CSCC01: Sprint-3 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/EditAccount.js

Responsibilities:

- Can change email
- Can change password
- Can deactivate account

Collaborators:

- components/UpdateEmail.js -> contains the popup component for updating email
- components/UpdateEmail.js -> contains the popup component for updating password
- components/UpdateEmail.js -> contains the popup component for deactivating account

File Name: components/UpdateEmail.js

Responsibilities:

- Popup for allowing the user to update their email

Collaborators:

- components/EditAccount.js -> uses the popup component when user clicks update button in settings
- routes/settings.js -> get current user settings, post updated user settings

File Name: components/UpdatePassword.js

Responsibilities:

- Popup for allowing the user to update their password

Collaborators:

- components/EditAccount.js -> uses the popup component when user clicks update button in settings
- routes/settings.js -> get current user settings, post updated user settings

File Name: components/DeactivateAccount.js

Responsibilities:

- Popup for allowing the user to deactivate their account

Collaborators:

- components/EditAccount.js -> uses the popup component when user clicks deactivate account button in settings
- routes/settings.js -> get current user settings, delete 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 and gives ability to navigate to profile

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/PostView.js

Responsibilities

- Allows user to see the full view of a post with all the comments
- Can post a comment
- Can agree/disagree the post

Collaborators

- routes/TheZone.js -> retrieve post data and sends any new comments made

File Name: components/ReportPost.js

Responsibilities:

- Shows a list of all posts that have been reported
- Can delete post if the post is inappropriate

Collaborators:

- components/Reports.js -> retrieve the list of post reports

File Name: components/ReportComment.js

Responsibilities:

- Shows a list of all comments that have been reported
- Can delete comment if the comment is inappropriate

Collaborators:

components/Reports.js -> retrieve the list of comment reports

File Name: components/Reports.js

Responsibilities:

- List of post/comments reported

Collaborators:

- routes/zone.js -> retrieves the list of reports from the database

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/Login.js -> Navigates to login page
- components/Profile.js -> Navigates to Profile page - initial route
- components/Trivia.js -> Navigates to Trivia page
- components/Settings.js -> Navigates to Settings page
- components/Citations.js -> Navigates to Citation page
- components/Queue.js -> Allows users to enter a queue for trivia
- components/PicksAndPredictions.js -> Navigates to Picks and Predictions
- components/AnalysisPost.js -> Navigates to a specific Analysis Post
- components/Analysis.js -> Navigates to Analysis page
- components/ReportPosts.js -> Allow for reporting posts
- components/ReportComments.js -> Allow for reporting comments
- components/PostView.js -> Navigate to a specific Zone post
- 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/Settings.js

Responsibilities:

- Sets up routes to all the settings pages

Collaborators:

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

File Name: components/TheZone.js

Responsibilities:

- Displays posts for users to see
- Allows to users to create posts
- Agree and disagree with posts

Collaborators:

- components/PostView.js -> linked to show post individually/ request
- routes/post.js -> uses route for posts
- components/TheZoneReportPopup.js -> popup for reporting posts

File Name: components/Trivia.js

Responsibilities:

- Controls the state for a Trivia game

Collaborators:

- components/InGameTrivia.js -> Displays the game, given state
- routes/Queue.js -> Displays the queue, given state
- routes/triviaSolo.js -> requests for solo trivia match
- routes/triviaHeadToHead.js -> get queue, requests for head to head trivia match

File Name: components/InGameTrivia.js

Responsibilities:

- Given the current Trivia state, displays the game (question, timer, answer buttons)
- Allows the user to click on an option to submit their answer for a Trivia question

Collaborators:

- components/Trivia.js -> Gets the state, displays the game
- components/PostTrivia.js -> At the end of the game, displays winner/loser and navigation options
- components/TriviaSidebar.js -> Given the state, displays additional information

File Name: components/TriviaSidebar.js

Responsibilities:

- Given the current Trivia state, displays an option of trivia modes or the status of the match (previous questions, scores, users in play)

Collaborators:

- components/InGameTrivia.js -> Displays the sidebar
- components/ProfilePicture.js -> Given the users, displays the profile pictures

File Name: components/PostTrivia.js

Responsibilities:

- Display the final scores of a Trivia match
- If the game mode is 1v1, displays the winner or loser
- Allows the user to select navigation options “play again”, “select mode”

Collaborators:

- components/ProfilePicture.js -> Given the users, displays the profile pictures
- components/InGameTrivia.js -> Is displayed when the Trivia game is over

File Name: components/Queue.js

Responsibilities:

- Manages users in queue of a trivia match

Collaborators:

- components/Trivia.js -> displays the queue screen
- routes/triviaHeadToHead.js -> Join and leave queue, find and create game

File Name: components/Citations.js

Responsibilities:

- Cite images that were used for this project

Collaborators: none

File Name: components/ProfilePicture.js

Responsibilities:

- Shows the user's profile picture with the SportCred logo around it, given username/url

Collaborators:

- routes/profile.js -> Fetch the url of the user's image if url not given
- components/TriviaSidebar.js -> Show the profile picture user, (opponent)
- components/PostTrivia.js -> Show the profile pictures of user, (opponent)

File Name: components/Analysis.js

Responsibilities:

- Shows all current and past debates

Collaborators:

- components/AnalysisPost.js -> individual component for each debate
- routes/analysis.js -> Handles requests to get information from database

File Name: components/AnalysisPost.js

Responsibilities:

- Shows each individual debate on the Analysis page and its details

Collaborators:

- components/Analysis.js -> displays multiple AnalysisPost components
- routes/analysis.js -> Handles requests to get information from database

File Name: components/AnalysisPostView.js

Responsibilities:

- Shows information of a specific debate (question, image, and comments)
- Handles the scoring of a particular post

Collaborators:

- components/Analysis.js -> sends id of debate over to AnalysisPostView
- routes/Analysis.js -> Handles requests to get information from database
- components/Slider.js -> display a slider, get information when slider is clicked
- components/Histogram.js -> display a histogram

File Name: components/Slider.js

Responsibilities:

- Display a slider for the user to score a value from 0 to 100 by hovering their mouse and clicking
- notify when the slider is clicked and propagate the value

Collaborators: none

File Name: components/Histogram.js

Responsibilities:

- Display the distribution of scores for a particular analysis post in the form of a histogram

Collaborators: none

File Name: components/TheZoneReportPopup.js

Responsibilities:

- Shows the report popup and gives the user the ability to report a post

Collaborators:

- components/TheZone.js → displayed in The Zone
- routes/post.js → api call to change report status

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
- routes/zone.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
- route/zone.js -> Requests comment data

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
- routes/zone.js -> Requests to get data in schema

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

File Name: models/headtoheadgame.js

Responsibilities:

- Holds the schema and represents the data in the head-to-head game collection that exists in MongoDB
- Handles queries for anything that deals with the head-to-head trivia
- Collaborators:
- routes/triviaHeadToHead.js -> create, access and update the data stored in headtoheadgame

File Name: models/queue.js

Responsibilities:

- Holds the schema and represents the data in the queue collection that exists in MongoDB
- Handles queries for anything that deals with matchmaking

Collaborators:

- routes/headtoheadgame.js -> create, access and update the data stored in queue

File Name: models/solotriviainstance.js

Responsibilities:

- Holds the schema and represents the data in the solo trivia game collection that exists in MongoDB
- Handles queries for anything that deals with solo trivia

Collaborators:

- routes/solotrivia.js -> create, access and update the data stored in solotriviainstance

File Name: models/trivia.js

Responsibilities:

- Holds the schema and represents the data in the trivia collection that exists in MongoDB
- Stores a pool of available trivia questions for solo trivia and head-to-head trivia

Collaborators:

- routes/triviaHeadToHead.js -> read a randomized 11 trivia questions from the pool
- routes/triviaSolo.js -> read randomized trivias questions from the pool

File Name: models/analysis.js

Responsibilities:

- Holds the schema and represents the data in the analysis collection that exists in MongoDB

Collaborators:

- routes/analysis.js -> read information in analysis
- routes/analysisPost.js -> read information in analysis

File Name: models/analysisPost.js

Responsibilities:

- Holds the schema and represents the data in the analysispost collection that exists in MongoDB
- Stores information in the posts of analysis

Collaborators:

- routes/analysisPost.js -> get information of posts in analysispost collection

File Name: models/analysisQuestion.js

Responsibilities:

- Holds the schema and represents the data in the analysisquestion collection that exists in MongoDB
- Stores a pool of questions and corresponding images to build analyses

Collaborators:

- None

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 and account settings

Collaborator:

- models/profile.js -> Obtains and modifies data in the model
- models/user.js -> Obtains and modifies user in the model
- models/acs.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
- model/acs.js -> Creates an ACS model 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

File Name: routes/zone.js

Responsibilities: Set up the route to handle all requests regarding “The Zone”

Collaborators:

- models/acs -> Get ACS data from certain users
- models/post -> Get post data from posters and insert data into posts
- models/profile -> Get images from posters
- models/comment -> Get comment data and insert data into comments
- components/TheZone -> Calls requests
- components/TheZonePostView -> Calls requests
- components/ReportPosts -> Calls requests
- components/ReportComments -> Calls requests
- Server.js -> Sets the URIs for all routes

File Name: routes/triviaHeadToHead.js

Responsibilities:

- maintaining and handles information stores in model headtoheadgame

Collaborators:

- model/headtoheadgame.js -> stores information relating with head-to-head trivia game
- model/trivia -> stores trivia questions for initializing head-to-head trivia game

File Name: routes/solotrivia.js

Responsibilities:

- Sets up routes and handles the REST requests related to solo trivia

Collaborators:

- model/solotriviainstance.js -> stores information relating to a solo trivia game
- model/trivia -> stores trivia questions for initializing a solo trivia game
- server.js -> Master backend router, all REST URIs

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