**CAPSTONE PLANNING DOCUMENTATION:**

**“Ask your Coaches” Web App**

**Summary**

The idea behind this website came to me during practice… I’m a high school nordic skiing coach. Often the only time we can have a good dialogue with our athletes about things like technique adjustments, waxing, etc. is during practice or right before a race while we have the wax benches going. I thought it would be a value add for our team to have a site they could visit where they could post questions and our team of coaches could respond.

It would be the best idea (in my opinion) to have a separate page for admins to answer questions. This would help to maintain a clear separation of responsibilities and access levels between regular users and admins.

To set up the system so that regular users cannot access the admin page, you could implement user authentication and authorization. When a user logs in, the system should verify their identity and determine whether they are a regular user or an admin. If the user is an admin, they would be directed to the admin page. If the user is a regular user, they would be directed to the page where they can post questions.

To ensure that the admin's answers populate back on the user's page, I will use a database to store the questions and answers. When a user posts a question, the question will be stored in the database with a unique identifier. When an admin answers the question on the admin page, the answer will also be stored in the database with the same identifier as the question.

On the user's page, the system will retrieve the questions and answers from the database and display them to the user. This will require some programming on the backend to retrieve the data from the database and some front-end programming to display the data to the user.

**Features for the app include:**

* The ability to post anonymously - enabling newer skiers to ask meaningful questions despite the nerves that come with being new to the sport/younger than their peers.
  + Posts are made on the front end and are handled on the server side by GET and POST requests to serve the front end. The server code sends the post data to the db using Sequelize/PostgreSQL
* The use of a database to collect questions and answers from coaching staff.
  + Seeded db via code in the server file. DB values include Strings and Booleans.
* An FAQ section which listens for certain keywords in questions then populates different tables for the user on the page, e.g. “wax,” “technique,” “racing strategy,” etc., allowing site users to easily scroll through FAQs that relate to their questions. This would hopefully create a living resource that grows over time.
  + To ensure that the admin's answers populate back on the user's page, I will use a database to store the questions and answers. When a user posts a question, the question would be stored in the database with a unique identifier. When an admin answers the question on the admin page, the answer would also be stored in the database with the same identifier as the question.
  + On the user's page, the system will retrieve the questions and answers from the database and display them to the user. This will require programming on the backend to retrieve the data from the database and some front-end programming to display the data to the user.

**Security Features:**

* It will be important to ensure that the system is secure and that user data is protected. This will require implementing appropriate security measures such as encryption, user authentication, and authorization, and data validation

**Current State/CSS Styling:**

I’ve started building out the app already and the following is a screenshot of the current styling/appearance on the user page, which is similar to the admin page. This styling incorporates our team colors and a photo I took of the trails where we practice. Not all CSS styling is apparent in the screenshot (submit button changes to red and has a shadow effect when hovered over, FAQ tables take on styling once data is received back from the server. Still working on these features.

