CSCC01: Sprint-1 System Design Document
Team: BBQED
Professor: Ilir Dema

Team Members:
Deon Li - lideon - 1003349263
Cherie Kong - kongcher - 1004800986
Raymond Chen - chenray5 - 1005333337
Andy PhyLim - phyliman - 1005157937
Alex Wan - wanalex- 1004382562
Alexis Yang - yangale8 - 1005350111
Hongfei Chen - chenho50 - 1002917025

Contents:

CRC Cards

Component CRC Cards

File Name: components/Signup.js
Responsibilities:
- Allows the user to give information to signup: first name, last name, email, password, age, gender, etc

Collaborators:
- routes/signup.js -> get existing username/email, post user information on completion
- components/ImageSelect.js -> use the ImageSelect component to get favorite teams data

File Name: components/ImageSelect.js
Responsibilities:
- Given a list of name/url pairs, shows icons in a grid format
- Can click icons to select them
- Can click a button to return a list of names of selected icons
- Shows an error message if the maximum number of selected icons has been reached

Collaborators:
- components/Signup.js -> uses ImageSelect to select favorite teams
- components/Profile.js -> uses ImageSelect to modify favorite teams
- components/EditProfile.js -> uses ImageSelect to show favorite teams

File Name: components/EditProfile.js
Responsibilities:
- Can change profile picture by giving url
- Can edit status, about me
- Can edit favorite teams by using ImageSelect

Collaborators:
- components/ImageSelect.js -> uses the ImageSelect component to show favorite teams
- routes/team.js -> get a list of all teams for ImageSelect
- routes/settings.js -> get current user settings, post updated user settings

File Name: components/Header.js
Responsibilities:
- Shows a navbar that allows the user to navigate to various pages: The Zone, Trivia, Analysis, Picks/Predictions
- Shows the user's profile picture and username

Collaborators:
- route/profile.js -> get profile picture, username
- Context/AuthContext.js -> read the global variable for user's data

File Name: components/Profile.js
Responsibilities:
- Shows the user that is logged in their Status, About, Interest, Current & History Picks , ACS Score & History, Radar List
- Can view/edit current user's profile

Collaborators:
- routes/profile.js -> all information related to the user will be taken from routes
- components/ImageSelect.js -> used to display user interest
- components/EditProfile.js -> any changed information will influence profile display
- components/ProfilePostPopup.js -> can access popup to make posts
- components/ProfileRadarListPopup.js -> can access to see full list of radar list
- Context/AuthContext.js -> read the global variable for user's data

File Name: components/ProfilePostPopup.js
Responsibilities:
- A popup that appears that user can write post

Collaborators:

- routes/post.js -> sends the post submitted to database

File Name: components/RadarList.js
Responsibilities:
- Shows the list of people and their ACS score that the current user follows

Collaborators:
- routes/profile.js -> retrieves radar list data from router

File Name: components/ProfileRadarListPopup.js
Responsibilities:
- A popup that appears so that current user can view the full list of radar list

Collaborators:
- components/RadarList.js -> retrieve the radar list

File Name: components/Login.js
Responsibilities:
- Provides the UI portion of Authentication
- Allows signed up users to gain access to the rest of the app with correct credentials
- Keeps track of username and password

Collaborators:
- components/Signup.js -> users can sign up through login page
- context/AuthContext.js -> read the global variable for user's data

File Name: components/MasterRouter.js
Responsibilities:
- Sets up routes/navigation to all pages

Collaborators:
- components/Header.js -> A visual header for the user to navigate to specific pages
- components/Signup.js -> Navigates to sign up page
- components/Login.js -> Navigates to login page
- components/Profile.js -> Navigates to Profile page - initial route
- components/Registration.js -> Navigates to Registration page
- components/Settings.js -> Navigates to Settings page
- hocs/PrivateRoute.js -> Sets specific pages to require authentication
- hocs/PublicRoute.js -> Sets specific pages to not require authentication

File Name: index.js
Responsibilities:
- The start of the web app. Sets up routes and authorization

Collaborators:
- components/MasterRouter.js -> Sets up route structures
- Context/AuthContext.js -> Sets up structure for authorization

File Name: components/EditProfile.js
Responsibilities:
- Allows user to modify their profile image, status, about, and interests

Collaborator:
- routes/settings.js -> all information related to the user requested from this route
- components/profile.js -> modified information will display on profile

File Name: components/Settings.js
Responsibilities:
- Sets up routes to all the settings pages

Collaborators:
- components/EditProfile.js -> One of the routes that is apart of settings

Model CRC Cards

File Name: models/login
Responsibilities:
- Authentication
- Gets a user's username and checks if the password is identical

Collaborators:
- components/Login.js
- components/Header.js-> header for website
- components/signup.js -> can lead to signup

File Name: models/team.js
Responsibilities:
- Holds the schema for the name and reference to a team's logo

Collaborators:
- components/Signup.js -> signing up uses the schema for the names
- components/Profile.js -> the profile page uses the schema for the names
- routes/team -> route is used

File Name: models/profile.js
Responsibilities:
- Holds the schema and represents the data in profiles collection that exists in MongoDB
- Handles queries for anything that deals with the profile collection

Collaborators:
- components/profile.js -> Stores the data for the content on the profile
- routes/profile.js -> Requests the data stored in the schema
- routes/settings.js -> Requests the data stored in schema so can modify

File Name: models/acs.js
Responsibilities:
- Holds the schema and represents the data in acs collection that exists in MongoDB
- Handles queries for anything that deals with the acs collection

Collaborators:
- components/profile.js -> ACS history is shown on profile
- routes/profile.js -> Requests the data stored in the schema

File Name: models/comment.js
Responsibilities:
- Holds the schema and represents the data in the comments collection that exists in MongoDB.
- Handles queries for anything that deals with the Comments collection

Collaborators:
- model/post.js -> All comments must relate to a post
- model/user.js -> All comments must relate to a user

File Name: models/post.js
Responsibilities:
- Holds the schema and represents the post collection that exists in MongoDB.
- Handles queries for anything that deals with the post collection

Collaborators:
- model/comment.js -> Each post should have an array of comments
- model/user.js -> All posts must relate to a user
- routes/post.js -> Requests to create new posts

File Name: models/user.js
Responsibilities:
- Holds the schema and represents the data in the user collection that exists in MongoDB
- Handles queries for anything that deals with the user collection, specifically signup and signin

Collaborators:
- model/profile.js -> Each user must have only a single profile attached
- route/signup.js -> Adds user on successful signup

- route/signin.js -> Checks user data to see if login was successful

Route CRC Cards

File Name: routes/profile.js
Responsibilities:
- Sets up routes and handles the REST requests related to profile

Collaborators:
- models/profile.js -> Obtains data in the model
- models/acs.js -> Obtains data in the model
- models/team.js -> Obtains data in the model
- server.js -> Master backend router, all REST URIs
- Passport -> check if the user has the token

File Name: routes/settings.js
Responsibilities:
- Sets up routes and handles the REST requests related to profile

Collaborator:
- models/profile.js -> Obtains and modifies data in the model
- server.js -> Master backend router, all REST URIs

File Name: routes/team.js
Responsibilities:
- Sets up routes and handles the REST requests related to teams

Collaborators:
- models/team.js -> uses route
- components/Signup.js -> user is prompted to select favourite team
- components/Profile.js -> interest section is populated
- server.js -> Master backend router, all REST URIs are set here

File Name: routes/signup.js
Responsibilities:
- Sets up routes and handles the REST requests related to signup

Collaborators:
- model/user.js -> Creates a user on successful request
- model/user.js -> Creates a profile on successful request
- server.js -> Master backend router, all REST URIs are set here
- components/Login.js -> routes are called here

File Name: routes/post.js
Responsibilities:
- Sets up routes and handles the REST requests related to posts

Collaborators:
- model/post.js -> Creates post on a successful request (/add)
- server.js -> Master backend router, all REST URIs
- components/Profile.js -> Profile is able to create new posts

File Name: server.js
Responsibilities:
- Sets up all REST URIs using the routes

Collaborators:
- routes/. Sets up the URI for all routers using the route: "/."

File Name: routes/logout.js
Responsibilities: set up the route to handle the get request of logging out the user
Collaborators:
- Passport.js -> checking if user has the access token

File Name: routes/login.js
Responsibilities: set up the route to handle the get request of logging in the user
Collaborators:
- Passport.js -> allows for authentication to occur

Authentication CRC Cards

File Name: Passport.js
Responsibilities: provide local and jwt strategy for the passport
Collaborators: None

File Name: Context/AuthContext.js
Responsibilities: generate the provider and consumer of global variable (user {username, email, permissions}, isAuthenticated) in the frontend
Collaborators:

- ServiceAuthService.js -> check is the user is authenticated

File Name: hocs/PrivateRoute.js
Responsibilities:
Collaborators:

- Context/AuthContext.js -> read the global variable to determine if user is authenticated

File Name: hocs/PublicRoute.js
Responsibilities:
Collaborators:

- Context/AuthContext.js -> read the global variable to determine if user is authenticated

File Name: Services/AuthService.js
Responsibilities: Handler to interact with backend relating with authentication
Collaborators: None

Software Architecture Diagram