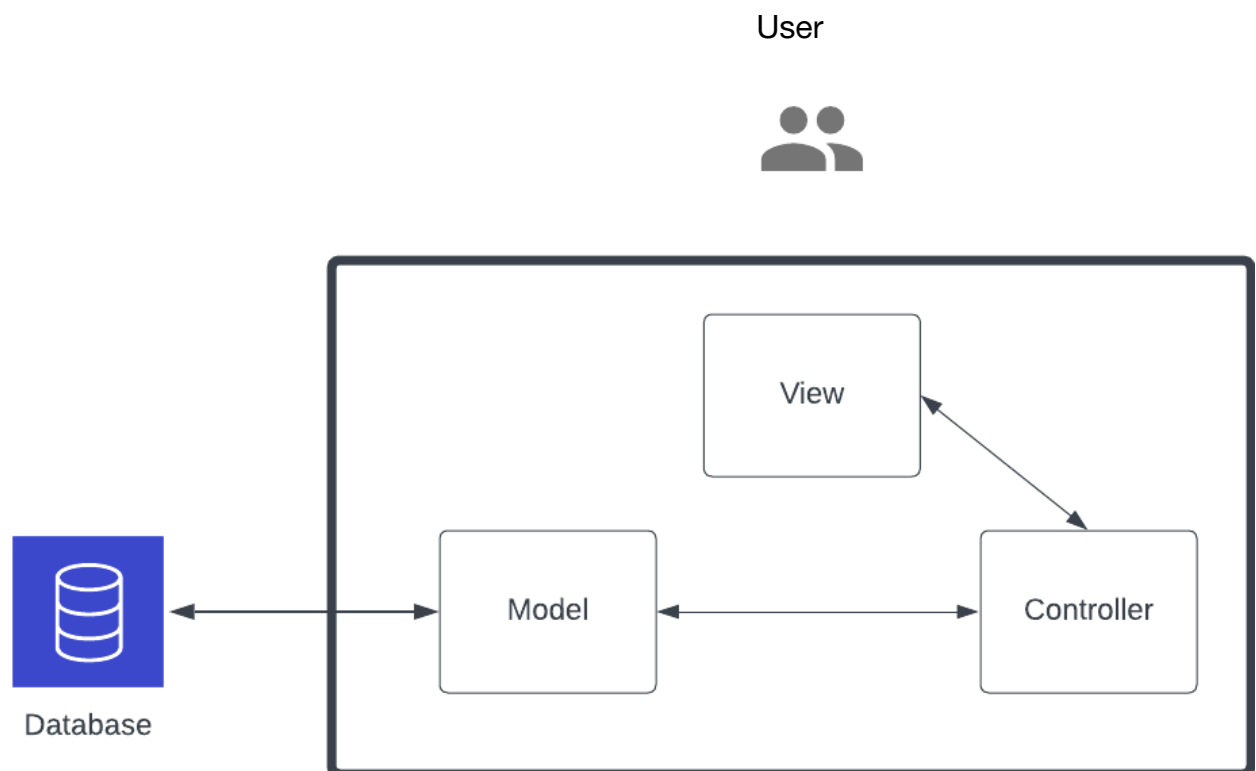


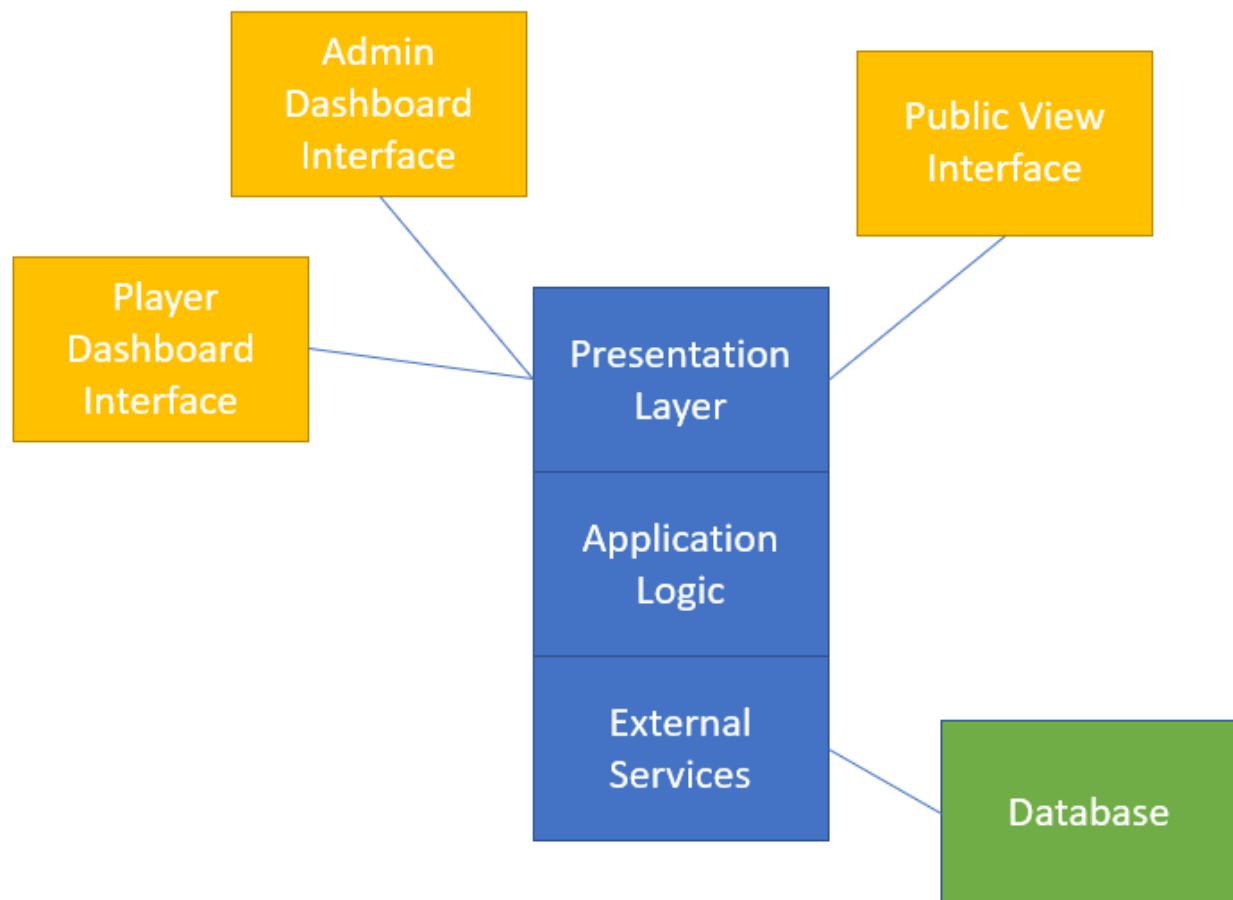
Figure: Context diagram for the application. The thick black border marks the boundary of the system. The system interacts externally only with the database.

Guides to the Main Architectural Views

We have employed a logical view and a development view for the architecture for our system.

Logical View:

Primary Diagram:



Element Catalog:

Admin Dashboard Interface (Owner: Rupal): this provides the interactive web pages for the tournament organizer user which includes functionalities including tournament management, player roster management, Seeding and drawing management, and match management. The tournament organizer may edit and publish a tournament, edit

player information, assign seeding to event entries, make draws for each event, and operate with the matches on the tournament day.

Player Dashboard Interface (Owner: Junfeng): this provides the interactive web page for the tournament player users. A player may edit their profile information and register for the events they want to participate in for the current active tournament.

Public View Interface (Owner: Junfeng): this provides the static web pages for the general public (tournament spectators, tournament participants, etc) to view all the tournament related information including tournament details, announcements, players, event brackets, match progress, and much more. Authorized users may login to the system on this interface.

Presentation Layer (Owner: Enfa): this takes care of the rendering for all the web page interfaces.

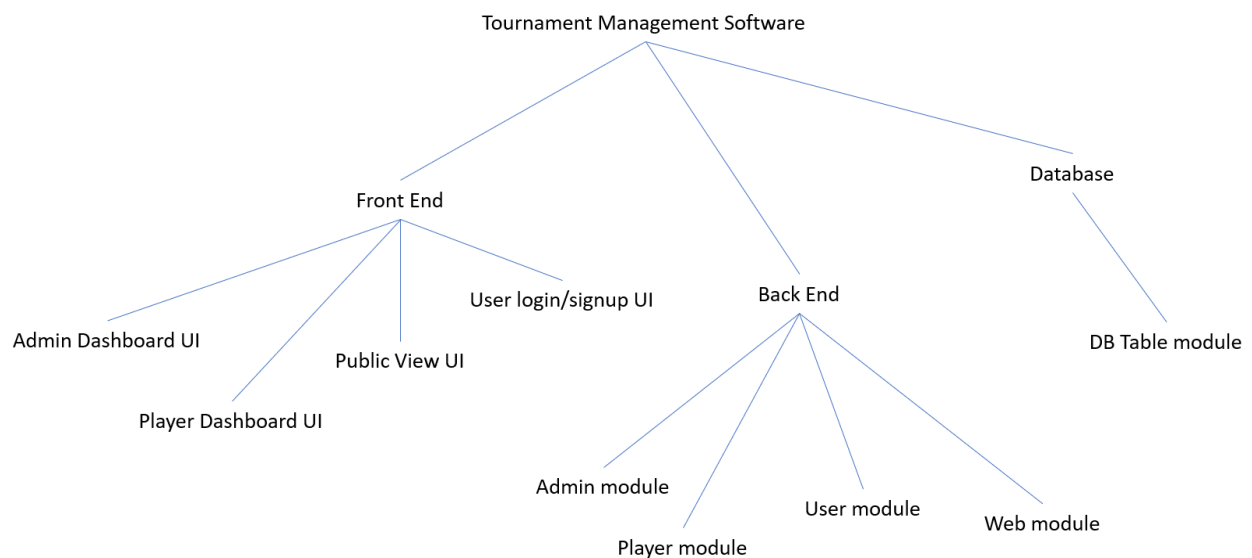
Application Logic (Owner: Enfa): this provides the support of communication between the Presentation Layer and the External Service. Includes requesting the information from the External Service (database) and sending to the Presentation Layer to display, storing the information passed in from the Presentation Layer and storing to the External Service (database).

External Services (Owner: Urvika): this provides the support for setting up the external database for our application, deploying the database on a distributed server, creating all the database tables.

Database (Owner: Urvika): this provides the storage space on cloud for our application needs including player profiles, tournament details, events, draws, and matches.

Development View:

Primary Diagram:



Element Catalog:

User login/signup UI (Owner: Junfeng): this includes the implementation of the user login and sign up pages. Users may sign up for a new account via it. Authentication will be performed when users try to gain access to the application.

Admin Dashboard UI (Owner: Rupal): this includes the implementation of tournament management page, player roster page, events management page, and match management page. The admin user may use it to interact with all the admin capabilities for the application.

Player Dashboard UI (Owner: Junfeng): this includes the implementation of the player profile management and tournament registration page. The player user may use it to interact with all the player user capabilities for the application.

Public View UI (Owner: Junfeng): this includes the implementation of the public web pages including home page, login/signup page, player roster display page, event display page, draw display page, and match display page. The general public user may use it to browse all the tournament related public information.

Admin module (Owner: Enfa): this module processes the data received from the admin front end UI and also sends the requested data to the admin front end UI to display. This module interacts with the database. This also routes all the admin dashboard web pages.

Player module (Owner: Enfa): this module processes the data received from the player front end UI and also sends the requested data to the player front end UI to display. This module interacts with the database. This also routes all the player dashboard web pages.

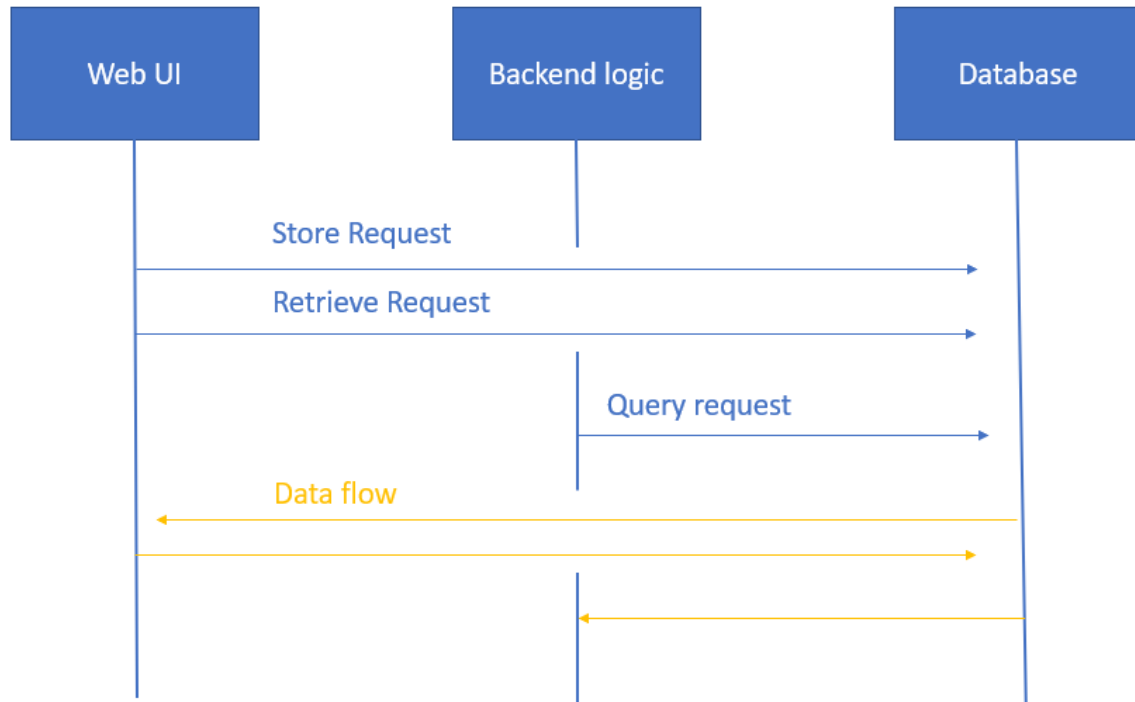
User module (Owner: Enfa): this module processes the user log in data received from the login front end UI and authenticates the user. This module processes the user sign up data received from the account sign up front end UI and validates it. This module interacts with the database. This also sends the requested data to the front end UI to display. This also routes all the login/sign up web pages.

Web module (Owner: Enfa): this module sends all the requested data for the public view page UI to display. This module interacts with the database. This also routes all the public view pages.

DB Table module (Owner: Urvika): this module creates all the required database tables for this application including user login table, permission table, player table, event table, tournament table, draw table, match table, etc. Our production database is hosted on Amazon RDS and uses PostgreSQL; it is linked to an EC2 instance. Our database is accessible over the internet seamlessly just by installing postgres client on our systems, and it's used by all team members for development. This avoided the installation of database software on each team member's local PC. This saved significant time for the team.

Dynamic Views

Primary Diagram:



Element Catalog:

Store request (owner: Junfeng): player profile information and event registration information will be stored to the database from the player dashboard UI. Tournament information and event/match information will be stored to the database from the admin dashboard UI. Player account information will be stored to the database from the user sign up UI.

Retrieve request (owner: Enfa): all the data displaying on different UI's we have will need to retrieve the data from the database

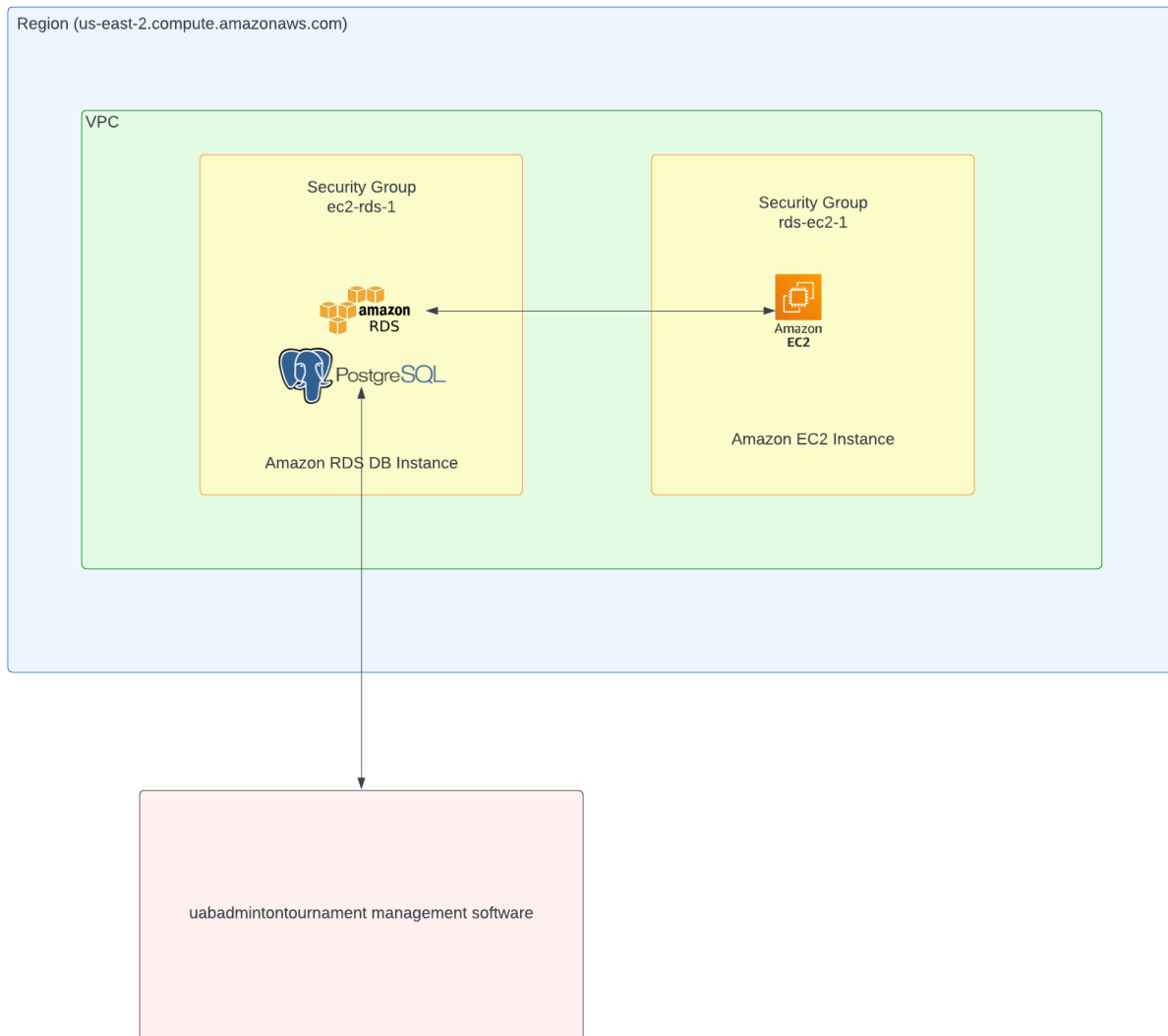
Query request (owner: Enfa): the user login function will need to query the user table from the database and authenticate the user login credentials.

Dataflow (Owner: Urvika): the data flow will be corresponded to the request types listed above.

External Interactions - Deployment Views

For the current scope of our project, the deployment view consists of deployed servers running on Amazon RDS and Amazon EC2. Our software runs locally but it communicates to the servers on Amazon cloud. The below diagram despite the geographical boundaries of the servers. Amazon RDS automatically takes care of replication internally for scalability, performance, and availability.

Primary Diagram:



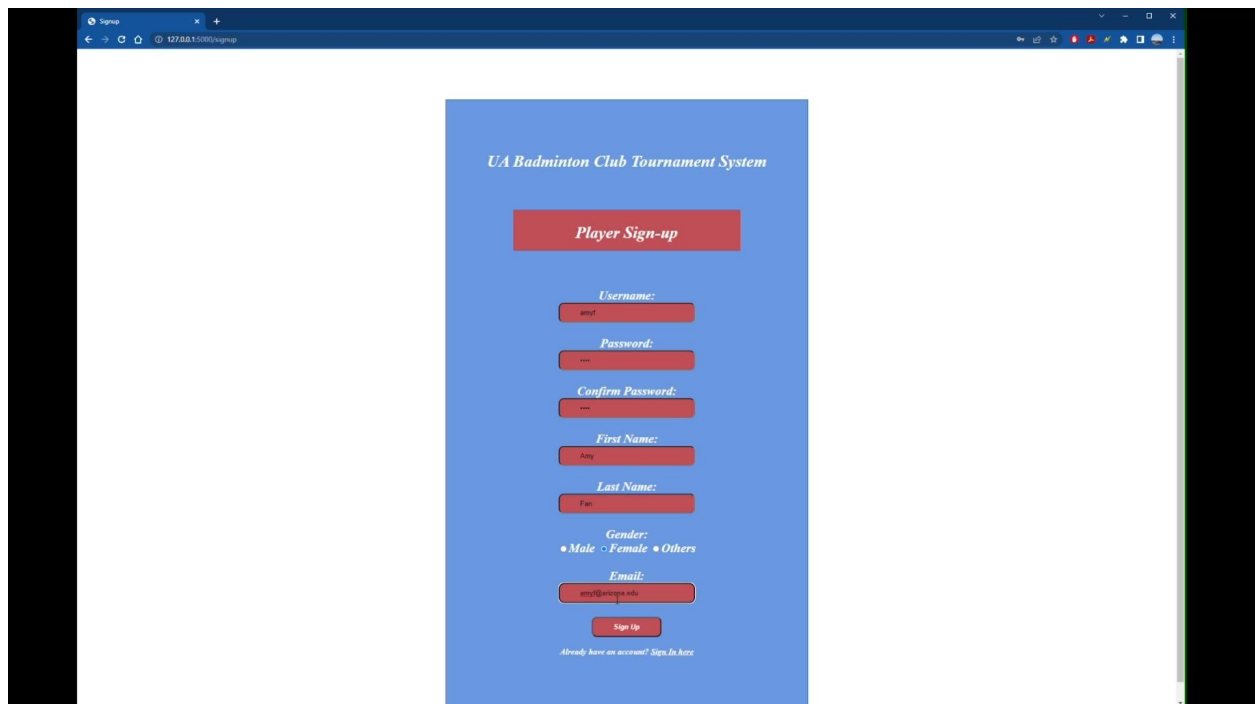
5. Final Status

Screenshots

We have mainly three types of actors. One is the administrator, another is the player and lastly a member of the public.

Actor : Player

1. Player can sign up



The screenshot shows a web browser window displaying the 'UA Badminton Club Tournament System' Player Sign-up form. The form is centered on a light blue background. It includes the following fields and options:

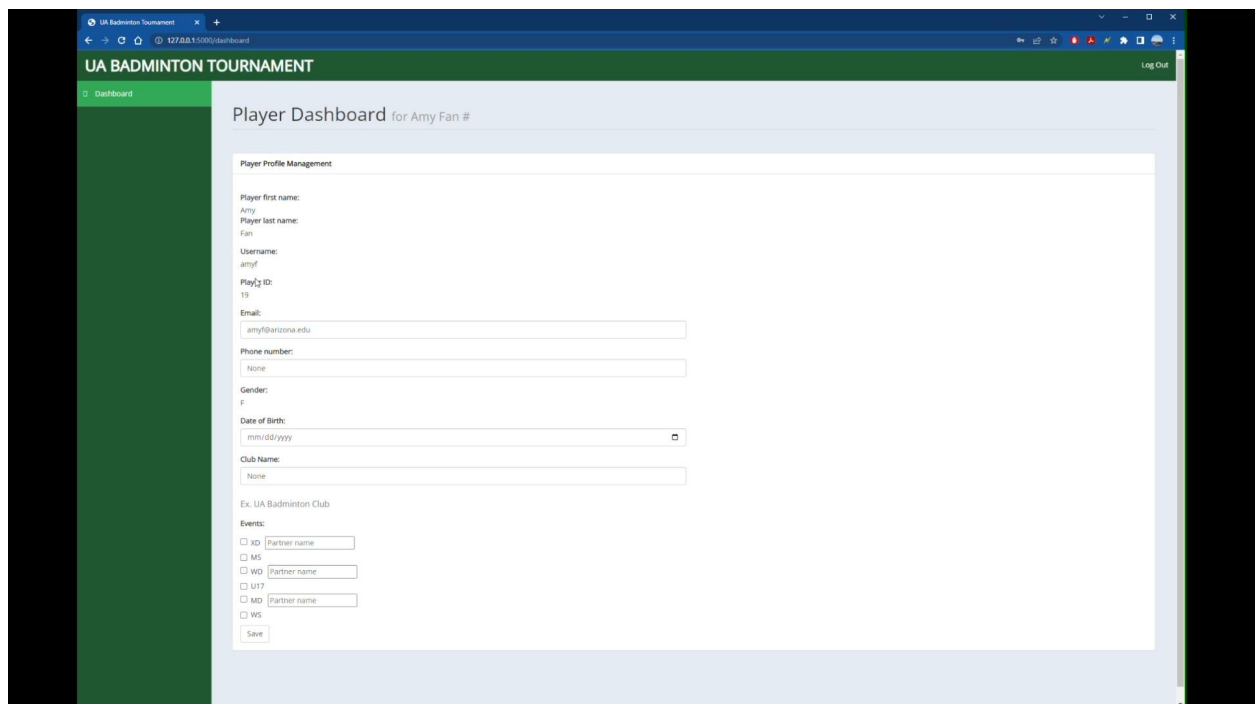
- Username:** Text input field with the value 'amr'.
- Password:** Text input field with the value 'amr'.
- Confirm Password:** Text input field with the value 'amr'.
- First Name:** Text input field with the value 'amr'.
- Last Name:** Text input field with the value 'amr'.
- Gender:** Radio button options: ☒ Male, ☐ Female, ☐ Others.
- Email:** Text input field with the value 'amr@amr.edu'.
- Sign Up:** A red button to submit the form.
- Already have an account? Sign In here**: A link at the bottom of the form.

Player signs up on the website

2. Player can securely sign in to the player dashboard



Player signs in on the website



Player successfully signs in to their dashboard/account

3. Player can edit profile details

The screenshot shows a web browser window with the URL `127.0.0.1:5000/dashboard`. The page title is "UA BADMINTON TOURNAMENT" and the user is logged out. The left sidebar has a "Dashboard" link. The main content area is titled "Player Dashboard for Amy Fan #". Below this is a "Player Profile Management" form. The form contains the following fields and values:

- Player first name: Amy
- Player last name: Fan
- Username: amyf
- Player ID: 19
- Email: amy@arizona.edu
- Phone number: 520-345-2233
- Gender: F
- Date of Birth: 02/12/1996
- Club Name: UA Badminton Club

Below the form, there is an example: "Ex. UA Badminton Club". Under the "Events:" section, there are five checkboxes, each with a "Partner name" field:

- ☐ XD Partner name
- ☐ MS Partner name
- ☐ WD Partner name
- ☐ U17 Partner name
- ☐ MS Partner name

A "Save" button is located at the bottom of the form.

Player is able to add/edit details

4. Player can sign up for tournament events

This screenshot is identical to the previous one, but with the "Events:" section updated. The checkboxes are now:

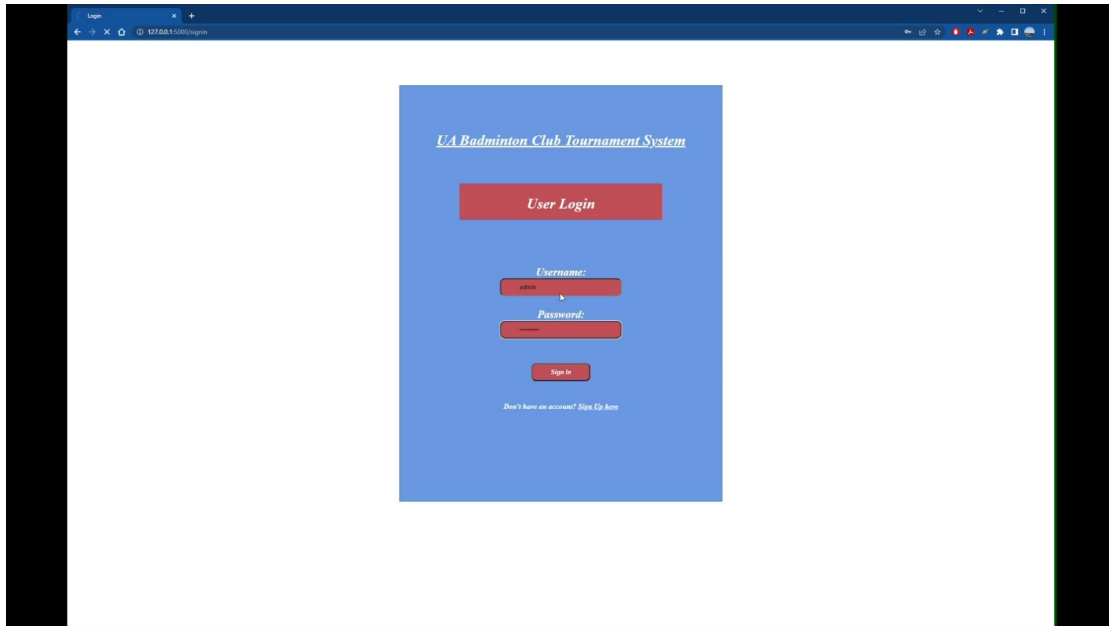
- ☒ XD Partner name
- ☐ MS Partner name
- ☒ WD Partner name
- ☒ U17 Partner name
- ☒ MS Partner name

The "Save" button remains at the bottom.

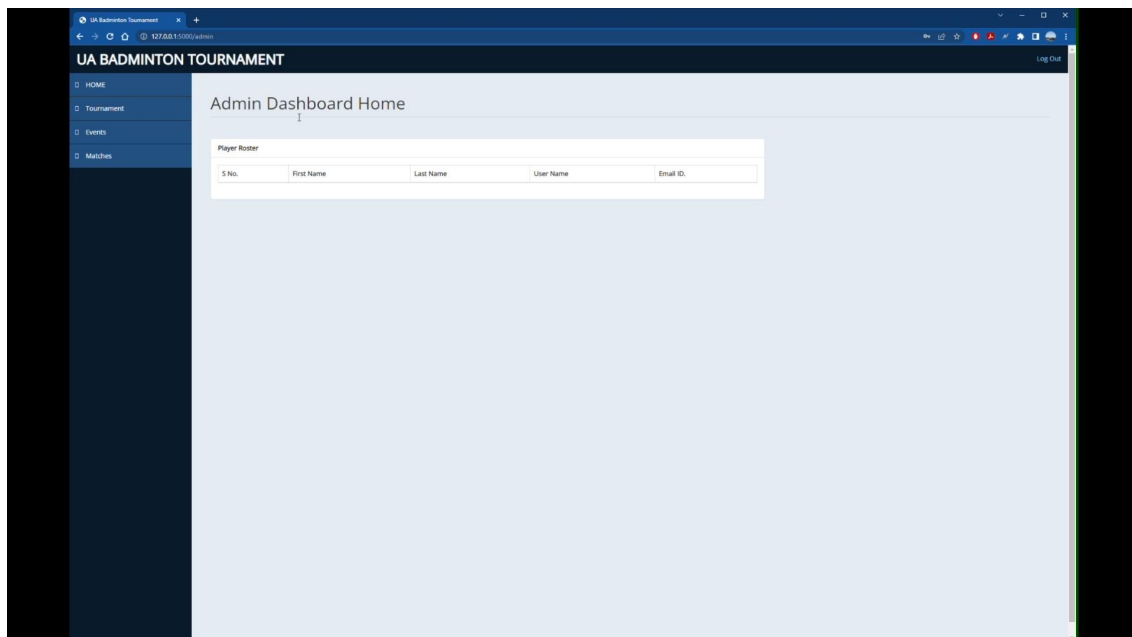
Player registers for events, and can also include their partner

Actor : Admin

1. Admin can Login



Screenshot of admin logging in



Admin successfully logged into their account and can now view their dashboard

2. Admin Create Tournament

The screenshot shows the 'UA BADMINTON TOURNAMENT' admin dashboard. The left sidebar contains links for HOME, Tournament, Events, and Matches. The main content area is titled 'Tournament' and contains a form with the following sections:

- Details:**
 - Name: 2023 Smashcat Tournament
 - Location: UA South Rec center
 - Date/Time:
 - Registration Open: 04/25/2023 11:26 PM
 - Registration Closed: 04/26/2023 11:26 PM
 - Tournament Start Date: 04/28/2023 11:26 PM
 - Tournament End Date: 04/29/2023 11:26 PM
- Events:**
 - ☒ Mens Single (MS) - Gender Allowed: M - Max Participants Allowed: 20
 - ☒ Mens Double (MD) - Gender Allowed: M - Max Participants Allowed: 20
 - ☒ Womens Single (WS) - Gender Allowed: F - Max Participants Allowed: 20
 - ☒ Womens Double (WD) - Gender Allowed: F - Max Participants Allowed: 20
 - ☒ Mix Doubles (XD) - Gender Allowed: M,F - Max Participants Allowed: 20
 - ☐ Under 19 (U19) - Gender Allowed: - Max Participants Allowed: -
 - ☒ Under 17 (U17) - Gender Allowed: M,F - Max Participants Allowed: 20
- Announcements:**
 - This is the UA badminton tournament system. Please sign up soon!
 - Buttons: Submit, Reset

Admin creates a tournament via dashboard

The screenshot shows the 'UA BADMINTON TOURNAMENT' admin dashboard with the 'Active Tournament Details' page. The left sidebar contains links for HOME, Tournament, Players, Events, Drives, and Matches. The main content area is titled 'Active Tournament Details' and contains the following sections:

- General Information:**
 - 2023 Smashcat Tournament
 - Date: 2023-04-28T00:00
 - Location: Tucson, USA
 - Venue: UA South Rec center
 - Contact: Julie Fan (juliefan@arizona.edu)
- Important Dates:**
 - Registration Open: 04/25/2023
 - Entry Deadline: 04/26/2023
 - Tournament: 04/28/2023
 - End of tournament: 04/29/2023
- Events:**
 - MS
 - MD
 - WS
 - WD
 - XD
 - U17
- Venue Direction:**
 - UA Sijuth Rec center
- Contact:**
 - UA Badminton Club Organization
 - Person of Contact: Julie Fan
 - Email: juliefan@arizona.edu
 - Phone: 520-123-3344

Tournament successfully created

3. Admin assigns seeding score

The screenshot shows a web application for the 'UA Badminton Tournament'. The browser address bar shows '127.0.0.1:5000/uaadmin/seed'. The interface is divided into three sections for different age groups: 'Under 19 (U19)', 'Under 17 (U17)', and 'Under 15 (U15)'. Each section has a table with columns for '#', 'Player 1', and 'Seed'. The 'Under 19 (U19)' section has 8 rows with players like 'Picaro Leo', 'Urnika Gola', 'Jugal Patel', 'John Roche', 'Amy Fan', 'Chinnai Bhowanj', 'Yuangma Hwe', and 'Kata Prince'. The 'Under 17 (U17)' section has 6 rows with players like 'Emma Fluer', 'Rupal Jan', 'Enfa Rose', 'Jimmy Yao', 'Erik Peterson', and 'Amy Fan'. The 'Under 15 (U15)' section is currently empty. Each section has a 'Save' button at the bottom.

#	Player 1	Seed	
4	Picaro Leo	Chinnai Bhowanj	4
5	Urnika Gola	Yuangma Hwe	5
6	Jugal Patel	Kata Prince	6
7	John Roche	Julie Fan	7
8	Amy Fan	Albert Dennis	8

Under 19 (U19)

#	Player 1	Seed
---	----------	------

Under 17 (U17)

#	Player 1	Seed
1	Emma Fluer	1
2	Rupal Jan	2
3	Enfa Rose	3
4	Jimmy Yao	4
5	Erik Peterson	5
6	Amy Fan	6

Under 15 (U15)

#	Player 1	Seed
---	----------	------

Admin can view all players and assign them a seed score

4. Admin can create draws automatically

The screenshot shows the 'UA BADMINTON TOURNAMENT' admin interface. The left sidebar has a menu with 'HOME', 'Tournament', 'Players', 'Events', 'Draws', and 'Matches'. The 'Draws' section is selected, showing a 'Draws' page. The page displays three sections: 'MS' (Main Singles), 'MD' (Main Doubles), and 'WS' (Women's Singles). Each section shows a list of players and their seed numbers, with lines indicating the draw structure. The 'MS' section has 8 players, 'MD' has 4 players, and 'WS' has 6 players. The draw structure is automatically generated based on the seed values.

UA BADMINTON TOURNAMENT

Draws

MS

1	Jurifeng Su
2	Albert Dennis
3	Chinnai Bhowanj
4	John Roche
5	Jugal Patel
6	Yuangma Hwe
7	Erik Peterson
8	Jimmy Yao

MD

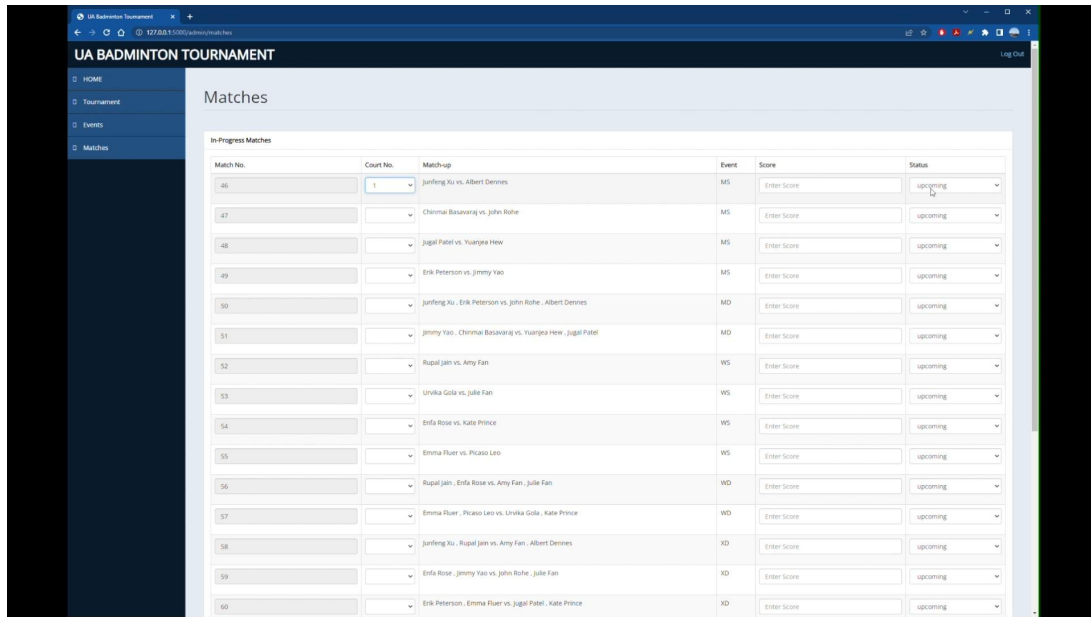
1	Jurifeng Su - Erik Peterson
2	John Roche - Albert Dennis
3	Jimmy Yao - Chinnai Bhowanj
4	Yuangma Hwe - Jugal Patel

WS

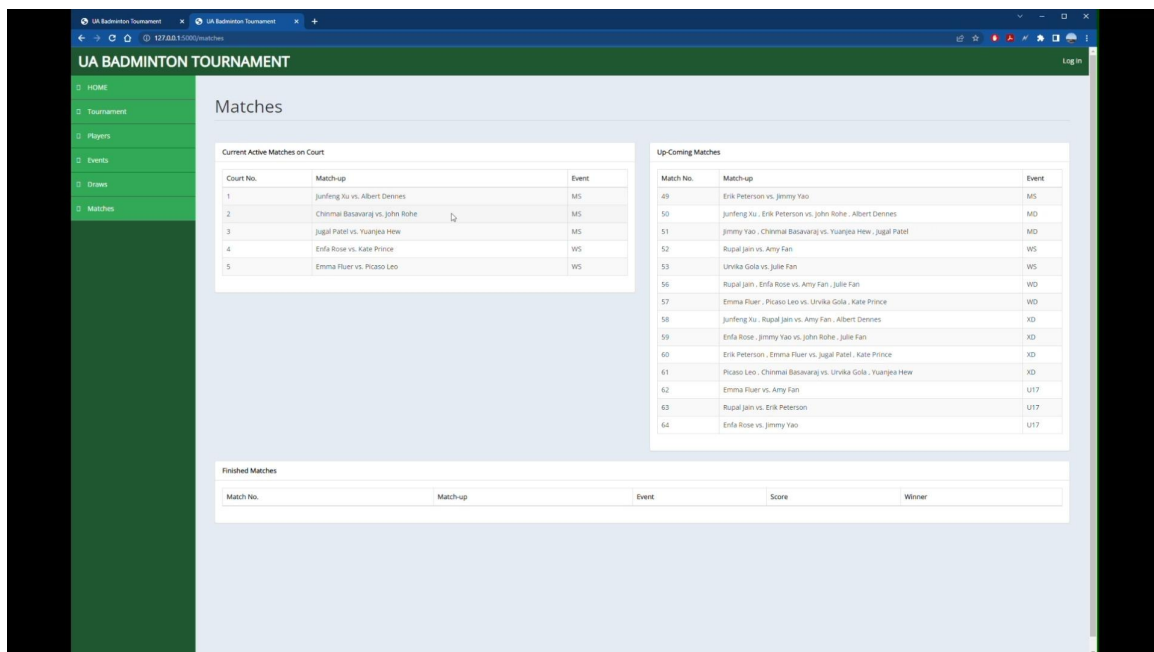
1	Rupal Jan
2	Amy Fan
3	Urnika Gola
4	Julie Fan
5	Enfa Rose
6	Kata Prince

Given the seed values, the system automatically generated draw for all matches

5. Admin can update match info



Admin can view all matches auto generated, and can add/edit details



Details are reflected on the website

6 . Admin can update match results

UA BADMINTON TOURNAMENT

Matches

In-Progress Matches

Match No.	Court No.	Match-up	Event	Score	Status
46	1	Junfeng Xu vs. Albert Dennes	MS	21-14	finished
47	2	Chinnai Basavaraj vs. John Rohe	MS	15-21	finished
48	3	Jugal Patel vs. Yungea Hew	MS	Enter Score	in progress
54	4	Enfa Rose vs. Kate Prince	WS	Enter Score	in progress
55	5	Emma Fluor vs. Picaso Leo	WS	Enter Score	in progress
49		Erik Peterson vs. Jimmy Yao	MS	Enter Score	upcoming
50		Junfeng Xu , Erik Peterson vs. John Rohe , Albert Dennes	MD	Enter Score	upcoming
51		Jimmy Yao , Chinnai Basavaraj vs. Yungea Hew , Jugal Patel	MD	Enter Score	upcoming
52		Rupal Jain vs. Amy Fan	WS	Enter Score	upcoming
53		Unvika Gola vs. Julie Fan	WS	Enter Score	upcoming
56		Rupal Jain , Enfa Rose vs. Amy Fan , Julie Fan	WD	Enter Score	upcoming
57		Emma Fluor , Picaso Leo vs. Unvika Gola , Kate Prince	WD	Enter Score	upcoming
58		Junfeng Xu , Rupal Jain vs. Amy Fan , Albert Dennes	XD	Enter Score	upcoming
59		Enfa Rose , Jimmy Yao vs. John Rohe , Julie Fan	XD	Enter Score	upcoming
60		Erik Peterson , Emma Fluor vs. Jugal Patel , Kate Prince	XD	Enter Score	upcoming

Admin updates the final match score on the dashboard

UA BADMINTON TOURNAMENT

Matches

In-Progress Matches

Match No.	Court No.	Match-up	Event	Score	Status
51		Jimmy Yao , Chinnai Basavaraj vs. Yungea Hew , Jugal Patel	MD	Enter Score	upcoming
52		Rupal Jain vs. Amy Fan	WS	Enter Score	upcoming
53		Unvika Gola vs. Julie Fan	WS	Enter Score	upcoming
56		Rupal Jain , Enfa Rose vs. Amy Fan , Julie Fan	WD	Enter Score	upcoming
57		Emma Fluor , Picaso Leo vs. Unvika Gola , Kate Prince	WD	Enter Score	upcoming
58		Junfeng Xu , Rupal Jain vs. Amy Fan , Albert Dennes	XD	Enter Score	upcoming
59		Enfa Rose , Jimmy Yao vs. John Rohe , Julie Fan	XD	Enter Score	upcoming
60		Erik Peterson , Emma Fluor vs. Jugal Patel , Kate Prince	XD	Enter Score	upcoming
61		Picaso Leo , Chinnai Basavaraj vs. Unvika Gola , Yungea Hew	XD	Enter Score	upcoming
62		Emma Fluor vs. Amy Fan	U17	Enter Score	upcoming
63		Rupal Jain vs. Erik Peterson	U17	Enter Score	upcoming
64		Enfa Rose vs. Jimmy Yao	U17	Enter Score	upcoming

Save

Finished Matches

Match No.	Match-up	Event	Score	Winner
46	Junfeng Xu vs. Albert Dennes	MS	21-14	Junfeng Xu
47	Chinnai Basavaraj vs. John Rohe	MS	15-21	John Rohe

Save

Admin can view finished matches and mark the winner

UA BADMINTON TOURNAMENT

Log in

Matches

Current Active Matches on Court

Court No.	Match-up	Event
3	Jugal Patel vs. Yuangsa Hiew	MS
4	Enfa Rose vs. Kate Prince	WS
5	Emma Ruer vs. Pico Leo	WS

Up-Coming Matches

Match No.	Match-up	Event
49	Erik Peterson vs. Jimmy Yao	MS
50	Junfeng Xu, Erik Peterson vs. John Rohe, Albert Dennes	MD
51	Jimmy Yao, Chinnai Basavaraj vs. Yuangsa Hiew, Jugal Patel	MD
52	Rupal Jain vs. Amy Fan	WS
53	Unvika Gola vs. Julie Fan	WS
56	Rupal Jain, Enfa Rose vs. Amy Fan, Julie Fan	WD
57	Emma Ruer, Pico Leo vs. Unvika Gola, Kate Prince	WD
58	Junfeng Xu, Rupal Jain vs. Amy Fan, Albert Dennes	XD
59	Enfa Rose, Jimmy Yao vs. John Rohe, Julie Fan	XD
60	Erik Peterson, Emma Ruer vs. Jugal Patel, Kate Prince	XD
61	Pico Leo, Chinnai Basavaraj vs. Unvika Gola, Yuangsa Hiew	XD
62	Emma Ruer vs. Amy Fan	U17
63	Rupal Jain vs. Erik Peterson	U17
64	Enfa Rose vs. Jimmy Yao	U17

Finished Matches

Match No.	Match-up	Event	Score	Winner
46	Junfeng Xu vs. Albert Dennes	MS	21-14	Junfeng Xu
47	Chinnai Basavaraj vs. John Rohe	MS	15-21	John Rohe

Results reflected on the website

Actor : Public

May include some screenshots from earlier where it was displayed to show a successful completion of a task, but shown here to demonstrate all the views for the public.

1. Anyone can view tournament details including events

UA BADMINTON TOURNAMENT

Log in

Active Tournament Details

General Information

2023 Smashat Tournament
 Date: 2023-04-28/05-05
 Location: Tucson, USA
 Venue: UA South Rec Center
 Contact: Jule Fan (julefan@uadsmashat.com)

Important Dates

Registration Open: 04/29/2023
 Entry Deadline: 04/29/2023
 Tournament: 04/30/2023
 End of Tournament: 04/29/2023

Venue Location

UA South Rec Center

Events

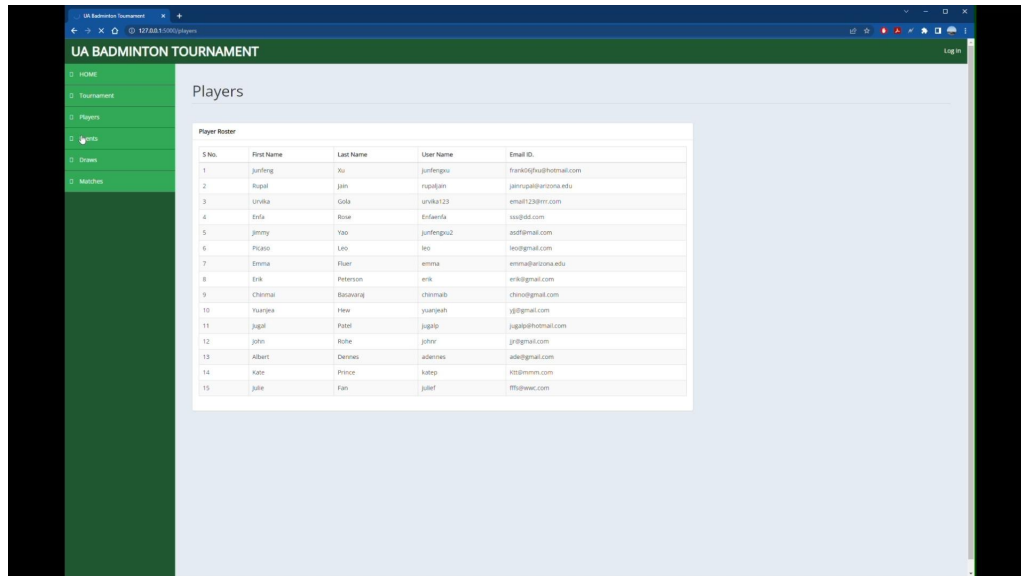
MS
 MD
 WS
 WD
 XD
 U17

Contact

UA Badminton Club Organization
 Person of Contact: Jule Fan
 Email: julefan@uadsmashat.com
 Phone: 520-123-3344

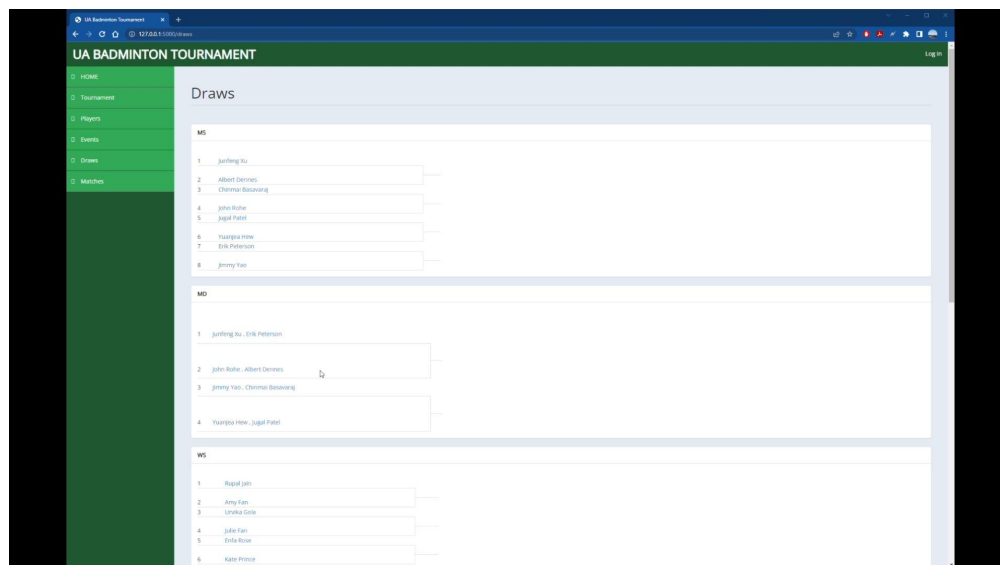
Tournament details is up for public display

2. Anyone can view all registered players



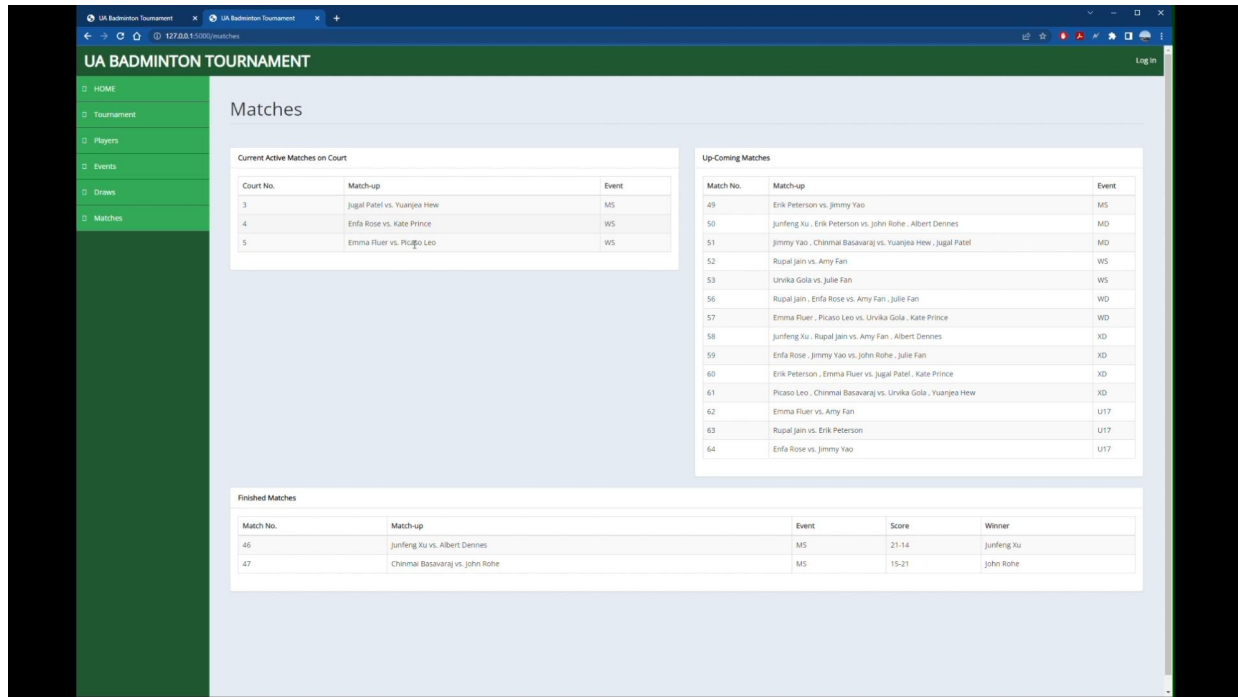
Anyone can view all the registered players for the tournament, and their username as each player register one by one

3. Anyone can see initial draw for all events



Anyone can view the draws for every event

4. Anyone can see match info and results updated live



Anyone can view all matches results live on the website

Tests Performed and Adequacy

1. **Manually functional testing** with dummy values on a local server.
2. **Unit testing** for front end UI receiving data from or sending data to the back end controller. We have performed unit testing on all the front end user interfaces we have implemented so far which consist of HTML and CSS code. Scaffolding was used in order to invoke the front end code since the backend support is not completed yet.

3. **Unit testing using the Pytest framework.** An example: Added a test to add a permission object to the permission table in the database and verify that it was successfully committed to the database using the app context and database session. This test helps ensure that our app's database functionality is working as expected and can handle adding new data to the database.
4. **Function verification testing** for each functionality were implemented.

Adequacy

We covered all tables in the database and had manual tests for other functions. We estimate at least 70% of the code base to be covered in the testing.

Code Contribution

The final repository has 12259 lines of code, of which HTML was 59.0%, Python was 31.8% and CSS was 9.2%. But on the whole, combining all the lines of code written by each contributor during the development is 39,780

6. Project Management

Development process:

The development process followed for our software engineering project was a combination of Scrum, Agile practices, and Extreme Programming. This approach allowed us to effectively manage our workflow, prioritize tasks, and adapt to changes as needed.

Tools and Methodologies

1. Jira:

We utilized Jira as a project management tool to plan our sprints in accordance with Agile and Extreme programming methodologies. Jira enabled us to prioritize user stories, track progress, and manage time efficiently to ensure we stayed on track towards achieving our sprint goals.

Backlog

Search backlog

UG JX RJ

Only My Issues

Recently Updated

UBT Sprint 1 4 issues

Project Design and Setup Signup and Sign In Player dashboard and admin dashboard front-end interface.
20/Feb/23 7:10 PM • 15/Mar/23 7:10 PM

View linked pages

JX RJ UG

Signup and Sign In

None

UBT-1

Project Design and Setup

None

UG UBT-11

Player Dashboard front end

None

JX UBT-27

Admin Dashboard front end

None

RJ UBT-26

Backlog 6 issues

Create sprint

Home Page

None

UBT-24

Player seeding

None

UBT-2

Draw making

None

UBT-3

Player profile creation

None

UBT-4

UBT-11

Project Design and Setup

Description

Add a description...

Subtasks

88% Done

UBT-12 What are the interfaces n...

JX DONE

UBT-13 Interfaces Design

RJ DONE

UBT-14 Database Design

EG DONE

UBT-15 Create a da...

UG IN PROGRESS

UBT-16 Test Project is Loading - ...

JX DONE

UBT-17 Test Project is Loa...

UG DONE

UBT-18 Test Project is Loading - ...

RJ DONE

UBT-19 Test Project is Loading - ...

EG DONE

UBT-20 Create an ER Diag...

UG DONE

UBT Sprint 3

Search this board

UG EG JX RJ

Only My Issues

Recently Updated

TO DO

IN PROGRESS

DONE

UBT-86 DONE 4 sub-tasks Adding db creation and make sure the current main branch code works

Foreign key relationship for Player with match and event, when there there is no tournament

UBT-100 UG

Bug fix user sign up

UBT-102 EG

Drop db and ensure new db is created properly according to new structure

UBT-103 EG

fix duplicate db initialization

UBT-104 EG

Other Issues 15 issues

Admin dashboard backend SQL for home/players page

2

UBT-85 EG

Player dashboard event registration

2

UBT-87 RJ

Tournament creation on admin dashboard.

3

UBT-88 RJ

Store seeding information of a player for different events.

2

UBT-89 UG

Populating the types of events to register for on the player dashboard once tournament

2. Confluence:

We created a Daily Dashboard in Confluence to track progress, monitor team performance, and identify potential roadblocks. This platform allowed for efficient documentation and collaboration among team members.

 Tuesday <April 18> (Sprint 3 Scrum 2)

Name	What did I work on yesterday 🤖
@Urvika Gola	<ul style="list-style-type: none">Participated in 1:1 meeting with Junfeng on UBT-89: Store seeding information of a player for different events. DONEImplemented EventPlayerSeed Table ua-badminton-tournament: Seeding table CLOSEDClosed bug UBT-100: Foreign key relationship for Player with match and event, when there there is no tournament DONEAdded announcement column. UBT-100: Foreign key relationship for Player with match and event, when there there is no tournament DONE UBT-98: Adding Announcement column in the tournament table DONE
@Enfa Rose George	<ul style="list-style-type: none">Player roster for admin and public.<ul style="list-style-type: none">Public - ua-badminton-tournament: Public player roster CLOSEDAdmin - ua-badminton-tournament: add : admin dashboard view player roster CLOSEDCreating draws
@Junfeng Xu	<ol style="list-style-type: none">Admin dashboard event page, grab data from event table.Held 1:1 meeting with Urvika on UBT-89: Store seeding information of a player for different events. DONE
@Rupal Jain	<ol style="list-style-type: none">Tournament Creation Admin Dashboard: grab the information from the user and pass it to the backend. Populate the required tables and return the information to display the information again on frontendTournament Updation on Admin Dashboard. ua-badminton-tournament: Tournament Creation and Updation on Admin Dashboard CLOSED

Monday <April 24> (Sprint 3 Scrum 3)

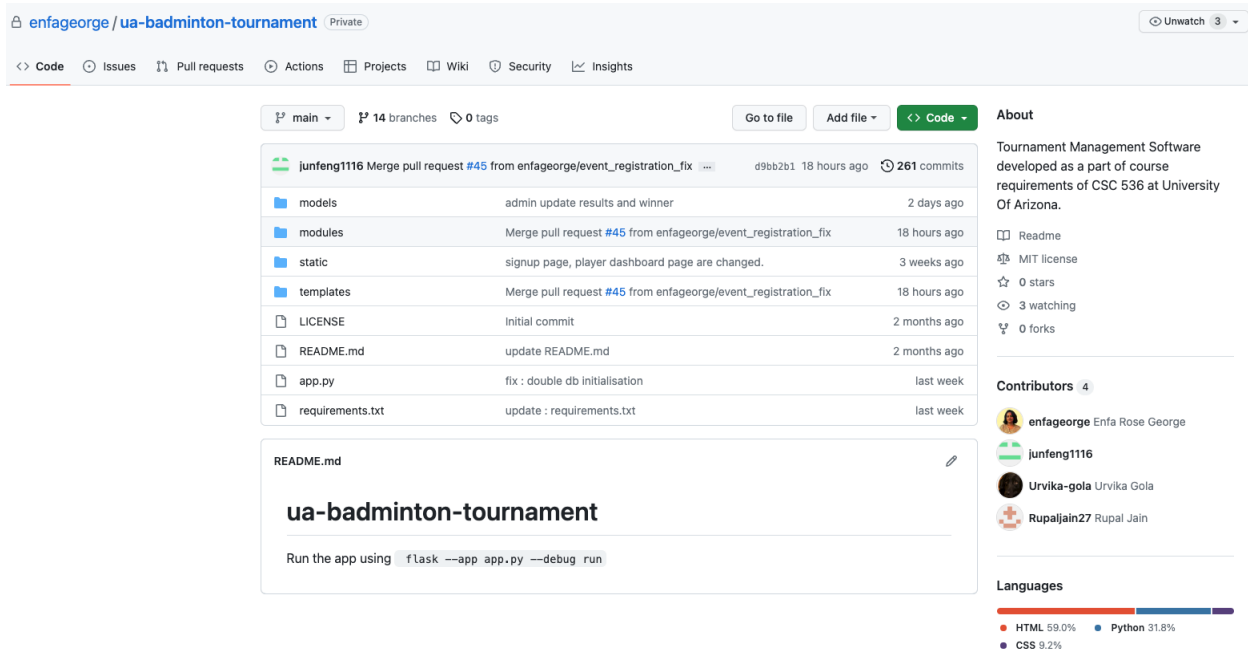
Name	What did I work on yesterday 🤖
@Urvika Gola	<p>1. Completed Admin dashboard match page, match table and result table get updated. https://ugola.atlassian.net/browse/UBT-93</p> <p>ua-badminton-tournament: admin update results and winner CLOSED</p> <p>ua-badminton-tournament: Update matches page CLOSED</p>
@Enfa Rose George	<p>1. Completed initial draw making functionality, make draws and update the match table. UBT-93: Match page on admin dashboard DONE</p> <p>2. Bug fix, user sign up ua-badminton-tournament: Bug fix user signup CLOSED</p>
@Junfeng Xu	<p>1. Bugs fixed for Demo and Demo Preparation ua-badminton-tournament: bugs fixed CLOSED</p> <p>2. Completed public match page ua-badminton-tournament: Public match page CLOSED ua-badminton-tournament: Public and Admin event pages, Public Match page CLOSED</p> <p>3. Helped Urvika to understand frontend to middleware.</p>
@Rupal Jain	<p>1. Public home page and Tournament description #46 ua-badminton-tournament: Public home page and Tournament description CLOSED</p> <p>2. Bug fixes ua-badminton-tournament: Event registration fix CLOSED</p>

Tuesday <April 4> (Sprint 2 Scrum 3)

1	Name	What did I work on yesterday 🤖	What am I working on today? 🤖	What issues are blocking me? 🤖
2	@Urvika Gola	<ul style="list-style-type: none"> Resolved EC2 DB issue by terminating the old instance, starting afresh with Amazon DBS with PostgreSQL and connecting this to EC2 instance. UBT-60: Create Amazon DBS with PostgreSQL and connect it to EC2 DONE Configured for public access. Checked public access thorough my local machine. UBT-59: Install and Run PostgreSQL on Mac (localhost) DONE 	<ul style="list-style-type: none"> connected psq on Amazon DBS with SQL Alchemy ua-badminton-tournament: connected psq on Amazon DBS with SQL Alchemy CLOSED https://ugola.atlassian.net/browse/UBT-60 Report and PPT discussion, I am finishing the report and presentation for Status report 2. Reviewed Rupals PR ua-badminton-tournament: Added backend code for Admin Dashboard CLOSED 	
3	@Enfa Rose George	<ul style="list-style-type: none"> Connected the PostgreSQL db set up by Urvika to the project via SQL alchemy so that db object is available throughout the app. Inserted init table value needs such as admin login via psq Wrote db operations for signup, loading player dashboard details Ensured that dashboards cannot be accessed by anyone outside of the soleowner using python decorator. Reviewed PR by team mates and recommended corrections/best practises before PR merge 	<ul style="list-style-type: none"> Wrote db operations for signup, and updating almost all player details Ensured that dashboards cannot be accessed by anyone outside of the soleowner using python decorator. Implemented the logout functionality Reviewed PR by team mates and recommended corrections/best practises before PR merge 	<ul style="list-style-type: none"> [SOLVED] Signup sheet wasn't collecting all details needed to create a user in db. This was due to communication gap and was solved in the meeting.
4	@Junfeng Xu	<ul style="list-style-type: none"> I completed integrating the user login and sign up into the home page. ua-badminton-tournament: Homepage sign in CLOSED I completed implementing the following functionalities to the player dashboard front end UI <ul style="list-style-type: none"> reading backend variables and displaying them on the front end UI for player dashboard. The inputs to the player dashboard front end UI are also able to be passed to backend controller UBT-63: Backend: Player dashboard receiving data from frontend DONE 	<ul style="list-style-type: none"> Working on status update 2 Created Architectural Diagrams organizing how to split the presentation and report to each team member preparing on my assignment of the presentation writing my assigned sections of the report Incorporated new changes by middleware: ua-badminton-tournament: signup page, player dashboard page are changed. CLOSED 	
5	@Rupal Jain	<p>I was working on the Admin dashboard Events page so that the events page should receive data from backend. UBT-66: Admin Dashboard Tournament C reation Button connection to backend DONE</p>	<p>1. I am finishing the report and presentation for Status report 2.</p> <p>2. I will be concluding the backend connection for all the pages of Admin Dashboard</p>	<p>I have to reformat the data received from backend in necessary format readable to HTML. ua-badminton-tournament: Added connection for Admin dashboard for pages: Home, Tournament, Matches, Events CLOSED</p>

3. Github:

We used Github for version control and managing releases. This allowed us to maintain a clear history of code changes, facilitate collaboration, and streamline the deployment process.



Sprint Planning and Execution

Before sprint planning, we made sure that the product backlog was well-defined, containing a prioritized list of user stories. The priorities corresponding to each user story were highlighted in a red box.

During sprint planning, we focused on three critical areas:

Sprint objectives: It is critical to get the team to think about the heart of the sprint and what value we will produce for our clients once the sprint is completed.

Sprint tasks: Following that, we defined the exact activities and user stories to be selected from the product backlog to assist in meeting the Sprint Goal.

Sprint execution: We planned how the tasks would be completed by breaking them down into subtasks. and who would be in charge of each

Change Log contains the Major events during the project

Date	Description	Motivation	Implementation
03/02/2023	UI Design Change	To make the admin dashboard more user-friendly and easier to update.	Updated UI design to have an admin dashboard in a single screen instead of multiple screens. This will make it easy for the Admin to make updates.
03/07/2023	Database redesign	To fix errors with primitive data types in initial design.	Reworked the structure of the database as previously we found some design error, and created a new ER diagram.

03/12/2023	Customer desired over all experience change	To satisfy the customer's updated requirements that we have learned from the customer's feedback.	Given our current design for the project, it is still able to address customer's new surfaced problems.
------------	---	---	---

Anything else?

By following a development process that combined Scrum, Agile practices, and Extreme Programming, our team was able to manage tasks efficiently, adapt to changing requirements, and collaborate effectively. The use of Jira, Confluence, and Github as key tools further supported our efforts in delivering a high-quality software project. We also incorporated 1, 2, 3 Story Points in our Jira dashboard after learning about story points in class.

Team Coordination

When and how often did the team meet?

Throughout the development period, our team met on a frequent basis to ensure beneficial communication and collaboration. We conducted three sprints in all, with daily stand-up meetings held either in person or online. When making significant choices, we chose to meet in person, for example our initial meeting, in accordance with Extreme Programming ideals.

In addition to our daily stand-up meetings, we also practiced a few sessions of pair programming online. This method involves one developer producing code while

another reviewed it for faults and brainstormed programming techniques, boosting knowledge sharing and better code quality.

How else did you communicate?

Apart from status update meetings, our team used a the following of communication channels to stay connected and share information, such as:

1. Google Meet: For status update video conferencing and customer meetings
2. WhatsApp Group: For quick and instant responses
3. Face-to-face: for in-person meetings to handle complex topics or make important decisions.

What did you accomplish during the meetings?

We kicked off the meeting by discussing the following main areas:

What did I do yesterday: Each team member communicated their day's progress and accomplishments, allowing the team to stay informed of individual work.

What do I plan to do next: Team members discussed their plans for the following day, promoting goal alignment and work prioritizing.

Any issues or roadblocks: Any issues or barriers encountered were acknowledged, allowing the team to explore solutions and offer help to one another, establishing a culture of collaboration and information sharing.

7. Team

Backgrounds

Junfeng Xu is a Master's student in the computer science program. He has some work experience in software engineering. He is proficient in C and assembly. He had not built anything like this project before.

Enfa George is a Master's student in the computer science program. They have worked in various internships and full-time roles in both startup and corporate environments where they gained expertise in Data Analytics and Research. Prior to this, they briefly worked as a backend developer too. They have built applications using python and flask in hackathon-like environments.

Rupal Jain is a Master's student in the computer science program. She has 3 years of experience in software development. She has worked on supporting and introducing new features for an application used for the hiring process. She is good at programming languages like Python, JavaScript and HTML/CSS. The framework used in this project was new to her.

Urvika Gola is a graduate student in computer science pursuing a Masters degree. She has 4 years of industry experience as a Software Engineer, including a leadership role managing a team of two at Intel. Urvika's current position as a graduate research assistant has provided her with exposure to cloud technologies such as Docker and Containers. She is familiar with Python.

Roles

During this iteration, our team had a well-defined set of roles and responsibilities. **Junfeng Xu** served as the Product Owner and contributed to the designing and development of the UI. **Urvika Gola** acted as the Scrum Master, she managed Jira dashboard and confluence pages. Urvika Gola designed the postgresql database and deployed it on Amazon RDS and Amazon EC2. **Rupal Jain** was one of the developers and contributed to the design and development of the system's UI. **Enfa Rose George** served as the Software Architect and backend engineer, and she was instrumental in designing the structure of the system.

Team Member Contributions:

1. **Junfeng** served as our Product Owner and played a crucial role in designing the overall high-level architecture based on customer needs. He also communicated with the customer to collect feedback, designed the user interface, and performed code reviews on both the code and design aspects. Junfeng also contributed to the implementation of the public view front-end interface web pages and the admin dashboard back-end supports and made changes to the player dashboard front-end interface. Junfeng was also responsible for performing code reviews for frontend and middleware related pull requests.
2. **Urvika** served as the Scrum Master and was responsible for project management. She contributed to the database design, creating an ER diagram that represented the new entities, relationships, and constraints of the database.

Urvika also worked on configuring the sprint on the Jira board and maintaining the daily dashboard. Additionally, she acted as the backend developer and designed, created, and configured the entire database on the cloud using Amazon RDS. She installed, configured PostgreSQL on Amazon RDS, and connected them to Amazon EC2. This saved the team significant time to setup the database locally. Urvika also worked on the front end and middleware components for the Admin's matches page. Using this, admin can update the match's status (upcoming, in-progress, or completed), update scores, and update the match's result by assigning the winning team. The data was synchronized with the database tables (matches and results). Urvika was also responsible for performing code reviews for backend-related pull requests.

3. **Rupal** served as one of the developers, she was responsible for designing the front end of the complete system. She began by brainstorming the overall flow of the application, starting from tournament creation to player signup, registering for events, making draws, and concluding the tournament. Rupal also contributed to finalizing the UI design, including the color scheme, tables, tabs, and other elements. She developed the Admin Dashboard, which included sub-pages such as tournament creation, displaying events, and matches under each event, as well as tracking in-progress and finished matches. Rupal was also responsible for connecting the backend to the front-end page and populating the dummy information structure. This includes completing the development of intermediate between Python and HTML.
4. **Enfa** played a crucial role in setting up the basic project in GitHub with instructions so that everyone could start off the project. She extensively brainstormed with Urvika about database design, and together they built the first

iteration of the database design. She set up the first database integration using the tables Urvika created. She also wrote the first scripts that used data from the database, and performed read and write operations by building the the signin and signup functionalities for users and admin, and displaying the player roster. She also built the functionality that automatically creates draws and matches according to customer specification, from a list of players and their seed score. She also created safeguards to ensure that unauthorized operations, such as unauthorized access to the admin dashboard was not possible.

Together, our team worked cohesively to produce a high-quality product that meets customer requirements.

Using the Element Catalogs to identify individual contributions.

Admin Dashboard Interface (Owner: Rupal): The tournament organizer may edit and publish a tournament, edit player information, assign seeding to event entries, make draws for each event, and operate with the matches on the tournament day.

Player Dashboard Interface (Owner: Junfeng, Rupal): This provides the interactive web page for the tournament player users. A player may edit their profile information and register for the events they want to participate in for the current active tournament.

Public View Interface (Owner: Junfeng): View all the tournament related information including tournament details, announcements, players, event brackets, match progress, and much more.

Presentation Layer (Owner: Enfa): This takes care of the rendering for all the web page interfaces.

Application Logic (Owner: Enfa): This includes Communication between the Presentation Layer and the External Service, requesting the information from the External Service (database) and sending to the Presentation Layer to display, storing the information passed in from the Presentation Layer and storing to the External Service (database).

External Services (Owner: Urvika): This provides the support for setting up the external database for our application, deploying the database on a distributed server, creating all the database tables.

Database (Owner: Urvika): This provides the storage space on Amazon RDS and Amazon EC2 for our application needs including admin profiles, player profiles, login details, permissions assigned, tournament details, events supported in the tournament, draws, matches and results stored in a postgresSQL database.

Each team member contributions to the project

Based on our team's estimates, it appears that each team member made significant contributions to the application's success. Rough estimates of the percentage contributions indicate that all team members contributed equally, with each member accounting for 25% of the project's overall progress. This demonstrates the collaborative and cooperative nature of our team, and the commitment of each member to ensuring the project's success. By sharing ideas, supporting one another, and working together, we were able to accomplish our goals and deliver a high-quality outcome of the sprint.

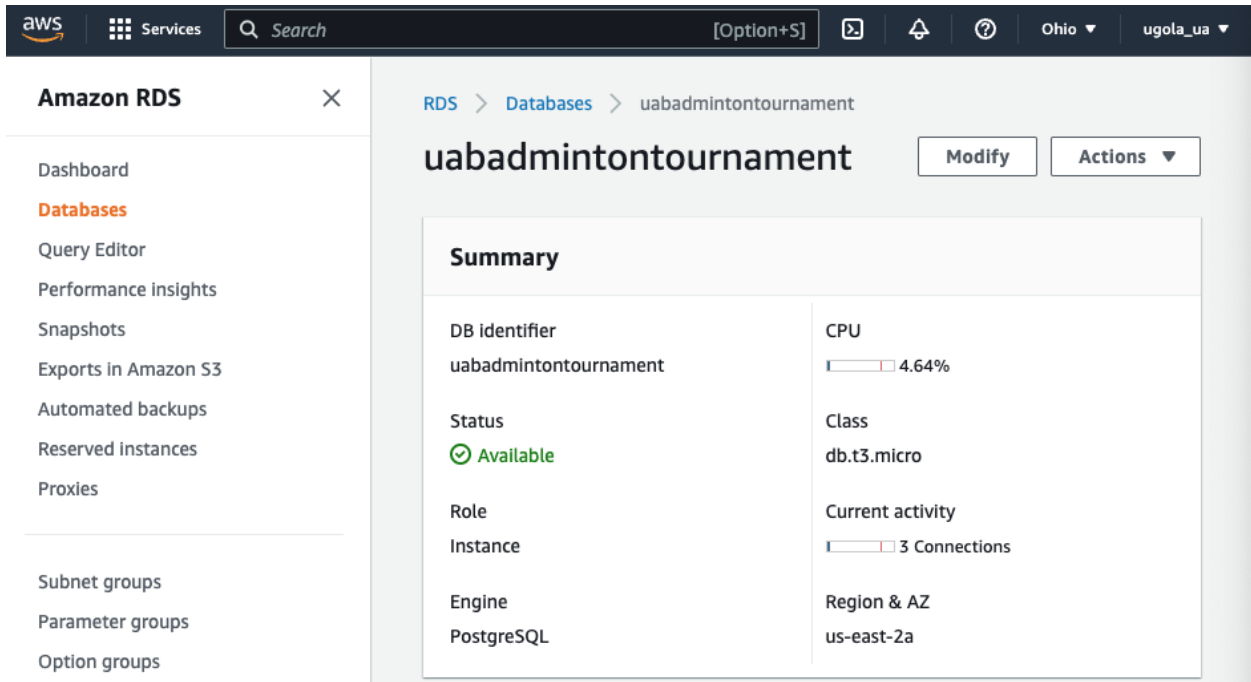
8. Constraints and Risks

Constraints

1. **Ethical constraints:** The tool prioritizes user privacy, data security, and safety. The tool collects participant Personal Identification Details, store, and use it in a responsible manner that is compliant with privacy laws and regulations. The tool is designed with security features in place to prevent unauthorized access, hacking, or any other security breach.
2. **Policy constraints:** The tool complies with all relevant policies and regulations related to data protection such as GDPR.
3. **Legal constraints:** The tool also complies with legal requirements, such as accessibility laws. The tool does not infringe on any patents, trademarks, copyrights, or other proprietary rights.

Access to the data, services, and resources needed

We are storing data on the cloud using Amazon RDS with configured PostgreSQL and connected to Amazon EC2.



The screenshot shows the Amazon RDS console interface. On the left is a navigation menu with options like Dashboard, Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, and Option groups. The main panel displays the details for the database instance 'uabadmintournament'. It includes a 'Summary' section with the following information:

Property	Value
DB Identifier	uabadmintournament
CPU	4.64%
Status	Available
Class	db.t3.micro
Role	
Instance	
Current activity	3 Connections
Engine	PostgreSQL
Region & AZ	us-east-2a

```
uabadmintournamentdb-> \dt+
```

Schema	Name	Type	Owner	Persistence	Access method	Size
public	event	table	postgres	permanent	heap	0 bytes
public	event_player	table	postgres	permanent	heap	0 bytes
public	login	table	postgres	permanent	heap	8192 bytes
public	match	table	postgres	permanent	heap	0 bytes
public	permission	table	postgres	permanent	heap	8192 bytes
public	player	table	postgres	permanent	heap	8192 bytes
public	result	table	postgres	permanent	heap	0 bytes
public	tournament	table	postgres	permanent	heap	0 bytes
public	user	table	postgres	permanent	heap	0 bytes
public	user_permission	table	postgres	permanent	heap	0 bytes

Anything else needed

No, there are no additional requirements or needs at this time.

9. Reflection

Lessons learned from this project

1. **Clear communication and collaboration are essential for project success:**

We discovered that effective communication among team members, stakeholders, and clients is crucial for understanding project requirements, resolving issues, and ensuring everyone is on the same page.

2. **Thorough planning and requirement analysis help in avoiding scope creep:**

We learned that investing time and effort in detailed planning and requirement analysis upfront can prevent scope creep and ensure a more streamlined development process.

3. **Regular testing and bug fixing contribute to a more stable product:**

We realized the importance of continuous testing and bug fixing throughout the development cycle to identify and address issues early on, resulting in a more stable and reliable product.

4. **Proper documentation aids in understanding and maintaining the**

codebase: We found that maintaining comprehensive documentation, including code documentation and project requirements, facilitates better understanding, smoother collaboration, and easier maintenance of the codebase.

5. **Comprehensive database design is essential for data management:**

We realized the significance of completing the database design phase thoroughly to ensure efficient data storage, retrieval, and manipulation within the project.

6. **Well-designed front-end user interfaces and back-end controllers**

contribute to a seamless user experience: The completion of the design

phase for front-end user interfaces and back-end controllers enabled us to develop a user-friendly system that aligns with user expectations and requirements.

What went well?

1. **Successful setup of the project on project management tools:** We accomplished setting up the project on various project management tools, including Jira, Atlassian, and Github, allowing us to effectively manage and track project progress throughout the iteration.
2. **Completion of database design:** We successfully completed the design phase for the database, ensuring a solid foundation for data management within the project.
3. **Implementation of user sign-in and sign-up functions:** The implementation of the user sign-in and sign-up functions, including the user interface and back-end controller portions, was completed successfully, providing users with seamless authentication capabilities.
4. **Development of player dashboard and admin dashboard:** We successfully designed and implemented the front-end user interfaces for both the player dashboard and admin dashboard, enabling users to access and interact with the respective functionalities.
5. **Gathering customer feedback and adding it to the backlog:** We conducted a meeting with the customer at the end of the iteration, collected valuable feedback, and added the feedback items to the backlog for future implementation and improvement.

Best practices

1. **Thorough testing procedures:** We followed comprehensive testing practices, including functional testing, unit testing, and user testing, to ensure the quality and reliability of the implemented functionalities.
2. **Regular customer review meetings:** Conducting customer review meetings allowed us to showcase our progress, gather feedback, and align the project with customer expectations, leading to improved collaboration and customer satisfaction.

What didn't go well?

One of the challenges encountered was the incomplete event registration function on the player dashboard. This incomplete feature had the potential to disrupt the seamless experience for players. It emphasized the need for improved task distribution and enhanced time management within the team to ensure timely completion of all planned functionalities.

What isn't working and how did you work around it?

Efficient backend support integration with front-end UI: By successfully integrating the backend support with the front-end user interfaces for user login/signup functions, player dashboard functionalities, and admin dashboard functionalities, we ensured the smooth operation of these features despite any potential challenges encountered during development.

For the features that were not implemented, what were the issues?

No specific issues were mentioned regarding unimplemented features, indicating successful completion of the planned functionalities within the respective iterations.

Recommendations

What would you do differently?

While we were able to achieve our goals for this sprint, we believe there is always room for improvement. We plan to implement more rigorous testing procedures to catch any issues before they get logged as bugs. Additionally, we will continue to look for ways to optimize the application's performance to manage heavy traffic, something that we expect during an ongoing tournament.

What advice do you have for other teams?

1. **Continuously prioritize and address the backlog:** To ensure a streamlined development process, we recommend regularly reviewing and prioritizing the backlog, addressing the customer's feedback, and incorporating necessary improvements and enhancements into future iterations.
2. **Maintain effective communication with the customer:** We would advise others to have genuine consumers and keep open lines of communication with them throughout the project since we understand the value of real customer input and feedback at regular intervals to align the project with their expectations and requirements.