Project Milestone 4
Group 5 - Depressed Umbrella

Criteria	Details
Revised List of Features	Organizer Account has now become a smaller role: ● Creates a league ● Sets maximum number of teams & number of players on each team "Player" has now been split into 2 types: Coach and Team Member ● Coach can create a team and records scores, which must be approved by another coach (we will create forms) ● Team Members can join teams (until capacity) Bracket Generation has stayed the same ● Randomly generates brackets for team competitions, dynamic, so different numbers of teams can compete against them Visualization if time permits: ● Where teams are in their brackets, we will have a bracket image with the teams in their respective places.
Architecture Diagram	Included at end of file
Front End design	Included at end of file

Web Service Design	● We do not plan on using any 3rd party Web Services. Data that we need to access will be stored in a mySQL database, and accessed using Node.js, then the frontend will be dynamically generated via Pug. ● Since we are using Node.js in our index.js file, there will be JSON (JavaScript Object Notation) objects created in that file to handle data coming from the database, which will then be passed to various views that pug will take advantage of to dynamically render web pages containing information from the database. ● Our project is going to take full advantage of the npm(node package manager) for Javascript. Npm packages are defined in files called package.json. In order for our project to work, we will have our project include the base Node.js packages, pug packages, express packages, and MySQL package. For editing purposes, we have also included the nodemon package, so we do not need to restart the server everytime our index.js file changes.
Database design	Included at end of file

Architecture Diagram

All data hosted on Personal Server owned by one of our group members (Chris). Represented in yellow, for it will host our entire project.

The protocols we will use to connect NodeJS to HTML:
1. Set-up a server to listen to the requests we send from HTML pages
2. Connect it by setting the right variable paths, for example:

```
var http = require('http');
```

Send the SQL data to our webpage. This will populate our webpages, such as keeping track of winning teams, etc. in a table.

Finally, we will be able to connect our HTML pages to NodeJS, and by extension the database. Thus, NodeJS will act as our linker between the Webpage and the Database.

We will use NodeJS to pull our SQL data and pass it to our Webpages. This will be placed on the personal server we will also be storing the database.

Send information from the forms back to NodeJS

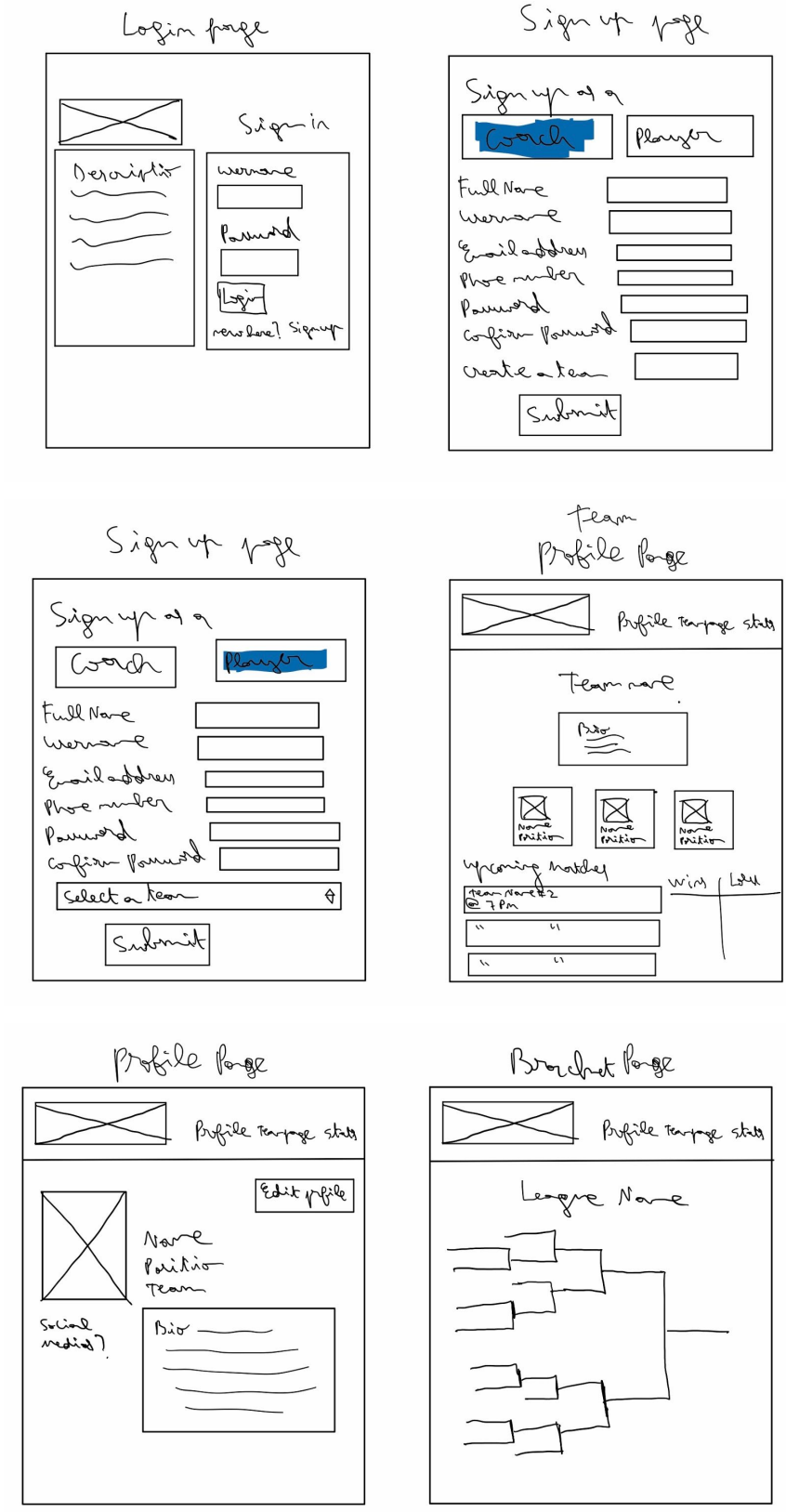
NodeJS will send data collected from values entered into webpages. We can then add rows to our database for wins, losses, new users, etc.

Personal Server hosting a MYSQL database of our user data.

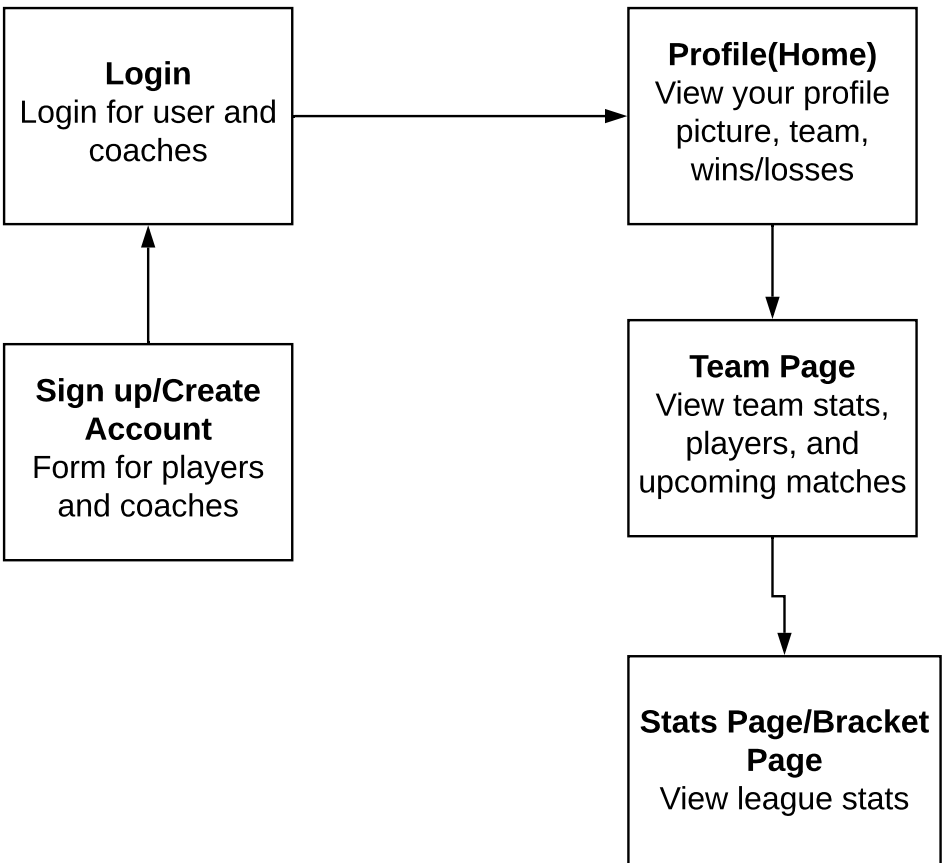
NodeJS pages will run queries to request data from the database. MYSQL will send the requested data.

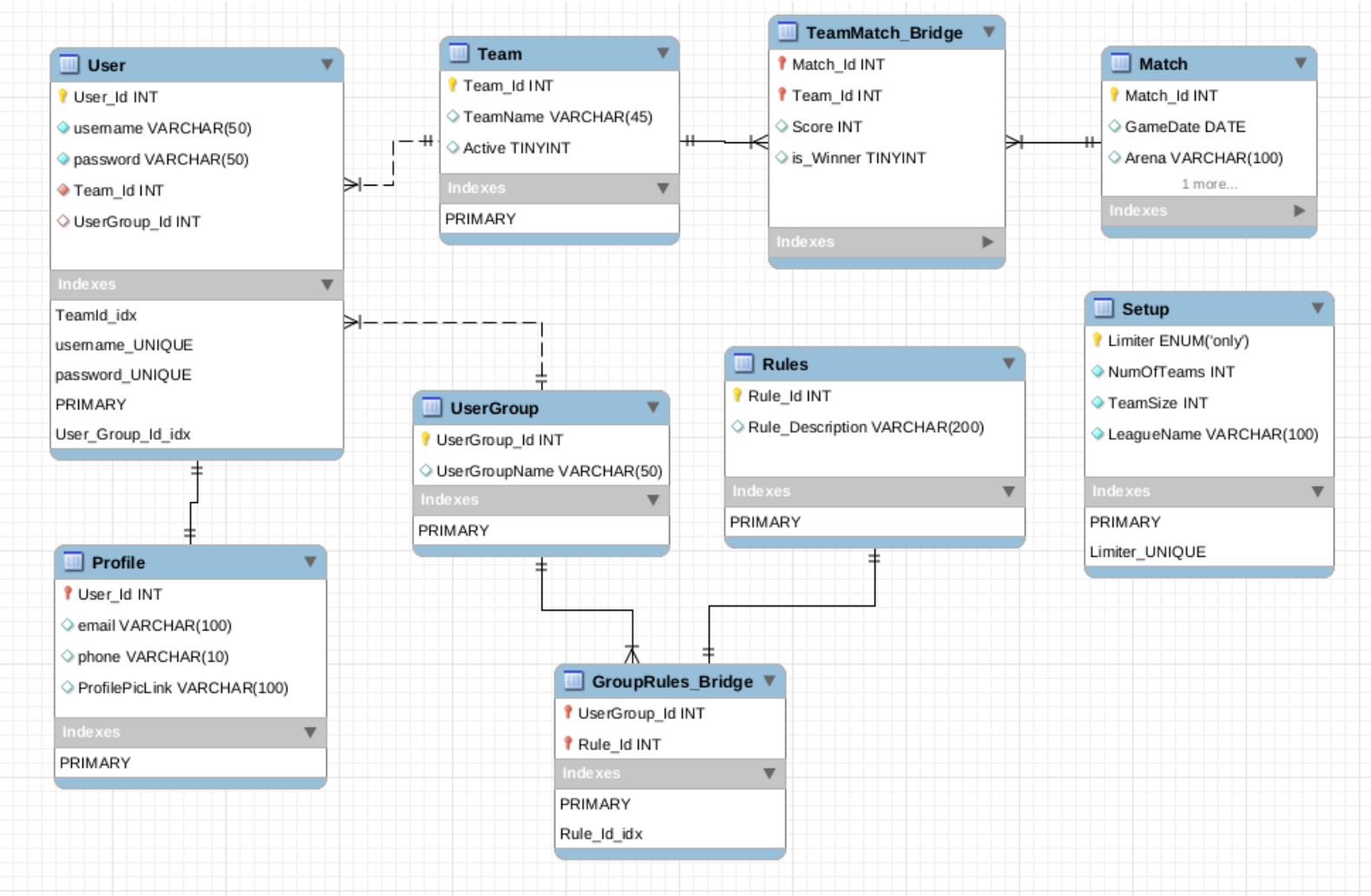
The protocols we will use to connect NodeJS to MYSQL:
1. use a mysql driver
2. we can create a connection in NodeJs using mysql.createConnection()

Front End design



Features


Login
Login for user and coaches
Profile(Home)
View your profile picture, team, wins/losses
Sign up/Create Account
Form for players and coaches
Team Page
View team stats, players, and upcoming matches
Stats Page/Bracket Page
View league stats



- Database Type: mySQL
- **Design Logic:**
    - **Table:**
        - **User**
            - Contains login information and what team user belongs to
        - **Profile**
            - Contains information used by player profile page
        - **Team**
            - Contains team information
        - **UserGroup/GroupRules_Bridge/Rules**
            - Contains login rules, and various other permissions on what a userGroup can do
        - **Setup**
            - A table of variables that only need to be defined once, as such the limiter column, allows for one and only one row to be input into the table
        - **Match/TeamMatch_Bridge**
            - Contains match information and results from match, as well as a Boolean (TINYINT) is_winner, which is used for coaches to confirm or deny result submissions from other coaches