CS482/495/496 Software Project Proposal: Football Team Management

Nyrique' Butler, Andrew Dominquez, Leslie Kim, Chloe Miranda, Jack Wehberg

2025-10-10

1 Client Information

By sharing this client information and the rest of this document, you are stating that this client has provided this project as something they want (not something you created and asked if they wanted), and that they are interested in having you complete this project for your capstone.

- Client name: Dr. Cui
- Client title: stockholder
- Client email address: lcui@loyola.edu
- Client employer: Kids' Activity Management Organization
- How you know the client: Our client, Dr. Cui, is acting as a stockholder for a teens soccer league.

2 Project Description

2.1 Overview

[Add a few paragraphs describing your project succinctly. What problem are you trying to solve, what is the purpose of your project? Why does your client want this project?]

Our client, Dr. Cui is a main stockholder of Kids' Activity Management Organization. He is requesting a web application specifically for a local soccer league to allow league administrators to manage seasons and for the adults registered in the league to have a place to view game schedules for the whole season. This will also be used as a platform for registered adults in the league to connect with each other online as a community.

2.2 Key Features

[At this point you should have a basic understanding of your client's needs. List out the key features of the software system the client wants you to build.]
Key features:

- Login
- calendar of matches that spans for the season
- match generator at start of season
- match manager as season progresses into playoffs

social media

- picture/video feed
- commenting on posts
- liking posts

2.3 Why this Project is Interesting

This app centralizes access to the league not only for registered parents but for prospective parents who are looking to enroll their kids in an activity. This app is also enticing because it uses database management and a social media style interface on its main page. For example, user accounts are stored in the database, the admin can update the scoreboard and playoff bracket, and registered adults can post comments, videos, or images on the feed.

Additionally, this project is meaningful because it makes it easier for parents and their kids to stay informed about matches, team performance, and schedule. It is also interactive for registered parents to share media and engage with others. The app supports the league community by providing an online space to connect and stay informed throughout the season.

2.4 Areas of CS required

The most relevant subfields are web development, database management, and security.

2.5 Potential Concerns and Questions

All of us have limited to no experience in web programming. We anticipate that implementing the back-end with MongoDB and Node.js will be the most difficult part. So we might have questions on how to set up the brackets or the scoreboard later on in this project.

3 Requirements

3.1 Non-Functional Requirements

ID	NFR Title	Category	Description
NFR1	Blue theme	Usability	Website should have a blue color theme
NFR2	Include logo	Usability	A custom logo should be present in the website
NFR3	Color-coded calendar	Usability	The calendar should have color-coded dates
NFR4	Post with minor protection	Security	All posted content with minors should be blurred for unregistered users
NFR5	Password security	Security	Accounts should be encouraged to have strong password
NFR6	Admin control	Security	Only the admin should handle any sort of live data

Table 1: Non-Functional requirements

3.2 Functional Requirements (User Stories)

[In CS482, all functional requirements are written as User Stories. In CS496, some projects may use a different template to write the requirements. The table below is an example of writing the Stories. Adapt accordingly to different templates or if you want to display more info.]

ID	Story Title	Points	Description
S1	Story Example 1	5	As a user, I want to write a user story example, so that people will understand them.
S2	Story Example 2	2	As a user, I want to write a user story example, so that people will understand them.

Table 2: Functional requirements as User Stories.

4 System Design

4.1 Architecture

Our team is following the MVC architecture. In our design, the "View" layer would handle the user interface with HTML, CSS, and JavaScript. The "Model" layer would handle the database using MongoDB. The "Controller" layer would manage incoming requests by using Node.js features. This includes applying user account permission and account authentication. We will use the Express module to handle requests.

4.2 Diagrams

[CS482, on sprints/iterations 2-3, you need to create and update a diagram (check the assignment for which type of diagram). On CS496, since before sprint/iteration 1 you should have a class diagram and keep it up-to-date.]

4.3 Technology

The programming language our team will be using is JavaScript, along with HTML & CSS for webpage development. JavaScript allows us to use the same language for both client and server side, making development more efficient. We will be using Node.js as our main framework as it allows us to use JavaScript on the server side as well. Our team will be using MongoDB as our database because it integrates easily with JavaScript. To test our code, we will be using Jest as it is simple to write & run tests, helping us to meet our coverage goal.

4.4 Coding Standards

The team will follow consistent coding standards to ensure readability, maintainability, and reliability throughout development. Only working code will be committed to the repository, and each commit message must clearly describe what was changed or added. Code will be pushed in small increments to make testing and debugging easier, and any commented-out or developer-only notes will be removed before release. All variable declarations will use const or let, instead of var, and complex methods will include inline comments explaining their purpose. File names will follow the lowercase-hyphen format, variables and functions will use camelCase, and constants will be written in UPPERCASE_WITH_UNDERSCORES. Test names will clearly describe the function being tested and the expected outcome.

Functions will be kept short, serving a single clear purpose, and each will include JSDoc-style comments explaining its parameters and return values. Code formatting will follow a consistent style, including two-space indentation, single quotes for strings (unless otherwise necessary), one blank line between functions, and removal of unused variables or imports. Testing will be conducted using Jest, with at least 60% code coverage required for commits and a final goal of 80% coverage at project completion. Project documentation will be maintained alongside code updates to ensure accuracy and alignment with the current implementation.

4.5 Data

Here is a description of the entities that will be stored in our database for the football league.

The football league is comprised of a league administrator, teams, team managers, registered adults and their registered kids. A league administrator or "admin" has a unique id and their name, username, email, phone number, date of birth and login password stored. A league is comprised of teams. A team has a unique id, a team manager, team name, and a roster of registered kids. Team managers have a unique id, their name, username, email, phone, date of birth, login password and their team's id. Registered kids are registered by their parents and they input their children's name, and date of birth. A kid cannot register itself. Parents/guardians of kids must be registered adults. A registered adult has their name, username, email, phone number, date of birth and login password stored as well as the id of their registered kid.

In the league, there is a season for football. A season has a unique id, a start date, an end date and an eventual champion. Seasons are comprised of games. A game has a unique id, date, start time, end time, place, weather, status in the season (normal, playoff, championship), a winning team and a losing team.

Live chats are recorded on the landing page during games and are comprised of messsages. Messages have a unique ID, a username from registered adult, date and time.

resulting tables

- League_Admin(@id_admin, name, email, phone, date_of_birth, password, username)
- Team_Manager(@id_manager, name, email, phone, date_of_birth, password, #id_team, username)
- Registered_Adult(@id_adult, name, email, phone, date_of_birth, password, username)
- Registered_Kid(@id_kid, name, date_of_birth, #id_adult, #id_team)
- Team(@id_team, name_team, #id_manager, wins, losses)
- Season(@id_season, start_date, end_date, #id_champion_team)
- Match(@id_game, date, start_time, end_time, final_score, place, weather, status, #id_team1, #id_team2, #id_season)

4.6 UI Mocks

The entire sketch so far is hypothetical and is subject to change upon testing or further discussion as to what should be prioritized upon first loading the website. When the page is loaded, a guest/user/admin will see the home page that has a feed centered in the center with an active standings tab to the left, a scaled down schedule display on the right and another widget just below that. There would be a static bar at the topic that will prompt users to log in at the top right. Once a user presses log in, there would be an option to enter your email and password. Otherwise, a guest will create a new account via a sign up button that will appear at the bottom of the card. The sign up card would ask for a whole bunch of information that will serve as an identifier to ensure not just anybody can sign up for the website. It was originally planned for the administrator dashboard to be its on page, but upon discussion it was decided that the administrator dashboard would be appear on the profile of the admin and be hidden on all non-administrator accounts.

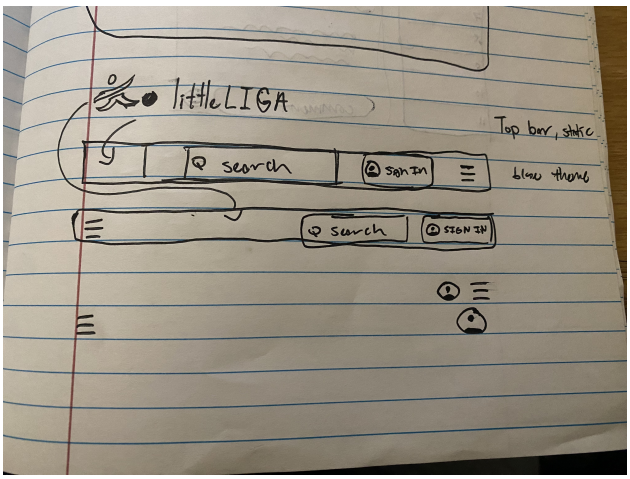

Figure 1: Static Bar

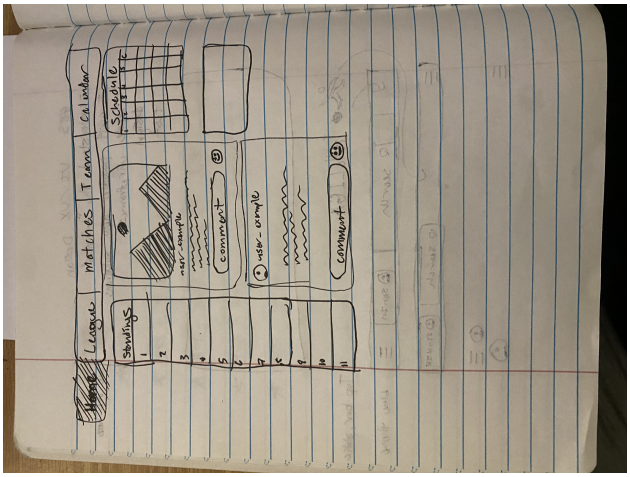

Figure 2: Main landing page

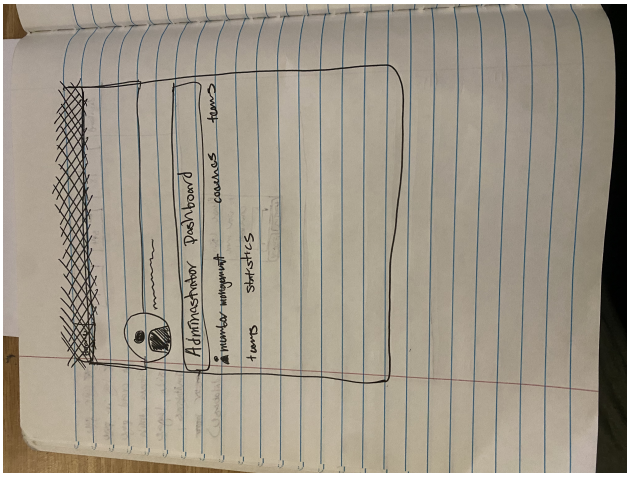

Figure 3: very rough Admin dashboard on profile page

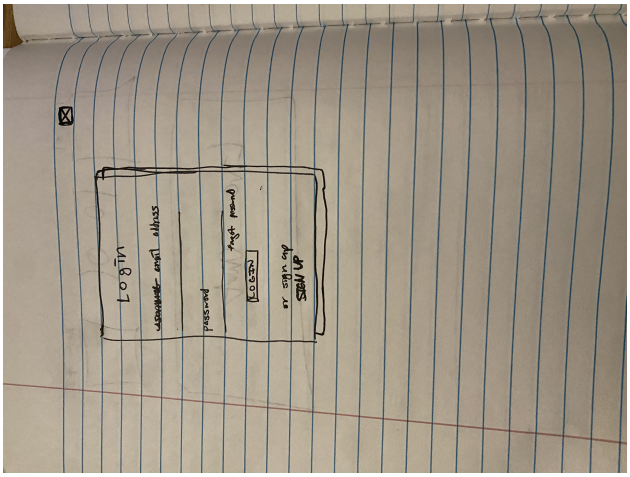

Figure 4: part 1 of log in page(login)

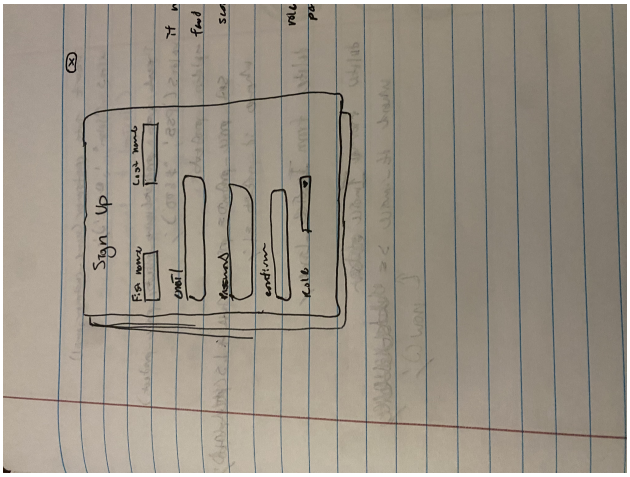

Figure 5: part 2 of log in page(sign up)

5 Iterations

5.1 Iteration Planning

[In CS496, you plan all iterations beforehand. In CS482, you update the planning here at each iteration.]

Iteration	Dates	Stories	Points
1	01/01 - 02/01	S1 Story Example, S2 Story Example 2	07
2	02/01 - 03/01	S3 Story Title, S4 Story Title, S5 Story Title, S6 Story Title	17
3	03/01 - 04/01	S7 Story Title, S8 Story Title, S9 Story Title, S10 Story Title, S11 Story Title	21
4	04/01 - 05/01	S12 Story Title, S13 Story Title, S14 Story Title, S15 Story Title	19
5	05/01 - 06/01	S16 Story Title, S17 Story Title	06
		Total:	70

Table 3: Iteration Planning for Incremental Deliveries

5.2 Iteration/Sprint 1

5.2.1 Planning

[Which stories did you plan for this iteration/sprint. Add the total points for this plan. You can also explain the reason behind your planning, and what major feature(s) your team is focusing on delivering by completing these stories. You may use a table for a summary display of the planning, but elaborate in text more detail in your focus and feature plan.]

5.2.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.2.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some

artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.2.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

5.3 Iteration/Sprint 2

5.3.1 Planning

[Which stories did you plan for this iteration/sprint. Add the total points for this plan. You can also explain the reason behind your planning, and what major feature(s) your team is focusing on delivering by completing these stories. You may use a table for a summary display of the planning, but elaborate in text more detail in your focus and feature plan.]

5.3.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.3.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.3.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

5.4 Iteration/Sprint 3

5.4.1 Planning

[Which stories did you plan for this iteration/sprint. Add the total points for this plan. You can also explain the reason behind your planning, and what major feature(s) your team is focusing on delivering by completing these stories. You may use a table for a summary display of the planning, but elaborate in text more detail in your focus and feature plan.]

5.4.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.4.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.4.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

5.5 Iteration/Sprint 4

[CS496 has 5 sprints. CS482 only has only 3 sprints (remove Iterations 4 and 5 from this doc if you are writing a doc for 482]

5.5.1 Planning

[Which stories did you plan for this iteration/sprint. Add the total points for this plan. You can also explain the reason behind your planning, and what major feature(s) your team is focusing on delivering by completing these stories. You may use a table for a summary display of the planning, but elaborate in text more detail in your focus and feature plan.]

5.5.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.5.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.5.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

5.6 Iteration/Sprint 5

5.6.1 Planning

[Which stories did you plan for this iteration/sprint. Add the total points for this plan. You can also explain the reason behind your planning, and what major feature(s) your team is focusing on delivering by completing these stories. You may use a table for a summary display of the planning, but elaborate in text more detail in your focus and feature plan.]

5.6.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.6.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.6.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

6 Final Remarks

6.1 Overall Progress

[Have you completed everything? If so, present evidence on how you brought value to your client, and the overall client satisfaction. Otherwise, estimate how much progress you done and how long it would take to finish this project.]

6.2 Project Reflection

[Your personal reflection on the project. What lessons did you learned. What would you have done differently. How can you do better work in future projects? You may write this as a team or per person (or both)]

Appendix

[Appendix section if needed]