System Design - SPORTCRED

The Code Manglers

Renzo Reyes (reyesren), Ian Gu (guian), Victor Guo (guovicto), Kevin Oh (ohkevin2), Farhan Chowdhury (chowd601)
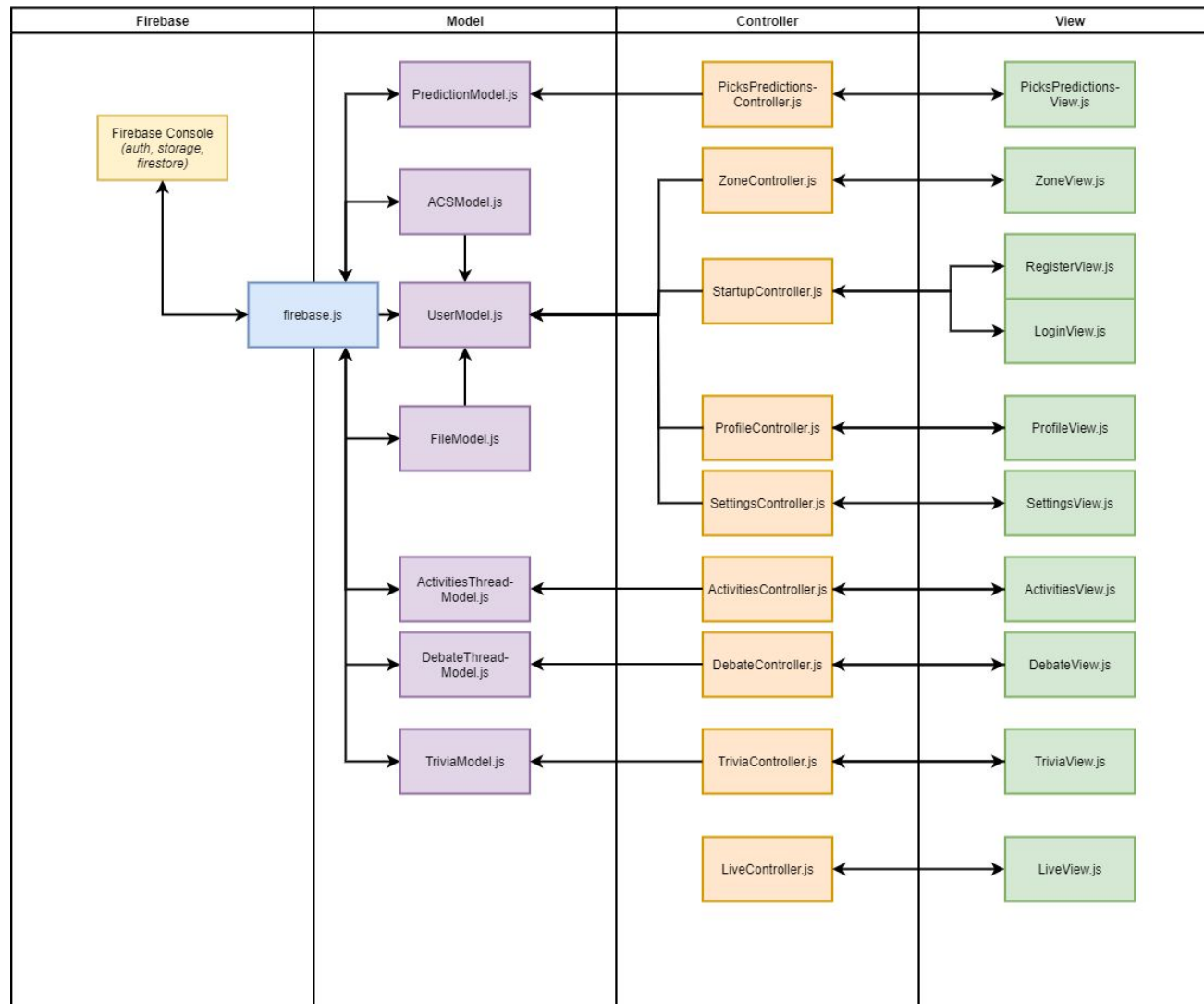
Course: CSC C01 @ University of Toronto Scarborough

Professor: Illir Dema

Table of Contents

Architecture Diagram

Firebase	Model	Controller	View
Firebase Console *(auth, storage, firestore)*	PredictionModel.js	PicksPredictions-Controller.js	PicksPredictions-View.js
firebase.js	ACSModel.js	ZoneController.js	ZoneView.js
	UserModel.js	StartupController.js	RegisterView.js
			LoginView.js
	FileModel.js	ProfileController.js	ProfileView.js
		SettingsController.js	SettingsView.js
	ActivitiesThread-Model.js	ActivitiesController.js	ActivitiesView.js
	DebateThread-Model.js	DebateController.js	DebateView.js
	TriviaModel.js	TriviaController.js	TriviaView.js
		LiveController.js	LiveView.js

Structure

Class name:	App.js
Parent class:	None
Subclasses:	None
Responsibilities:	• This is the entry point of the application. It serves the navigation component.
Collaborators:	• AppNavigator.js

Class name:	AppNavigator.js
Parent class:	None
Subclasses:	None
Responsibilities:	• AppNavigator.js holds all the controllers, and allows the user to switch between each page.
Collaborators:	All controllers

Class name:	Firebase.js

Parent class:	None
Subclasses:	
Responsibilities:	• Firebase.js provides the basic connection and configuration for using the firebase service
Collaborators:	All models

Models

Class name:	UserModel.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class handles any requests to Firebase related to User Information (e.g. Username, Password, Custom ACS score, etc.)
Collaborators:	● Firebase.js ● FileModel.js

Class name:	ACSModel.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class handles any requests to Firebase related to a User's ACS score
Collaborators:	● UserModel.js ● Firebase.js

Class name:	PostModel.js
Parent class:	None
Subclasses:	None
Responsibilities:	• This class handles requests to Firebase related to an activity thread.
Collaborators:	• UserModel.js • Firebase.js

Class name:	DebateThreadModel.js
Parent class:	None
Subclasses:	None
Responsibilities:	• This class handles requests to Firebase related to a debate thread.js
Collaborators:	• UserModel.js • Firebase.js

Class name:	TriviaModel.js

Parent class:	None
Subclasses:	None
Responsibilities :	● This class handles requests to Firebase related to trivia questions and answers, and user history
Collaborators:	● UserModel.js ● Firebase.js

Class name:	TriviaChallengeModel.js
Parent class:	None
Subclasses:	None
Responsibilities :	● This class handles requests to Firebase related to user challenges and history
Collaborators:	● UserModel.js ● Firebase.js

Class name:	PicksAndPredictionsModel.js
Parent class:	None

Subclasses:	None
Responsibilities:	● This class handles requests to Firebase to retrieve new prediction data to be given to the user
Collaborators:	● Firebase.js

Class name:	PlayoffsModel.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class handles requests to Firebase related to a playoffs predictions
Collaborators:	● UserModel.js ● Firebase.js

Class name:	DailyPicksModel.js
Parent class:	None
Subclasses:	None
Responsibilities:	This class handles requests to Firebase related to a daily picks model
Collaborators:	Firebase.js

Controllers

Class name:	PicksPredictionsTabController.js
Parent class:	None
Subclasses:	None
Responsibilities:	- The class will get the necessary teams and brackets to provide to the view - The class will retrieve the picks and predictions from the view, to then send it to the PredictionModel
Collaborators:	None

Class name:	PlayoffsController.js
Parent class:	None
Subclasses:	None
Responsibilities:	- The class will get the necessary teams and brackets to provide to the view - The class will retrieve the picks and predictions from the view, to then send it to the PlayoffsModel

Collaborators:	• PlayoffsView.js • PlayoffsModel.js

Class name:	ZoneController.js
Parent class:	None
Subclasses:	List all the subclasses separated by a comma.
Responsibilities :	• The class will listen for user to choose which part of the app to navigate to while on ZoneView
Collaborators:	• ZoneView.js • UserModel.js

Class name:	StartupController.js
Parent class:	None
Subclasses:	None
Responsibilities :	• The class will send login information to the UserModel for login • The class will send registration information to the UserModel

<table>
<tr>
<td></td>
<td>

for registration
- The class will provide the necessary information to either RegisterView or LoginView for display

</td>
</tr>
<tr>
<td>Collaborators:</td>
<td>

- RegisterView.js
- LoginView.js
- UserModel.js

</td>
</tr>
</table>

<table>
<tr>
<td>Class name:</td>
<td>ProfileController.js</td>
</tr>
<tr>
<td>Parent class:</td>
<td>None</td>
</tr>
<tr>
<td>Subclasses:</td>
<td>None</td>
</tr>
<tr>
<td>Responsibilities
:</td>
<td>

- The class will get the user's information from the UserModel to then populate the view of the profile
- The class will listen for profile edits, and update the UserModel of those edits the user has made.

</td>
</tr>
<tr>
<td>Collaborators:</td>
<td>

- ProfileView.js
- UserModel.js

</td>
</tr>
</table>

<table>
<tr>
<td>Class name:</td>
<td>SettingsController.js</td>
</tr>
<tr>
<td>Parent class:</td>
<td>None</td>
</tr>
<tr>
<td>Subclasses:</td>
<td>None</td>
</tr>
</table>

Responsibilities:	● The class will retrieve the user's custom settings that they want from the app ● The class will get any updated user settings and update the UserModel of these settings
Collaborators:	● SettingsView.js ● UserModel.js

Class name:	ActivitiesController.js
Parent class:	None
Subclasses:	None
Responsibilities:	● The class allows users to contribute to discussions through ActivitiesThreadModel
Collaborators:	● ProfileView.js ● ThreadModel.js

Class name:	DebateController.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class allows users to contribute to discussions through DebateThreadModel

Collaborators:	● DebateView.js ● ThreadModel.js

Class name:	TriviaController.js
Parent class:	None
Subclasses:	None
Responsibilities :	● This class feeds a "trivia game" to the view ● It will listen for user input and write to TriviaModel accordingly
Collaborators:	● TriviaView.js ● TriviaModel.js

Class name:	TriviaHeadToHeadController.js
Parent class:	None
Subclasses:	None
Responsibilities :	● This class displays the tabs for a user to challenge another user, their incoming challenges, and their history.
Collaborators:	● ChallengeUsersView.js ● IncomingChallengesView.js

	● CompletedChallengesView.js ● TriviaChallengeModel.js ● UserModel.js ● TriviaModel.js

Class name:	TriviaStartGameController.js
Parent class:	None
Subclasses:	None
Responsibilities	● Navigate to the loading screen when user presses start game
Collaborators	● TriviaLoadingScreenView

Class name:	TriviaLoadingScreenController.js
Parent class:	None
Subclasses:	None
Responsibilities	● Show a loading screen until data is retrieved from the database ● Route to trivia game after loading is done
Collaborators	● TriviaLoadingScreenView

	● TriviaModel

Class name:	TriviaMainGameController.js
Parent class:	None
Subclasses:	None
Responsibilities	● Send information to the main trivia game view so the game can start ● End the game and navigate to the results page
Collaborators	● TriviaMainGameView

Class name:	TriviaResultsController.js
Parent class:	None
Subclasses:	None
Responsibilities	● Send results to the results view and render it ● Record the results in the database ● Navigate to the home page after user presses continue
Collaborators	● TriviaResultsView ● TriviaModel

Class name:	LiveController.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class is a placeholder for a future feature
Collaborators:	● LiveView.js

Class name:	CreatePostController.js
Parent class:	None
Subclasses:	None
Responsibilities :	This class is responsible for processing data from CreatePostView and communicating with the database to retrieve/add data
Collaborators:	- CreatePostView.js - Firebase.js - UserModel.js - PostModel.js

Class name:	FullPostController.js
Parent class:	None
Subclasses:	None
Responsibilities :	This class is responsible for retrieving all post data from the backend and sending it to be displayed
Collaborators:	- FullPostView.js - Firebase.js - PostModel.js

Class name:	TheZoneController.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class is responsible for retrieving post data from the database to display in TheZoneView.js
Collaborators:	● TheZoneView.js ● Firebase.js

Class name:	TheZoneTabsController.js
Parent class:	None
Subclasses:	None
Responsibilities:	● This class is responsible for managing the Zone and Radar views as tabs
Collaborators:	● TheZoneView.js ● RadarView.js

Views

Class name:	PlayoffView.js
Parent class:	None
Subclasses:	None
Responsibilities :	• represent the UI components for playoffs Picks & Predictions in the application
Collaborators:	• PicksPredictionsController.js

Class name:	RegisterView.js
Parent class:	None
Subclasses:	List all the subclasses separated by a comma.
Responsibilities :	• the UI component for Register in the application
Collaborators:	• StartupController.js

Class name:	LoginView.js

Parent class:	None
Subclasses:	None.
Responsibilities:	● the UI component for Login in the application
Collaborators:	● LoginController.js

Class name:	ProfileView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for Profile in the application
Collaborators:	● ProfileController.js

Class name:	SettingsView.js
Parent class:	None
Subclasses:	None

Responsibilities:	● the UI component for Settings in the application
Collaborators:	● SettingsController.js

Class name:	ActivitiesView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for Activities in the application
Collaborators:	● ActivitiesController.js

Class name:	DebateView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for Debate in the application
Collaborators:	● DebateController.js

Class name:	TriviaView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for Trivia in the application
Collaborators:	● TriviaController.js

Class name:	TriviaLandingView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● The landing view for Trivia in the application
Collaborators:	● TriviaController.js

Class name:	TriviaStartGameView.js
Parent class:	None
Subclasses:	None
Responsibilities	● Show a start game button the user
Collaborators:	● TriviaStartGameController

Class name:	TriviaLoadingScreenView.js
Parent class:	None
Subclasses:	None
Responsibilities	● Show a loading screen to the user
Collaborators:	● TriviaLoadingScreenController

Class name:	TriviaMainGameView.js

Parent class:	None
Subclasses:	None
Responsibilities	- Show the main trivia game to the user - Show question count - Show score - Show question - Show options - Show if user pressed correct answer or wrong answer
Collaborators:	TriviaMainGameController

Class name:	TriviaResultsView.js
Parent class:	None
Subclasses:	None
Responsibilities	- Show results view to the user - Show final score to the user - Show ACS change to the user
Collaborators:	TriviaResultsController

Class name:	ChallengeUsersView.js

Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for users to challenge another user to trivia
Collaborators:	● TriviaHeadToHeadController.js

Class name:	IncomingChallengesView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component to view a users incoming challenges
Collaborators:	● TriviaHeadToHeadController.js

Class name:	CompletedChallengesView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for a user to view their completed challenges

Collaborators:	● TriviaHeadToHeadController.js

Class name:	LiveView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for Live View in the application
Collaborators:	● LiveController.js

Class name:	CreatePostView.js
Parent class:	None
Subclasses:	None
Responsibilities:	● the UI component for creating posts
Collaborators:	● CreatePostController.js

Class name:	FullPostView.js
Parent class:	None
Subclasses:	None
Responsibilities :	● the UI component for viewing full posts
Collaborators:	● FullPostController.js

Class name:	TheZoneView.js
Parent class:	None
Subclasses:	None
Responsibilities :	● the UI component for TheZone (should list all relevant posts user would see and allow them to go to create post and navigate to other pages)
Collaborators:	● TheZoneController.js