Hospital Management Game

Getting Started

Clone this repository and execute the following commands in a terminal:

- `git checkout master`
- `npm install`

Running application

You can use the following two methods to run the application.

Method 1

- Start mongoDB on the default port using cmd: `mongod`
- Dev? `npm run start`
- Staging or Production? `npm run build` and `npm run serve`
- Make sure you do `npm run test` before building the app.

Method 2

- `sudo chmod 777 run.sh`
- `sudo ./run.sh`

Project Structure

You will notice a few files/directories within this project:

1. `lib/app` - Server side (backend) stuff.
2. `lib/app/models` - Contains all the schema for the collections.
3. `public/app` - Front end stuff. Where the angular controllers, views, routes are located.
4. `public/assets` - Where libraries for angular, bootstrap and jquery are located.
5. `lib/index.js` - the primary configuration file for the server

Components

The major components of the application are the following:

- Game
    - Player Gameplay
    - Agent Gameplay
- Questionnaire
    - Pre-Game Questionnaire
    - Post-Game Questionnaire
- Admin
    - Report Generation
    - Admin Management
    - Game Configuration Management

Game

The game component consist of the following sub-components:

- Player Gameplay
- Agent Gameplay

The files relating to the game component are as follows:

Client Side

- `public/app/views/pages/game/game.html` - Template of the game view.
- `public/app/controllers/gamePageCtrl.js` - Used for rendering data in the game view.
- `public/app/services/gamePageServices.js` - Contains the necessary functions like patient queueing algorithm, assigning patients to room, maintaining room timers and other utility functions needed for playing the game.
- `public/app/services/gameStateServices.js` - Contains structure of game state.
- `public/app/services/roomServices.js` - Contains structure for a room.
- `public/app/services/userStatsService.js` - Provides structure for maintaining user statistics like moves and utility functions.
- `public/app/services/circleService.js` - Provides structure for circle object.
- `public/app/services/agentServices.js` - Contains the necessary functions for agent gameplay such as agent sharing algorithm. All functionality of the agent gameplay is contained in this file.

Server Side

- `lib/app/routes/api.js` - Uses API route (`/api/game/updateUserStatistics`) to update user statistics collection with user moves data.

Models

- `lib/app/models/userStatistics.js` - Contains schema for the user statistics collection.

Questionnaire

The questionnaire component has two major sub-components, namely

- Pre-Game Questionnaire
- Post-Game Questionnaire

The files relating to the pre-game and post-game questionnaires are as follows:

Client Side

- `public/app/views/pages/game/demographics.html` - Template of the pre-game demographics questionnaire.
- `public/app/controllers/preGameQuestionnaireCtrl.js` - Used for rendering data in the pre-game questionnaire view.
- `public/app/views/game/trustTaskQuestionnaire.html` - Template of the post-game trust and task questionnaire.
- `public/app/controllers/postGameQuestionnaireCtrl.js` - Used for rendering data in the post-game questionnaire view.
- `public/app/services/questionnaireServices.js` - Contains functions for updating user statistics with questionnaire data.

Server Side

- `lib/app/configs/demographic-questions.js` - Contains questions of the pre-game questionnaire.
- `lib/app/configs/trust-task-questions.js` - Contains questions of the post-game questionnaire.
- `lib/app/routes/api.js` - Uses API route (`/api/game/updateUserStatistics`) to update user statistics collection with player's pre-game and post-game questionnaire responses.

Admin

The admin module consists of four major sub-components, namely

- Report Generation
- Admin Management
- Game Configuration Management

Report Generation

The reporting component consists of two major sub-components, namely

- Admin reports
- Game reports

The files relating to reporting component are as follows:

- `public/app/views/pages/admin/reporting.html` - Template to pick date range.
- `public/app/controllers/reportCtrl.js` - Controller associated with reporting view.
- `public/app/services/reportServices.js` - Contains functions to get and set log data.

Admin Management

The admin management component is used to add or delete admins. The files associated with the admin management module are as follows:

- `public/app/views/pages/admin/manageAdmin.html` - Template used for managing admins
- `public/app/controllers/manageAdminCtrl.js` - Contains functions for adding, deleting and listing admins in the manage

admin view.

- `public/app/services/manageAdminServices.js` _ Contains functions for adding, deleting admins from the collection.

Game Configuration Management

The game configuration management component is used to create and maintain game configurations. The files relating to game configuration management are as follows:

- `public/app/views/pages/admin/gameConfigPage.html` - Template for game configuration view.
- `public/app/controllers/gameConfigCtrl.js` - Controllers used to render addition and deletion of game configuration data in the game configuration view.
- `public/app/services/gameConfigServices.js` - Functions used to add or delete configurations in the collection.