

SWENG 500: Software Engineering Studio
Test Plan and Report

Group 5

Jeffrey Chiampi

Brandon Hessler

Kyle Hughes

Ryan McDonald

Overview

This document contains the results of white box testing the Science Olympiad application. The application was designed in two modules. The first module, the server API, is a fully functional API, which interacts with the system's database. The second module is responsible for the user interface (UI) and utilizes the API. It is primarily responsible for the presentation and input of data. Since these modules were developed with separate programming languages, their testing strategies and methods varied.

In total, 252 tests were developed. The continuous integration and continuous delivery development strategy we employed was facilitated by a Jenkins server. Jenkins is an open source automation server written in Java. It ingested our source code from GitHub, built our applications, ran the tests, and provided reports. If the tests passed, Jenkins deployed our app to production. The nature of this process meant that all tests had to pass in order for the application to be deployed.

Testing the Server API

In order to reach 93% instruction coverage, 95% method coverage, and 100% class coverage 54 API tests were developed. Instruction coverage refers to the smallest unit JaCoCo counts. This is equivalent to a single Java byte code instruction. A complete coverage summary can be view in the following section. While this coverage percentage is not 100% it does represent testing every critical component.

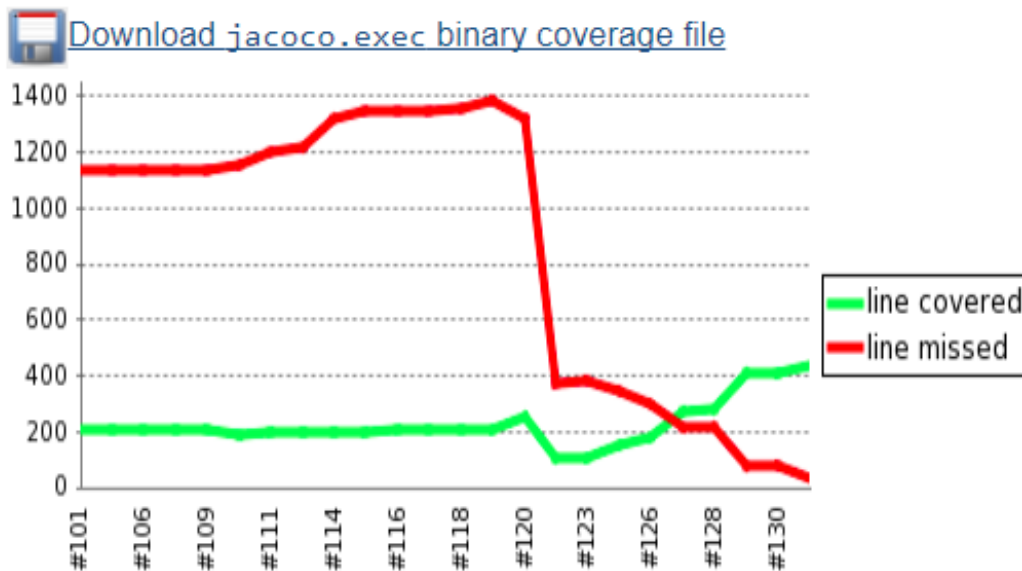
Coverage Summary

The following table shows coverage percentages from the Server API's controllers.

Instruction	Branch	Complexity	Line	Method	Class
93%	89%	87%	92%	95%	100%

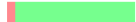
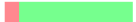
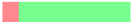
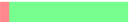
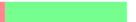
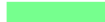
The following chart shows the coverage growth. Note that the drop off was created when we figure out how to successfully exclude library files from the testing report. These files were skewing our coverage report and making it seem like our tests were not effective.

JaCoCo Coverage Report





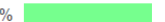







































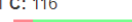




Coverage breakdown by source code file

The following screen show shows a more detailed breakdown by source file.

name	instruction	branch	complexity	line	method	class
edu.pennstate.science_olympiad.controllers	M: 134 C: 1846 93% 	M: 21 C: 165 89% 	M: 21 C: 137 87% 	M: 36 C: 433 92% 	M: 3 C: 61 95% 	M: 0 C: 8 100% 

Coverage Breakdown by Source File

name	instruction	branch	complexity	line	method	class
AuthController	M: 5 C: 134 96% 	M: 4 C: 14 78% 	M: 4 C: 8 67% 	M: 2 C: 33 94% 	M: 0 C: 3 100% 	M: 0 C: 1 100% 
BuildingController	M: 0 C: 142 100% 	M: 2 C: 18 90% 	M: 2 C: 14 88% 	M: 0 C: 34 100% 	M: 0 C: 6 100% 	M: 0 C: 1 100% 
CommsController	M: 21 C: 128 86% 	M: 2 C: 6 75% 	M: 2 C: 6 75% 	M: 5 C: 25 83% 	M: 1 C: 3 75% 	M: 0 C: 1 100% 
EventController	M: 22 C: 385 95% 	M: 3 C: 35 92% 	M: 3 C: 32 91% 	M: 6 C: 87 94% 	M: 1 C: 15 94% 	M: 0 C: 1 100% 
RoomController	M: 2 C: 111 98% 	M: 1 C: 11 92% 	M: 1 C: 10 91% 	M: 1 C: 30 97% 	M: 0 C: 5 100% 	M: 0 C: 1 100% 
SchoolController	M: 0 C: 151 100% 	M: 0 C: 14 100% 	M: 0 C: 13 100% 	M: 0 C: 37 100% 	M: 0 C: 6 100% 	M: 0 C: 1 100% 
TeamController	M: 5 C: 307 98% 	M: 1 C: 21 95% 	M: 1 C: 20 95% 	M: 1 C: 71 99% 	M: 0 C: 9 100% 	M: 0 C: 1 100% 
UsersController	M: 79 C: 488 86% 	M: 8 C: 46 85% 	M: 8 C: 34 81% 	M: 21 C: 116 85% 	M: 1 C: 14 93% 	M: 0 C: 1 100% 

Test Results

The following outline is an expert from the Jenkins server's console. It shows that all 54 server API tests successfully passed.

```
org.springframework.test.context.junit4.spring.SpringContextLoader$ContextLoaderServlet
[info] Using TestExecutionListeners: [org.springframework.test.context.web.ServletTestExecutionListener@1e40d2b7,
org.springframework.test.context.support DirtiesContextBeforeModesTestExecutionListener@2a694c7c,
org.springframework.test.context.support.DependencyInjectionTestExecutionListener@73a5b00e,
org.springframework.test.context.support DirtiesContextTestExecutionListener@3fe900b0,
org.springframework.test.context.transaction.TransactionalTestExecutionListener@27a4a890,
org.springframework.test.context.jdbc.SqlScriptsTestExecutionListener@474e0ab4]
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.022 sec
[info] Closing org.springframework.web.context.support.GenericWebApplicationContext@3a01773b: startup date [Wed Apr 18 21:58:41 EDT 2018]; root of context
hierarchy
```

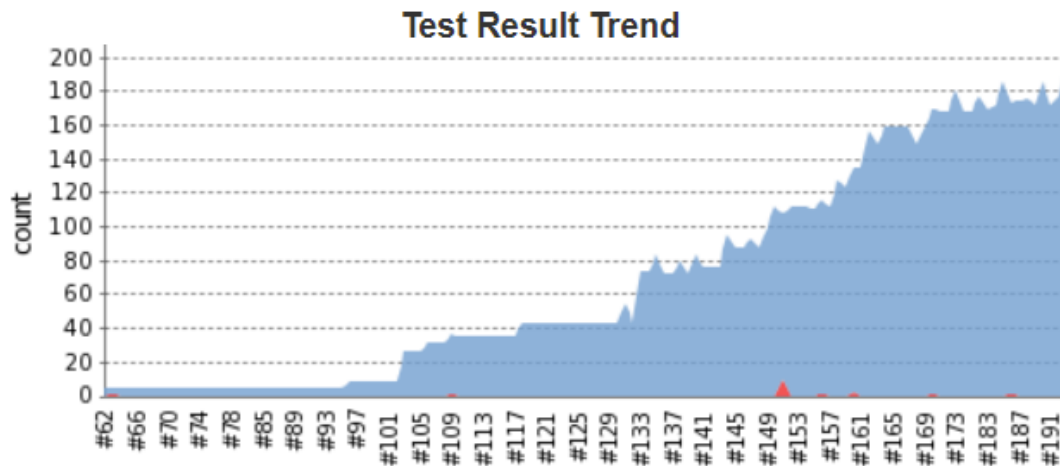
Results :

Tests run: 54, Failures: 0, Errors: 0, Skipped: 0

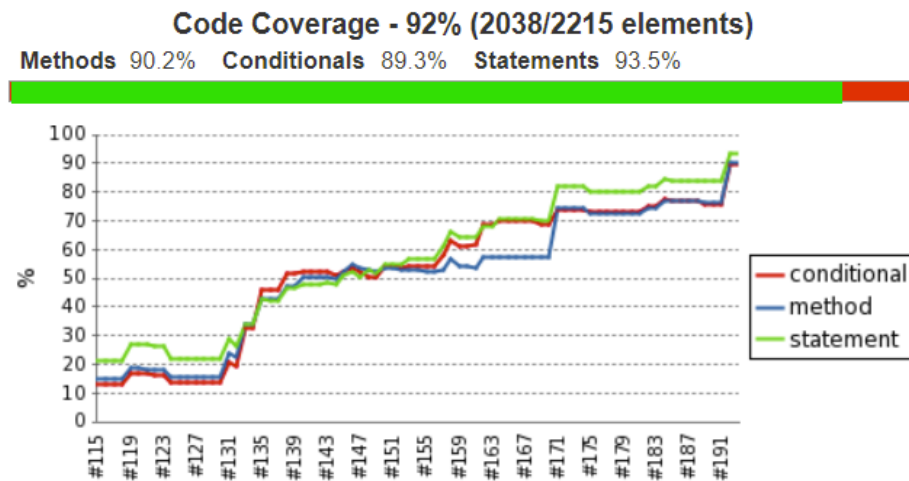
```
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.0:report (post-unit-test) @ ScienceOlympiad-Server ---
[INFO] Loading execution data file /var/lib/jenkins/workspace/SWENG500_API_Server/target/jacoco.exec
[INFO] Analyzed bundle 'ScienceOlympiad-Server' with 49 classes
[INFO]
```

Testing the UI

In order to reach 93.56% coverage 198 UI tests were developed. The following graphs show how our test coverage evolved as the project was developed. As you can see we began testing early in development and continued a steady growth as features were added.



The next graph shows the type of coverage we were able to achieve. While the 93.56% coverage is less than we had hoped to achieve we feel that it represents a substantial accomplishment, which covers the primary system functionality. The asynchronous nature of JavaScript meant that some components were difficult to test due to their state changes.



Test Results

The following report shows that app 198 tests successfully passed when the project was built.

All Tests

Package	Duration	Fail (diff)	Skip (diff)	Pass (diff)	Total (diff)
/src/adapters/___tests___/HttpRequest.test	11 ms	0	0	12	12
/src/components/Buildings/___tests___/BuildingSelector.test	12 ms	0	0	4	4
/src/components/Cards/___tests___/Card.test	38 ms	0	0	2	2
/src/components/Cards/___tests___/StatsCard.test	5 ms	0	0	1	1
/src/components/Cards/___tests___/UserCard.test	9 ms	0	0	1	1
/src/components/Footer/___tests___/Footer.test	6 ms	0	0	1	1
/src/components/Header/___tests___/Header.test	8 ms	0	0	4	4
/src/components/Header/___tests___/HeaderLinks.test	6 ms	0	0	3	3
/src/components/HttpExample/___tests___/ChuckNorris.test	21 ms	0	0	4	4
/src/components/Login/___tests___/Forgot.test	18 ms	0	0	5	5
/src/components/Login/___tests___/Login.test	35 ms	0	0	8 +8	8 +8
/src/components/Login/___tests___/Signup.test	0.19 sec	0	0	11	11
/src/components/Schools/___tests___/SchoolSelector.test	12 ms	0	0	4	4
/src/components/Sidebar/___tests___/Sidebar.test	16 ms	0	0	3	3
/src/components/Students/___tests___/StudentAdder.test	0.23 sec	0	0	4	4
/src/components/Students/___tests___/StudentViewer.test	77 ms	0	0	7	7
/src/components/Teams/___tests___/TeamAdder.test	39 ms	0	0	4	4
/src/components/Teams/___tests___/TeamViewer.test	0.36 sec	0	0	7	7
/src/containers/App/___tests___/App.test	0.11 sec	0	0	3	3
/src/containers/Login/___tests___/LoginContainer.test	37 ms	0	0	3	3
/src/elements/CustomButton/___tests___/CustomButton.test	2 ms	0	0	1	1
/src/elements/CustomSelector/___tests___/CustomDropdown.test	57 ms	0	0	4	4
/src/routes/___tests___/AuthenticatedRoute.test	13 ms	0	0	2	2
/src/routes/___tests___/app.test	1 ms	0	0	1	1
/src/utills/___tests___/AuthService.test	16 ms	0	0	10	10
/src/utills/___tests___/constants.test	1 ms	0	0	3	3
/src/utills/___tests___/validator.test	1 ms	0	0	2	2
/src/views/Dashboard/___tests___/Dashboard.test	44 ms	0	0	10	10
/src/views/EasterEgg/___tests___/sudoku.test	1 ms	0	0	1	1
/src/views/Events/___tests___/Events.test	0.87 sec	0	0	18	18
/src/views/Extras/___tests___/Extras.test	42 ms	0	0	6	6
/src/views/Locations/___tests___/Building.test	0.16 sec	0	0	2	2
/src/views/Locations/___tests___/Room.test	0.19 sec	0	0	10	10
/src/views/Schools/___tests___/School.test	0.19 sec	0	0	13 +13	13 +13
/src/views/Teams/___tests___/StudentTeamCreator.test	15 ms	0	0	3	3
/src/views/Teams/___tests___/TeamManagement.test	9 ms	0	0	3	3
/src/views/UserProfile/___tests___/UserProfile.test	0.3 sec	0	0	18	18

UI Test

The next section lists each UI test. You can click on any test for more information.

All files

83.29% Statements 1266/1520 75.4% Branches 426/565 76.32% Functions 390/511 83.58% Lines 1257/1504

File		Statements		Branches		Functions		Lines	
src	<div></div>	100%	1/1	100%	0/0	100%	0/0	100%	1/1
src/adapters	<div></div>	100%	45/45	100%	27/27	100%	13/13	100%	44/44
src/components/Buildings	<div></div>	100%	21/21	66.67%	8/12	84.62%	11/13	100%	20/20
src/components/Cards	<div></div>	100%	3/3	100%	14/14	100%	3/3	100%	3/3
src/components/Footer	<div></div>	100%	1/1	100%	0/0	100%	1/1	100%	1/1
src/components/Header	<div></div>	93.55%	29/31	87.5%	7/8	100%	12/12	93.55%	29/31
src/components/Login	<div></div>	95.59%	130/136	87.5%	49/56	89.74%	35/39	96.24%	128/133
src/components/Schools	<div></div>	100%	16/16	50%	4/8	90%	9/10	100%	15/15
src/components/Sidebar	<div></div>	100%	11/11	83.33%	10/12	100%	6/6	100%	11/11
src/components/Students	<div></div>	100%	76/76	100%	19/19	100%	37/37	100%	76/76
src/components/Teams	<div></div>	100%	105/105	100%	15/15	100%	46/46	100%	105/105
src/containers/App	<div></div>	96.77%	30/31	95%	19/20	100%	7/7	96.77%	30/31
src/containers/Login	<div></div>	100%	29/29	100%	13/13	100%	10/10	100%	29/29
src/elements/CustomButton	<div></div>	100%	4/4	100%	0/0	100%	1/1	100%	4/4
src/elements/CustomSelector	<div></div>	100%	26/26	78.57%	11/14	100%	12/12	100%	24/24
src/routes	<div></div>	100%	4/4	100%	4/4	100%	2/2	100%	4/4
src/utils	<div></div>	100%	104/104	100%	50/50	100%	28/28	100%	104/104
src/views/Dashboard	<div></div>	73.17%	60/82	37.5%	6/16	71.05%	27/38	78.95%	60/76
src/views/EasterEgg	<div></div>	100%	1/1	100%	0/0	100%	1/1	100%	1/1
src/views/Events	<div></div>	65.56%	238/363	63.79%	74/116	36.26%	33/91	65.56%	238/363
src/views/Extras	<div></div>	86.05%	37/43	57.14%	8/14	93.75%	15/16	86.05%	37/43
src/views/Locations	<div></div>	56.11%	101/180	37.5%	27/72	42.11%	24/57	56.11%	101/180
src/views/Schools	<div></div>	85.87%	79/92	61.76%	21/34	60.71%	17/28	85.87%	79/92
src/views/Teams	<div></div>	100%	19/19	100%	8/8	100%	12/12	100%	19/19
src/views/UserProfile	<div></div>	100%	96/96	96.97%	32/33	100%	28/28	100%	94/94

All files src

100% Statements 1/1 100% Branches 0/0 100% Functions 0/0 100% Lines 1/1

File ▲		Statements ▾	▾	Branches ▾	▾	Functions ▾	▾	Lines ▾	▾
setupTests.js	<div></div>	100%	1/1	100%	0/0	100%	0/0	100%	1/1

All files src/adapters

100% Statements 45/45 100% Branches 27/27 100% Functions 13/13 100% Lines 44/44

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
httpRequest.js	<div></div>	100%	45/45	100%	27/27	100%	13/13	100%	44/44

All files src/components/Buildings

100% Statements 21/21 66.67% Branches 8/12 84.62% Functions 11/13 100% Lines 20/20

File		Statements		Branches		Functions		Lines	
BuildingSelector.js	<div></div>	100%	21/21	66.67%	8/12	84.62%	11/13	100%	20/20

All files src/components/Cards

100% Statements 3/3 100% Branches 14/14 100% Functions 3/3 100% Lines 3/3

File ▲		Statements ⇅	⇅	Branches ⇅	⇅	Functions ⇅	⇅	Lines ⇅	⇅
Card.js	<div></div>	100%	1/1	100%	14/14	100%	1/1	100%	1/1
StatsCard.js	<div></div>	100%	1/1	100%	0/0	100%	1/1	100%	1/1
UserCard.js	<div></div>	100%	1/1	100%	0/0	100%	1/1	100%	1/1

All files src/components/Footer

100% Statements 1/1 100% Branches 0/0 100% Functions 1/1 100% Lines 1/1

File		Statements		Branches		Functions		Lines	
Footer.js	<div></div>	100%	1/1	100%	0/0	100%	1/1	100%	1/1

All files src/components/Header

93.55% Statements 29/31 87.5% Branches 7/8 100% Functions 12/12 93.55% Lines 29/31

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
Header.js	<div><div></div></div>	91.3%	21/23	87.5%	7/8	100%	6/6	91.3%	21/23
HeaderLinks.js	<div><div></div></div>	100%	8/8	100%	0/0	100%	6/6	100%	8/8

All files src/components/Login

95.59% Statements 130/136 87.5% Branches 49/56 89.74% Functions 35/39 96.24% Lines 128/133

File		Statements		Branches		Functions		Lines	
Forgot.js	<div></div>	100%	17/17	100%	2/2	100%	9/9	100%	17/17
Login.js	<div></div>	100%	24/24	100%	12/12	100%	10/10	100%	23/23
Signup.js	<div></div>	93.68%	89/95	83.33%	35/42	80%	16/20	94.62%	88/93

All files src/components/Schools

100% Statements 16/16 50% Branches 4/8 90% Functions 9/10 100% Lines 15/15

File		Statements		Branches		Functions		Lines	
SchoolSelector.js	<div></div>	100%	16/16	50%	4/8	90%	9/10	100%	15/15

All files src/components/Sidebar

100% Statements 11/11 83.33% Branches 10/12 100% Functions 6/6 100% Lines 11/11

File ▲		Statements ▼		Branches ▼		Functions ▼		Lines ▼	
Sidebar.js	<div></div>	100%	11/11	83.33%	10/12	100%	6/6	100%	11/11

All files src/components/Students

100% Statements 76/76 100% Branches 19/19 100% Functions 37/37 100% Lines 76/76

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
StudentAdder.js	<div></div>	100%	33/33	100%	10/10	100%	13/13	100%	33/33
StudentViewer.js	<div></div>	100%	43/43	100%	9/9	100%	24/24	100%	43/43

All files src/components/Teams

100% Statements 105/105 100% Branches 15/15 100% Functions 46/46 100% Lines 105/105

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
TeamAdder.js	<div></div>	100%	31/31	100%	8/8	100%	12/12	100%	31/31
TeamViewer.js	<div></div>	100%	74/74	100%	7/7	100%	34/34	100%	74/74

All files src/containers/App

96.77% Statements 30/31 95% Branches 19/20 100% Functions 7/7 96.77% Lines 30/31

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
App.js	<div></div>	96.77%	30/31	95%	19/20	100%	7/7	96.77%	30/31

All files src/containers/Login

100% Statements 29/29 100% Branches 13/13 100% Functions 10/10 100% Lines 29/29

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
LoginContainer.js	<div></div>	100%	29/29	100%	13/13	100%	10/10	100%	29/29

All files src/elements/CustomButton

100% Statements 4/4 100% Branches 0/0 100% Functions 1/1 100% Lines 4/4

File ▲		Statements ▾	▾	Branches ▾	▾	Functions ▾	▾	Lines ▾	▾
CustomButton.js	<div></div>	100%	4/4	100%	0/0	100%	1/1	100%	4/4

All files src/elements/CustomSelector

100% Statements 26/26 78.57% Branches 11/14 100% Functions 12/12 100% Lines 24/24

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
CustomDropdown.js	<div></div>	100%	26/26	78.57%	11/14	100%	12/12	100%	24/24

All files src/routes

100% Statements 4/4 100% Branches 4/4 100% Functions 2/2 100% Lines 4/4

File ▲		Statements ▾	▾	Branches ▾	▾	Functions ▾	▾	Lines ▾	▾
AuthenticatedRoute.js	<div></div>	100%	3/3	100%	4/4	100%	2/2	100%	3/3
app.js	<div></div>	100%	1/1	100%	0/0	100%	0/0	100%	1/1

All files src/utils

100% Statements 104/104 100% Branches 50/50 100% Functions 28/28 100% Lines 104/104

File		Statements		Branches		Functions		Lines	
AuthService.js	<div></div>	100%	69/69	100%	32/32	100%	22/22	100%	69/69
constants.js	<div></div>	100%	10/10	100%	6/6	100%	4/4	100%	10/10
validator.js	<div></div>	100%	25/25	100%	12/12	100%	2/2	100%	25/25

All files src/views/Dashboard

73.17% Statements 60/82 37.5% Branches 6/16 71.05% Functions 27/38 78.95% Lines 60/76

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
CoachCount.js	<div><div></div></div>	78.57%	11/14	50%	1/2	71.43%	5/7	84.62%	11/13
Dashboard.js	<div><div></div></div>	37.5%	3/8	16.67%	1/6	100%	1/1	37.5%	3/8
EventCount.js	<div><div></div></div>	75%	12/16	50%	1/2	62.5%	5/8	85.71%	12/14
JudgeCount.js	<div><div></div></div>	78.57%	11/14	50%	1/2	71.43%	5/7	84.62%	11/13
TeamCount.js	<div><div></div></div>	78.57%	11/14	50%	1/2	71.43%	5/7	84.62%	11/13
TodaysEvents.js	<div><div></div></div>	75%	12/16	50%	1/2	75%	6/8	80%	12/15

All files src/views/EasterEgg

100% Statements 1/1 100% Branches 0/0 100% Functions 1/1 100% Lines 1/1

File ▲		Statements ▾	▾	Branches ▾	▾	Functions ▾	▾	Lines ▾	▾
sudoku.js	<div></div>	100%	1/1	100%	0/0	100%	1/1	100%	1/1

All files src/views/Events

65.56% Statements 238/363 63.79% Branches 74/116 36.26% Functions 33/91 65.56% Lines 238/363

File ▲		Statements ▼		Branches ▼		Functions ▼		Lines ▼	
EventDetail.js	<div><div></div></div>	41.24%	40/97	26.67%	8/30	25.64%	10/39	41.24%	40/97
Events.js	<div><div></div></div>	74.44%	198/266	76.74%	66/86	44.23%	23/52	74.44%	198/266

All files src/views/Extras

86.05% Statements 37/43 57.14% Branches 8/14 93.75% Functions 15/16 86.05% Lines 37/43

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
Extras.js	<div></div>	86.05%	37/43	57.14%	8/14	93.75%	15/16	86.05%	37/43

All files src/views/Locations

56.11% Statements 101/180 37.5% Branches 27/72 42.11% Functions 24/57 56.11% Lines 101/180

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
Buildings.js	<div><div></div></div>	27.59%	24/87	13.89%	5/36	25.93%	7/27	27.59%	24/87
Rooms.js	<div><div></div></div>	82.8%	77/93	61.11%	22/36	56.67%	17/30	82.8%	77/93

All files src/views/Schools

85.87% Statements 79/92 61.76% Branches 21/34 60.71% Functions 17/28 85.87% Lines 79/92

File		Statements		Branches		Functions		Lines	
Schools.js	<div><div></div></div>	85.87%	79/92	61.76%	21/34	60.71%	17/28	85.87%	79/92

All files src/views/Teams

100% Statements 19/19 100% Branches 8/8 100% Functions 12/12 100% Lines 19/19

File		Statements		Branches		Functions		Lines	
StudentTeamCreator.js	<div></div>	100%	10/10	100%	0/0	100%	7/7	100%	10/10
TeamManagement.js	<div></div>	100%	9/9	100%	8/8	100%	5/5	100%	9/9

All files src/views/UserProfile

100% Statements 96/96 96.97% Branches 32/33 100% Functions 28/28 100% Lines 94/94

File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
UserProfile.js	<div></div>	100%	96/96	96.97%	32/33	100%	28/28	100%	94/94

All files / **src** setupTests.js

100% Statements 1/1 100% Branches 0/0 100% Functions 0/0 100% Lines 1/1

```
1  import { configure } from 'enzyme';  
2  import Adapter from 'enzyme-adapter-react-16';  
3  
4  37x configure( { adapter: new Adapter() } );
```

All files / src/adapters httpRequest.js

100% Statements 45/45

100% Branches 27/27

100% Functions 13/13

100% Lines 44/44

```
1  /** http_request.js
2      *
3      * Wrapper for Axios library for making HTTP Requests and handling sessions
4      */
5      import AuthService from '../utils/AuthService';
6      import axios from 'axios';
7      28x var emptyHeader = {};
8
9      28x axios.interceptors.response.use(response => {
10     27x     return response;
11     }, error => {
12     29x         var filteredErr = HttpRequest.errorHandler(error);
13     29x         return Promise.reject(filteredErr);
14     })
15
16
17     export default class HttpRequest {
18
19         static httpRequest(url, httpMethod, httpHeader, requestData, needsSession = false) {
20     58x             if (httpHeader == null) httpHeader = emptyHeader;
21
22     58x             var promise = new Promise(function (resolve, reject) {
23
24                 //IF request requires session, check if User is still Authorized
25                 // and update timestamp if they are
26     58x                 if (needsSession && !AuthService.isAuthenticated(true)) {
27     1x                     window.location = '/#/login';
28     1x                     reject('ERROR: Must be an authorized user. Please login.');
```

```

55 3x      resolve({status: result.status, body: result.data});
56      }).catch(function (error) {
57 3x      reject(error);
58      })
59  } else {
60 1x      reject('ERROR: The requested HTTP method ' + httpMethod + ' is not supported');
61  }
62  });
63
64 58x  return promise;
65  }
66
67  static errorHandler(error) {
68 34x  var errResponse = {};
69 34x  errResponse.status = 500;
70
71 34x  if (error.response) {
72 33x      if (error.response.status === 401) {
73 3x          if(!error.response.data.includes("incorrect")) {
74 2x              AuthService.revokeAuth(true);
75 2x              window.location = '/#/login';
76          }
77      }
78
79 33x      if (error.response.status) {
80 32x          errResponse.status = error.response.status;
81      }
82
83 33x      if (error.response.data) {
84 19x          errResponse.message = error.response.data;
85      } else {
86 14x          errResponse.message = "SOMETHING BAD HAPPENED WITH RESPONSE FROM BACKEND.";
87      }
88  } else {
89 1x      errResponse.message = "SOMETHING BAD HAPPENED WITH RESPONSE FROM BACKEND: " + error;
90  }
91
92 34x  return errResponse;
93  }
94  }
95

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Buildings BuildingSelector.js

100% Statements 21/21 66.67% Branches 8/12 84.62% Functions 11/13 100% Lines 20/20

```
1 import React from 'react';
2 import SelectField from 'material-ui/SelectField';
3 import MenuItem from 'material-ui/MenuItem';
4 import HttpRequest from '../../adapters/httpRequest';
5 import constants from '../../utils/constants';
6
7 class BuildingSelector extends React.Component {
8
9     constructor(props) {
10         4x super(props);
11
12         4x this.state = {
13             errorMsg: this.props.errorMsg,
14             buildingList: [],
15             roomList: [],
16         }
17     }
18
19     sortByKey(array, key) {
20         6x return array.sort(function(a, b) {
21             6x var x = a[key]; var y = b[key];
22             6x return ((x < y) ? -1 : ((x > y) ? 1 : 0));
23         });
24     }
25
26     componentWillMount() {
27         3x var _this = this;
28         3x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getAllRooms',
29         3x 'GET', constants.useCredentials(), null).then(function (result) {
30             _this.setState({roomList: _this.sortByKey(result.body, "roomName")})
31         }).catch(function (error) {
32
33         })
34         3x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getBuildings',
35         3x 'GET', constants.useCredentials(), null).then(function (result) {
36             _this.setState({buildingList: _this.sortByKey(result.body, "building")})
37         }).catch(function (error) {
38
39         })
40     }
41
42     componentWillReceiveProps(nextProps) {
43         1x
44         1x
45         E if(JSON.stringify(this.props.errorMsg) !== JSON.stringify(nextProps.errorMsg))
46             this.setState({errorMsg: nextProps.errorMsg})
47     }
48     1x
49     updateValues = (event, index, value) => {
50         this.props.callBack(event, index, value);
51     }
52     12x
53     5x render() {
54         if (Object.keys(this.state.buildingList).length !== 0 && Object.keys(this.state.roomList).length
```

```

55     !== 0 ) {
56         return (
57             <SelectField
58                 name={"buildings"}
59                 hintText={this.props.hintText}
60                 errorText={this.state.errorMsg}
61                 floatingLabelText={this.props.labelText}
62                 onChange={this.updateValues}
63                 fullWidth={true}
64                 value={this.props.selected}>
65                 {
66                     Object.keys(this.state.buildingList).map(function (key) {
67                         var items = Object.keys(this.state.roomList).map(function (roomKey){
68
69                             if(this.state.buildingList[key].id.indexOf(this.state.roomList[roomKey].buildingID) > -1) {
70                                 return (
71                                     <MenuItem key={this.state.roomList[roomKey].id}
72                                         primaryText={this.state.buildingList[key].building + ' --
73                                     ' + this.state.roomList[roomKey].roomName}
74                                     // secondaryText={this.state.roomList[roomKey].roomName}
75                                     value={this.state.roomList[roomKey].id}/>
76                                 )
77                             }, this)
78                         }, this)
79                         return (
80                             items
81                         )
82                     }, this)
83                 }
84             </SelectField>
85         )
86     }
87 }
88 else
    return ("ERROR LOADING BUILDINGS")
}

export default BuildingSelector;

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Cards Card.js

100% Statements

1/1

100% Branches

14/14

100% Functions

1/1

100% Lines

1/1

```
1 import React, { Component } from 'react';
2
3
4 export class Card extends Component{
5   render(){
6     13x return (
7       <div className={"card"+(this.props.plain ? " card-plain":"")} style={this.props.style}>
8         <div className={"header"
9           + (this.props.hCenter ? " text-center":"")}>
10          <h4 className="title">{this.props.title}</h4>
11          <p className="category">{this.props.category}</p>
12        </div>
13        <div className={"content"
14          + (this.props.ctAllIcons ? " all-icons":"" )
15          + (this.props.ctTableFullWidth ? " table-full-width":"" )
16          + (this.props.ctTableResponsive ? " table-responsive":"" )
17          + (this.props.ctTableUpgrade ? " table-upgrade":"" )}>
18
19          {this.props.content}
20
21          <div className="footer">
22            {this.props.legend}
23            {this.props.stats != null ? <hr />:""}
24            <div className="stats">
25              <i className={this.props.statsIcon}></i> {this.props.stats}
26            </div>
27          </div>
28        </div>
29      </div>
30    );
31  }
32 }
33
34 export default Card;
35
```

All files / src/components/Cards StatsCard.js

100% Statements 1/1

100% Branches 0/0

100% Functions 1/1

100% Lines 1/1

```
1 import React, { Component } from 'react';
2 import { Row, Col } from 'react-bootstrap';
3
4 export class StatsCard extends Component{
5   render(){
6     11x    return (
7       <div className="card card-stats" style={{'text-align':'left'}}>
8         <div className="content">
9           <div className="numbers">
10            <p style={{'text-align':'left','font-weight':'bold','font-size':'14pt'}}>
11 {this.props.bigIcon} {this.props.statsText}</p>
12            </div>
13            <div className="numbers">
14              <p style={{'text-align':'left'}}>{this.props.statsValue}</p>
15            </div>
16          </div>
17        </div>
18      </div>
19    );
20  }
21 }
22 export default StatsCard;
```

All files / src/components/Cards UserCard.js

100% Statements 1/1 100% Branches 0/0 100% Functions 1/1 100% Lines 1/1

```
1  import React, { Component } from 'react';
2
3
4  export class UserCard extends Component{
5      render(){
6  1x      return (
7          <div className="card card-user">
8              <div className="image">
9                  <img src={this.props.bgImage} alt="..." />
10             </div>
11             <div className="content">
12                 <div className="author">
13                     <a href="#/app/user">
14                         <img className="avatar border-gray" src={this.props.avatar} alt="..." />
15                         <h4 className="title">
16                             {this.props.name}
17                         <br />
18                         <small>{this.props.userName}</small>
19                     </h4>
20                     </a>
21                 </div>
22                 <p className="description text-center">
23                     {this.props.description}
24                 </p>
25             </div>
26             <hr />
27             <div className="text-center">
28                 {this.props.socials}
29             </div>
30         </div>
31     );
32 }
33
34
35 export default UserCard;
36
```


All files / src/components/Footer Footer.js

100% Statements 1/1

100% Branches 0/0

100% Functions 1/1

100% Lines 1/1

```
1  import React, {Component} from 'react';
2  import { Grid } from 'react-bootstrap';
3
4  class Footer extends Component {
5      render() {
6          3x      return (
7              <footer className="footer">
8                  <Grid>
9                      <nav className="pull-left">
10                         <ul>
11                             <li>
12                                 <a href="/#/app/dashboard">
13                                     Home
14                                 </a>
15                             </li>
16                             <li>
17                                 <a href="http://www.psu.edu">
18                                     PSU
19                                 </a>
20                             </li>
21                             <li>
22                                 <a href="https://www.soinc.org/">
23                                     Science Olympiad
24                                 </a>
25                             </li>
26                         </ul>
27                     </nav>
28                     <p className="copyright pull-right">
29                         &copy; {(new Date()).getFullYear()} - SWENG500 - Group 5
30                     </p>
31                 </Grid>
32             </footer>
33         );
34     }
35 }
36
37 export default Footer;
38
```

All files / src/components/Header Header.js

91.3% Statements

21/23

87.5% Branches

7/8

100% Functions

6/6

91.3% Lines

21/23

```
1 import React, { Component } from 'react';
2 import { Navbar } from 'react-bootstrap';
3
4 import HeaderLinks from './HeaderLinks.js';
5
6 import appRoutes from '../../routes/app.js';
7
8 class Header extends Component{
9   constructor(props){
10     5x super(props);
11     5x this.mobileSidebarToggle = this.mobileSidebarToggle.bind(this);
12     5x this.state = {
13       sidebarExists: false
14     };
15   }
16   mobileSidebarToggle(e){
17     2x if(this.state.sidebarExists === false){
18       1x this.setState({
19         sidebarExists : true
20       });
21     }
22     2x e.preventDefault();
23     2x document.documentElement.classList.toggle('nav-open');
24     2x var node = document.createElement('div');
25     2x node.id = 'bodyClick';
26     2x node.onclick = function(){
27       1x this.parentElement.removeChild(this);
28       document.documentElement.classList.toggle('nav-open');
29     };
30     2x document.body.appendChild(node);
31   }
32   getBrand(){
33     8x var name = null;
34     8x appRoutes.map((prop,key) => {
35       80x if(prop.redirect){
36         8x if(prop.path === this.props.location.pathname){
37           name = prop.name;
38         }
39       }
40       }else{
41         72x if(prop.path === this.props.location.pathname){
42           5x name = prop.name;
43         }
44       }
45     }
46     80x return null;
47   })
48   8x return name;
49 }
50 render(){
51   8x return (
52     <Navbar fluid>
53       <Navbar.Header>
54         <Navbar.Brand>
55           <div id="brand-display">{this.getBrand()}</div>
```

```
55         </Navbar.Brand>
56         <Navbar.Toggle onClick={this.mobileSidebarToggle}/>
57     </Navbar.Header>
58     <Navbar.Collapse>
59         <HeaderLinks />
60     </Navbar.Collapse>
61 </Navbar>
62 );
63 }
64 }
65
66 export default Header;
67
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Header HeaderLinks.js

100% Statements 8/8

100% Branches 0/0

100% Functions 6/6

100% Lines 8/8

```
1  import React, {Component} from 'react';
2  import { NavItem, Nav } from 'react-bootstrap';
3
4  import AuthService from '../../utils/AuthService';
5
6
7  class HeaderLinks extends Component{
8
9      constructor(props) {
10  4x        super(props);
11
12  4x        this.onLogoutClick = this.onLogoutClick.bind(this);
13      }
14
15      onLogoutClick(event) {
16  2x        event.preventDefault();
17
18  2x        AuthService.logout().then(function (result) {
19  1x            console.log(result);
20        }).catch(function (error) {
21  1x            console.log(error);
22        }).then( () => {
23  2x            window.location = '/';
24        });
25
26      }
27
28      render(){
29  5x        return (
30            <div>
31                <Nav pullRight>
32                    <NavItem eventKey={1} href="/#/app/user">Account</NavItem>
33                    <NavItem eventKey={3} onClick={this.onLogoutClick}>Log out</NavItem>
34                </Nav>
35            </div>
36        );
37      }
38  }
39
40  export default HeaderLinks;
41
```

All files / src/components/Login Forgot.js

100% Statements 17/17

100% Branches 2/2

100% Functions 9/9

100% Lines 17/17

```
1 import React, { Component } from 'react';
2 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
3 import RaisedButton from 'material-ui/RaisedButton';
4 import TextField from 'material-ui/TextField';
5 import AppBar from 'material-ui/AppBar';
6 import NotificationSystem from 'react-notification-system';
7
8 import {Redirect} from 'react-router-dom';
9
10 import AuthService from '../../utils/AuthService';
11 import HttpRequest from "../../adapters/httpRequest";
12 import constants from "../../utils/constants";
13 import {style} from "../../variables/Variables";
14 import $ from "jquery";
15 import Login from "../Login";
16 import PropTypes from "prop-types";
17
18 class Forgot extends Component {
19   constructor(props) {
20     super(props);
21
22     this.state = {
23       email: ''
24     };
25
26     this.handleClick = this.handleClick.bind(this);
27   }
28
29   handleClick(event) {
30
31     if(this.state.email.trim() !== '') {
32       var body = {};
33       var _this = this;
34
35       body.emailAddress = this.state.email.trim();
36
37       _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
38         '/sweng500/resetPassword', 'POST', constants.useCredentials(), body).then(function (result) {
39         _this.setState({email : "", emailRequired: null}, () => {
40           _this.props.notify(
41             "Success: Password reset email sent",
42             "success",
43             "tc",
44             5
45           );
46           $('#reset-container').addClass('collapse');
47         });
48       }).catch(function (error) {
49         _this.setState({
50           email: _this.state.email.trim(),
51           emailRequired: "Provide valid email address."
52         }, () => {
53           _this.props.notify(
54             "Error: Could not send password reset email. Provide a valid email address.",
```

```

55         "error",
56         "tc",
57         5
58     );
59     });
60 })
61
62 1x    } else {
63        this.setState({
64            email: this.state.email.trim(),
65            emailRequired: "An email address is required."
66        })
67    }
68 }
69
70 10x  render() {
71      return (
72          <div id='login-div'>
73              <MuiThemeProvider>
74                  <div>
75                      <AppBar showMenuIconButton={false} title="Password Recovery"/>
76                      <div id='reset-container'>
77                          <TextField
78                              hintText="Enter your email address"
79                              errorText={this.state.emailRequired}
80 1x          floatingLabelText="Email Address"
81                              onChange={(event, newValue) => this.setState({email: newValue})}
82                              required={true}
83                          />
84                          <br/>
85 4x          <RaisedButton label="Submit" primary={true} style={{margin:15}}
86                              onClick={(event) => this.handleClick(event)}/>
87                      </div>
88                  </div>
89              </MuiThemeProvider>
90          </div>
91      )
92  }
93 }
94
95 export default Forgot;

```

All files / src/components/Login Login.js

100% Statements 24/24

100% Branches 12/12

100% Functions 10/10

100% Lines 23/23

```
1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
4 import RaisedButton from 'material-ui/RaisedButton';
5 import TextField from 'material-ui/TextField';
6 import AppBar from 'material-ui/AppBar';
7 import PasswordField from 'material-ui-password-field'
8
9 import {Redirect} from 'react-router-dom';
10
11 import AuthService from '../utils/AuthService';
12
13 class Login extends Component {
14   constructor(props) {
15     8x super(props);
16
17     8x this.state = {
18       emailAddress: '',
19       password: '',
20       emailRequired: undefined,
21       passRequired: undefined,
22       redirect: false
23     }
24
25     8x this.handleClick = this.handleClick.bind(this);
26   }
27
28   handleClick(event) {
29     5x if(event) { event.preventDefault(); }
30
31     5x if(this.state.emailAddress.trim()) {
32       4x if(this.state.password.trim()) {
33
34         3x AuthService.login(this.state.emailAddress.toLowerCase(),
35 this.state.password).then(function (result) {
36
37         2x   }).catch(function (error) {
38           console.log(error);
39         3x   }).then(() => {
40           1x     if (AuthService.isLoggedIn()) {
41             this.setState({
42               redirect: true
43             })
44           2x     } else {
45             this.props.notify(
46               "ERROR: Please provide valid login credentials.",
47               "error",
48               "tc",
49               6
50             )
51           }
52         }
53       );
54     1x   } else {
```

```

55         this.setState({
56             emailRequired: undefined,
57             passRequired: "Password is required."
58         })
59     }
60     1x } else {
61         this.setState({
62             emailAddress: this.state.emailAddress.trim(),
63             emailRequired: "An email address is required.",
64             passRequired: undefined
65         })
66     }
67 }
68
69 8x componentDidMount() {
70     1x if(AuthService.isLoggedIn()) {
71         this.setState({
72             redirect: true
73         })
74     }
75 }
76
77 19x render() {
78     16x if(!this.state.redirect) {
79         return (
80             <div id='login-div'>
81                 <MuiThemeProvider>
82                     <div style={{width:275}}>
83                         <AppBar showMenuIconButton={false} title="Login"/>
84                         <TextField
85                             name={"email"}
86                             type="Email Address"
87                             hintText="Enter your email address"
88                             errorText={this.state.emailRequired}
89                             2x floatingLabelText="Email Address"
90                             onChange={(event, newValue) => this.setState({emailAddress: newValue})}
91                             required={true}
92                             fullWidth={true}
93                         />
94                         <br/>
95                         <PasswordField
96                             name={"password"}
97                             type="password"
98                             errorText={this.state.passRequired}
99                             2x floatingLabelText="Password"
100                             onChange={(event, newValue) => this.setState({password: newValue})}
101                             fullWidth={true}
102                             required={true}
103                         />
104                         <br/>
105                         7x <RaisedButton label="Submit" primary={true} style={{margin:15}}
106                             onClick={(event) => this.handleClick(event)}/>
107                     </div>
108                 </MuiThemeProvider>
109             </div>
110         )
111     3x } else {
112         return (
113             <Redirect from="/" to="/app/dashboard" />
114         )

```



```
115     }
116   }
117 }
118 3x
119 Login.propTypes = {
120   notify: PropTypes.any,
121 3x }
122 Login.defaultProps = {
123   notify : null
124 }
125
126 export default Login;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Login Signup.js

93.68% Statements 89/95 83.33% Branches 35/42 80% Functions 16/20 94.62% Lines 88/93

```
1 import React from 'react';
2 import {
3     Step,
4     Stepper,
5     StepLabel,
6     StepContent,
7 } from 'material-ui/Stepper';
8 import RaisedButton from 'material-ui/RaisedButton';
9 import FlatButton from 'material-ui/FlatButton';
10 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
11 import TextField from 'material-ui/TextField';
12 import AppBar from 'material-ui/AppBar';
13 import InputMask from 'react-input-mask';
14 import {Row, Col, Grid} from 'react-bootstrap';
15 import HttpRequest from '../adapters/httpRequest';
16 import constants from '../utils/constants';
17 import PasswordField from 'material-ui-password-field';
18 import SchoolSelector from '../Schools/SchoolSelector';
19
20 class Signup extends React.Component {
21
22     constructor(props) {
23         11x super(props);
24
25         11x this.state = {
26             finished: false,
27             stepIndex: 0,
28             firstName: '',
29             lastName: '',
30             phoneNumber: '',
31             emailAddress: '',
32             password: '',
33             confirm: '',
34             school: '',
35             httpResponse: '',
36             accountMessage: '',
37             schoolList: {}
38         };
39
40         11x this.callBack = this.callBack.bind(this);
41     }
42
43     componentDidMount() {
44         11x var _this = this;
45
46         11x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getSchools',
47 'GET', null, null).then(function (result) {
48         11x
49             _this.state.schoolList = result.body;
50
51             }).catch(function (error) {
52
53             })
54     }
```

```

55
56 // Checks to see if an email has a host, @ symbols, and domain.
57 2x validEmail(text) {
58 2x     let reg = /^w+([\.-]?\w+)*@w+([\.-]?\w+)*(\.w{2,3})+$/;
59 1x     if (reg.test(text) === false)
60         return true;
61 1x     else
62         return false;
63 }
64
65 // Check if a password is valid. 8 characters, uppercase, lowercase, and numbers
66 5x validPassword(text) {
67 2x     if (text.length < 8) return false;
68 2x     var hasUpperCase = /[A-Z]/.test(text);
69 2x     var hasLowerCase = /[a-z]/.test(text);
70 2x     var hasNumbers = /\d/.test(text);
71 2x     var hasNonalphas = /\W/.test(text);
72     if (hasUpperCase + hasLowerCase + hasNumbers + hasNonalphas < 3) return false;
73 2x
74     return true;
75 }
76
77 callback(event, index, value) {
78
79     this.setState({school: value});
80 }
81
82 // Handles the next button
83 9x handleNext = () => {
84 9x     const {stepIndex} = this.state;
85 9x     var _this = this;
86     var body = {};
87
88 9x     // Flag to indicate if we can proceed
89     var missingInfo = false;
90
91
92 9x     // Form page 1
93     if (stepIndex === 0) {
94
95         // Checks first name
96         1x if (this.state.firstName.trim()) {
97             this.setState({
98                 firstName: this.state.firstName.trim(),
99                 firstNameRequired: undefined
100             })
101         }
102         2x else {
103             this.setState({
104                 firstName: this.state.firstName.trim(),
105                 firstNameRequired: "First name is required."
106             })
107             missingInfo = true;
108         }
109
110         // Checks last name
111         1x if (this.state.lastName.trim()) {
112             this.setState({
113                 lastName: this.state.lastName.trim(),
114                 lastNameRequired: undefined

```

```

115     })
116   }
117   2x else {
118     this.setState({
119       lastName: this.state.lastName.trim(),
120       lastNameRequired: "Last name is required."
121     })
122     missingInfo = true;
123   }
124
125   3x // Checks phone number
126   1x if (this.state.phoneNumber.trim()) {
127     this.setState({
128       phoneNumber: this.state.phoneNumber.trim(),
129       phoneNumberRequired: undefined
130     })
131   }
132   2x else {
133     this.setState({
134       phoneNumber: this.state.phoneNumber.trim(),
135       phoneNumberRequired: "A phone number is required."
136     })
137     missingInfo = true;
138   }
139
140   3x // Checks emails address is not blank
141   if (this.state.emailAddress.trim()) {
142
143     2x // If the address is not valid
144     1x if (this.validEmail(this.state.emailAddress.trim())) {
145       this.setState({
146         emailAddress: this.state.emailAddress.trim(),
147         emailAddressRequired: "A valid email address is required."
148       })
149       1x
150       missingInfo = true;
151     }
152     else {
153
154       // Check to see if the email address already exists
155       1x
156       body.emailAddress = this.state.emailAddress.toLowerCase();
157       1x
158       _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
159       '/sweng500/emailAvailable', 'POST', null, body ).then(function (result) {
160       1x
161
162         _this.setState({
163           emailAddress: _this.state.emailAddress.trim(),
164           emailAddressRequired: "",
165           httpResponse: result,
166         })
167       1x
168       E if (missingInfo === false)
169       {
170         _this.setState({
171           stepIndex: stepIndex + 1
172         })
173       }
174

```

```

175         }).catch(function (error) {
176
177
178         if (_this.state.httpResponse.status !== 200)
179             missingInfo = true;
180
181         _this.setState({
182             emailAddress: _this.state.emailAddress.trim(),
183             emailAddressRequired: "This address has already been registered.",
184             httpResponse: error
185         })
186     })
187 }
188
189 1x }
190 else {
191     this.setState({
192         emailAddress: this.state.emailAddress.trim(),
193         1x emailAddressRequired: "An email address is required."
194     })
195     missingInfo = true;
196 }
197
198 6x } // Form page 2
199
200 else if (stepIndex === 1) {
201     6x
202     1x // Checks to see if the passwords are equal to each other
203     if (this.state.password.trim() !== this.state.confirm.trim()) {
204         this.setState({
205             confirmRequired: "Passwords must match.",
206             1x passwordRequired: "Passwords must match."
207         })
208         this.props.notify(
209             "ERROR: Your passwords do not match.",
210             "error",
211             "tc",
212             1x 6
213         );
214         missingInfo = true;
215     }
216     5x else {
217         2x // Tests the password complexity
218         if (this.validPassword(this.state.password.trim())) {
219             this.setState({
220                 password: this.state.password.trim(),
221                 passwordRequired: undefined
222             })
223         }
224         3x else {
225             this.props.notify(
226                 "ERROR: Your password must be 8 or more characters, contain capital letters, lower
227 case letters, and at least one number.",
228                 "error",
229                 3x "tc",
230                 10
231             );
232             this.setState({
233                 confirmRequired: "A complex password is required.",
234                 3x passwordRequired: "A complex password is required."

```

```

235         })
236
237         missingInfo = true;
238     }
239 }
240
241 if (this.state.school.trim() === '')
242 {
243     this.setState({
244         schoolRequired: "Please select your school district."
245     })
246 }
247 else {
248     this.setState({schoolRequired: undefined})
249 }
250
251 if (!missingInfo) {
252
253     this.setState({
254         stepIndex: stepIndex + 1,
255         finished: stepIndex >= 2,
256     });
257
258     // Create the user account
259     var cleanPhoneNumber = this.state.phoneNumber;
260     cleanPhoneNumber = cleanPhoneNumber.replace(/\s/g, ''); // Remove spaces
261     cleanPhoneNumber = cleanPhoneNumber.replace(/\(|\)/g, ''); // Remove ( and )
262     cleanPhoneNumber = cleanPhoneNumber.replace(/-/g, ""); // Remove -
263     cleanPhoneNumber = '+' + cleanPhoneNumber; // Add +
264
265     body = {};
266
267     body.firstName = this.state.firstName;
268     body.lastName = this.state.lastName;
269     body.emailAddress = this.state.emailAddress.toLowerCase();
270     body.phoneNumber = cleanPhoneNumber;
271     body.password = this.state.password;
272     body.minutesBeforeEvent = 10;
273     var schoolID = this.state.school;
274
275     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
276     '/sweng500/addUser/?userType=COACH&schoolID=' + _this.state.school, 'POST', null, body ).then(function
277 (result) {
278
279         if (result.status === 200)
280             _this.setState({accountMessage: "Congratulations! Your account has been created.
281 Please return to the login screen."})
282
283         }).catch(function (error) {
284
285             _this.setState({accountMessage: "There was an error creating your account. Please try
286 again later."})
287
288         })
289     }
290 }
291
292 }
293
294 }

```

```

    };

    handlePrev = () => {
      const {stepIndex} = this.state;

      if (stepIndex > 0) {
        this.setState({stepIndex: stepIndex - 1});
      }
    };

    render() {
      const {finished, stepIndex} = this.state;
      const contentStyle = {margin: '0 16px'};
      return (
        <MuiThemeProvider>
          <div>
            <AppBar showMenuIconButton={false} title="Account Registration"/>
            <Stepper activeStep={stepIndex} orientation={'vertical'} style={{maxWidth: 700,
margin: '0',border: '1 solid black'}}>
              <Step>
                <StepLabel>Personal Information</StepLabel>
                <StepContent>
                  <Grid>
                    <Row className="show-grid">
                      <Col xs={7} md={3}>
                        <TextField
                          name="fname"
                          hintText="Enter your first name"
                          errorText={this.state.firstNameRequired}
                          floatingLabelText="First name"
                          onChange={(event, newValue) => this.setState({firstName:
newValue})}

                          value={this.state.firstName}
                          fullWidth={true}
                          required={true}/>
                      </Col>
                      <Col xs={7} md={3}>
                        <TextField
                          name="lname"
                          hintText="Enter your last name"
                          errorText={this.state.lastNameRequired}
                          floatingLabelText="Last name"
                          onChange={(event, newValue) => this.setState({lastName:
newValue})}

                          value={this.state.lastName}
                          fullWidth={true}
                          required={true}/>
                      </Col>
                    </Row>
                    <Row className="show-grid">
                      <Col xs={7} md={3}>
                        <TextField
                          name="phone"
                          errorText={this.state.phoneNumberRequired}
                          floatingLabelText="Phone number"
                          onChange={(event, newValue) => this.setState({phoneNumber:
newValue})}

                          value={this.state.phoneNumber}
                          fullWidth={true}
                          required={true}>

```

```

355 1x
356         <InputMask mask="1 (999) 999-9999" maskChar="#"
357             value={this.state.phoneNumber}/>
358     </TextField>
359 </Col>
360 <Col xs={7} md={3}>
361     <TextField
362         name="email"
363         hintText="Enter your email address"
364         errorText={this.state.emailAddressRequired}
365         floatingLabelText="Email address"
366         onChange={(event, newValue) =>
367 this.setState({emailAddress: newValue})}
368         value={this.state.emailAddress}
369         fullWidth={true}
370         required={true}/>
371     </Col>
372 </Row>
373 </Grid>
374 </StepContent>
375 </Step>
376 1x <Step>
377     <StepLabel>Account Information</StepLabel>
378     <StepContent>
379         <Grid>
380             <Row className="show-grid">
381                 <Col xs={7} md={3}>
382                     <PasswordField
383                         name={"password"}
384                         type="password"
385                         hintText="Enter your password"
386                         errorText={this.state.passwordRequired}
387                         floatingLabelText="Password"
388 1x         onChange={(event, newValue) => this.setState({password:
389 newValue})}
390                         value={this.state.password}
391                         fullWidth={true}
392                         required={true}/>
393                 </Col>
394                 <Col xs={7} md={3}>
395                     <PasswordField
396                         name={"confirm"}
397                         type="password"
398                         hintText="Confirm your password"
399                         errorText={this.state.confirmRequired}
400                         floatingLabelText="Confirm your password"
401                         onChange={(event, newValue) => this.setState({confirm:
402 newValue})}
403                         value={this.state.confirm}
404                         fullWidth={true}
405                         required={true}/>
406                 </Col>
407             </Row>
408             <Row className="show-grid">
409                 <Col xs={7} md={3}>
410                     <SchoolSelector selected={this.state.school} errorMsg=
411 {this.state.schoolRequired}
412                         callback={this.callBack} labelText={"School"}
413                         hintText={"Select your school"}/>
414                 </Col>

```



```

415         </Row>
416     </Grid>
417 </StepContent>
418 </Step>
419 <Step>
420     <StepLabel>Account Creation</StepLabel>
421     <StepContent>
422         <Grid>
423             <Row className="show-grid">
424                 <Col>
425                     <div>{this.state.accountMessage}</div>
426                 </Col>
427             </Row>
428         </Grid>
429     </StepContent>
430 </Step>
431 </Stepper>
432 <div style={contentStyle} className={stepIndex === 2 ? 'collapse' : ''}>
433     <div>
434         <div style={{marginTop: 12}}>
435             <FlatButton
436                 label="Back"
437                 disabled={stepIndex === 0 || stepIndex === 2}
438                 onClick={this.handlePrev}
439                 style={{marginRight: 12}}/>
440             <RaisedButton
441                 label={stepIndex === 2 ? 'Return to login' : 'Next'}
442                 primary={true}
443                 onClick={this.handleNext}/>
444         </div>
445     </div>
446 </div>
447 </div>
448 </MuiThemeProvider>
449 );
450 }
451 }
452
453 export default Signup;

```

All files / src/components/Schools SchoolSelector.js

100% Statements 16/16 50% Branches 4/8 90% Functions 9/10 100% Lines 15/15

```
1 import React from 'react';
2 import SelectField from 'material-ui/SelectField';
3 import MenuItem from 'material-ui/MenuItem';
4 import HttpRequest from '../../adapters/httpRequest';
5 import constants from '../../utils/constants';
6
7 class SchoolSelector extends React.Component {
8
9     constructor(props) {
10         4x super(props);
11
12         4x this.state = {
13             errorMsg: this.props.errorMsg,
14             schoolList: [],
15         }
16     }
17
18     sortByKey(array, key) {
19         3x return array.sort(function(a, b) {
20             3x var x = a[key]; var y = b[key];
21             3x return ((x < y) ? -1 : ((x > y) ? 1 : 0));
22         });
23     }
24
25     componentWillMount() {
26         3x var _this = this;
27
28         3x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getSchools',
29         3x 'GET', null, null).then(function (result) {
30             _this.setState({schoolList: _this.sortByKey(result.body, "schoolName")})
31         }).catch(function (error) {
32
33         })
34     }
35
36     componentWillReceiveProps(nextProps) {
37         1x
38         1x if(JSON.stringify(this.props.errorMsg) !== JSON.stringify(nextProps.errorMsg))
39             this.setState({errorMsg: nextProps.errorMsg})
40     }
41
42     updateValues = (event, index, value) => {
43         1x this.props.callBack(event, index, value);
44     }
45
46     render() {
47         9x
48         5x if (Object.keys(this.state.schoolList).length !== 0) {
49             return (
50                 <SelectField
51                     name={"schools"}
52                     hintText={this.props.hintText}
53                     errorText={this.state.errorMsg}
54                     floatingLabelText={this.props.labelText}
```

```

55         onChange={this.updateValues}
56         maxHeight={200}
57         value={this.props.selected}>
58         {
59     10x         Object.keys(this.state.schoolList).map(function (key) {
60             return (
61                 <MenuItem key={this.state.schoolList[key].id}
62                     primaryText={this.state.schoolList[key].schoolName}
63                     value={this.state.schoolList[key].id}/>
64             )
65         }, this)
66     }
67     </SelectField>
68 )
69 }
70 }
71 4x     else
72         return ("ERROR LOADING SCHOOLS")
73     }
74 }
75
export default SchoolSelector;

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Sidebar Sidebar.js

100% Statements 11/11 83.33% Branches 10/12 100% Functions 6/6 100% Lines 11/11

```
1 import React, {Component} from 'react';
2 import { NavLink } from 'react-router-dom';
3
4 import HeaderLinks from '../Header/HeaderLinks.js';
5
6 import imagine from '../../assets/img/Mountain.jpg';
7 import logo from '../../assets/img/reactlogo.png';
8
9 import appRoutes from '../../routes/app.js';
10 import AuthService from "../../utils/AuthService";
11
12 class Sidebar extends Component{
13   constructor(props){
14     4x super(props);
15     4x this.state = {
16       width: window.innerWidth
17     }
18   }
19   activeRoute(routeName) {
20     47x return this.props.location.pathname.indexOf(routeName) > -1 ? 'active' : '';
21   }
22   updateDimensions(){
23     4x this.setState({width:window.innerWidth});
24   }
25   componentDidMount() {
26     4x this.updateDimensions();
27     4x window.addEventListener("resize", this.updateDimensions.bind(this));
28   }
29   render(){
30     9x const sidebarBackground = {
31       backgroundImage: 'url(' + imagine + ')'
32     };
33     9x return (
34       <div id="sidebar" className="sidebar" data-color="black" data-image={imagine}>
35         <div className="sidebar-background" style={sidebarBackground}></div>
36         <div className="logo">
37           <a href="#dashboard" className="simple-text logo-mini App-logo">
38             <div className="logo-img">
39               <img src={logo} alt="logo_image"/>
40             </div>
41
42             </a>
43             <a href="#dashboard" className="simple-text logo-normal">
44               Science Olympiad
45             </a>
46           </div>
47         <div className="sidebar-wrapper">
48           <ul className="nav">
49             { this.state.width <= 991 ? (<HeaderLinks />):null }
50             {
51               appRoutes.map((prop,key) => {
52                 //DO NOT RENDER SIDEBAR LINK IF USER IS NOT ALLOWED TO VIEW
53                 90x if(prop.redirect || prop.linkOnly ||(prop.users &&
54                   43x !AuthService.isUserRoleAllowed(prop.users))) {
```

```

55         return null;
56     }
57     47x
58     return (
59         <li className={prop.upgrade ? "active active-pro" :
60         this.activeRoute(prop.path)}
61             key={key}>
62             <NavLink to={prop.path} className="nav-link"
63             activeClassName="active">
64                 <i className={prop.icon}></i>
65                 <p>{prop.name}</p>
66             </NavLink>
67         </li>
68     );
69     })
70     }
71     </ul>
72     </div>
73     </div>
74     );
75     }
76 }

export default Sidebar;

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Students StudentAdder.js

100% Statements 33/33 100% Branches 10/10 100% Functions 13/13 100% Lines 33/33

```
1 import React, {Component} from 'react';
2 import {MuiThemeProvider, AppBar, TextField, RaisedButton, FontIcon} from 'material-ui';
3 import {Grid, Row, Col} from 'react-bootstrap';
4
5 import CustomDropdown from "../../elements/CustomSelector/CustomDropdown";
6 import constants from "../../utils/constants";
7 import Validator from "../../utils/validator";
8 import HttpRequest from "../../adapters/httpRequest";
9
10 class StudentAdder extends Component {
11   constructor(props) {
12     4x super(props);
13
14     4x this.state = {
15       firstName: "",
16       lastName: "",
17       emailAddress: "",
18       selectedSchool: undefined,
19       errors: {}
20     }
21
22     4x this.selectedSchool = this.selectedSchool.bind(this);
23     4x this.validateStudentForm = this.validateStudentForm.bind(this);
24   }
25
26   selectedSchool(value) {
27     1x this.setState({
28       selectedSchool: value
29     })
30   }
31
32   addStudent() {
33     2x var body = {};
34     2x body.firstName = this.state.firstName;
35     2x body.lastName = this.state.lastName;
36     2x body.emailAddress = this.state.emailAddress;
37
38     2x var _this = this;
39
40     2x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/addUser/?
41     1x userType=STUDENT&schoolID=' + _this.state.selectedSchool.id, 'POST', constants.useCredentials(), body,
42     true).then(function (result) {
43       _this.setState({
44         firstName: "",
45         lastName: "",
46         emailAddress: "",
47         1x selectedSchool: undefined
48         1x }, () => {
49           _this.props.addNotification(<div><b>{body.firstName + ' ' + body.lastName}</b> has been
50           created.</div>, 'success');
51           _this.props.updateTable();
52         });
53         1x
54         // alert(result.body);
```

```

    }).catch(function (error) {
      _this.props.addNotification(<div><b>{body.firstName}</b> could not be created because: <em>
{error.message}</em></div>, 'error');
    });
  }

  validateStudentForm() {
    var errors = {};

    if(this.state.firstName.trim() === "") {
      errors.firstNameError = "First name is required";
    }
    if(this.state.lastName.trim() === "") {
      errors.lastNameError = "Last name is required";
    }

    let result = Validator.validateEmail(this.state.emailAddress);

    if(!result.isValid) {
      errors.emailAddressError = result.message;
    }

    if( !this.state.selectedSchool ){
      errors.schoolError = "School is required"
    }

    this.setState({errors: errors}, () => {
      if(constants.isEmpty(this.state.errors)) {
        this.addStudent();
      }
    });
  }

  render() {
    return (
      <MuiThemeProvider>
        <div style={{textAlign: 'center'}}>
          <AppBar showMenuIconButton={false} title="Register Student" style={{zIndex: 10}}/>
          <Grid>
            <Row className="text-center" >
              <Col xs={7} md={3}>
                <TextField
                  name="fname"
                  hintText="Student's First Name"
                  errorText={this.state.errors.firstNameError}
                  floatingLabelText="First name"
                  onChange={(event, newValue) => this.setState({firstName:
newValue.trim()})}

                  value={this.state.firstName}
                  fullWidth={true}
                  required={true}
                  style={{textAlign: 'left'}}
                />
              </Col>
              <Col xs={7} md={3}>
                <TextField
                  name="lname"
                  hintText="Student's Last Name"
                  errorText={this.state.errors.lastNameError}
                  floatingLabelText="Last name"

```

```

115         onChange={(event, newValue) => this.setState({lastName:
116     newValue.trim()})}}
117
118         value={this.state.lastName}
119         fullWidth={true}
120         required={true}
121         style={{textAlign: 'left'}}
122     />
123 </Col>
124 </Row>
125 2x <Row className="show-grid">
126     <Col xs={7} md={3}>
127         <TextField
128             name="email"
129             hintText="Student's Email Address"
130             errorText={this.state.errors.emailAddressError}
131             floatingLabelText="Email address"
132             onChange={(event, newValue) => this.setState({emailAddress:
133     newValue.trim()})}}
134
135             value={this.state.emailAddress}
136             fullWidth={true}
137             required={true}
138             style={{textAlign: 'left'}}
139         />
140     </Col>
141     <Col xs={7} md={3}>
142         <CustomDropdown
143             textAlign="left"
144             name="school"
145             labelText="School"
146             hintText="Select School"
147             errorText={this.state.errors.schoolError}
148             selected={this.state.selectedSchool}
149             endpoint="/sweng500/getSchools"
150             sortKey="schoolName"
151             textKeys=["schoolName"]}
152             selectedValue={this.selectedSchool}
153         />
154     </Col>
155 </Row>
156 <Row className="center-block">
157     <Col sm={6} smOffset={1} style={{maxWidth: 200}}>
158         <RaisedButton icon={<FontIcon className="pe-7s-close-circle" />}
159     label="Cancel" onClick={event => {this.props.togglePanel("")} } />
160     </Col>
161     <Col sm={6} style={{maxWidth: 200}}>
162         <RaisedButton icon={<FontIcon className="pe-7s-like2" />} primary={true}
163     onClick={this.validateStudentForm} label="Confirm"/>
164     </Col>
165 </Row>
166 </Grid>
167 </div>
168 </MuiThemeProvider>
169 )
170 }
171 }
172
173 export default StudentAdder;

```


All files / src/components/Students StudentViewer.js

100% Statements 43/43

100% Branches 9/9

100% Functions 24/24

100% Lines 43/43

```
1 import React, {Component} from 'react';
2 import {MenuItem, RaisedButton, SelectField, MuiThemeProvider} from 'material-ui';
3 import Button from '../elements/CustomButton/CustomButton';
4 import {Panel} from 'react-bootstrap';
5 import FontIcon from 'material-ui/FontIcon';
6
7 import ReactTable from 'react-table';
8 import constants from "../../utils/constants";
9 import HttpRequest from "../../adapters/httpRequest";
10 import matchSorter from "match-sorter";
11
12 class StudentViewer extends Component {
13   constructor(props) {
14     8x super(props);
15
16     8x this.state = {
17       studentsSelectable: [],
18       selectedStudents: [],
19       openStudentSelector: false
20     };
21
22     8x this.studentMenuItems = this.studentMenuItems.bind(this);
23     8x this.selectionRenderer = this.selectionRenderer.bind(this);
24     8x this.toggleStudentSelector = this.toggleStudentSelector.bind(this);
25     8x this.handleSubmit = this.handleSubmit.bind(this);
26   }
27
28   toggleStudentSelector() {
29     1x this.setState({openStudentSelector: false, selectedStudents: []});
30   }
31
32   handleSubmit() {
33     2x this.setState({
34       openStudentSelector: false
35     });
36
37     2x var body = JSON.parse(JSON.stringify(this.props.teamProp));
38
39     2x this.state.selectedStudents.forEach(function (student) {
40       2x body.students.push(student);
41     });
42
43     2x var _this = this;
44     2x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
45     1x '/sweng500/updateStudentsInTeam', 'POST', constants.useCredentials(), body, true).then(function (result) {
46       var updatedTeam = result.body;
47     1x
48     1x _this.props.updateTeam(updatedTeam, _this.props.viewIndex);
49     _this.props.addNotification(<div><b>{body.name}</b> has been updated.</div>, 'success');
50     1x }).catch(function (error) {
51     1x _this.props.addNotification(<div><b>{body.name}</b> could not be updated at this time.</div>,
52     'error');
53     console.log(error.message);
54   });
```

```

55     }
56     2x
57     getStudentsInSchoolDistrict(schoolId) {
58     2x         var promise = new Promise(function (resolve, reject) {
59     1x
60             HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getStudentsFromSchool/?
61     1x schoolId=' + schoolId, 'GET', constants.useCredentials(), null, true).then(function (result) {
62                 resolve(result.body);
63             }).catch(function (error) {
64                 reject(error.message);
65     2x             })
66         });
67
68         return promise;
69     2x     }
70
71     2x     clickAddStudentToTeam(team) {
72     1x         var _this = this;
73
74         _this.getStudentsInSchoolDistrict(team.school.id).then(function (students) {
75             _this.setState({
76                 studentsSelectable: students,
77     1x                 openStudentSelector: true
78             })
79         }).catch(function (error) {
80             _this.setState({
81                 studentsSelectable: [],
82     1x                 openStudentSelector: true
83             });
84
85             console.log(error);
86         });
87     19x     }
88     7x
89     renderSelectFieldIfContent() {
90         if(this.state.studentsSelectable && this.state.studentsSelectable.length > 0) {
91             return (
92                 <SelectField
93                     multiple={true}
94                     autoWidth={true}
95     1x                     hintText="Select Students"
96                     selectionRenderer={this.selectionRenderer}
97                     value={this.state.selectedStudents}
98                     onChange={ (event, index, values) => this.setState({selectedStudents: values})}
99                 >
100                     {this.studentMenuItems()}
101     12x                 </SelectField>
102             )
103         } else {
104             return (
105                 <div><b>No students are currently available to be assigned to a new team within this
106 district</b></div>
107             )
108     3x         }
109     }
110     1x
111     selectionRenderer() {
112     1x         switch (this.state.selectedStudents.length) {
113             case 0:
114     1x                 return '';

```

```

115         case 1:
116             return this.state.selectedStudents[0].firstName + " " +
117 this.state.selectedStudents[0].lastName + " selected";
118         default:
119 7x             return `${this.state.selectedStudents.length} names selected`;
120 7x         }
121     }
122
123     studentMenuItems() {
124         return this.state.studentsSelectable.map((student) => (
125             <MenuItem
126                 key={student.id}
127                 insetChildren={true}
128                 checked={this.state.selectedStudents.indexOf(student) > -1}
129                 value={student}
130                 primaryText={student.firstName + " " + student.lastName}
131 19x             />
132         ));
133     }
134
135     render() {
136         return (
137             <MuiThemeProvider>
138                 <div style={{padding: '20px'}}>
139                     <div style={{textAlign: 'center', fontSize: 'large'}}>
140                         <em>Students in team: <b>{this.props.teamProp.name}</b></em>
141                     </div>
142                     <ReactTable
143                         data={this.props.teamProp.students}
144                         columns={this.columns2}
145                         filterable
146                         defaultPageSize={this.props.teamProp.students.length}
147                         showPagination={false}
148 2x                         className="-striped -highlight"
149                         defaultSorted={[{id: "firstName"}]}
150                     />
151                     <br />
152                     <div>
153                         <Button fill bsStyle="info" onClick={event =>
154 {this.clickAddStudentToTeam(this.props.teamProp)}} disabled={this.state.openStudentSelector}>
155                             Add Students to <b>{this.props.teamProp.name}</b>
156                         </Button>
157                         <br />
158                         <Panel id="student-to-team-collapsible-panel" expanded=
159 {this.state.openStudentSelector} onToggle={ () => {}}>
160                             style={{border: 0, width: '50%', marginLeft: '25%', marginRight: '25%'}}>
161                                 <Panel.Body collapsible>
162                                     {
163                                         this.renderSelectFieldIfContent()
164                                     }
165                                 <br />
166                                 <RaisedButton icon={<FontIcon className="pe-7s-close-circle" />}
167 label="Cancel"
168                                     onClick={this.toggleStudentSelector}/>&nbsp;&nbsp; 
169                                 <RaisedButton icon={<FontIcon className="pe-7s-like2" />} primary={true}
170 label="Add Students"
171                                     onClick={this.handleSubmit} disabled=
172 {this.state.selectedStudents.length === 0 ? true : false}/>
173                                 </Panel.Body>
174                             </Panel>

```

```

175     1x         </div>
176         </div>
177     </MuiThemeProvider>
178 )
179 }
180
181 columns2 = [
182     1x     {
183         Header: 'First Name',
184         filterMethod: (filter, rows) =>
185             matchSorter(rows, filter.value, {keys: ["firstName"]}),
186         filterAll: true,
187         accessor: 'firstName' // String-based value accessors!
188     },
189     1x     {
190         Header: 'Last Name',
191         filterMethod: (filter, rows) =>
192             matchSorter(rows, filter.value, {keys: ["lastName"]}),
193         filterAll: true,
194         accessor: 'lastName' // String-based value accessors!
195     },
196     {
197         Header: 'Email',
198         filterMethod: (filter, rows) =>
199             matchSorter(rows, filter.value, {keys: ["emailAddress"]}),
200         filterAll: true,
201         accessor: 'emailAddress' // String-based value accessors!
202     },
203     {
204         Header: 'Actions',
205         accessor: 'menuActions', // String-based value accessors!
206         style: {textAlign: 'center'},
207         sortable: false,
208         filterable: false
209     }
210 ];
211
212 export default StudentViewer;

```

All files / src/components/Teams TeamAdder.js

100% Statements 31/31 100% Branches 8/8 100% Functions 12/12 100% Lines 31/31

```
1 import React, {Component} from 'react';
2 import {MuiThemeProvider, AppBar, TextField, RaisedButton, FontIcon} from 'material-ui';
3 import {Grid, Row, Col} from 'react-bootstrap';
4
5 import CustomDropdown from "../../elements/CustomSelector/CustomDropdown";
6 import Validator from "../../utils/validator";
7 import constants from "../../utils/constants";
8 import HttpRequest from "../../adapters/httpRequest";
9
10 class TeamAdder extends Component {
11   constructor(props) {
12     4x super(props);
13
14     4x this.state = {
15       selectedSchool: undefined,
16       selectedCoach: undefined,
17       teamName: "",
18       unallocatedStudents: [],
19       errors: {}
20     }
21
22     4x this.selectedCoach = this.selectedCoach.bind(this);
23     4x this.selectedSchool = this.selectedSchool.bind(this);
24     4x this.validateTeamForm = this.validateTeamForm.bind(this);
25   }
26
27   selectedCoach(value) {
28     1x this.setState({
29       selectedCoach: value
30     })
31   }
32
33   selectedSchool(value) {
34     1x this.setState({
35       selectedSchool: value
36     })
37   }
38
39   addTeam() {
40     2x var body = {};
41     2x body.name = this.state.teamName;
42     2x body.coach = this.state.selectedCoach;
43     2x body.school = this.state.selectedSchool;
44
45     2x var _this = this;
46
47     2x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/addTeam/',
48     1x 'POST', constants.useCredentials(), body, true).then(function (result) {
49       _this.setState({
50         selectedSchool: undefined,
51         selectedCoach: undefined,
52         teamName: ""
53       }, () => {
54         1x _this.props.addNotification(<div><b>{body.name}</b> has been created.</div>, 'success');
```

```

        _this.props.updateTable()
    });

    }).catch(function (error) {
        _this.props.addNotification(<div><b>{body.name}</b> could not be created because: <em>
{error.message}</em></div>, 'error');
    });
}

validateTeamForm() {
    var errors = {};

    let result = Validator.validateTeamName(this.state.teamName);

    if(!result.isValid) {
        errors.teamNameError = result.message;
    }

    if( !this.state.selectedSchool ){
        errors.schoolError = "School is required"
    }

    if( !this.state.selectedCoach ){
        errors.coachError = "Coach is required"
    }

    this.setState({errors: errors}, () => {
        if(constants.isEmpty(this.state.errors)) {
            this.addTeam();
        }
    });
}

render() {
    return (
        <MuiThemeProvider>
            <div style={{textAlign: 'center'}}>
                <AppBar showMenuIconButton={false} title="Register Team" style={{zIndex: 10}}/>
                <Grid>
                    <Row className="text-center">
                        <Col xs={14} md={6}>
                            <TextField
                                name="tname"
                                hintText="Team's Name"
                                errorText={this.state.errors.teamNameError}
                                floatingLabelText="Team Name"
                                onChange={(event, newValue) => this.setState({teamName: newValue})}
                                value={this.state.teamName}
                                fullWidth={true}
                                required={true}
                                style={{margin: '1px', textAlign: 'left'}}
                            />
                        </Col>
                        <Col xs={7} md={3}></Col>
                    </Row>
                    <Row className="show-grid">
                        <Col xs={7} md={3}>
                            <CustomDropdown
                                name={"school"}
                                labelText={"School"}

```

```

115         hintText={"Select School"}
116         errorText={this.state.errors.schoolError}
117         selected={this.state.selectedSchool}
118         endpoint={"/sweng500/getSchools"}
119         sortKey={"schoolName"}
120         textKeys={["schoolName"]}
121         selectedValue={this.selectedSchool}
122     />
123 </Col>
124 <Col xs={7} md={3}>
125     <CustomDropdown
126         name={"coach"}
127         labelText={"Coach"}
128         hintText={"Select Coach"}
129         errorText={this.state.errors.coachError}
130         selected={this.state.selectedCoach}
131         endpoint={"/sweng500/getCoaches"}
132         sortKey={"lastName"}
133         textKeys={["firstName", "lastName"]}
134         selectedValue={this.selectedCoach}
135     />
136 </Col>
137 </Row>
138 1x <Row className="center-block">
139     <Col sm={6} smOffset={1} style={{maxWidth: 200}}>
140         <RaisedButton icon={<FontIcon className="pe-7s-close-circle" />}
141         label="Cancel" onClick={event => {this.props.togglePanel("")}} />
142     </Col>
143     <Col sm={6} style={{maxWidth: 200}}>
144         <RaisedButton icon={<FontIcon className="pe-7s-like2" />} primary={true}
145         onClick={this.validateTeamForm} label="Confirm"/>
146     </Col>
147 </Row>
148 </Grid>
149 </div>
150 </MuiThemeProvider>
151 )
152 }

}

export default TeamAdder;

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/components/Teams TeamViewer.js

100% Statements 74/74

100% Branches 7/7

100% Functions 34/34

100% Lines 74/74

```
1 import React, {Component} from 'react';
2 import {RaisedButton, AppBar, FontIcon, FlatButton, MuiThemeProvider} from 'material-ui';
3 import ReactTable from 'react-table';
4 import constants from "../../utils/constants";
5 import HttpRequest from "../../adapters/httpRequest";
6 import matchSorter from "match-sorter";
7 import StudentViewer from "../Students/StudentViewer";
8
9 import Loader from 'react-loader'
10
11 import {Modal} from 'react-bootstrap';
12
13 class TeamViewer extends Component {
14   constructor(props) {
15     8x super(props);
16
17     8x this.state = {
18       teams: [],
19       selectedTeam: null,
20       selectedStudent: null,
21       studentToRemoveFromTeam: null,
22       modal: false,
23       modalInfo: constants.getEmptyModalInfo(),
24       expanded: {},
25       isLoading: false
26     };
27
28     8x this.updateTeam = this.updateTeam.bind(this);
29     8x this.closeModal = this.closeModal.bind(this);
30     8x this.deleteTeamButtonClicked = this.deleteTeamButtonClicked.bind(this);
31     8x this.handleRowExpanded = this.handleRowExpanded.bind(this);
32   }
33
34   componentWillReceiveProps(nextProps) {
35     2x if(this.props.tableUpdateToggler !== nextProps.tableUpdateToggler) {
36       1x this.getTeams();
37     }
38   }
39
40   componentWillMount() {
41     8x this.getTeams();
42   }
43
44   getTeams() {
45     10x var _this = this;
46
47     10x _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getTeams',
48     9x 'GET', constants.useCredentials(), null, true).then(function (result) {
49       let resultTeams = result.body;
50
51     9x resultTeams.forEach(function (team) {
52     54x _this.addButtonsToTeam(team);
53   });
54   9x
```

```

55         _this.setState({
56             isLoading:true,
57             teams: resultTeams,
58             expanded: {},
59             selectedTeam: null,
60             selectedStudent: null,
61             modal:false,
62             modalInfo: constants.getEmptyModalInfo(),
63         })
64     1x    }).catch(function (error) {
65     1x        _this.props.addNotification(<div>Could not retrieve teams at this time. Try again later.
66        </div>, 'error');
67            console.log(error);
68        })
69    }
70    2x
71    2x    deleteTeam(team) {
72        var _this = this;
73    2x        var removeId = team.id;
74    1x
75    6x        _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/removeTeam/" +
76        removeId, "DELETE", constants.useCredentials(), null, true).then(function (result) {
77            var tempTeams = _this.state.teams.filter(t => {
78    1x                return t !== team;
79            });
80
81            _this.setState({
82                teams: tempTeams,
83                selectedTeam: null,
84                selectedStudent: null,
85                modal:false,
86                modalInfo: constants.getEmptyModalInfo(),
87    1x                expanded: {}
88            });
89    1x
90            _this.props.addNotification(<div>Team <b>{team.name}</b> has been deleted.</div>, 'success');
91        }).catch(function (error) {
92            _this.props.addNotification(<div>Team <b>{team.name}</b> could not be deleted because: <em>
93        {error.message}</em></div>, 'error');
94        })
95    2x    }
96    2x
97
98    2x    deleteStudent(student) {
99    1x        var _this = this;
100        var studentId = student.id;
101    1x
102        _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
103        "/sweng500/deleteStudent/" + studentId, "DELETE", constants.useCredentials(), null, true).then(function
104    1x (result) {
105    1x        _this.props.addNotification(<div>Student <b>{student.firstName + ' ' + student.lastName}</b>
106        has been deleted.</div>, 'success');
107
108        _this.getTeams();
109
110        }).catch(function (error) {
111    2x            console.log(error);
112    2x            _this.props.addNotification(<div>Student <b>{student.firstName + ' ' + student.lastName}</b>
113        could not be deleted because: <em>{error.message}</em></div>, 'error');
114        })

```

```

115 2x    }
116
117 2x    removeStudentFromTeam(student, team) {
118 2x
119 1x        var tempStudents = team.students.filter(s => {
120 1x            return s !== student;
121 1x        });
122 6x
123        team.students = tempStudents;
124
125 1x        var _this = this;
126 1x        _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
127 '/sweng500/updateStudentsInTeam', 'POST', constants.useCredentials(), team, true).then(function (result) {
128 1x            var updatedTeam = result.body;
129
130            var tempTeams = _this.state.teams.filter(t => {
131                return t.id !== team.id;
132            });
133
134            _this.addButtonsToTeam(updatedTeam);
135            tempTeams.push(updatedTeam);
136
137 1x            _this.setState({
138                teams: tempTeams,
139 1x                selectedTeam: null,
140 1x                selectedStudent: null,
141                modal:false,
142                modalInfo: constants.getEmptyModalInfo(),
143                expanded: {}
144            });
145 1x
146            _this.props.addNotification(<div><b>{student.firstName + ' ' + student.lastName}</b> has been
147 1x removed from <b>{team.name}</b></div>, 'success');
148 6x            }).catch(function (error) {
149 1x                _this.props.addNotification(<div><b>{team.name}</b> could not be updated at this time.</div>,
150 'error');
151 1x                console.log(error.message);
152            });
153        }
154
155 1x    updateTeam(updatedTeam, viewIndex) {
156        var tempTeams = this.state.teams;
157
158        tempTeams.forEach(function (team) {
159 1x            if(team.id === updatedTeam.id) {
160                this.addButtonsToTeam(updatedTeam);
161
162                team.students = updatedTeam.students;
163            }
164 1x        }, this);
165
166        this.setState({
167            teams: tempTeams
168        });
169
170        this.handleRowExpanded({}, viewIndex);
171    }
172
173 2x    // Close the modal
174    closeModal() {

```

```

175         this.setState({
176             modal: false,
177             selectedTeam: null,
178             selectedStudent: null,
179             modalInfo: constants.getEmptyModalInfo()
180         })
181     }
182
183     deleteTeamButtonClicked(team) {
184         this.setState({
185             selectedTeam: team,
186             modal: true,
187             modalInfo: {
188                 title: 'Delete Team',
189                 body:
190                 2x         <div style={{textAlign: 'center'}}>
191                             <b>Are you sure you wish to delete team <u><em>{team.name}</em></u>?</b>
192                             <br />
193                             Note: This action cannot be undone.
194                         </div>,
195                 modalAction: 'DELETE'
196             }
197         });
198     }
199
200     deleteStudentClicked(student) {
201         this.setState({
202             selectedStudent: student,
203             modal: true,
204             modalInfo: {
205                 title: 'Delete Student',
206                 body:
207                 2x         <div style={{textAlign: 'center'}}>
208                             <b>Are you sure you wish to delete student <u><em>{student.firstName + ' ' +
209 student.lastName}</em></u>?</b>
210                             <br />
211                             Note: This action cannot be undone.
212                         </div>,
213                 modalAction: 'DELETESTUDENT'
214             }
215         })
216     }
217
218     removeStudentFromTeamButtonClicked(student, team) {
219         this.setState({
220             selectedStudent: student,
221             selectedTeam: team,
222             modal: true,
223             6x         modalInfo: {
224                 title: 'Update Team',
225                 2x         body:
226                 2x         <div style={{textAlign: 'center'}}>
227                             <b>Are you sure you wish to remove student <u><em>{student.firstName + ' ' +
228 2x student.lastName}</em></u> from team <u><em>{team.name}</em></u>?</b>
229                             </div>,
230                 modalAction: 'REMOVESTUDENTFROMTEAM'
231             }
232         })
233     }
234

```

```

235     handleModalActionClicked(modalAction) {
236         switch(modalAction) {
237             56x case 'DELETE':
238                 this.deleteTeam(this.state.selectedTeam);
239                 break;
240             case 'REMOVEDSTUDENTFROMTEAM':
241                 this.removeStudentFromTeam(this.state.selectedStudent, this.state.selectedTeam);
242             56x break;
243             81x case 'DELETESTUDENT':
244                 this.deleteStudent(this.state.selectedStudent);
245                 break;
246         }
247     }
248
249     addButtonsToTeam(team) {
250         team.menuActions =
251             <div>
252                 <RaisedButton style={{minWidth: '30%'}} icon={<FontIcon className="pe-7s-trash" />}
253             3x secondary={true} onClick={this.deleteTeamButtonClicked.bind(this, team)} />
254             </div>;
255
256         for(let studIndex in team.students) {
257             team.students[studIndex].menuActions =
258                 <div >
259             31x <RaisedButton style={{minWidth: '40%'}} icon={<FontIcon className="pe-7s-less" />}
260                 backgroundColor="#FFC300" onClick={this.removeStudentFromTeamButtonClicked.bind(this,
261                 team.students[studIndex], team)} />
262                 <RaisedButton style={{minWidth: '40%'}} icon={<FontIcon className="pe-7s-trash" />}
263                 secondary={true} onClick={this.deleteStudentClicked.bind(this, team.students[studIndex])} />
264                 </div>;
265             }
266         }
267
268         //Handling Row Expansion
269         handleRowExpanded(newExpanded, index) {
270             this.setState({
271                 expanded: {[index]: !this.state.expanded[index]}
272             })
273             1x }
274
275         render() {
276             return (
277                 <MuiThemeProvider>
278                     <div>
279                         <Loader color="#3498db" loaded={this.state.isLoaded}>
280                         <ReactTable
281                             data={this.state.teams}
282                             columns={this.columns}
283                             expanded={this.state.expanded}
284                             onExpandedChange={this.handleRowExpanded}
285                             filterable
286                             defaultPageSize={10}
287                             className="-striped -highlight"
288                             defaultSorted={[{id: "name"}]}
289                             SubComponent={row => (
290                                 <StudentViewer teamProp={row.original} viewIndex={row.viewIndex}
291                                 updateTeam={this.updateTeam} updateTable={this.props.updateTable} addNotification=
292                                 {this.props.addNotification}/>
293                             6x )}
294                             />

```

```

295         </Loader>
296         <Modal show={this.state.modal} onHide={this.closeModal}>
297             <Modal.Header>
298                 <Modal.Title> <AppBar
299                     iconElementRight={<FlatButton label="Close"/>}
300                     showMenuIconButton={false}
301                     onRightIconButtonClick={this.closeModal}
302                     title={this.state.modalInfo.title}
303                 /></Modal.Title>
304             </Modal.Header>
305             <Modal.Body>
306                 {this.state.modalInfo.body}
307             </Modal.Body>
308             <Modal.Footer>
309                 <RaisedButton icon={<FontIcon className="pe-7s-close-circle" />}
310                 label="Cancel"
311                     onClick={this.closeModal}/>&nbsp;&nbsp;&
312                 <RaisedButton icon={<FontIcon className="pe-7s-like2" />} primary={true}
313                 label="Confirm"
314                     onClick={ () =>
315 this.handleModalActionClicked(this.state.modalInfo.modalAction)} />
316             </Modal.Footer>
317         </Modal>
318     </div>
319 </MuiThemeProvider>
320 )
321 }
322
323 columns = [
324     {
325         Header: 'Team Name',
326         filterMethod: (filter, rows) =>
327             matchSorter(rows, filter.value, {keys: ["name"]}),
328         filterAll: true,
329         accessor: 'name' // String-based value accessors!
330     }, {
331         Header: 'Coach',
332         id: 'coachName',
333         filterMethod: (filter, rows) =>
334             matchSorter(rows, filter.value, {keys: ["coachName"]}),
335         filterAll: true,
336         accessor: 'coach.name' // String-based value accessors!
337     }, {
338         Header: 'School',
339         id: 'schoolName',
340         filterMethod: (filter, rows) =>
341             matchSorter(rows, filter.value, {keys: ["schoolName"]}),
342         filterAll: true,
343         accessor: 'school.schoolName' // String-based value accessors!
344     },
345     {
346         Header: 'Actions',
347         accessor: 'menuActions', // String-based value accessors!
348         style:{textAlign:'center'},
349         sortable: false,
350         filterable: false
351     }
352 ];
353 }

```

```
export default TeamViewer;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/containers/App App.js

96.77% Statements 30/31 95% Branches 19/20 100% Functions 7/7 96.77% Lines 30/31

```
1 import React, { Component } from 'react';
2 import NotificationSystem from 'react-notification-system';
3 import {
4     Switch,
5     Redirect
6 } from 'react-router-dom';
7
8 import {style} from "../../variables/Variables.js";
9 import AuthenticatedRoute from '../../routes/AuthenticatedRoute';
10 import Header from '../../components/Header/Header';
11 import Footer from '../../components/Footer/Footer';
12 import Sidebar from '../../components/Sidebar/Sidebar';
13 import appRoutes from '../../routes/app.js';
14
15
16 class App extends Component {
17     constructor(props){
18         2x super(props);
19         2x this.componentDidMount = this.componentDidMount.bind(this);
20         2x this.notify = this.notify.bind(this);
21         2x this.state = {
22             _notificationSystem: null
23         };
24     }
25
26     notify(message, level, position, autoDismiss, optionalTitle){
27         2x this.state._notificationSystem.addNotification({
28             title: optionalTitle ? optionalTitle : (<span data-notify="icon" className="pe-7s-gift">
29 </span>),
30             message: (
31                 <div>
32                     {message}
33                 </div>
34             ),
35             level: level ? level : 'info',
36             position: position ? position : 'tc',
37             autoDismiss: autoDismiss ? autoDismiss : 10,
38         });
39     }
40
41     6x static getRandoNotification() {
42         6x     var notifyJson = {};
43         6x     notifyJson.position = 'tr';
44         notifyJson.autoDismiss = 10;
45
46         6x     let color = Math.floor((Math.random() * 4) + 1);
47         switch (color) {
48             1x     case 1:
49                 1x         notifyJson.level = 'success';
50                 break;
51             1x     case 2:
52                 1x         notifyJson.level = 'warning';
53                 break;
54             1x     case 3:
```



```

        notifyJson.level = 'error';
        break;
    case 4:
        notifyJson.level = 'info';
        break;
    default:
        notifyJson.level = 'info';
        break;
}

return notifyJson;
}

componentDidMount(){
    this.setState({_notificationSystem: this.refs.notificationSystem});
    var _notificationSystem = this.refs.notificationSystem;

    let notifyJson = App.getRandoNotification();

    _notificationSystem.addNotification({
        title: (<span data-notify="icon" className="pe-7s-gift"></span>),
        message: (
            <div>
                Welcome to <b>Science Olympiad Dashboard</b> - a central hub for Science Olympiad
                Competitions.
            </div>
        ),
        level: notifyJson.level,
        position: notifyJson.position,
        autoDismiss: notifyJson.autoDismiss,
    });
}

componentDidUpdate(e){
    if(window.innerWidth < 993 && e.history.location.pathname !== e.location.pathname &&
    document.documentElement.className.indexOf('nav-open') !== -1){
        document.documentElement.classList.toggle('nav-open');
    }
}

render() {
    return (
        <div className="wrapper">
            <NotificationSystem ref="notificationSystem" style={style}/>
            <Sidebar {...this.props} />
            <div id="main-panel" className="main-panel">
                <Header {...this.props}/>
                <Switch>
                    {
                        appRoutes.map((prop,key) => {
                            if(prop.redirect)
                                return (
                                    <Redirect from={prop.path} to={prop.to} key={key}/>
                                );
                            return (
                                <AuthenticatedRoute path={prop.path} component={prop.component}
                                users={prop.users} key={key} notify={this.notify}/>
                            );
                        })
                    }
                </Switch>
            </div>
        </div>
    );
}

```

```
115
116
117
118
119
120
    </Switch>
    <Footer />
  </div>
</div>
);
}
}

export default App;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/containers/Login LoginContainer.js

100% Statements 29/29

100% Branches 13/13

100% Functions 10/10


100% Lines 29/29

```
1 import React, {Component} from 'react';
2 import Login from "../../components/Login/Login";
3 import Signup from "../../components/Login/Signup";
4 import Forgot from "../../components/Login/Forgot";
5 import $ from 'jquery';
6 import NotificationSystem from 'react-notification-system';
7
8 import mountainBackground from '../../assets/img/SmallerMountain.jpg';
9
10 import {style} from "../../variables/Variables";
11 1x const backgroundStyle = {
12     background: 'url(' + mountainBackground + ') no-repeat center center fixed',
13     backgroundSize: 'cover',
14     backgroundPosition: 'center center no-repeat'
15 };
16
17 class LoginContainer extends Component {
18
19     constructor(props) {
20 3x         super(props);
21
22 3x         this.state = {
23             view: "login",
24             _notificationSystem: null
25         };
26
27 3x         this.addNotification = this.addNotification.bind(this);
28     }
29
30     toggleLoginView(event) {
31 5x         event.preventDefault();
32
33 5x         var value = event.target.attributes.getNamedItem('data-type').value;
34
35 5x         switch (value) {
36             case 'register':
37 1x                 $('#login-container-slider').addClass('collapse');
38 1x                 $('#signup-container-slider').removeClass('collapse');
39 1x                 break;
40             case 'forgot':
41 1x                 $('#login-container-slider').addClass('collapse');
42 1x                 $('#forgot-container-slider').removeClass('collapse');
43 1x                 break;
44             case 'slogin':
45 1x                 $('#login-container-slider').removeClass('collapse');
46 1x                 $('#signup-container-slider').addClass('collapse');
47 1x                 break;
48             case 'flogin':
49 1x                 $('#login-container-slider').removeClass('collapse');
50 1x                 $('#forgot-container-slider').addClass('collapse');
51 1x                 break;
52             default :
53 1x                 break;
54     }
```

```

55     }
56
57     addNotification(message, level, position, autoDismiss) {
58 6x       if(this.state._notificationSystem) {
59 2x         this.state._notificationSystem.addNotification({
60           title: (<span data-notify="icon" className="pe-7s-door-lock"></span>),
61           message: (
62             <div>
63               {message}
64             </div>
65           ),
66           level: level ? level : 'error',
67           position: position ? position : 'tc',
68           autoDismiss: autoDismiss ? autoDismiss : 10,
69         });
70       }
71     }
72
73     componentDidMount() {
74 4x       this.setState({_notificationSystem: this.refs.notificationSystem}, () => {
75 4x         this.addNotification(
76           <span>Please <b>Login</b> to the Science Olympiad System.</span>,
77           'info',
78           'tc',
79           10
80         )
81       })
82     }
83
84     render() {
85
86 9x       return (
87         <div id='login-container' className="animated fadeIn login-center" style={backgroundStyle}>
88           <NotificationSystem ref="notificationSystem" style={style}/>
89           <div id='login-container-card' className="login-card row">
90             <div id='login-container-slider' className="animated fadeIn">
91               <Login notify={this.addNotification}/>
92 1x               <a data-type={'forgot'} onClick={event => this.toggleLoginView(event)}
93               className="col-sm-10">Forgot your password?</a>
94 1x               <br/>
95               <a data-type={'register'} onClick={event => this.toggleLoginView(event)}
96               className="col-sm-10">Register for a new account</a>
97               <br/>
98             </div>
99             <div id='signup-container-slider' className="animated collapse fadeIn">
100               <Signup notify={this.addNotification}/>
101 2x               <br/>
102               <a data-type={'slogin'} onClick={event => this.toggleLoginView(event)}
103               className="col-sm-9">Return to the login screen</a>
104             </div>
105             <div id='forgot-container-slider' className="animated collapse fadeIn">
106 1x               <Forgot notify={this.addNotification}/>
107               <br/>
108               <a data-type={'flogin'} onClick={event => this.toggleLoginView(event)}
109               className="col-sm-9">Return to the login screen</a>
110             </div>
111             </div>
112           </div>
113         </div>
114       )

```

```
115 
116  }
```

```
}
```

```
export default LoginContainer;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/elements/CustomButton CustomButton.js

100% Statements 4/4 100% Branches 0/0 100% Functions 1/1 100% Lines 4/4

```
1 import React, { Component } from 'react';
2 import { Button } from 'react-bootstrap';
3 import cx from 'classnames';
4 import PropTypes from 'prop-types';
5
6 class CustomButton extends Component {
7   render() {
8     2x const { fill, simple, pullRight, round, block, ...rest } = this.props;
9
10    2x const btnClasses = cx({
11      'btn-fill': fill,
12      'btn-simple': simple,
13      'pull-right': pullRight,
14      'btn-block': block,
15      'btn-round': round
16    });
17
18    2x return (
19      <Button
20        className={btnClasses}
21        {...rest}
22      />
23    );
24  }
25 }
26
27 10x CustomButton.propTypes = {
28   fill: PropTypes.bool,
29   simple: PropTypes.bool,
30   pullRight: PropTypes.bool,
31   block: PropTypes.bool,
32   round: PropTypes.bool
33 }
34
35 export default CustomButton;
36
```

All files / src/elements/CustomSelector CustomDropdown.js

100% Statements 26/26 78.57% Branches 11/14 100% Functions 12/12 100% Lines 24/24

```
1 import React, {Component} from 'react';
2 import SelectField from 'material-ui/SelectField';
3 import MenuItem from 'material-ui/MenuItem';
4 import HttpRequest from '../..../adapters/httpRequest';
5 import constants from '../..../utils/constants';
6
7 class CustomDropdown extends Component {
8
9     constructor(props) {
10 4x      super(props);
11
12 4x      this.state = {
13          errorText: this.props.errorText,
14          list: [],
15      }
16
17 4x      this.updateValues = this.updateValues.bind(this);
18
19      //INCOMING PROPS:
20      //selected - "the item selected"
21      //endpoint - "/sweng500/getCoaches"
22      //sortKey - "lastName"
23      //hintText - "the label before clicking"
24      //labelText - "the label"
25      //textKeys - ["firstName", "lastName"]
26    }
27
28    sortByKey(array, key) {
29 3x      if(key === undefined) return array;
30
31 2x      return array.sort(function(a, b) {
32 2x          var x = a[key]; var y = b[key];
33 2x          return ((x < y) ? -1 : ((x > y) ? 1 : 0));
34      });
35    }
36
37    buildLabel(listItem) {
38 8x      var label = "";
39 8x      if(this.props.textKeys && this.props.textKeys.length > 0) {
40 6x          this.props.textKeys.forEach(function (key) {
41 6x              label += (listItem[key] + " ");
42          });
43
44 6x          return label.trim();
45      }
46
47 2x      return " - ";
48    }
49
50    componentDidMount() {
51 4x      var _this = this;
52
53 4x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + this.props.endpoint,
54 3x      'GET', constants.useCredentials(), null, true).then(function (result) {
```

```

55         _this.setState({list: _this.sortByKey(result.body, _this.props.sortKey)})
56 1x     }).catch(function (error) {
57         _this.setState({
58             errorText: "Failed to get data"
59         });
60     })
61 }
62
63 2x componentWillReceiveProps(nextProps) {
64 1x     if(this.state.errorText !== nextProps.errorText)
65         this.setState({errorText: nextProps.errorText})
66 }
67
68 1x updateValues(event, index, value) {
69     this.props.selectedValue(value);
70 }
71
72 9x render() {
73 4x     if (this.state.list.length !== 0) {
74         return (
75             <SelectField
76                 name={this.props.name}
77                 hintText={this.props.hintText}
78                 errorText={this.state.errorText}
79                 floatingLabelText={this.props.labelText}
80                 onChange={this.updateValues}
81                 value={this.props.selected}
82                 maxHeight={200}
83                 fullWidth={true}
84                 style={{textAlign: 'left'}}>
85                 {
86 8x                     Object.keys(this.state.list).map(function (key) {
87                         return (
88                             <MenuItem key={this.state.list[key].id}
89                                 primaryText={this.buildLabel(this.state.list[key])}
90                                 value={this.state.list[key]}>
91                         )
92                     }, this)
93                 }
94             </SelectField>
95         )
96     }
97     else
98 5x         return <span></span>
99
100 }
101 }
102
export default CustomDropdown;

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/routes AuthenticatedRoute.js

100% Statements 3/3

100% Branches 4/4

100% Functions 2/2

100% Lines 3/3

```
1  import React from 'react';
2  import {
3      Route,
4      Redirect
5  } from 'react-router-dom';
6
7  import AuthService from "../utils/AuthService";
8
9  /**
10   * We have to make our own Authenticated Route.... So here it is --
11   * @param Component
12   * @param rest
13   * @returns {*}
14   * @constructor
15   */
16  2x const AuthenticatedRoute = ({ component: Component, ...rest }) => (
17  3x    <Route {...rest} render={props => (
18  3x      AuthService.isLoggedIn() && AuthService.isUserRoleAllowed(rest.users)? (
19          <Component {...props}/>
20      ) : (
21          <Redirect to={{
22              pathname: '/login',
23              state: { from: props.location }
24          }}/>
25      )
26  )}/>
27  )
28
29  export default AuthenticatedRoute;
```

All files / src/routes app.js

100% Statements 1/1

100% Branches 0/0

100% Functions 0/0

100% Lines 1/1

```
1 import Dashboard from '../views/Dashboard/Dashboard';
2 import UserProfile from '../views/UserProfile/UserProfile';
3 import Events from '../views/Events/Events';
4 import Schools from '../views/Schools/Schools';
5 import Buildings from '../views/Locations/Buildings';
6 import Rooms from '../views/Locations/Rooms';
7 import TeamManagement from "../views/Teams/TeamManagement";
8 import Scoring from "../views/Scoring/Scoring";
9 import Extras from "../views/Extras/Extras";
10
11 4x const appRoutes = [
12     { path: "/app/dashboard", name: "Dashboard", icon: "pe-7s-graph", component: Dashboard },
13     { path: "/app/events", name:"Events", icon:"pe-7s-global", component :Events},
14     { path: "/app/schools", name:"Schools", icon:"pe-7s-home", component :Schools, users:['ADMIN']},
15     { path: "/app/buildings", name:"Buildings", icon:"pe-7s-map", component: Buildings, users:['ADMIN']},
16     { path: "/app/rooms", name:"Rooms", icon:"pe-7s-ribbon", component: Rooms, users:['ADMIN']},
17     { path: "/app/teams", name: "Team Management", icon: "pe-7s-users", component : TeamManagement, users:
18     ['ADMIN', 'COACH']},
19     { path: "/app/scoring", name: "Event Scores", icon: "pe-7s-medal", component : Scoring },
20     { path: "/app/extras", name:"Extras", icon:"pe-7s-gleam", component :Extras, users:['ADMIN']},
21     { linkOnly: true, path: "/app/user", name:"User Profile", icon:"pe-7s-global", component :UserProfile},
22     { redirect: true, path:"/", to:"/app/dashboard", name: "Dashboard" }
23 ];
24
25 export default appRoutes;
```

All files / src/utils AuthService.js

100% Statements 69/69

100% Branches 32/32

100% Functions 22/22

100% Lines 69/69

```
1 import jwt from 'jwt-simple';
2 import moment from 'moment';
3
4 import constants from './constants';
5 import HttpRequest from '../adapters/httpRequest';
6
7 export default class AuthService {
8
9     static login(username, password) {
10         var promise = new Promise(function (resolve, reject) {
11             var body = {};
12             body.emailAddress = username;
13             body.password = password;
14
15             //var _this = this;
16             HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/auth/login', 'POST',
17 constants.useCredentials(), body ).then(function (result) {
18                 let toEncode = {
19                     emailAddress: result.body.emailAddress,
20                     role: result.body.role,
21                     expiration: moment().add(20, 'minutes').format('YYYY-MM-DDTHH:mm:ss.SSS')
22                 }
23
24                 var encoded = jwt.encode(toEncode, result.body.session);
25
26                 AuthService.setSession(result.body.session);
27                 AuthService.setToken(encoded);
28
29                 resolve({status: result.status, message: "Authorized", emailAddress:
30 result.body.emailAddress});
31             }).catch(function (error) {
32                 console.log(error);
33                 reject(error);
34             })
35         });
36
37         return promise;
38     }
39
40     static logout() {
41         var promise = new Promise(function (resolve, reject) {
42             var body = {};
43
44             const token = AuthService.getToken();
45             const session = AuthService.getSession();
46             if(token && session) {
47
48                 body.emailAddress = jwt.decode(token, session).emailAddress;
49
50                 HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/auth/logout', 'POST',
51 constants.useCredentials(), body ).then(function (result) {
52                     console.log(result);
53                     AuthService.revokeAuth();
54                 })
55             }
56         });
57     }
58 }
```

```

55 1x         resolve({status: result.status, message: result.body});
56 1x     }).catch(function (error) {
57         AuthService.revokeAuth();
58         console.log(error);
59 1x         resolve({status: error.status, message: error.message});
60 1x     })
61     } else {
62         AuthService.revokeAuth();
63         resolve({status: 410, message: "Token is expired."});
64 3x     }
65     });
66
67     return promise;
68 1x }
69
70 static isLoggedIn() {
71     return AuthService.isAuthenticated();
72 4x }
73 4x
74 4x static isAuthenticated(isServerCall) {
75 3x     const token = AuthService.getToken();
76 3x     const session = AuthService.getSession();
77 1x     if (token && session) {
78 1x         var decoded = AuthService.decodeSessionVars();
79         if (!decoded || moment(decoded.expiration).isBefore(moment())) { // Checking if token is
80 expired.
81 2x             AuthService.revokeAuth(true);
82 1x             return false;
83         }
84         else {
85 2x             if(isServerCall) {
86                 AuthService.updateTimeStamp(decoded);
87             }
88 1x
89 1x             return true;
90         }
91     } else {
92         AuthService.revokeAuth();
93         return false;
94 3x     }
95 2x }
96
97 static isUserRoleAllowed(listOfAllowedRoles) {
98 1x     if(listOfAllowedRoles && listOfAllowedRoles.length > 0){
99         return listOfAllowedRoles.indexOf(AuthService.getUserRole()) > -1;
100     }
101
102 7x     return true;
103 7x }
104
105 7x static revokeAuth(withAlert) {
106     AuthService.revokeToken();
107 2x     AuthService.endSession();
108
109     if(withAlert) {
110         //DO SOMETHING BETTER THAN THIS
111         alert("Your session has ended. Please log back in to continue");
112 4x     }
113 4x }
114

```

```

115 static getUserRole() {
116     let decoded = AuthService.decodeSessionVars();
117     2x return (decoded && decoded.role) ? decoded.role : undefined;
118     2x }
119
120 static getUserEmail() {
121     let decoded = AuthService.decodeSessionVars();
122     11x return (decoded && decoded.emailAddress) ? decoded.emailAddress : undefined;
123     11x }
124
125 static decodeSessionVars() {
126     8x const session = AuthService.getSession();
127     8x const token = AuthService.getToken();
128
129     3x if(token && session) {
130         var sessionInfo = jwt.decode(token, session);
131         return sessionInfo;
132     } else {
133         return undefined;
134     } 2x
135 }
136
137 2x static updateTimeStamp(decoded) {
138     decoded.expiration = moment().add(20, 'minutes').format('YYYY-MM-DDTHH:mm:ss.SSS');
139     2x
140     let encoded = jwt.encode(decoded, AuthService.getSession());
141     AuthService.setToken(encoded);
142
143     11x return true;
144 }
145
146 static setSession(session) {
147     23x localStorage.setItem('blob', session);
148 }
149
150 static getSession() {
151     8x return localStorage.getItem('blob');
152 }
153
154 static endSession() {
155     22x localStorage.removeItem('blob');
156 }
157
158 static getToken() {
159     13x return localStorage.getItem('id_token');
160 }
161
162 static setToken(idToken) {
163     8x localStorage.setItem('id_token', idToken);
164 }
165
166 static revokeToken() {
167     localStorage.removeItem('id_token');
168 }
169
170 }

```

All files / [src/utils](#) constants.js

100% Statements 10/10

100% Branches 6/6

100% Functions 4/4

100% Lines 10/10

```
1
2   29x  module.exports.getServerUrl = function () {
3   125x      if(process.env.NODE_ENV && process.env.NODE_ENV.toLowerCase() === 'development') {
4       1x          return 'http://localhost:8080';
5              } else {
6   124x          return window.location.protocol + '//server.sweng500.com';
7              }
8      };
9
10   29x  module.exports.useCredentials = function () {
11   176x      return {withCredentials: true};
12      };
13
14   29x  module.exports.getEmptyModalInfo = function () {
15   20x      return {
16          title: '',
17          body: null,
18          modalAction: ''
19      };
20      };
21
22   29x  module.exports.isEmpty = function (obj) {
23   12x      return Object.keys(obj).length === 0 && obj.constructor === Object;
24      };
25
26
```

All files / src/utils validator.js

100% Statements 25/25

100% Branches 12/12

100% Functions 2/2

100% Lines 25/25

```
1
2 9x module.exports.validateTeamName = function (teamName) {
3 8x   var result = {isValid: false};
4
5 8x   if(!teamName) {
6 3x     result.message = "Team Name is required.";
7 3x     return result;
8   }
9
10 5x   if(teamName.trim().length < 5) {
11 1x     result.message = "Team Name is too short.";
12 1x     return result;
13   }
14
15 4x   let reg = /^[a-z0-9|\s]+$/i;
16 4x   if(!reg.test(teamName.trim())){
17 1x     result.message = "May only include Alphanumeric";
18 1x     return result;
19   }
20
21 3x   result.isValid = true;
22 3x   return result;
23 };
24
25 // Checks to see if an email has a host, @ symbols, and domain.
26 9x module.exports.validateEmail = function(text) {
27 11x   var result = {isValid: false};
28
29 11x   if(!text || text.trim() === "") {
30 5x     result.message = "Email address is required";
31 5x     return result;
32   }
33
34 6x   let reg = /^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*(\\.\\w{2,3})+$/;
35 6x   if(!reg.test(text)) {
36 2x     result.message = "Must be proper Email address";
37 2x     return result;
38   }
39
40 4x   result.isValid = true;
41 4x   return result;
42 };
```

All files / src/views/Dashboard CoachCount.js

78.57% Statements 11/14 50% Branches 1/2 71.43% Functions 5/7 84.62% Lines 11/13

```
1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import HttpRequest from "../../adapters/httpRequest";
4 import constants from "../../utils/constants";
5 import { StatsCard } from "../../components/Cards/StatsCard.js";
6
7 class CoachCount extends Component {
8
9     constructor(props) {
10 1x      super(props);
11
12 1x      this.state = {
13          result: []
14      };
15
16 1x      this.getCoaches();
17
18 1x      this.getCoaches = this.getCoaches.bind(this);
19    }
20
21    getCoaches(event) {
22 1x      if(event) {event.preventDefault()};
23
24 1x      var _this = this;
25 1x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/getCoaches",
26 1x      "get", constants.useCredentials(), null, true).then(function (result) {
27          _this.setState({
28              result: result.body
29          })
30          console.log(result.body);
31      }).catch(function (error) {
32      console.log(error);
33      })
34    }
35
36    formatBody() {
37 2x
38        return (
39            <div>
40                {this.state.result.length}
41                <div className="footer">
42                    <hr />
43                    <div className="stats">
44                        <a style={{cursor:'pointer'}} onClick={event => this.getCoaches(event)}><i
45      className="fa fa-refresh"></i> Update Now</a>
46                    </div>
47                </div>
48            </div>
49        );
50    }
51
52 2x    render() {
53
54        return (
```



```
55         <StatsCard
56             bigIcon={<i className="pe-7s-graph1 text-warning"></i>}
57             statsText="Number of Coaches"
58             statsValue={this.formatBody()}
59         />
60
61     );
62 }
63
64 }
65
export default CoachCount;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Dashboard Dashboard.js

37.5% Statements 3/8 16.67% Branches 1/6 100% Functions 1/1 37.5% Lines 3/8

```
1 import React, { Component } from 'react';
2 import { Grid, Row, Col } from 'react-bootstrap';
3 import ChuckParent from '../../components/HttpExample/ChuckParent.js'
4 import TodaysEvents from '../../views/Dashboard/TodaysEvents.js'
5 import EventCount from "../EventCount";
6 import TeamCount from "../TeamCount";
7 import CoachCount from "../CoachCount";
8 import JudgeCount from "../JudgeCount";
9 import AuthService from "../../utils/AuthService";
10
11 class Dashboard extends Component {
12
13     render() {
14
15         2x const userType = AuthService.getUserRole();
16
17         2x E if (userType === "ADMIN") {
18
19             2x return (
20                 <div className="content">
21                     <Grid fluid>
22                         <Row>
23                             <Col lg={3} sm={6}>
24                                 <EventCount/>
25                             </Col>
26                             <Col lg={3} sm={6}>
27                                 <TeamCount/>
28                             </Col>
29                             <Col lg={3} sm={6}>
30                                 <CoachCount/>
31                             </Col>
32                             <Col lg={3} sm={6}>
33                                 <JudgeCount/>
34                             </Col>
35                         </Row>
36                         <Row>
37                             <Col lg={3} sm={6}>
38                                 <TodaysEvents/>
39                             </Col>
40                             <Col lg={3} sm={6}>
41                                 <ChuckParent/>
42                             </Col>
43                         </Row>
44                     </Grid>
45                 </div>
46             );
47         }
48         else if (userType == "COACH") {
49
50             return (
51                 <div className="content">
52                     <Grid fluid>
53                         <Row>
54                             <Col lg={3} sm={6}>
```

```

55         <EventCount/>
56     </Col>
57     <Col lg={3} sm={6}>
58         <TeamCount/>
59     </Col>
60 </Row>
61 <Row>
62     <Col lg={3} sm={6}>
63         <TodaysEvents/>
64     </Col>
65 </Row>
66 </Grid>
67 </div>
68 );
69 }
70 else if (userType == "JUDGE") {
71     return (
72         <div className="content">
73             <Grid fluid>
74                 <Row>
75                     <Col lg={3} sm={6}>
76                         <EventCount/>
77                     </Col>
78                     <Col lg={3} sm={6}>
79                         <TeamCount/>
80                     </Col>
81                     <Col lg={3} sm={6}>
82                         <CoachCount/>
83                     </Col>
84                 </Row>
85                 <Row>
86                     <Col lg={3} sm={6}>
87                         <TodaysEvents/>
88                     </Col>
89                 </Row>
90             </Grid>
91         </div>
92     );
93 }
94 else
95 {
96     return (
97         <div className="content">
98             <Grid fluid>
99                 <Row>
100                     <Col lg={3} sm={6}>
101                         <TodaysEvents/>
102                     </Col>
103                 </Row>
104             </Grid>
105         </div>
106     );
107 }
108
109 }
110 }
111
112 export default Dashboard;
113

```


All files / src/views/Dashboard EventCount.js

75% Statements 12/16 50% Branches 1/2 62.5% Functions 5/8 85.71% Lines 12/14

```
1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import HttpRequest from "../../adapters/httpRequest";
4 import constants from "../../utils/constants";
5 import { StatsCard } from "../../components/Cards/StatsCard.js";
6
7 class EventCount extends Component {
8
9     constructor(props) {
10 1x      super(props);
11
12 1x      this.state = {
13          result: []
14      };
15
16 1x      this.getEvents();
17
18 1x      this.getEvents = this.getEvents.bind(this);
19    }
20
21    getEvents(event) {
22 1x      if(event) {event.preventDefault()};
23
24 1x      var _this = this;
25 1x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/events", "get",
26 1x      constants.useCredentials(), null, true).then(function (result) {
27          _this.setState({
28              result: result.body
29          })
30          console.log(result.body);
31      }).catch(function (error) {
32          console.log(error);
33      })
34    }
35
36    formatBody() {
37 2x      return (
38          <div>
39              {this.state.result.length}
40              <div className="footer">
41                  <hr />
42                  <div className="stats">
43                      <a style={{cursor:'pointer'}} onClick={event => this.getEvents(event)}><i
44                      className="fa fa-refresh"></i> Update Now</a>
45                  </div>
46              </div>
47          </div>
48      );
49    }
50
51    render() {
52 2x      var refresh = <a style={{cursor:'pointer'}} onClick={event => this.props.getEvents(event)}><i
```

```
55     className="fa fa-refresh"></i> Update Now</a>
56
57     return (
58         <StatsCard
59             bigIcon={<i className="pe-7s-graph1 text-warning"></i>}
60             statsText="Number of Events"
61             statsValue={this.formatBody()}
62         />
63     );
64 }
65
66
67 }

export default EventCount;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Dashboard JudgeCount.js

78.57% Statements 11/14 50% Branches 1/2 71.43% Functions 5/7 84.62% Lines 11/13

```
1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import HttpRequest from "../../adapters/httpRequest";
4 import constants from "../../utils/constants";
5 import { StatsCard } from "../../components/Cards/StatsCard.js";
6
7 class CoachCount extends Component {
8
9     constructor(props) {
10 1x      super(props);
11
12 1x      this.state = {
13          result: []
14      };
15
16 1x      this.judgeCount();
17
18 1x      this.judgeCount = this.judgeCount.bind(this);
19    }
20
21    judgeCount(event) {
22 1x      if(event) {event.preventDefault()};
23
24 1x      var _this = this;
25 1x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/getJudges",
26 1x      "get", constants.useCredentials(), null, true).then(function (result) {
27          _this.setState({
28              result: result.body
29          })
30          console.log(result.body);
31      }).catch(function (error) {
32          console.log(error);
33      })
34    }
35
36    formatBody() {
37 2x      return (
38          <div>
39              {this.state.result.length}
40              <div className="footer">
41                  <hr />
42                  <div className="stats">
43                      <a style={{cursor:'pointer'}} onClick={event => this.judgeCount(event)}><i
44                      className="fa fa-refresh"></i> Update Now</a>
45                  </div>
46              </div>
47          </div>
48      );
49    }
50
51    render() {
52 2x      return (
53
54
```

```
55         <StatsCard
56             bigIcon={<i className="pe-7s-graph1 text-warning"></i>}
57             statsText="Number of Judges"
58             statsValue={this.formatBody()}
59         />
60
61     );
62 }
63
64 }
65
export default CoachCount;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Dashboard TeamCount.js

78.57% Statements 11/14 50% Branches 1/2 71.43% Functions 5/7 84.62% Lines 11/13

```
1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import HttpRequest from "../../adapters/httpRequest";
4 import constants from "../../utils/constants";
5 import { StatsCard } from "../../components/Cards/StatsCard.js";
6
7 class TeamCount extends Component {
8
9     constructor(props) {
10 1x      super(props);
11
12 1x      this.state = {
13          result: []
14      };
15
16 1x      this.getTeams();
17
18 1x      this.getTeams = this.getTeams.bind(this);
19    }
20
21    getTeams(event) {
22 1x      if(event) {event.preventDefault()};
23
24 1x      var _this = this;
25 1x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/getTeams",
26 1x      "get", constants.useCredentials(), null, true).then(function (result) {
27          _this.setState({
28              result: result.body
29          })
30          console.log(result.body);
31      }).catch(function (error) {
32          console.log(error);
33      })
34    }
35
36    formatBody() {
37 2x      return (
38          <div>
39              {this.state.result.length}
40              <div className="footer">
41                  <hr />
42                  <div className="stats">
43                      <a style={{cursor:'pointer'}} onClick={event => this.getTeams(event)}><i
44                      className="fa fa-refresh"></i> Update Now</a>
45                  </div>
46              </div>
47          </div>
48      );
49    }
50
51    render() {
52 2x      return (
53
54
```

```
55         <StatsCard
56             bigIcon={<i className="pe-7s-graph1 text-warning"></i>}
57             statsText="Number of Teams"
58             statsValue={this.formatBody()}
59         />
60     );
61 }
62
63 }
64
export default TeamCount;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Dashboard TodaysEvents.js

75% Statements 12/16 50% Branches 1/2 75% Functions 6/8 80% Lines 12/15

```
1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import HttpRequest from "../../adapters/httpRequest";
4 import constants from "../../utils/constants";
5 import {StatsCard} from '../../components/Cards/StatsCard.js';
6
7 class TodaysEvents extends Component {
8
9     constructor(props) {
10 1x      super(props);
11
12 1x      this.state = {
13          result: []
14      };
15
16 1x      this.getEvents();
17
18 1x      this.getEvents = this.getEvents.bind(this);
19    }
20
21    getEvents(event) {
22 1x      if(event) {event.preventDefault()};
23
24 1x      var _this = this;
25 1x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/events", "get",
26 1x      constants.useCredentials(), null, true).then(function (result) {
27          _this.setState({
28              result: result.body
29          })
30          console.log(result.body);
31 1x      }).catch(function (error) {
32          console.log(error);
33      })
34    }
35
36    formatBody(){
37 2x
38      var namesList = this.state.result.map(function(user, i) {
39          return <li key={i}>{user.name}</li>
40      }
41    )
42 1x
43      return (
44          <div id="contact-chuck-container-div" key="contact-chuck-container-div"
45          className="chuckNorrisClass">
46              <ol> {namesList} </ol>
47              <div className="footer">
48                  <hr />
49                  <div className="stats">
50                      <a style={{cursor:'pointer'}} onClick={event => this.getEvents(event)}><i className="fa
51 fa-refresh"></i> Update Now</a>
52                  </div>
53              </div>
54          </div>
```

```
55     );
56   }
57
58   render() {
59     2x
60
61
62     return (
63       <StatsCard
64         bigIcon={<i className="pe-7s-global text-warning"></i>}
65         statsText="Upcoming Events"
66         statsValue={this.formatBody()}
67       />
68     );
69   }
70 }
71
72 }

```

`export default TodaysEvents;`

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/EasterEgg sudoku.js

100% Statements 1/1 100% Branches 0/0 100% Functions 1/1 100% Lines 1/1

```
1 import React, {Component} from 'react';
2
3 import Sudoku from 'sudoku-react-component';
4
5 class EasterEgg extends Component {
6
7     render() {
8 1x       return (
9           <div id='sudoku-container'>
10              <Sudoku />
11          </div>
12      )
13  }
14
15 }
16
17 export default EasterEgg;
```

All files / src/views/Events EventDetail.js

41.24% Statements 40/97 26.67% Branches 8/30 25.64% Functions 10/39 41.24% Lines 40/97

```
1 import React, {Component} from 'react';
2 import { Modal, PageHeader, Panel, Grid, Col, Row,} from 'react-bootstrap';
3 import HttpRequest from "../../adapters/httpRequest";
4 import constants from "../../utils/constants";
5 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
6 import RaisedButton from 'material-ui/RaisedButton';
7 import "react-table/react-table.css";
8 import {Map, Marker, GoogleApiWrapper} from 'google-maps-react';
9 import Loader from 'react-loader'
10 import Card from '../../components/Cards/Card.js';
11 import Divider from 'material-ui/Divider';
12 import AppBar from 'material-ui/AppBar'
13 import FlatButton from 'material-ui/FlatButton';
14 import FontIcon from 'material-ui/FontIcon';
15 import Dialog from 'material-ui/Dialog';
16
17
18
19 import {
20     Table,
21     TableBody,
22     TableHeader,
23     TableHeaderColumn,
24     TableRow,
25     TableRowColumn,
26 } from 'material-ui/Table';
27 import CustomDropdown from "../../elements/CustomSelector/CustomDropdown";
28 import AuthService from "../../utils/AuthService";
29
30
31 class EventDetail extends Component {
32     constructor(props) {
33         1x super(props);
34         1x this.formatTimeString = this.formatTimeString.bind(this);
35         1x this.selectedTeam = this.selectedTeam.bind(this);
36         1x this.closeTeamModal = this.closeTeamModal.bind(this);
37         1x this.removeJudge = this.removeJudge.bind(this);
38         1x this.removeTeam = this.removeTeam.bind(this);
39         //for the map
40         1x this.divStyle = {
41             height: '300px',
42             width: '450px',
43             position: 'relative',
44             overflow: 'scroll'
45         };
46
47         1x this.state = {
48             loading: false,
49
50             //needed so initial backend call will not have blank eventid
51             eventId: props.eventId,
52             latitude: '',
53             longitude: '',
54             eventDetail: {},
```

```

55         eventDate: '',
56         startTime: '',
57         endTime: '',
58         buildingName: '',
59         judgesDetail: {},
60
61         //teams
62         teamModal: false,
63         //custom dropdown returns an object
64         selectedTeamValue: null,
65         teamSelectorError: '',
66         teamsDetail: {},
67         loadAddTeam: true,
68         //delete
69         showDeleteBtn:true,
70         deleteTeam:false,
71         deleteJudge:false,
72         deleteId:'',
73         confirmDialog:false,
74         confirmMessage:''
75     };
76
77 }
78
79 // If there is a latitude and longitude then display it
80 addMarker = () => {
81     if (this.state.latitude !== undefined)
82         return (
83             <Marker name='Current location' position={{lat: this.state.latitude, lng:
84 this.state.longitude}}/>);
85     }
86
87 //This component gets called initially with main events page, need to update when this is called with a
88 2x no prop
89 2x componentWillReceiveProps(nextProps) {
90     var _this = this;
91     _this.setState({
92         eventId: nextProps.eventId
93     });
94 }
95 1x
96 componentDidMount() {
97 1x     var _this = this;
98     //hide things from non admins
99 1x     let modifyRoles = ['ADMIN', 'COACH'];
100 1x     //eventually only allow an admin to delete judges and teams
101 1x     let deleteRole = ['ADMIN'];
102 1x     let allowModify = AuthService.isUserRoleAllowed(modifyRoles);
103     let allowDelete = AuthService.isUserRoleAllowed(deleteRole);
104     _this.setState({
105         showRegisterBtn: allowModify,
106         showDeleteBtn:allowDelete
107     });
108 1x
109 1x
110 1x     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + "/sweng500/event/" +
111 1x this.state.eventId, "get", constants.useCredentials(), null, true).then(function (result) {
112         const start = _this.formatTimeString(new Date(result.body.startTime));
113         const endT = _this.formatTimeString(new Date(result.body.endTime));
114         const building = result.body.room.buildingID;

```

```

115         _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
116         "/sweng500/getBuilding/" + building, "get", constants.useCredentials(), null, true).then(function
117         (buildResult) {
118
119             _this.setState({
120                 eventDetail: result.body,
121                 latitude: buildResult.body.lat,
122                 longitude: buildResult.body.lng,
123                 buildingName: buildResult.body.building,
124                 eventDate: new Date(result.body.eventDate).toDateString(),
125                 startTime: start,
126                 endTime: endT,
127                 loading: true
128             })
129             }).catch(function (error) {
130 1x         })
131 1x
132         }).catch(function (error) {
133         })
134
135         _this.serverRequestJudge = HttpRequest.httpRequest(constants.getServerUrl() +
136         "/sweng500/event/judges/" + _this.state.eventId, "get", constants.useCredentials(), null,
137         true).then(function (judgeResult) {
138 1x         _this.setState({
139 1x             judgesDetail: judgeResult.body,
140         })
141
142         }).catch(function (error) {
143         })
144
145         _this.serverRequestJudge = HttpRequest.httpRequest(constants.getServerUrl() +
146         "/sweng500/event/teams/" + _this.state.eventId, "get", constants.useCredentials(), null,
147         true).then(function (teamResult) {
148 2x         _this.setState({
149 2x             teamsDetail: teamResult.body,
150 2x         })
151 2x
152 2x         }).catch(function (error) {
153 2x         })
154     }
155
156     formatTimeString(date) {
157         var hours = date.getHours();
158         var minutes = date.getMinutes();
159         var ampm = hours >= 12 ? 'pm' : 'am';
160         hours = hours % 12;
161         hours = hours ? hours : 12;
162         return hours + ":" + minutes + " " + ampm;
163     }
164
165     closeTeamModal() {
166         this.setState({loadAddTeam: true, teamModal: false, teamSelectorError: ''})
167     }
168
169     selectedTeam(value) {
170         this.setState({selectedTeamValue: value})
171     }
172
173     registerTeam() {
174

```



```

175     var _this = this;
176     //custom dropdown stores the entire object so just get the idea
177     if (_this.state.selectedTeamValue === null) {
178         _this.setState({teamSelectorError: 'Please select a team'});
179     } else {
180         var body = {teamName: _this.state.selectedTeamValue.name, eventName:
181         _this.state.eventDetail.name};
182
183         _this.serverRequestJudge = HttpRequest.httpRequest(constants.getServerUrl() +
184         "/sweng500/event/" + _this.state.eventId + "/" + _this.state.selectedTeamValue.id, "POST",
185         constants.useCredentials(), body, true).then(function (judgeResult) {
186 6x         _this.props.addNotification("Success, the team has been registered ", "success", "tc", 6);
187         _this.setState({loadAddTeam: true, teamModal: false, teamSelectorError: ''})
188         _this.componentDidMount();
189
190         }).catch(function (error) {
191         _this.setState({teamSelectorError: 'Team already registered'});
192     })
193
194     }
195 }
196
197 renderIfEventFound() {
198     if (this.state.eventDetail !== null && Object.keys(this.state.eventDetail).length !== 0) {
199         //only display delete events if an admin
200         let showActionBar=null;
201         if(this.state.showDeleteBtn) {
202             showActionBar = <TableHeaderColumn>Action</TableHeaderColumn>
203         }
204         return (
205             <MuiThemeProvider>
206                 <Row>
207                     <Col md={8} mdOffset={2} xs={7}>
208                         <Panel bsStyle="info">
209                             <Panel.Heading>Event </Panel.Heading>
210                             <Panel.Body>
211                                 <PageHeader>
212                                     {this.state.eventDetail.name} <br/>
213                                     <Divider/>
214                                     <small
215                                     style={{'overflow-wrap': 'break-word'}}>
216 {this.state.eventDetail.description}</small>
217                                 </PageHeader>
218                             </Panel.Body>
219                         </Panel>
220                     </Col>
221                     <Divider/>
222                 </Row>
223                 <Row>
224                     <Col md={6} xs={8}>
225                         <Panel bsStyle="info">
226                             <Panel.Heading>Event Details</Panel.Heading>
227                             <Panel.Body><h4>Event Date : {this.state.eventDate} </h4><br/><br/>
228                                 <Divider/>
229                                 <h4>Start Time : {this.state.startTime} </h4> <br/> <br/>
230                                 <Divider/><br/>
231                                 <h4> End Time : {this.state.endTime} </h4></Panel.Body>
232                             </Panel>
233                         </Col>
234                     <Col md={6} xs={8}>

```

```

235     <Panel bsStyle="info">
236       <Panel.Heading>Building: {this.state.buildingName} ----
237       Room: {this.state.eventDetail.room.roomName}</Panel.Heading>
238     <Panel.Body>
239       <div id="map"
240         style={{height: 300, width: 400, marginLeft: 'auto', marginRight:
241         'auto'}}>
242         <Map
243           style={this.divStyle}
244           google={this.props.google}
245           zoomControl={true}
246           initialCenter={{
247             lat: 41.306610,
248             lng: -76.015437
249           }}
250           zoom={16}
251           clickableIcons={false}>
252           {this.addMarker()}
253         </Map>
254       </div>
255     </Panel.Body>
256   </Panel>
257 </Col>
258 </Row>
259 <Row>
260   <Col md={6} xs={8}>
261     <Panel bsStyle="info">
262       <Panel.Heading>Judges</Panel.Heading>
263       <Panel.Body><Table selectable={false}>
264         <TableHeader displaySelectAll={false}
265           adjustForCheckbox={false}>
266           <TableRow>
267             <TableHeaderColumn>First Name</TableHeaderColumn>
268             <TableHeaderColumn>Last Name</TableHeaderColumn>
269             {showActionBar}
270           </TableRow>
271         </TableHeader>
272         {this.renderIfJudgesFound()}
273       </Table></Panel.Body>
274     </Panel>
275   </Col>
276   <Col md={6} xs={8}>
277     <Panel bsStyle="info">
278       <Panel.Heading>Teams</Panel.Heading>
279       <Panel.Body><Table selectable={false}>
280         <TableHeader displaySelectAll={false}
281           adjustForCheckbox={false}>
282           <TableRow>
283             <TableHeaderColumn>Team Name</TableHeaderColumn>
284             <TableHeaderColumn>School Name</TableHeaderColumn>
285             {showActionBar}
286           </TableRow>
287         </TableHeader>
288         {this.renderIfTeamsFound()}
289       </Table></Panel.Body>
290     </Panel>
291   </Col>
292 </Row>
293 </Row>
294 </Row>

```

```

295         </Panel>
296     </Col>
297
298     </Row>
299 </MuiThemeProvider>
300 )
301 }
302 }
303
304 //Return the judges for this event, only show first and last name
305 renderIfJudgesFound() {
306     if (this.state.judgesDetail !== null && Object.keys(this.state.judgesDetail).length !== 0) {
307         let deleteBtnJudge = null;
308         return (
309             <TableBody displayRowCheckbox={false}
310                 showRowHover={true}>
311                 {
312                     Object.keys(this.state.judgesDetail).map(function (key) {
313                         if(this.state.showDeleteBtn) {
314                             deleteBtnJudge= <TableRowColumn><RaisedButton icon={<FontIcon
315 className="pe-7s-trash" />}
316                                     secondary={true} label="Delete"
317                                     onClick={this.confirmJudgeDelete.bind(this,
318 this.state.judgesDetail[key])}/>
319                             </TableRowColumn>
320                         }
321                         return (
322                             <TableRow key={key}>
323                                 <TableRowColumn>{this.state.judgesDetail[key].firstName}
324 </TableRowColumn>
325                                 <TableRowColumn>{this.state.judgesDetail[key].lastName}
326 </TableRowColumn>
327                                 {deleteBtnJudge}
328
329                             </TableRow>
330                         )
331
332                     }, this)
333                 }
334             </TableBody>
335         )
336     }
337 }
338
339 confirmJudgeDelete = (s) => {
340     this.setState({
341         confirmDialog:true,
342         confirmMessage:'Are you sure you want to delete the following judge: ' + s.firstName + " " +
343 s.lastName,
344         deleteTeam:false,
345         deleteJudge:true,
346         deleteId:s.id
347     })
348 }
349
350 removeJudge() {
351
352     var _this = this;
353     var removeId = _this.state.deleteId;
354     //just making remove a post for now

```

```

355     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
356     "/sweng500/event/"+this.state.eventId+"/removeJudge/" + removeId, "DELETE", constants.useCredentials(),
357     null, true).then(function (result) {
358         _this.setState({confirmDialog: false});
359         _this.props.addNotification(
360             "Success: The judge has been removed",
361             "info",
362             "tc",
363             6
364         );
365         _this.componentDidMount();
366     }).catch(function (error) {
367         _this.setState({confirmDialog: false});
368         _this.props.addNotification(
369             "Error: The judge could not be removed",
370             "error",
371             "tc",
372             6
373         );
374     })
375 }
376
377 //Return the judges for this event, only show first and last name
378 renderIfTeamsFound() {
379     if (this.state.teamsDetail !== null && Object.keys(this.state.teamsDetail).length !== 0) {
380         let deleteBtnTeam = null;
381
382         return (
383             <TableBody displayRowCheckbox={false}
384                 showRowHover={true}>
385                 {
386                     Object.keys(this.state.teamsDetail).map(function (key) {
387                         if(this.state.showDeleteBtn) {
388                             deleteBtnTeam= <TableRowColumn> <RaisedButton icon={<FontIcon
389 className="pe-7s-trash" />}
390                                     secondary={true} label="Delete"
391                                     onClick={this.confirmTeamDelete.bind(this,
392 this.state.teamsDetail[key])}/>
393                             </TableRowColumn>
394                         }
395                         return (
396                             <TableRow key={key}>
397                                 <TableRowColumn>{this.state.teamsDetail[key].name}</TableRowColumn>
398                                 <TableRowColumn>{this.state.teamsDetail[key].school.schoolName}
399 </TableRowColumn>
400                                 {deleteBtnTeam}
401
402                             </TableRow>
403                         )
404
405                     }, this)
406                 }
407             </TableBody>
408         )
409     }
410 }
411 confirmTeamDelete = (s) => {
412     this.setState({
413         confirmDialog:true,
414         confirmMessage:'Are you sure you want to delete the following team: ' + s.name,

```

```

415         deleteTeam:true,
416         deleteJudge:false,
417         deleteId:s.id
418     })
419 }
420
421 removeTeam() {
422     var _this = this;
423     var removeId = _this.state.deleteId;
424     //just making remove a post for now
425     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
426 6x  "/sweng500/event/"+this.state.eventId+"/removeTeam/" + removeId, "DELETE", constants.useCredentials()),
427 6x  null, true).then(function (result) {
428     _this.setState({confirmDialog: false});
429     _this.props.addNotification(
430         "Success: The team has been removed",
431         "info",
432         "tc",
433         6
434     );
435     _this.componentDidMount();
436     }).catch(function (error) {
437     _this.setState({confirmDialog: false});
438     _this.props.addNotification(
439         "Error: The team could not be removed",
440         "error",
441 6x  "tc",
442         6
443     );
444     })
445 }
446
447 closeConfirmDialog = () => {
448     this.setState({confirmDialog: false});
449 }
450
451 render() {
452     let deleteActions=null;
453     I if(this.state.deleteTeam) {
454 6x     deleteActions = [
455         <FlatButton
456             label="Cancel"
457             primary={true}
458             onClick={this.closeConfirmDialog}
459         />,
460         <FlatButton
461 6x     label="Delete"
462         primary={true}
463 6x     onClick={this.removeTeam}
464 5x     />,
465     ];
466     } else {
467         deleteActions = [
468             <FlatButton
469                 label="Cancel"
470                 primary={true}
471 1x     onClick={this.closeConfirmDialog}
472             />,
473             <FlatButton
474 6x     label="Delete"

```

```

        primary={true}
        onClick={this.removeJudge}
      />,
    ];
  }
  let modalTitleBar = <AppBar>
    iconElementRight={<FlatButton label="Close"/>}
    showMenuIconButton={false}
    onRightIconButtonClick={(event) => this.closeTeamModal()}
    title="Register a team to an event"
  />

  let cardButtons=null;
  //only show register buttons to admins and coaches
  if (this.state.showRegisterBtn) {
    cardButtons = <div><RaisedButton icon={<FontIcon className="pe-7s-angle-left-circle"/>}
primary={true}
                                label="Go back to events"
                                onClick={event => this.props.showEvents(event)}/>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~
                                <RaisedButton icon={<FontIcon className="pe-7s-study"/>} primary={true} label="Register a
Team"
                                onClick={event => this.showTeamModal(event)}/></div>

    } else {
      cardButtons = <RaisedButton icon={<FontIcon className="pe-7s-angle-left-circle" />} primary=
{true} label="Go back to events"
                                onClick={event => this.props.showEvents(event)}/>
    }
  return (
    <div className="content">
      <MuiThemeProvider>
        <Grid fluid>
          <Row>
            <Col md={12}>
              <Card>
                hCenter={this.state.eventDetail.eventName}
                // title={this.state.eventDetail.eventName}
                category={
                  <div>
                    {cardButtons}
                  </div>
                }
                content={
                  <Loader color="#3498db" loaded={this.state.loading}>
                    {this.renderIfEventFound()}
                  </Loader>
                }
              </Card>
            </Col>
          </Row>
        </Grid>
        <Modal bsSize="medium" show={this.state.teamModal} onHide={this.closeTeamModal}>
          <Modal.Header>
            <Modal.Title> {modalTitleBar}</Modal.Title>
          </Modal.Header>

          <Modal.Body>
            <Grid>
              <Row>
                <Col xs={7} md={3}>
                  <CustomDropdown

```

```

535         name={"team"}
536         labelText={"Team"}
537         hintText={"Select team"}
538         selected={this.state.selectedTeamValue}
539         endpoint={"/sweng500/getTeamsByUser"}
540         sortKey={"name"}
541         textKeys={["name"]}
542         selectedValue={this.selectedTeam}
543         errorText={this.state.teamSelectorError}
544     />
545 </Col>
546 </Row>
547 </Grid>
548 </Modal.Body>
549
550 <Modal.Footer>
    <Loader loaded={this.state.loadAddTeam}></Loader>
    <RaisedButton icon={<FontIcon className="pe-7s-like2"/>} primary={true}
        label="Register Team"
        onClick={event => this.registerTeam()}/>
</Modal.Footer>
</Modal>
<Dialog
    actions={deleteActions}
    modal={false}
    open={this.state.confirmDialog}
    onRequestClose={this.closeConfirmDialog}
>
    {this.state.confirmMessage}
</Dialog>
</MuiThemeProvider>

</div>

    )
}

showTeamModal(event) {
    this.setState({teamModal: true})
}

export default GoogleApiWrapper({
    apiKey: "AIzaSyC7xiiV97LyRQd-GB9aBmiJaYFGW5DVIbM"
})(EventDetail);

```

All files / src/views/Events Events.js

74.44% Statements 198/266 76.74% Branches 66/86 44.23% Functions 23/52 74.44% Lines 198/266

```
1 import React, {Component} from 'react';
2 import {PageHeader, Well, Alert, Grid, Col, Row, Modal} from 'react-bootstrap';
3 import Loader from 'react-loader';
4 import {
5     Step,
6     Stepper,
7     StepLabel,
8     StepContent,
9 } from 'material-ui/Stepper';
10 import $ from 'jquery';
11 import HttpRequest from "../../adapters/httpRequest";
12 import constants from "../../utils/constants";
13 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
14 import RaisedButton from 'material-ui/RaisedButton';
15 import TextField from 'material-ui/TextField';
16 import DatePicker from 'material-ui/DatePicker';
17 import TimePicker from 'material-ui/TimePicker';
18 import AppBar from 'material-ui/AppBar';
19 import {blue500} from 'material-ui/styles/colors';
20 import FlatButton from 'material-ui/FlatButton';
21 import FontIcon from 'material-ui/FontIcon';
22 import ReactTable from 'react-table';
23 import "react-table/react-table.css";
24 import matchSorter from 'match-sorter';
25 import SelectField from 'material-ui/SelectField';
26 import MenuItem from 'material-ui/MenuItem';
27 import Dialog from 'material-ui/Dialog';
28 import EventDetail from './EventDetail';
29 import NotificationSystem from 'react-notification-system';
30 import {style} from "../../variables/Variables";
31 import BuildingSelector from "../../components/Buildings/BuildingSelector";
32 import * as Promises from "axios";
33 import AuthService from "../../utils/AuthService";
34
35 class Events extends Component {
36     constructor(props) {
37         super(props);
38         this.closeModal = this.closeModal.bind(this);
39         this.createNewEvent = this.createNewEvent.bind(this);
40         this.createEventPost = this.createEventPost.bind(this);
41         this.removeEvent = this.removeEvent.bind(this);
42         this.nextStep = this.nextStep.bind(this);
43         this.previousStep = this.previousStep.bind(this);
44         this.eventDetails = this.eventDetails.bind(this);
45         this.showEvents = this.showEvents.bind(this);
46         this.addJudgeInputs = this.addJudgeInputs.bind(this);
47         this.removeNewJudge = this.removeNewJudge.bind(this);
48         this.addNotification = this.addNotification.bind(this);
49         this.buildingCallback = this.buildingCallback.bind(this);
50
51         this.state = {
52             events: {},
53             editMode: false,
54             editEventId: '',
```



```

55         loading: false,
56         loadCreateEvent: true,
57         modal: false,
58         confirmDialog: false,
59         confirmMessage: '',
60         deleteID: '',
61         stepIndex: 0,
62         //used to send to event detail as a prop
63         eventId: '',
64
65         eventName: '',
66         eventDate: '',
67         startTime: '',
68         endTime: '',
69         eventLocation: '',
70         eventDescription: '',
71         //for adding judges
72         judgeInputs: [],
73         //keep track of the actual new Judges
74         judgeCount: 0,
75         //hold the existing judges
76         existingJudgeValues: [],
77         existingJudgeEmails: [],
78         _notificationSystem: null
79
80
81     };
82 }
83
84
85     buildingCallback(event, index, value) {
86
87         this.setState({eventLocation: value});
88     }
89
90     addNotification(message, level, position, autoDismiss) {
91         if(this.state._notificationSystem) {
92             this.state._notificationSystem.addNotification({
93                 title: (<span data-notify="icon" className="pe-7s-note2"></span>),
94                 message: (
95                     <div>
96                         {message}
97                     </div>
98                 ),
99                 level: level ? level : 'info',
100                 position: position ? position : 'tc',
101                 autoDismiss: autoDismiss ? autoDismiss : 10,
102             });
103         }
104     }
105
106     //launch the modal to enter an event or edit one
107     createNewEvent = (mode) => {
108         const _this = this;
109         if(mode.status === "edit") {
110             _this.serverRequestJudge = HttpRequest.httpRequest(constants.getServerUrl() +
111                 "/sweng500/event/judges/" + mode.id, "get", constants.useCredentials(), null, true).then(function
112                 (judgeResult) {
113                     let judgeVals = judgeResult.body;
114                     //have the judges for this event already selected

```

```

115         var selectedJudgeIds = [];
116         for (let value in judgeVals) {
117             selectedJudgeIds.push(judgeVals[value].id);
118         }
119         _this.setState({
120             modal: true,
121             stepIndex: 0,
122             editEventId: mode.id,
123             editMode: true,
124             eventName: mode.name,
125             eventDate: new Date(mode.eventDate),
126             startTime: new Date(mode.startTime),
127             endTime: new Date(mode.endTime),
128             eventDescription: mode.description,
129             eventLocation: mode.room.id,
130             existingJudgeValues: selectedJudgeIds,
131             judgeInputs: [],
132             judgeCount: 0,
133         })
134     })
135
136     }).catch(function (error) {
137         //no judges assigned to the event yet
138         _this.setState({
139             editEventId: mode.id,
140             modal: true,
141             stepIndex: 0,
142             editMode: true,
143             eventName: mode.name,
144             eventDate: new Date(mode.eventDate),
145             startTime: new Date(mode.startTime),
146             endTime: new Date(mode.endTime),
147             eventDescription: mode.description,
148             eventLocation: mode.room.id,
149             existingJudgeValues: [],
150             judgeInputs: [],
151             judgeCount: 0,
152         })
153     })
154 }
155 } else {
156     //creating new event
157     _this.setState({
158         modal: true,
159         editMode: false,
160         editEventId: '',
161         stepIndex: 0,
162         eventName: '',
163         eventDate: '',
164         startTime: '',
165         endTime: '',
166         eventDescription: '',
167         eventLocation: '',
168         existingJudgeValues: [],
169         judgeInputs: [],
170         judgeCount: 0,
171     })
172 }
173 }

```

```

175 2x createEventPost() {
176 2x     var _this = this;
177 2x     var body = {};
178     var gatherjudges = {};
179 2x     var newJudgeList = [];
180 1x     var i;
181 1x     for (i = 1; i <= this.state.judgeCount; i++) {
182 1x         var tempFname = "#judgefname" + i;
183         var tempLname = "#judgelname" + i;
184 1x         var tempEmail = "#judgemail" + i;
185 1x         // alert($(tempFname).val() + " " + $(tempLname).val() + " " + $(tempEmail).val());
186 1x         gatherjudges.fname = $(tempFname).val();
187         gatherjudges.lname = $(tempLname).val();
188 1x         gatherjudges.email = $(tempEmail).val();
189 1x
190         newJudgeList.push(gatherjudges);
191         gatherjudges = {};
192
193
194 2x     }
195
196 2x     var event = {};
197 2x
198 2x     event.name = this.state.eventName;
199 2x     event.description = this.state.eventDescription;
200 2x     event.eventDate = this.state.eventDate;
201 2x     event.room = this.state.eventLocation;
202     event.startTime = this.state.startTime;
203 2x     event.endTime = this.state.endTime;
204     //the actual string
205 2x     body.eventJson = event;
206 2x     //existing judge values has an extra pair of quotes around each item in the list...
207 2x     body.existingJudgeValues = this.state.existingJudgeValues;
208 2x     body.newJudgeValues = newJudgeList;
209     console.log(JSON.stringify(body));
210     _this.setState({
211         //show the spinner
212 2x         loadCreateEvent: false
213 2x     });
214     E if(this.state.editMode) {
215         _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
216 2x "/sweng500/updateEvent/"+this.state.editEventId, "POST", constants.useCredentials(), body,
217 true).then(function (result) {
218         //show some success and then clear the fields
219         //refresh after an add to show new event in table
220         _this.setState({
221             modal: false,
222             stepIndex: 0,
223             loadCreateEvent: true,
224             eventName: '',
225             eventDescription: '',
226             //this is a date object
227             eventDate: '',
228             startTime: '',
229             endTime: '',
230             eventLocation: '',
231             judgeInputs: [],
232             judgeCount: 0,
233             existingJudgeValues: [],
234 2x             existingJudgeEmails: [],

```

```

235         renderDetails: false,
236
237     });
238     _this.addNotification(
239         "Success: The event has been saved.",
240         "success",
241         "tc",
242         6
243     );
244     _this.componentDidMount();
245
246     }).catch(function (error) {
247         _this.setState({
248             loadCreateEvent: true,
249         });
250         _this.addNotification(
251             "Error: There was a problem editing the event.",
252             "error",
253             "tc",
254             6
255         );
256         _this.componentDidMount();
257     })
258     } else {
259         _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
260         "/sweng500/addEvent/", "POST", constants.useCredentials(), body, true).then(function (result) {
261             //show some success and then clear the fields
262             //refresh after an add to show new event in table
263             _this.setState({
264                 modal: false,
265                 stepIndex: 0,
266                 loadCreateEvent: true,
267                 eventName: '',
268                 eventDescription: '',
269                 //this is a date object
270                 eventDate: '',
271                 startTime: '',
272                 endTime: '',
273                 eventLocation: '',
274                 judgeInputs: [],
275                 judgeCount: 0,
276                 existingJudgeValues: [],
277                 existingJudgeEmails: [],
278                 renderDetails: false,
279
280             });
281             _this.addNotification(
282                 "Success: The event has been added.",
283                 "success",
284                 "tc",
285                 6
286             );
287             _this.componentDidMount();
288
289             }).catch(function (error) {
290                 _this.setState({
291                     loadCreateEvent: true
292                 });
293                 _this.addNotification(
294                     "Error: There was a problem creating the event.",

```

```

295         "error",
296         "tc",
297         6
298     );
299     _this.componentDidMount();
300 })
301 1x    }
302    }
303
304    closeModal() {
305        //reset when closing modal
306        this.setState({
307            modal: false,
308            editMode: false,
309            editEventId: ''
310        })
311    }
312 1x
313
314 3x    // Checks to see if an email has a host, @ symbols, and domain.
315    validEmail(text) {
316        let reg = /^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*\\.\\w{2,3}+$/;
317        if (reg.test(text) === false)
318            return true;
319        else
320            return false;
321    }
322 6x
323
324 6x    nextStep() {
325        //validate event entries here
326
327        2x    var _this = this;
328        2x    var missingInfo = false;
329        1x    const {eventName, stepIndex} = this.state;
330        1x    // Checks first name
331    if (stepIndex == 0) {
332        var promises = [];
333        if (this.state.eventName.length < 1) {
334            missingInfo = true;
335            this.setState({
336                eventName: this.state.eventName.trim(),
337                eventNameError: "Event name is required"
338            })
339            //only check if duplicate event if we are creating a new event
340        } else if (!this.state.editMode){
341            promises.push(_this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
342            2x    "/sweng500/verifyEvent/" + eventName, "get", constants.useCredentials(), null, true));
343        } else {
344            1x    this.setState({
345                1x    eventNameError: undefined
346            })
347        }
348
349        1x    if (this.state.eventDate.length < 1) {
350            missingInfo = true;
351            this.setState({
352                eventDateError: "Event date is required"
353            })
354        } else {
355            this.setState({
356                eventDateError: undefined
357            })

```

```

355     }
356     1x if (this.state.startTime.length < 1) {
357         missingInfo = true;
358         this.setState({
359             startTimeError: "Event start time is required"
360         })
361     } else {
362     1x         this.setState({
363             startTimeError: undefined
364         })
365     }
366     1x if (this.state.endTime.length < 1) {
367         missingInfo = true;
368         this.setState({
369             endTimeError: "Event end time is required"
370         })
371     } else {
372     1x         this.setState({
373             endTimeError: undefined
374         })
375     }
376     if (this.state.eventLocation.length < 1) {
377     1x         missingInfo = true;
378         this.setState({
379             eventLocationError: "Please select a building for the event"
380         })
381     }
382     2x else {
383     1x         this.setState({
384     1x             eventLocationError: undefined
385         })
386     }
387
388     1x if (this.state.eventDescription.length < 1) {
389         missingInfo = true;
390         this.setState({
391             eventDescriptionError: "Event description is required"
392         })
393     } else {
394     1x         this.setState({
395     1x             eventDescriptionError: undefined
396         })
397     }
398
399     if (!missingInfo) {
400         E if(promises.length > 0) {
401             Promises.all(promises).then(function () {
402
403                 _this.setState({
404                     eventNameError: undefined,
405                     stepIndex: stepIndex+1
406                 })
407
408             }).catch(function (error) {
409                 _this.setState({
410                     eventNameError: "Event Name already exists"
411                 });
412                 console.log(error);
413             })
414     4x         } else {

```

```

415         _this.setState({
416             stepIndex: stepIndex + 1
417         })
418     }
419 }
420 } else if (stepIndex == 1) {
421     //no validation since we are just pulling judges from the db
422     //clear the new judge inputs since we don't keep state on them
423     _this.setState({
424         stepIndex: stepIndex + 1,
425         judgeInputs: [],
426         judgeCount: 0
427     })
428 } else {
429     //only verify the last email and send to create event post
430
431     var validateFields = false;
432     let judgeCnt = _this.state.judgeCount;
433     let idname = "#judgemail" + judgeCnt;
434     var promises = [];
435     //do not run when displaying the first set of inputs
436     if (judgeCnt > 0) {
437         var email = $(idname).val();
438         var body = {};
439         E if (email != null && email.length > 0) {
440             E if (!this.validEmail(email.trim())) {
441                 body.emailAddress = email;
442                 promises.push(_this.serverRequest =
443                 HttpRequest.httpRequest(constants.getServerUrl() +
444                     '/sweng500/emailAvailable', 'POST', null, body));
445             } else {
446                 validateFields = true;
447                 $(idname).parent().parent().parent().parent().find(".errorText").text("Email is
448 not in the correct format");
449                 $(idname).parent().parent().parent().parent().find(".errorText").css("display",
450 "block");
451             }
452         } else {
453             validateFields = true;
454             $(idname).parent().parent().parent().parent().find(".errorText").text("Email is
455 required");
456             $(idname).parent().parent().parent().parent().find(".errorText").css("display",
457 "block");
458         }
459     }
460     //only continue to create event if this email is good
461     if (promises.length > 0) {
462         Promises.all(promises).then(function (results) {
463             $(idname).attr("disabled", true);
464             $(idname).parent().parent().parent().parent().find(".errorText").text("");
465             $(idname).parent().parent().parent().parent().find(".errorText").css("display",
466 "none");
467             //finally send the event request
468             _this.createEventPost();
469
470         }).catch(function (error) {
471             $(idname).parent().parent().parent().parent().find(".errorText").text("Email is
472 already in use");
473             $(idname).parent().parent().parent().parent().find(".errorText").css("display",
474 "block");

```

```

475     });
476   } else {
477     //just create event if no new judges and no syntax error
478     E if (!validateFields) {
479       if (_this.state.existingJudgeValues.length > 0) {
480         _this.createEventPost()
481       } else {
482         _this.addNotification(
483           "Error: Please assign at least one judge to the event or create a new one.",
484           "error",
485           "tc",
486           6
487         );
488         _this.componentDidMount();
489       }
490     }
491   }
492 }
493 1x
494
495 //go back a step while creating event
496 previousStep() {
497   1x const {stepIndex} = this.state;
498   1x this.setState({
499     stepIndex: stepIndex - 1
500   })
501 }
502
503 //go to the event page to show all of the information available to edit
504 //will need a similar method to go back to event main page..
505 eventDetails(id) {
506   //alert(id);
507   this.setState({
508     renderDetails: true,
509     eventId: id
510   })
511   $('#eventDetails').removeClass('collapse');
512   $('#eventPage').addClass('collapse');
513   1x
514   1x }
515
516 //called from the subclass to go back to events
517 1x showEvents(event) {
518   1x event.preventDefault();
519   1x this.setState({
520     renderDetails: false
521   })
522   $('#eventDetails').addClass('collapse');
523   $('#eventPage').removeClass('collapse')
524   1x }
525
526 removeEvent() {
527   var _this = this;
528   var removeId = _this.state.deleteID;
529   //just making remove a post for now
530   _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
531     "/sweng500/removeEvent/" + removeId, "DELETE", constants.useCredentials(), null, true).then(function
532     (result) {
533       _this.setState({confirmDialog: false});
534       _this.addNotification(

```



```

535         "Success: The event has been deleted.",
536     1x         "info",
537         "tc",
538         6
539     );
540     _this.componentDidMount();
541 }).catch(function (error) {
542     _this.setState({confirmDialog: false});
543     _this.addNotification(
544         "Error: The event has not been deleted.",
545     2x         "error",
546         "tc",
547     2x         6
548     );
549     2x _this.componentDidMount();
550     2x console.log(error);
551     2x })
552     _this.setState({confirmDialog: false});
553     2x
554     1x
555     1x }
556     1x
557     1x //dynamically add in new input fields when clicked, validate the previous entry first
558     1x addJudgeInputs() {
559     1x     //get a local copy so we dont set state here and re-render, update state after
560
561     var _this = this;
562     //control whether we execute the promise based on validation
563     var validateFields = true;
564
565     let judgeCnt = _this.state.judgeCount;
566     let idname = "#judgemail" + judgeCnt;
567     var promises = [];
568     //do not run when displaying the first set of inputs
569     if (judgeCnt > 0) {
570         var email = $(idname).val();
571         var body = {};
572     2x         E if (email != null && email.length > 0) {
573     2x             E if (!this.validEmail(email.trim())) {
574                 body.emailAddress = email;
575                 promises.push(_this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl()
576 + '/sweng500/emailAvailable', 'POST', null, body));
577             } else {
578                 //not the prettiest code...
579                 $(idname).parent().parent().parent().parent().find(".errorText").text("Email is not
580 in the correct format");
581                 $(idname).parent().parent().parent().parent().find(".errorText").css("display",
582 "block");
583     2x                 validateFields = false;
584             }
585         } else {
586             $(idname).parent().parent().parent().parent().find(".errorText").text("Email is
587 required");
588             $(idname).parent().parent().parent().parent().find(".errorText").css("display",
589 "block");
590     2x             validateFields = false;
591         }
592     }
593     E if (validateFields) {
594         judgeCnt = judgeCnt + 1;

```

```

595
596     const fnameJudge = [
597         <TextField
598             id={"judgefname" + judgeCnt}
599             floatingLabelText="First Name"
600             required={true}
601             autoFocus={true}
602             fullWidth={true}
603         />];
604     const lnameJudge = [
605         <TextField
606             id={"judgelname" + judgeCnt}
607             floatingLabelText="Last Name"
608             required={true}
609             fullWidth={true}
610         />];
611     const emailJudge = [
612         <TextField
613             id={"judgemail" + judgeCnt}
614             floatingLabelText="Email"
615             required={true}
616             fullWidth={true}
617         />];
618
619     //only add the new row if the previous email is valid or
620     if (promises.length > 0) {
621         Promises.all(promises).then(function (results) {
622             $(idname).attr("disabled", true);
623             $(idname).parent().parent().parent().parent().find(".errorText").text("");
624             $(idname).parent().parent().parent().parent().find(".errorText").css("display",
625 "none");
626
627             _this.setState({
628                 judgeCount: _this.state.judgeCount + 1,
629
630                 //add judges class to use jquery to loop over reach one
631                 judgeInputs: _this.state.judgeInputs.concat(<Row>
632                     <div className={"col-md-9 col-xs-7" + " row" + judgeCnt}><Well>
633                         <h3>New Judge {judgeCnt}</h3>
634                         <Row><Alert style={{display: "none"}} bsStyle="danger"
635                             className="errorText"></Alert></Row>
636                         <Row><Col md={4} xs={7}>{fnameJudge}</Col></Row>
637                         <Row><Col md={4} xs={7}>{lnameJudge}</Col></Row>
638                         <Row><Col md={4} xs={7}>{emailJudge}</Col></Row></Well>
639                     </div>
640                 </Row>)
641             });
642         }).catch(function (error) {
643             //ugly way to add error text
644
645             $(idname).attr("disabled", false);
646             $(idname).parent().parent().parent().parent().find(".errorText").text("Email is
647 already in use");
648             $(idname).parent().parent().parent().parent().find(".errorText").css("display",
649 "block");
650
651             });
652         } else {
653             _this.setState({
654                 judgeCount: _this.state.judgeCount + 1,

```

```

655 1x //add judges class to use jquery to loop over reach one
656 judgeInputs: _this.state.judgeInputs.concat(<Row>
657 1x <div className={"col-md-9 col-xs-7" + " row" + judgeCnt}><Well>
658 <h3>New Judge {judgeCnt}</h3>
659 <Row> <Alert style={{display: "none"}} className="errorText"
660 1x bsStyle="danger"></Alert></Row><Row>
661 1x <Col md={4} xs={7}>{fnameJudge}</Col></Row>
662 <Row><Col md={4} xs={7}>{lnameJudge}</Col></Row>
663 <Row><Col md={4} xs={7}>{emailJudge}</Col></Row></Well>
664 </div>
665 </Row>)
666 });
667 }
668 }
669 }
670 }
671 }
672 1x //Remove the text fields for the most recent judge
673 1x removeNewJudge = (judgeCount) => {
674 //alert("judge count" + judgeCount);
675 var _this = this;
676 var tempCount = _this.state.judgeCount;
677 var domId = ".row" + tempCount;
678 $(domId).remove();
679 tempCount -= 1;
680 domId = ".row" + tempCount;
681 //reenable the previous row
682 22x $(domId).find("#judgemail" + tempCount).attr("disabled", false);
683 22x
684 22x //remove from the array and decrement
685 _this.state.judgeInputs.pop();
686 _this.setState({
687 judgeCount: _this.state.judgeCount - 1,
688
689 22x })
690 22x
691
692 22x }
693 22x
694 22x // Delete the event
695 22x confirmEventDelete = (s) => {
696 // alert("Got here");
697 this.setState({deleteID: s.id});
698 this.setState({confirmMessage: "Are you sure you want to delete : " + s.name + " ?"},
699 confirmDialog: true));
700 }
701
702 closeConfirmDialog = () => {
703 22x this.setState({confirmDialog: false});
704 }
705
706 componentDidMount() {
707 //hide things from non admins
708 let modifyRoles = ['ADMIN'];
709 let allowModify = AuthService.isUserRoleAllowed(modifyRoles);
710 this.setState({
711 showDeletebtn: allowModify
712 });
713
714 148x //Make call out to backend

```

[illegible]

```
775     this.state.events[value]]})></div><div><div>  
776         <RaisedButton style={{minWidth: '30%'}} icon={<FontIcon className="pe-7s-trash"  
777             />}  
778             secondary={true} label="Delete"  
779             onClick={this.confirmEventDelete.bind(this, this.state.events[value])}/>  
780         </div>  
781     } else {  
782         this.state.events[value].menuActions = <div>  
783             <RaisedButton style={{minWidth: '30%'}} icon={<FontIcon className="pe-7s-note2"  
784                 />}  
785             primary={true} label="View Details"  
786             onClick={(event) => this.eventDetails(this.state.events[value].id)}></div>  
787         }  
788     }  
789  
790  
791     return (  
792         <ReactTable  
793             data={this.state.events}  
794             filterable  
795             defaultFilterMethod={({filter, row) =>  
796                 String(row[filter.id]) === filter.value}  
797             columns={columns}  
798             defaultPageSize={10}  
799             className="-striped -highlight"  
800             defaultSorted={[{id: "name"}]}  
801         />  
802     )  
803   } else {  
804       <h1>Not Found</h1>  
805   }  
806 }  
807  
808 render() {  
809     const styles = {  
810         formFields: {  
811             color: blue500,  
812             textAlign: 'center',  
813             width: '100%'  
814         }  
815     };  
816  
817     let actionButton = null;  
818     let backButton = null;  
819     //control which buttons show up in the modal  
820     if (this.state.stepIndex == 0) {  
821         actionButton =  
822             <RaisedButton icon={<FontIcon className="pe-7s-angle-down-circle"/>} primary={true}  
823             label="Judges"  
824             onClick={this.nextStep}  
825             />;  
826     } else if (this.state.stepIndex == 1) {  
827         backButton =  
828             <RaisedButton icon={<FontIcon className="pe-7s-angle-up-circle"/>} primary={true}  
829             label="Back to Info"  
830             onClick={this.previousStep}>>;  
831         actionButton = <RaisedButton icon={<FontIcon className="pe-7s-angle-down-circle"/>} primary=  
832         {true}  
833             label="Add new judges"  
834             onClick={this.nextStep}>>;
```

```

835 148x
836 9x      } else {
837          backButton =
838              <RaisedButton icon={<FontIcon className="pe-7s-angle-up-circle"/>} primary={true}
839                  label="Back to Existing"
840                  onClick={this.previousStep}/>;
841 148x      if(this.state.editMode) {
842 128x          actionButton = <RaisedButton icon={<FontIcon className="pe-7s-
843 like2"/>} primary={true} label="Save Event"
844                                  onClick={this.nextStep}/>;
845              } else {
846                  actionButton = <RaisedButton icon={<FontIcon className="pe-7s-
847 like2"/>} primary={true} label="Create Event"
848                                  onClick={this.nextStep}/>;
849              }
850
851 148x      }
852 39x      if (this.state.renderDetails) {
853          return (
854              <div class="content">
855                  <NotificationSystem ref="notificationSystem" style={style}/>
856                  <EventDetail addNotification={this.addNotification} showEvents={this.showEvents}
857 eventId={this.state.eventId}/>
858              </div>
859 109x          )
860      }
861
862      const deleteActions = [
863          <FlatButton
864              label="Cancel"
865              primary={true}
866 148x              onClick={this.closeConfirmDialog}
867          />,
868          <FlatButton
869              label="Delete"
870              primary={true}
871              onClick={this.removeEvent}
872          />,
873      ];
874      let removeJudgeBtn;
875      if (this.state.judgeCount > 0) {
876          removeJudgeBtn = <RaisedButton icon={<FontIcon className="pe-7s-close"/>}
877                                  secondary={true} label="Remove judge"
878                                  onClick={this.removeNewJudge.bind(this)}/>;
879      }
880      let createEventBtn;
881      if (this.state.showDeletebtn) {
882          createEventBtn = <Row className="show-grid">
883              <Col md={4} mdOffset={4}>
884                  <RaisedButton icon={<FontIcon className="pe-7s-display1" />} primary={true}
885 label="Create New Event"
886                                  onClick={this.createNewEvent}/>
887              <br/>
888              </Col>
889          </Row>
890      }
891      let modalTitleBar;
892      if(this.state.editMode) {
893          modalTitleBar= <AppBar
894              iconElementRight={<FlatButton label="Close"/>}

```

```

895         showMenuIconButton={false}
896         onRightIconButtonClick={(event) => this.closeModal()}
897         title="Modify Event"
898     />
899 } else {
900     modalTitleBar= <AppBar
901         iconElementRight={<FlatButton label="Close"/>}
902         showMenuIconButton={false}
903         onRightIconButtonClick={(event) => this.closeModal()}
904         title="Create New Event"
905     />
906 }
907 return (
908     <div className="content">
909         <NotificationSystem ref="notificationSystem" style={style}/>
910         <div id="eventPage" key="notFound-key" className="notFoundClass">
911             <MuiThemeProvider>
912                 <Grid fluid>
913                     {createEventBtn}
914                     <br/>
915                     <br/>
916                     <Row className="show-grid">
917
918                         <Loader color="#3498db" loaded={this.state.loading}>
919
920                             {this.renderIfEventsFound()}
921                         </Loader>
922
923                     </Row>
924                 </Grid>
925             </MuiThemeProvider>
926         </div>
927     <MuiThemeProvider>
928         <Modal bsSize="large" show={this.state.modal} onHide={this.closeModal}>
929             <Modal.Header>
930                 <Modal.Title> {modalTitleBar}</Modal.Title>
931             </Modal.Header>
932
933             <Modal.Body>
934                 <Grid>
935                     <Stepper activeStep={this.state.stepIndex} orientation={'vertical'}>
936                         <Step>
937                             <StepLabel>Event Information</StepLabel>
938                             <StepContent>
939
940                                 <Row className="show-grid">
941                                     <Col md={3} mdOffset={1} xs={7}>
942                                         <TextField
943                                             id={"eventName"}
944                                             floatingLabelText="Event Name"
945                                             errorText={this.state.eventNameError}
946                                             onChange={(event, newValue) =>
947                                                 this.setState({eventName: newValue})}
948                                             value={this.state.eventName}
949                                             required={true}
950                                             autoFocus={true}
951                                             fullWidth={true}
952                                         />
953                                     </Col>
954                                 </Row>
955                             </StepContent>
956                         </Step>
957                     </Stepper>
958                 </Grid>
959             </Modal.Body>
960         </Modal>
961     </MuiThemeProvider>
962 </div>
963 )

```

```
this.setState({eventDate: newValue})}
```

```
this.setState({startTime: newValue})}
```

```
this.setState({endTime: newValue})}
```

```
{this.state.eventLocation}
```

```
{this.state.eventLocationError}
```

```
building/room"/>
```

```
</Col>
<Col md={3} mdOffset={1} xs={7}>
  <DatePicker
    errorText={this.state.eventDateError}
    onChange={(event, newValue) =>

      value={this.state.eventDate}
      hintText="Event Date"
      floatingLabelText="Event Date"
      textFieldStyle={styles.formFields}

      mode="landscape"/>

  </Col>
</Row>
<br/>
<Row className="show-grid">

</Row>
<br/>
<Row className="show-grid">
  <Col md={3} mdOffset={1} xs={7}>
    <TimePicker
      errorText={this.state.startTimeError}
      floatingLabelText="Start Time"
      onChange={(event, newValue) =>

        value={this.state.startTime}
        textFieldStyle={styles.formFields}
        hintText="Start time"
        autoOk={true}

      />
    </Col>
    <Col md={3} mdOffset={1} xs={7}>
      <TimePicker
        errorText={this.state.endTimeError}
        floatingLabelText="End Time"
        onChange={(event, newValue) =>

          value={this.state.endTime}
          hintText="End time"
          textFieldStyle={styles.formFields}
          autoOk={true}

        />
      </Col>
    </Row>
    <Row className="show-grid">
      <Col md={5} mdOffset={1} xs={7}>
        <BuildingSelector selected=

          errorMsg=

          callBack={this.buildingCallback}
          labelText={"Building - Room"}
          hintText={"Select a

        </Col>
      </Row>
    <Row className={"show-grid"}>
      <Col md={5} mdOffset={1} xs={7}>
```



```

1015         <TextField
1016             id={"eventDescription"}
1017             floatingLabelText="Event Description"
1018             errorText={this.state.eventDescriptionError}
1019             onChange={(event, newValue) =>
1020                 this.setState({eventDescription: newValue})}
1021             value={this.state.eventDescription}
1022             required={true}
1023             fullWidth={true}
1024             multiline={true}
1025             rows={3}
1026         />
1027     </Col>
1028 </Row>
1029 </StepContent>
1030 </Step>
1031 <Step>
1032     <StepLabel>Assign Existing Judges</StepLabel>
1033     <StepContent>
1034         <Row>
1035             <Col md={7} xs={5}>
1036                 <Alert bsStyle="info">
1037                     Select an existing judge or move to next step to
1038                     create new judges
1039                 </Alert>
1040             </Col>
1041         </Row>
1042         <Row>
1043             <Col md={7} xs={5}>
1044                 <SelectField
1045                     multiple={true}
1046                     fullWidth={true}
1047                     autoWidth={true}
1048                     hintText="Existing judges"
1049                     value={this.state.existingJudgeValues}
1050                     onChange={this.judgeMenuClick}
1051                 >
1052                     {this.judgeMenuItems(this.state.existingJudgeValues)}
1053                 </SelectField>
1054             </Col>
1055         </Row>
1056     </StepContent>
1057 </Step>
1058 <Step>
1059     <StepLabel>Create New Judges</StepLabel>
1060     <StepContent>
1061         {this.state.judgeInputs}
1062         <Row>
1063             <Col md={3}>
1064                 <RaisedButton icon={<FontIcon className="pe-7s-
1065                     angle-down-circle"/>}
1066                     primary={true} label="Add new judge"
1067                     onClick={this.addJudgeInputs}/>
1068             </Col>

```

1075
1076

```

        <Col md={3}>
          {removeJudgeBtn}
        </Col>

      </Row>
    </StepContent>
  </Step>
</Stepper>
</Grid>
</Modal.Body>

    <Modal.Footer>
      <Loader loaded={this.state.loadCreateEvent}></Loader> {backButton}
    </Modal.Footer>
  </Modal>
  <Dialog
    actions={deleteActions}
    modal={false}
    open={this.state.confirmDialog}
    onRequestClose={this.closeConfirmDialog}
  >
    {this.state.confirmMessage}
  </Dialog>

  </MuiThemeProvider>
</div>
);
}

//adds in the menu items for existing judges
judgeMenuItems = (existingJudgeValues) => {
  E if (this.state.existingJudgeEmails != null) {
    return this.state.existingJudgeEmails.map((obj) => (
      <MenuItem
        key={obj.id}
        insetChildren={true}
        targetOrigin={{horizontal: "right", vertical: "bottom"}}
        checked={existingJudgeValues && existingJudgeValues.indexOf(obj.id) > -1}
        value={obj.id}
        primaryText={obj.firstName + "    " + obj.lastName + "  --  " + obj.emailAddress}
      />
    ));
  }
}

//allows the check mark to be applied next to the selections
judgeMenuClick = (event, index, values) => {
  this.setState({existingJudgeValues: values});
}
}

export default Events;
```

All files / src/views/Extras Extras.js

86.05% Statements 37/43 57.14% Branches 8/14 93.75% Functions 15/16 86.05% Lines 37/43

```
1 import React, { Component } from 'react';
2 import {
3   Grid, Row, Col,
4 } from 'react-bootstrap';
5
6 import InputMask from 'react-input-mask';
7 import Button from '../elements/CustomButton/CustomButton.js';
8
9 import { TextField } from "material-ui";
10 import HttpRequest from "../adapters/httpRequest";
11 import constants from "../utils/constants";
12 import NotificationSystem from 'react-notification-system';
13 import { style } from "../variables/Variables";
14 import AppBar from 'material-ui/AppBar';
15 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
16 import Sudoku from 'sudoku-react-component';
17 import { Panel, PanelGroup } from 'react-bootstrap';
18
19 class Extras extends React.Component {
20
21   constructor(props) {
22     6x super(props);
23
24     6x this.sendEmail = this.sendEmail.bind(this);
25     6x this.sendText = this.sendText.bind(this);
26     6x this.handleSelect = this.handleSelect.bind(this);
27
28     6x this.state = {
29       user : {},
30       _notificationSystem : null,
31       emailAddress : "",
32       phoneNumber : "",
33       activeKey: '1'
34     };
35   }
36
37   componentWillMount() {
38     6x var _this = this;
39     //This is good shit to remember for later
40     // AuthService.getUserEmail();
41     // AuthService.getUserRole();
42     //This sends our credentials in the header
43
44     6x HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getUserProfile', 'GET',
45       constants.useCredentials(), null).then(function (result) {
46       console.log(result);
47       _this.setState({
48         user: result.body
49       });
50
51     }).catch(function (error) {
52       6x console.log(error);
53     })
54     //here we will get the user type
```

```

55
56     }
57
58     componentDidMount() {
59         6x         this.setState({_notificationSystem: this.refs.notificationSystem});
60     }
61
62     notify(message, level, position, autoDismiss) {
63         6x         this.state._notificationSystem.addNotification({
64             title: (<span data-notify="icon" className="pe-7s-door-lock"></span>),
65             message: (
66                 <div>
67                     {message}
68                 </div>
69             ),
70             level: level ? level : 'error',
71             position: position ? position : 'tc',
72             autoDismiss: autoDismiss ? autoDismiss : 10,
73         });
74     }
75
76     handleSelect(activeKey) {
77         1x         this.setState({ activeKey });
78     }
79
80
81     sendEmail(e) {
82         2x         var body = {};
83         2x         var _this = this;
84
85         2x         body.emailAddress = this.state.emailAddress;
86         2x         _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
87             '/sweng500/sendTestEmail', 'POST',
88             constants.useCredentials(), body).then(function (result) {
89         1x
90         1x         E if (result.status === 200) {
91             _this.notify(
92                 "Test email sent",
93                 "success",
94                 "tc",
95                 5
96         1x             );
97             _this.setState( { emailAddress : "" } );
98         } else if(result.status === 409) {
99             _this.notify(
100                 "Could not send email",
101                 "error",
102                 "tc",
103                 10
104             );
105         }
106
107         1x     }).catch(function (error) {
108         1x         console.log(error);
109         _this.notify(
110             "Could not send email",
111             "error",
112             "tc",
113             10
114         );

```

```

115     })
116 }
117
118 2x sendText(e) {
119 2x     var body = {};
120     var _this = this;
121
122 2x     // Create the user account
123 2x     var cleanPhoneNumber = this.state.phoneNumber;
124 2x     cleanPhoneNumber = cleanPhoneNumber.replace(/\s/g, '');           // Remove spaces
125 2x     cleanPhoneNumber = cleanPhoneNumber.replace(/\(|\)/g, '');       // Remove ( and )
126 2x     cleanPhoneNumber = cleanPhoneNumber.replace(/-/g, "");           // Remove -
127     cleanPhoneNumber = '+' + cleanPhoneNumber;                         // Add +
128 2x
129 2x     body.phoneNumber = cleanPhoneNumber;
130     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/sendTestText',
131 'POST',
132 1x         constants.useCredentials(), body).then(function (result) {
133 1x
134     E if (result.status === 200) {
135         _this.notify(
136             "Test text sent",
137             "success",
138             "tc",
139 1x             5
140         );
141         _this.setState( { phoneNumber : "" } );
142     } else if(result.status === 409) {
143         _this.notify(
144             "Could not send text",
145             "error",
146             "tc",
147             10
148         );
149     }
150 1x
151 1x     }).catch(function (error) {
152         console.log(error);
153         _this.notify(
154             "Could not send text",
155             "error",
156             "tc",
157             10
158         );
159     })
160 }
161 24x
162 render() {
163     return (
164         <div className="content">
165             <NotificationSystem ref="notificationSystem" style={style}/>
166             <MuiThemeProvider>
167                 <Grid fluid>
168                     <PanelGroup id="group"
169                         accordion
170                         activeKey={this.state.activeKey}
171                         onSelect={this.handleSelect}>
172                         <Panel id="email-panel" eventKey="1">
173                             <Panel.Heading style={{backgroundColor: '#194287'}}>
174                                 <Panel.Title toggle style={{color: 'white'}}>

```

```

175         Test Email Functionality
176     </Panel.Title>
177 </Panel.Heading>
178 <Panel.Collapse>
179     <Panel.Body collapsible>
180         <TextField
181             name="emailAddress"
182             hintText="Email Address"
183             floatingLabelText="Email Address"
184             onChange={(event, newValue) => this.setState({emailAddress:
185                 newValue}})
186             value={this.state.emailAddress}
187         />
188         {
189             <Button
190                 bsStyle="info"
191                 fill
192                 type="submit"
193                 onClick={this.sendEmail}
194                 style={{backgroundColor: '#194287', borderColor : '#194287'}}
195             >
196                 Send Email
197             </Button>
198         }
199     </Panel.Body>
200 </Panel.Collapse>
201 </Panel>
202 <Panel id="text-panel" eventKey="2">
203     <Panel.Heading style={{backgroundColor: '#194287'}}>
204         <Panel.Title toggle style={{color: 'white'}}>
205             Test SMS Texting Functionality
206         </Panel.Title>
207     </Panel.Heading>
208     <Panel.Collapse>
209         <Panel.Body collapsible>
210             <TextField
211                 name="phone"
212                 floatingLabelText="Phone number"
213                 onChange={(event, newValue) => this.setState({phoneNumber:
214                     newValue}})
215                 value={this.state.phoneNumber}
216             >
217                 <InputMask mask="9 (999) 999-9999" maskChar="#"
218                     value={this.state.phoneNumber}/>
219             </TextField>
220             {
221                 <Button
222                     bsStyle="info"
223                     fill
224                     type="submit"
225                     onClick={this.sendText}
226                     style={{backgroundColor: '#194287', borderColor : '#194287'}}
227                 >
228                     Send Text
229                 </Button>
230             }
231         </Panel.Body>
232     </Panel.Collapse>
233 </Panel>
234 <Panel id="egg-panel" eventKey="3">
235     <Panel.Heading style={{backgroundColor: '#194287'}}>
236         <Panel.Title toggle style={{color: 'white'}}>

```

```
235         Easter Egg
236       </Panel.Title>
237     </Panel.Heading>
238     <Panel.Collapse>
239       <Panel.Body collapsible>
240         <div id='sudoku-container' style={{width : '70%',
241 paddingLeft:'10%', paddingRight:'0%'}}>
242           <Sudoku />
243         </div>
244       </Panel.Body>
245     </Panel.Collapse>
246   </Panel>
247 </PanelGroup>
248 </Grid>
249 </MuiThemeProvider>
250 </div>
251   );
252 }
253 }
```

```
export default Extras;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Locations Buildings.js

27.59% Statements 24/87 13.89% Branches 5/36 25.93% Functions 7/27 27.59% Lines 24/87

```
1 import React, {Component} from 'react';
2 import {Grid, Col, Row, Modal} from 'react-bootstrap';
3 import Loader from 'react-loader';
4 import HttpRequest from "../../adapters/httpRequest";
5 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
6 import RaisedButton from 'material-ui/RaisedButton';
7 import TextField from 'material-ui/TextField';
8 import AppBar from 'material-ui/AppBar';
9 import FlatButton from 'material-ui/FlatButton';
10 import FontIcon from 'material-ui/FontIcon';
11 import Card from "../../components/Cards/Card.js";
12 import ReactTable from 'react-table';
13 import matchSorter from 'match-sorter';
14 import Dialog from 'material-ui/Dialog';
15 import constants from "../../utils/constants";
16 import NotificationSystem from 'react-notification-system';
17 import {style} from "../../variables/Variables";
18 import {Map, Marker, GoogleApiWrapper} from 'google-maps-react';
19
20
21 class Buildings extends Component {
22   constructor(props) {
23     2x    super(props);
24
25     2x    this.closeModal = this.closeModal.bind(this);
26     2x    this.openModal = this.openModal.bind(this);
27     2x    this.addNotification = this.addNotification.bind(this);
28
29     2x    this.divStyle = {
30        height: '300px',
31        width: '450px',
32        position: 'relative',
33        overflow: 'scroll'
34    };
35
36     2x    this.state = {
37        loading: false,
38        modal: false,
39        modalTitle: '',
40        modalAction: '',
41        modalButton: '',
42        editModal: false,
43        confirmDialog: false,
44        confirmMessage: '',
45        deleteID: '',
46        buildingID: '',
47        building: '',
48        longitude: null,
49        latitude: null,
50        buildingRequired: '',
51        buildingList: [],
52        _notificationSystem: null
53    };
54 }
```



```

55
56 // When the map is clicked record the latitude and longitude
57 onMapClicked = (mapProps, map, event) => {
58
59     this.setState({
60         latitude: event.latLng.lat(),
61         longitude: event.latLng.lng()
62     });
63 }
64
65 // If there is a latitude and longitude then display it
66 addMarker = () => {
67     5x if (this.state.latitude !== undefined)
68     5x return(<Marker name='Current location' position={{lat: this.state.latitude, lng:
69 this.state.longitude}}/>);
70 }
71
72 // Display notification
73 addNotification(message, level, position, autoDismiss, optionalTitle){
74     this.state._notificationSystem.addNotification({
75         title: optionalTitle ? optionalTitle : (<span data-notify="icon" className="pe-7s-map">
76 </span>),
77         message: (
78             <div>
79                 {message}
80             </div>
81         ),
82         level: level ? level : 'info',
83         position: position ? position : 'tc',
84         autoDismiss: autoDismiss ? autoDismiss : 10,
85     });
86 }
87
88 // Open the modal
89 openModal = (mode) => {
90
91     // Change from add to edit mode
92     if (mode.status === "add")
93     {
94         this.setState({
95             modal: true,
96             modalAction: 'add',
97             modalTitle: "Add Building",
98             modalButton: "Add Building",
99             latitude: null,
100             longitude: null,
101             building: '',
102             buildingRequired: '',
103             markerRequired: '',
104         })
105     }
106     else if (mode.status === "edit")
107     {
108         this.setState({
109             modal: true,
110             modalAction: 'edit',
111             modalTitle: "Edit Building",
112             modalButton: "Update Building",
113             latitude: mode.lat,
114             longitude: mode.lng,

```

```

115         building: mode.building,
116         buildingID: mode.id,
117         buildingRequired: '',
118         markerRequired: '',
119     ))
120
121     this.addMarker();
122 }
123 }
124
125 // Close the modal
126 closeModal() {
127     this.setState({
128         modal: false
129     })
130 }
131
132 // Confirm Delete building
133 confirmBuildingDelete = (s) => {
134     this.setState({deleteID: s.id});
135     this.setState({confirmMessage: "Are you sure you want to delete " + s.building + "?"},
136     confirmDialog: true));
137 }
138
139 // Close the confirm dialog
140 closeConfirmDialog = () => {
141     this.setState({confirmDialog: false});
142 }
143
144 // Remove the building
145 deleteBuilding = () => {
146
147     var id = this.state.deleteID;
148     var _this = this;
149
150     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
151     '/sweng500/removeBuilding/' + id, 'DELETE', constants.useCredentials(), null, true).then(function (result)
152     {
153
154         _this.setState({confirmDialog: false});
155
156         if (result.status === 200) {
157
158             _this.addNotification(
159                 "Success: The building has been deleted.",
160                 "info",
161                 "tc",
162                 6
163             );
164
165             _this.componentDidMount();
166         }
167
168     }).catch(function (error) {
169         console.log(error);
170
171         _this.addNotification(
172             "Error: The building has not been deleted.",
173             "error",
174             "tc",

```

```

        6
    );
}

    _this.setState({confirmDialog: false});
})
}

// Check inputs and adds a new building
handleSubmit = () => {

    var blankInputs = false;
    var building = this.state.building.trim();
    var lat = this.state.latitude;
    var lng = this.state.longitude;

    // Checks first name
    if (building) {
        this.setState({buildingRequired: undefined})
    }
    else {
        this.setState({buildingRequired: "A building name is required"})
        blankInputs = true;
    }

    // Checks first name
    if (lat !== null && lng !== null) {
        this.setState({markerRequired: undefined})
    }
    else {
        this.setState({markerRequired: "You must identify the building by placing a marker on the
map."})
        blankInputs = true;
    }

    // If no information is missing we can add the building
    if (!blankInputs)
    {

        var _this = this;

        var body = {};

        body.building = building;
        body.lat = lat;
        body.lng = lng;

        if (this.state.modalAction === "add") {

            _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
'/sweng500/addBuilding', 'POST', constants.useCredentials(), body, true).then(function (result) {

                if (result.status === 200) {

                    // Post a notification
                    _this.addNotification(
                        "Success: The building has been added.",
                        "info",
                        "tc",
                        6
                    );
                }
            });
        }
    }
}

```

```

235         );
236
237         // Close the modal and show a success message
238         _this.setState({modal: false});
239
240         // Update the component
241         _this.componentDidMount();
242     }
243
244     }).catch(function (error) {
245         console.log(error);
246
247         _this.addNotification(
248             "Error: The building has not been added.",
249             "error",
250             "tc",
251             6
252         );
253
254         // Close the modal and show a failed message
255         _this.setState({modal: false});
256
257     })
258 }
259 else if (this.state.modalAction === "edit")
260 {
261
262     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
263 '/sweng500/updateBuilding/' + this.state.buildingID, 'POST', constants.useCredentials(), body,
264 true).then(function (result) {
265
266         if (result.status === 200) {
267
268             // Close the modal and show a success message
269             _this.setState({
270                 modal: false,
271             })
272
273             _this.addNotification(
274                 "Success: The building has been updated.",
275                 "info",
276                 "tc",
277                 6
278             );
279
280             _this.componentDidMount();
281
282         }
283
284     }).catch(function (error) {
285         console.log(error);
286
287         _this.addNotification(
288             "Error: The building has not been updated.",
289             "error",
290             "tc",
291             6
292         );
293
294         // Close the modal and show a failed message

```

```

295 2x      _this.setState({modal: false})
296
297      })
298 2x    }
299  }
300 2x  }
301
302 1x  // Fetch a list of buildings
303  componentDidMount() {
304      this.setState({_notificationSystem: this.refs.notificationSystem});
305
306      //Make call out to backend
307      var _this = this;
308 1x
309      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getBuildings',
310 'GET', constants.useCredentials(), null, true).then(function (result) {
311
312      _this.setState({
313          buildingList: result.body,
314          loading: true
315 5x      })
316
317 1x      }).catch(function (error) {
318          console.log(error);
319      })
320  }
321
322
323  // Generate the table if the fetch was successful
324  renderIfBuildingsFound() {
325      if (this.state.buildingList !== null && Object.keys(this.state.buildingList).length !== 0) {
326
327          const columns = [{
328              Header: 'Building Name',
329              filterMethod: (filter, rows) =>
330                  matchSorter(rows, filter.value, { keys: ["building"] })),
331              filterAll: true,
332              accessor: 'building' // String-based value accessors!
333          }, {
334              Header: 'Latitude',
335              filterMethod: (filter, rows) =>
336                  matchSorter(rows, filter.value, { keys: ["latitude"] })),
337              filterAll: true,
338              accessor: 'lat' // String-based value accessors!
339          }, {
340              Header: 'Longitude',
341              filterMethod: (filter, rows) =>
342                  matchSorter(rows, filter.value, { keys: ["longitude"] })),
343 1x          filterAll: true,
344 2x          accessor: 'lng' // String-based value accessors!
345 2x      }, {
346 2x          Header: 'Actions',
347          accessor: 'menuActions', // String-based value accessors!
348          style:{textAlign:'center'},
349          sortable: false,
350          filterable: false
351 1x      }]];
352
353      for(let value in this.state.buildingList) {
354          this.state.buildingList[value].status = "edit";

```

```

        this.state.buildingList[value].position = value;
        this.state.buildingList[value].menuActions = <div><RaisedButton
            primary={true} onClick={this.openModal.bind(this, this.state.buildingList[value])}
label="Edit"/>&nbsp;&nbsp;&&nbsp;<RaisedButton
            secondary={true} onClick=
{this.confirmBuildingDelete.bind(this, this.state.buildingList[value])} label="Delete"/></div>;
    }

    return(
        <ReactTable
            data={this.state.buildingList}
            filterable
            defaultFilterMethod={({filter, row} =>
                String(row[filter.id]) === filter.value)
            columns={columns}
            defaultPageSize={10}
            className="-striped -highlight"
            defaultSorted={[{id: "building"}]}
        />
    )

    }
    else
        return("Error loading building list.")
    }

    render() {

        const actions = [
            <FlatButton
                label="Cancel"
                primary={true}
                onClick={this.closeConfirmDialog}/>,
            <FlatButton
                label="Delete"
                primary={true}
                onClick={this.deleteBuilding}/>,
        ];
        var status = {"status" : "add"};
        return (
            <MuiThemeProvider>
                <div className="content">
                    <NotificationSystem ref="notificationSystem" style={style}/>
                    <Grid fluid>
                        <Row>
                            <Col md={12}>
                                <Card
                                    style={{margin:10}}
                                    title="Registered Buildings"
                                    category={
                                        <div>These locations are registered with the Science Olympiad
system.<br/>They will appear in the registration systems and reports.
                                        <br/><br/>
                                        <RaisedButton primary={true} label="Add a new building"
onClick={this.openModal.bind(this, status)}/>
                                    </div>
                                <ctTableFullWidth ctTableResponsive
                                    content={
                                        <Loader color="#3498db" loaded={this.state.loading}>
                                            {this.renderIfBuildingsFound()}
                                        </Loader>
                                    </ctTableFullWidth>
                                </Col>
                            </Row>
                        </Grid>
                </div>
            </MuiThemeProvider>
        )
    }
}

```

```

        </Loader>
      }>
    </Col>
  </Row>
</Grid>
<Modal show={this.state.modal} onHide={this.closeModal}>
  <Modal.Header>
    <Modal.Title>
      <AppBar>
        iconElementRight=<FlatButton label="Close"/>
        showMenuIconButton={false}
        onRightIconButtonClick={(event) => this.closeModal()}
        title={this.state.modalTitle}/>
      </Modal.Title>
    </Modal.Header>
    <Modal.Body>
      <div style={{overflow: 'auto'}}>
        <TextField>
          id={"buildingName"}
          floatingLabelText="Building Name"
          onChange={(event, newValue) => this.setState({building: newValue})}
          value={this.state.building}
          required={true}
          autoFocus={true}
          fullWidth={true}
          hintText="Enter a name for the building"
          errorText={this.state.buildingRequired}/>

          <div>Click on the map to place a marker on the build's location.</div>
          <br/>
          {this.state.markerRequired}

          <div id="map" style={{height:300, width:450, marginLeft: 'auto',
marginRight: 'auto'}}>
            <Map>
              style={this.divStyle}
              google={this.props.google}
              zoomControl={true}
              initialCenter={{
                lat: 41.306610,
                lng: -76.015437
              }}
              zoom={16}
              onClick={this.onMapClicked}
              clickableIcons={false}>
                {this.addMarker()}
              </Map>
            </div>
          </div>
        </Modal.Body>
        <Modal.Footer>
          <RaisedButton icon=<FontIcon className="pe-7s-close-circle" /> primary={true}
label="Cancel"
onClick={this.closeModal}/>&nbsp;&nbsp;&nbsp;&nbsp;&
          <RaisedButton icon=<FontIcon className="pe-7s-like2" /> primary={true} label=
{this.state.modalButton}
onClick={this.handleSubmit}/>
        </Modal.Footer>
      </Modal>
    </Dialog>

```

```

        actions={actions}
        modal={false}
        open={this.state.confirmDialog}
        onRequestClose={this.closeConfirmDialog}>
        {this.state.confirmMessage}
      </Dialog>
    </div>
  </MuiThemeProvider>
);
}
}

export default GoogleApiWrapper({
  apiKey: "AIzaSyC7xiiV97LyRQd-GB9aBmiJaYFGW5DVIbM"
})(Buildings)

```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Locations Rooms.js

82.8% Statements 77/93 61.11% Branches 22/36 56.67% Functions 17/30 82.8% Lines 77/93

```
1 import React, {Component} from 'react';
2 import {Grid, Col, Row, Modal} from 'react-bootstrap';
3 import Loader from 'react-loader';
4 import HttpRequest from "../../adapters/httpRequest";
5 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
6 import RaisedButton from 'material-ui/RaisedButton';
7 import TextField from 'material-ui/TextField';
8 import AppBar from 'material-ui/AppBar';
9 import FlatButton from 'material-ui/FlatButton';
10 import FontIcon from 'material-ui/FontIcon';
11 import Card from '../../components/Cards/Card.js';
12 import ReactTable from 'react-table';
13 import matchSorter from 'match-sorter';
14 import Dialog from 'material-ui/Dialog';
15 import constants from "../../utils/constants";
16 import NotificationSystem from 'react-notification-system';
17 import {style} from "../../variables/Variables";
18 import BuildingSelector from '../../components/Buildings/BuildingSelector';
19
20
21 class Rooms extends Component {
22   constructor(props) {
23     8x super(props);
24     8x this.closeModal = this.closeModal.bind(this);
25     8x this.openModal = this.openModal.bind(this);
26     8x this.addNotification = this.addNotification.bind(this);
27     8x this.buildingCallback = this.buildingCallback.bind(this);
28
29     8x this.state = {
30       loading: false,
31       modal: false,
32       modalTitle: '',
33       modalAction: '',
34       modalButton: '',
35       editModal: false,
36       confirmDialog: false,
37       confirmMessage: '',
38       roomName: '',
39       building: null,
40       buildingRequired: null,
41       roomCapacity: null,
42       buildingError: null,
43       deleteID: '',
44       roomID: '',
45       roomCapacityRequired: '',
46       roomList: [],
47       buildingList: [],
48       _notificationSystem: null
49     };
50   }
51
52   addNotification(message, level, position, autoDismiss, optionalTitle){
53     this.state._notificationSystem.addNotification({
54       title: optionalTitle ? optionalTitle : (<span data-notify="icon" className="pe-7s-ribbon">
```

```

55     </span>)),
56         message: (
57             <div>
58                 {message}
59             </div>
60         ),
61         level: level ? level : 'info',
62         position: position ? position : 'tc',
63         autoDismiss: autoDismiss ? autoDismiss : 10,
64     });
65 }
66
67 buildingCallback(event, index, value) {
68
69     this.setState({
70         building: value});
71 }
72
73 // Open the modal
74 openModal = (mode) => {
75     2x
76     if (mode.status === "add")
77     1x
78     {
79         this.setState({
80             modal: true,
81             modalAction: 'add',
82             modalTitle: "Add New Room",
83             modalButton: "Add Room",
84             roomCapacity: '',
85             roomName: '',
86             roomNameRequired: '',
87             roomCapacityRequired: '',
88             building: '',
89             buildingRequired: ''
90         })
91     }
92     1x
93     else if (mode.status === "edit")
94     1x
95     {
96         this.setState({
97             modal: true,
98             modalAction: 'edit',
99             modalTitle: "Edit Room",
100             modalButton: "Update Room",
101             roomCapacity: mode.capacity,
102             roomName: mode.roomName,
103             building: mode.buildingID,
104             roomID: mode.id,
105             roomNameRequired: '',
106             roomCapacityRequired: ''
107         })
108     }
109 }
110
111 // Close the modal
112 1x closeModal() {
113     this.setState({
114         modal: false
115     })
116 }

```

```

115 // Delete room
116 1x confirmRoomDelete = (s) => {
117 1x   this.setState({deleteID: s.id});
118   this.setState({confirmMessage: "Are you sure you want to delete " + s.roomName + "?"},
119   confirmDialog: true});
120 }
121
122   getBuildingName(id)
123 84x   {
124
125 136x     for(let value in this.state.buildingList) {
126
127       I if (this.state.buildingList[value].id === id)
128         return this.state.buildingList[value].building;
129 84x     }
130
131     return "Invalid Building ID";
132   }
133
134   closeConfirmDialog = () => {
135     this.setState({confirmDialog: false});
136   }
137
138 1x   deleteRoom = () => {
139 1x
140     var id = this.state.deleteID;
141 1x     var _this = this;
142
143 1x     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/removeRoom/'
144 + id, 'DELETE', constants.useCredentials(), null).then(function (result) {
145 1x
146         _this.setState({confirmDialog: false});
147 1x
148         E if (result.status === 200) {
149
150             _this.addNotification(
151                 "Success: The room has been deleted.",
152                 "info",
153                 "tc",
154 1x                 6
155             );
156
157             _this.componentDidMount();
158         }
159
160     }).catch(function (error) {
161
162         _this.addNotification(
163             "Error: The room has not been deleted.",
164             "error",
165             "tc",
166             6
167         );
168
169
170         _this.setState({confirmDialog: false});
171     })
172 }
173
174 3x // Check inputs and add a new room

```

```

handleSubmit = () => {

    var blankInputs = false;
    var roomName = this.state.roomName.trim();
    var roomCapacity = this.state.roomCapacity;
    var buildingID = this.state.building;

    // Checks first name
    if (roomName) {
        this.setState({roomNameRequired: undefined})
    }
    else {
        this.setState({roomNameRequired: "A room name or number is required"})
        blankInputs = true;
    }

    // Checks contact name
    if (roomCapacity) {
        this.setState({roomCapacityRequired: undefined})
    }
    else {
        this.setState({roomCapacityRequired: "A room capacity is required"})
        blankInputs = true;
    }

    // Checks for a building
    if (buildingID) {
        this.setState({buildingRequired: undefined})
    }
    else {
        this.setState({buildingRequired: "A building is required"})
        blankInputs = true;
    }

    // If no information is missing we can add the room
    if (!blankInputs)
    {

        var _this = this;

        var body = {};

        body.roomName = this.state.roomName.trim();
        body.capacity = this.state.roomCapacity;
        body.buildingID = this.state.building;

        if (this.state.modalAction === "add") {

            _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
'/sweng500/addRoom', 'POST', constants.useCredentials(), body).then(function (result) {

                if (result.status === 200) {

                    // Post a notification
                    _this.addNotification(
                        "Success: The room has been added.",
                        "info",
                        "tc",
                        6
                    );
                }
            });
        }
    }
}

```

```

235
236 1x // Close the modal and show a success message
237 _this.setState({modal: false})
238
239 // Update the component
240 _this.componentDidMount();
241
242 }
243
244 }).catch(function (error) {
245
246     _this.addNotification(
247         "Error: The room has not been added.",
248         "error",
249         "tc",
250         6
251     );
252
253     // Close the modal and show a failed message
254 1x _this.setState({modal: false})
255
256 1x })
257 }
258 1x else E if (this.state.modalAction === "edit")
259 {
260     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
261 1x '/sweng500/updateRoom/' + this.state.roomID, 'POST', constants.useCredentials(), body).then(function
262 (result) {
263
264     E if (result.status === 200) {
265 1x
266         // Close the modal and show a success message
267         _this.setState({
268             modal: false,
269         })
270
271         _this.addNotification(
272 1x         "Success: The room has been updated.",
273         "info",
274         "tc",
275         6
276     );
277
278     _this.componentDidMount();
279
280     }
281
282     }).catch(function (error) {
283
284     _this.addNotification(
285         "Error: The room has not been updated.",
286         "error",
287         "tc",
288         6
289     );
290
291     // Close the modal and show a failed message
292 1x _this.setState({modal: false})
293
294     })

```

```

295     11x         }
296     }
297 }
298 11x
299 // Fetch a list of rooms
300 11x componentDidMount() {
301 11x     this.setState({_notificationSystem: this.refs.notificationSystem});
302
303     //Make call out to backend to get a list of rooms
304     var _this = this;
305
306     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getAllRooms',
307 'GET', constants.useCredentials(), null).then(function (result) {
308         _this.setState({
309             roomList: result.body,
310 11x             loading: true
311         })
312 11x
313     }).catch(function (error) {
314     })
315
316     //Make call out to backend to get a list of buildings
317     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
318 '/sweng500/getBuildings', 'GET', constants.useCredentials(), null, true).then(function (result) {
319
320         _this.setState({
321             buildingList: result.body,
322             loading: true
323         })
324
325     }).catch(function (error) {
326 80x         console.log(error);
327     })
328 42x
329
330 }
331
332 // Generate the table if the fetch was successful
333 renderIfRoomsFound() {
334     if (this.state.roomList !== null && Object.keys(this.state.roomList).length !== 0) {
335
336         const columns = [{
337             Header: 'Room Name or Number',
338             filterMethod: (filter, rows) =>
339                 matchSorter(rows, filter.value, { keys: ["roomName"] })),
340             filterAll: true,
341             accessor: 'roomName' // String-based value accessors!
342         }, {
343             Header: 'Building Name',
344             filterMethod: (filter, rows) =>
345                 matchSorter(rows, filter.value, { keys: ["buildingName"] })),
346             filterAll: true,
347             accessor: 'buildingName' // String-based value accessors!
348         }, {
349             Header: 'Room Capacity',
350             filterMethod: (filter, rows) =>
351                 matchSorter(rows, filter.value, { keys: ["capacity"] })),
352             filterAll: true,
353             accessor: 'capacity' // String-based value accessors!
354 42x         }, {

```

```

355         Header: 'Actions',
356         42x         accessor: 'menuActions', // String-based value accessors!
357         style:{textAlign:'center'},
358         84x         sortable: false,
359         84x         filterable: false
360         84x     ]];
361
362     var templist = this.state.roomList;
363
364     for(let value in templist) {
365         42x
366             templist[value].status = "edit";
367             templist[value].buildingName =
368 this.getBuildingName(this.state.roomList[value].buildingID);
369             templist[value].menuActions = <div><RaisedButton
370                 primary={true} onClick={this.openModal.bind(this,this.state.roomList[value])}
371 label="Edit"/>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<RaisedButton
372                 secondary={true} onClick=
373 {this.confirmRoomDelete.bind(this,this.state.roomList[value])} label="Delete"/></div>;
374             }
375
376     return(
377         <ReactTable
378             data={templist}
379             filterable
380             38x             defaultFilterMethod={({filter, row}) =>
381                 String(row[filter.id]) === filter.value}
382             columns={columns}
383             defaultPageSize={10}
384             className="-striped -highlight"
385             80x             defaultSorted={[{id: "roomName"}]}
386             />
387     )
388
389     }
390     else
391         return("ERROR LOADING ROOMS")
392 }
393
394 render() {
395
396     const actions = [
397         80x         <FlatButton
398         80x             label="Cancel"
399             primary={true}
400             onClick={this.closeConfirmDialog}
401         />,
402         <FlatButton
403             label="Delete"
404             primary={true}
405             onClick={this.deleteRoom}
406         />,
407     ];
408     var status = {"status" : "add"};
409     return (
410         <MuiThemeProvider>
411             <div className="content">
412                 <NotificationSystem ref="notificationSystem" style={style}/>
413                 <Grid fluid>
414                     <Row>

```

```

415         <Col md={12}>
416             <Card
417                 style={{margin:10}}
418                 title="Registered Rooms"
419                 category={
420                     <div>These rooms are registered with the Science Olympiad
421 system.<br/>They are assigned to an existing building.
422                     <br/><br/>
423                     <RaisedButton primary={true} label="Create a new room"
424 onClick={this.openModal.bind(this,status)}/>
425                 </div>
426                 ctTableFullWidth ctTableResponsive
427                 content={
428                     <Loader color="#3498db" loaded={this.state.loading}>
429                     {this.renderIfRoomsFound()}
430                     </Loader>
431                 }/>
432             </Col>
433         </Row>
434     </Grid>
435     <Modal show={this.state.modal} onHide={this.closeModal}>
436         <Modal.Header>
437             <Modal.Title> <AppBar
438                 iconElementRight=<FlatButton label="Close"/>
439                 showMenuIconButton={false}
440                 onRightIconButtonClick={(event) => this.closeModal()}
441                 title={this.state.modalTitle}
442             /></Modal.Title>
443         </Modal.Header>
444         <Modal.Body>
445             <TextField
446                 id={"roomName"}
447                 floatingLabelText="Room Name or Number"
448                 onChange={(event, newValue) => this.setState({roomName: newValue})}
449                 value={this.state.roomName}
450                 required={true}
451                 autoFocus={true}
452                 fullWidth={true}
453                 hintText="Enter a room name or number"
454                 errorText={this.state.roomNameRequired}
455             />
456             <br/>
457             <TextField
458                 id={"roomCapacity"}
459                 floatingLabelText="Room Capacity"
460                 onChange={(event, newValue) => this.setState({roomCapacity: newValue})}
461                 value={this.state.roomCapacity}
462                 required={true}
463                 fullWidth={true}
464                 hintText="Enter the room's capacity"
465                 errorText={this.state.roomCapacityRequired}
466             />
467             <br/>
468             <BuildingSelector selected={this.state.building}
469                 errorMsg={this.state.buildingRequired}
470                 callBack={this.buildingCallback}
471                 labelText={"Building"}
472                 hintText={"Select a building"}/>
473         </Modal.Body>
474         <Modal.Footer>

```



```

475         <RaisedButton icon={<FontIcon className="pe-7s-close-circle" />} primary=
476             {true} label="Cancel"
477                 onClick={this.closeModal}/>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~
478                     <RaisedButton icon={<FontIcon className="pe-7s-like2" />} primary={true}
479                         label={this.state.modalButton}
480                             onClick={this.handleSubmit}/>
481                 </Modal.Footer>
482             </Modal>
483             <Dialog
                actions={actions}
                modal={false}
                open={this.state.confirmDialog}
                onRequestClose={this.closeConfirmDialog}
            >
                {this.state.confirmMessage}
            </Dialog>
        </div>
    </MuiThemeProvider>
  );
}
}
```

```
export default Rooms;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Schools Schools.js

85.87% Statements 79/92 61.76% Branches 21/34 60.71% Functions 17/28 85.87% Lines 79/92

```
1 import React, {Component} from 'react';
2 import {Grid, Col, Row, Modal} from 'react-bootstrap';
3 import Loader from 'react-loader';
4 import HttpRequest from "../../adapters/httpRequest";
5 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
6 import RaisedButton from 'material-ui/RaisedButton';
7 import TextField from 'material-ui/TextField';
8 import AppBar from 'material-ui/AppBar';
9 import FlatButton from 'material-ui/FlatButton';
10 import FontIcon from 'material-ui/FontIcon';
11 import Card from '../../components/Cards/Card.js';
12 import InputMask from 'react-input-mask';
13 import ReactTable from 'react-table';
14 import matchSorter from 'match-sorter';
15 import Dialog from 'material-ui/Dialog';
16 import constants from "../../utils/constants";
17 import NotificationSystem from 'react-notification-system';
18 import {style} from "../../variables/Variables";
19
20
21 class Schools extends Component {
22   constructor(props) {
23     12x super(props);
24     12x this.closeModal = this.closeModal.bind(this);
25     12x this.openModal = this.openModal.bind(this);
26     12x this.addNotification = this.addNotification.bind(this);
27
28     12x this.state = {
29       loading: false,
30       modal: false,
31       modalTitle: '',
32       modalAction: '',
33       modalButton: '',
34       editModal: false,
35       confirmDialog: false,
36       confirmMessage: '',
37       formattedPhone: '',
38       deleteID: '',
39       schoolID: '',
40       schoolName: '',
41       schoolContactPhone: '',
42       schoolContactName: '',
43       schoolNameRequired: '',
44       schoolContactPhoneRequired: '',
45       schoolContactNameRequired: '',
46       schoolList: [],
47       _notificationSystem: null
48     };
49   }
50
51   addNotification(message, level, position, autoDismiss, optionalTitle){
52     12x this.state._notificationSystem.addNotification({
53       title: optionalTitle ? optionalTitle : (<span data-notify="icon" className="pe-7s-home">
```

```

55     </span>),
56         message: (
57             <div>
58                 {message}
59             </div>
60         ),
61         level: level ? level : 'info',
62         position: position ? position : 'tc',
63         autoDismiss: autoDismiss ? autoDismiss : 10,
64     });
65 }
66
67 // Open the modal
68 openModal = (mode) => {
69     2x
70     if (mode.status === "add")
71     1x
72     {
73         this.setState({
74             modal: true,
75             modalAction: 'add',
76             modalTitle: "Add New School",
77             modalButton: "Add School",
78             schoolContactName: '',
79             schoolContactPhone: '',
80             schoolName: '',
81             schoolNameRequired: '',
82             schoolContactPhoneRequired: '',
83             schoolContactNameRequired: ''
84         });
85     }
86     1x
87     else if (mode.status === "edit")
88     {
89         this.setState({
90             modal: true,
91             modalAction: 'edit',
92             modalTitle: "Edit School",
93             modalButton: "Update School",
94             schoolContactName: mode.schoolContactName,
95             schoolContactPhone: mode.schoolContactPhone,
96             schoolName: mode.schoolName,
97             schoolID: mode.id,
98             schoolNameRequired: '',
99             schoolContactPhoneRequired: '',
100             schoolContactNameRequired: ''
101         });
102     }
103
104     // Close the modal
105     1x
106     closeModal() {
107         this.setState({
108             modal: false
109         });
110     }
111
112     // Delete school
113     1x
114     confirmSchoolDelete = (s) => {
115         1x
116         this.setState({deleteID: s.id});
117         this.setState({confirmMessage: "Are you sure you want to delete" + s.schoolName + "?"},
118             confirmDialog: true));

```

```

115     }
116
117     closeConfirmDialog = () => {
118         this.setState({confirmDialog: false});
119     }
120
121     deleteSchool = () => {
122         var id = this.state.deleteID;
123         var _this = this;
124
125         _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/removeSchool/'
126 + id, 'DELETE', constants.useCredentials(), null).then(function (result) {
127
128         _this.setState({confirmDialog: false});
129
130         if (result.status === 200) {
131
132             _this.addNotification(
133                 "Success: The school has been deleted.",
134                 "info",
135                 "tc",
136                 6
137             );
138
139             _this.componentDidMount();
140         }
141     }).catch(function (error) {
142
143         _this.addNotification(
144             "Error: The school has not been deleted.",
145             "error",
146             "tc",
147             6
148         );
149
150         _this.setState({confirmDialog: false});
151     })
152 }
153
154 // Check inputs and add a new school
155 handleSubmit = () => {
156     var blankInputs = false;
157     var schoolName = this.state.schoolName.trim();
158     var schoolContactName = this.state.schoolContactName.trim();
159     var schoolContactPhone = this.state.schoolContactPhone.trim();
160
161     // Checks first name
162     if (schoolName) {
163         this.setState({schoolNameRequired: undefined})
164     }
165     else {
166         this.setState({schoolNameRequired: "School name is required"})
167         blankInputs = true;
168     }
169
170     // Checks contact name

```

```

175     if (schoolContactName) {
176         1x         this.setState({schoolContactNameRequired: undefined})
177     1x     }
178     else {
179         this.setState({schoolContactNameRequired: "School contact name is required"})
180         blankInputs = true;
181     3x     }
182     2x
183     // Checks first name
184     if (schoolContactPhone) {
185         1x         this.setState({schoolContactPhoneRequired: undefined})
186     1x     }
187     else {
188         this.setState({schoolContactPhoneRequired: "School phone number is required"})
189         blankInputs = true;
190     3x     }
191
192     // If no information is missing we can add the school
193     2x     if (!blankInputs)
194     {
195         2x
196         var _this = this;
197         2x
198         var body = {};
199         2x
200         2x         var cleanPhoneNumber = this.state.schoolContactPhone.trim();
201         cleanPhoneNumber = cleanPhoneNumber.replace(/\s/g, '');           // Remove spaces
202         2x         cleanPhoneNumber = cleanPhoneNumber.replace(/\(|\)/g, ''); // Remove ( and )
203         2x         cleanPhoneNumber = cleanPhoneNumber.replace(/-/g, "");    // Remove -
204         2x
205         body.schoolName = this.state.schoolName.trim();
206         2x         body.schoolContactName = this.state.schoolContactName.trim();
207         body.schoolContactPhone = cleanPhoneNumber;
208         1x
209         if (this.state.modalAction === "add") {
210             1x
211                 _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
212                 '/sweng500/addSchool', 'POST', constants.useCredentials(), body).then(function (result) {
213                 1x
214                     E if (result.status === 200) {
215
216                         // Post a notification
217                         _this.addNotification(
218                             "Success: The school has been added.",
219                             "info",
220                             "tc",
221                 1x                             6
222                         );
223
224                 1x                         // Close the modal and show a success message
225                         _this.setState({modal: false})
226
227                         // Update the component
228                         _this.componentDidMount();
229
230                     }
231
232                 }).catch(function (error) {
233
234                     _this.addNotification(

```

```

235         "Error: The school has not been added.",
236         "error",
237         "tc",
238         6
239     );
240
241     // Close the modal and show a failed message
242     1x _this.setState({modal: false})
243
244     1x })
245 }
246 1x else E if (this.state.modalAction === "edit")
247 {
248     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
249 1x '/sweng500/updateSchool/' + this.state.schoolID, 'POST', constants.useCredentials(), body).then(function
250 (result) {
251
252     E if (result.status === 200) {
253 1x
254         // Close the modal and show a success message
255         _this.setState({
256             modal: false,
257         })
258
259         _this.addNotification(
260 1x             "Success: The school has been updated.",
261             "info",
262             "tc",
263             6
264         );
265
266         _this.componentDidMount();
267
268     }
269
270     }).catch(function (error) {
271
272     1x _this.addNotification(
273         "Error: The school has not been updated.",
274         "error",
275         "tc",
276         6
277     );
278
279     // Close the modal and show a failed message
280     1x _this.setState({modal: false})
281
282     1x })
283 14x }
284 }
285 }
286 14x
287 // Fetch a list of schools
288 14x componentDidMount() {
289 14x     this.setState({notificationSystem: this.refs.notificationSystem});
290
291     //Make call out to backend
292     var _this = this;
293
294     _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getSchools',

```

```

295 'GET', constants.useCredentials(), null).then(function (result) {
296     _this.setState({
297         schoolList: result.body,
298         loading: true
299     })
300
301     80x     }).catch(function (error) {
302     80x         })
303     80x     }
304
305     80x     // Reformat the phone number
306     formatPhoneNumber(str)
307     {
308         var part1 = str.substr(1, 3)
309         var part2 = str.substr(4, 3)
310     85x         var part3 = str.substr(7, 4)
311
312     40x         return("(" + part1 + ") " + part2 + "-" + part3)
313     }
314
315     // Generate the table if the fetch was successful
316     renderIfSchoolsFound() {
317         if (this.state.schoolList !== null && Object.keys(this.state.schoolList).length !== 0) {
318
319             const columns = [{
320                 Header: 'School Name',
321                 filterMethod: (filter, rows) =>
322                     matchSorter(rows, filter.value, { keys: ["schoolName"] })),
323                 filterAll: true,
324                 accessor: 'schoolName' // String-based value accessors!
325             }, {
326                 Header: 'School Contact',
327                 filterMethod: (filter, rows) =>
328                     matchSorter(rows, filter.value, { keys: ["schoolContactName"] })),
329                 filterAll: true,
330                 accessor: 'schoolContactName' // String-based value accessors!
331             }, {
332                 Header: 'School Phone Number',
333                 filterMethod: (filter, rows) =>
334                     matchSorter(rows, filter.value, { keys: ["schoolContactPhone"] })),
335                 filterAll: true,
336                 accessor: 'formattedPhone' // String-based value accessors!
337             }, {
338     40x                 Header: 'Actions',
339                 accessor: 'menuActions', // String-based value accessors!
340     80x                 style:{textAlign:'center'},
341     80x                 sortable: false,
342     80x                 filterable: false
343             }
344         ];
345
346         for(let value in this.state.schoolList) {
347     40x             this.state.schoolList[value].formattedPhone =
348             this.formatPhoneNumber(this.state.schoolList[value].schoolContactPhone);
349             this.state.schoolList[value].status = "edit";
350             this.state.schoolList[value].menuActions = <div><RaisedButton
351                 primary={true} onClick={this.openModal.bind(this,this.state.schoolList[value])}>
352             label="Edit"/>&nbsp;&nbsp;&nbsp;<RaisedButton
353                 secondary={true} onClick=
354             {this.confirmSchoolDelete.bind(this,this.state.schoolList[value])} label="Delete"/></div>;

```

```

355     }
356
357     return(
358         <ReactTable
359             data={this.state.schoolList}
360             filterable
361             defaultFilterMethod={({filter, row}) =>
362 45x                 String(row[filter.id]) === filter.value}
363             columns={columns}
364             defaultPageSize={10}
365             className="-striped -highlight"
366             defaultSorted={[{id: "schoolName"}]}
367 85x         />
368     )
369
370 }
371 else
372     return("ERROR LOADING SCHOOLS")
373 }
374
375 render() {
376
377     const actions = [
378         <FlatButton
379 85x             label="Cancel"
380 85x             primary={true}
381             onClick={this.closeConfirmDialog}
382         />,
383         <FlatButton
384             label="Delete"
385             primary={true}
386             onClick={this.deleteSchool}
387         />,
388     ];
389     var status = {"status": "add"};
390     return (
391         <MuiThemeProvider>
392             <div className="content">
393                 <NotificationSystem ref="notificationSystem" style={style}/>
394                 <Grid fluid>
395                     <Row>
396                         <Col md={12}>
397                             <Card
398                                 style={{margin:10}}
399                                 title="Registered Schools"
400                                 category={
401                                     <div>These schools are registered with the Science Olympiad
402 system.<br/>They will appear in the registration systems and reports.
403                                     <br/><br/>
404                                     <RaisedButton primary={true} label="Create a new school"
405 onClick={this.openModal.bind(this,status)}/>
406                                 </div>
407                                 ctTableFullWidth ctTableResponsive
408                                 content={
409                                     <Loader color="#3498db" loaded={this.state.loading}>
410                                         {this.renderIfSchoolsFound()}
411                                     </Loader>
412                                 }/>
413                         </Col>
414                     </Row>

```



```

</Grid>
<Modal show={this.state.modal} onHide={this.closeModal}>
  <Modal.Header>
    <Modal.Title> <AppBar
      iconElementRight={<FlatButton label="Close"/>}
      showMenuIconButton={false}
      onRightIconButtonClick={(event) => this.closeModal()}
      title={this.state.modalTitle}
    /></Modal.Title>
  </Modal.Header>
  <Modal.Body>
    <TextField
      id={"schoolName"}
      floatingLabelText="School Name"
      onChange={(event, newValue) => this.setState({schoolName: newValue})}
      value={this.state.schoolName}
      required={true}
      autoFocus={true}
      fullWidth={true}
      hintText="Enter a name for the school"
      errorText={this.state.schoolNameRequired}
    />
    <br/>
    <TextField
      id={"schoolContactName"}
      floatingLabelText="Contact Name"
      onChange={(event, newValue) => this.setState({schoolContactName:
newValue})}

      value={this.state.schoolContactName}
      required={true}
      fullWidth={true}
      hintText="Enter a first and last name for the school contact"
      errorText={this.state.schoolContactNameRequired}
    />
    <br/>
    <TextField
      id={"schoolContactPhone"}
      floatingLabelText="Phone Number"
      onChange={(event, newValue) => this.setState({schoolContactPhone:
newValue})}

      value={this.state.schoolContactPhone}
      required={true}
      fullWidth={true}
      hintText="Enter the school's phone number"
      errorText={this.state.schoolContactPhoneRequired}
    >
      <InputMask mask="1 (999) 999-9999" maskChar="#" value=
{this.state.schoolContactPhone}/>
    </TextField>
  </Modal.Body>
  <Modal.Footer>
    <RaisedButton icon={<FontIcon className="pe-7s-close-circle" />} primary=
{true} label="Cancel"
      onClick={this.closeModal}/>&nbsp;&nbsp; 
    <RaisedButton icon={<FontIcon className="pe-7s-like2" />} primary={true}
label={this.state.modalButton}
      onClick={this.handleSubmit}/>
  </Modal.Footer>
</Modal>
</Dialog>

```

```
        actions={actions}
        modal={false}
        open={this.state.confirmDialog}
        onRequestClose={this.closeConfirmDialog}
      >
        {this.state.confirmMessage}
      </Dialog>
    </div>
  </MuiThemeProvider>
);
}
}

export default Schools;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/Teams StudentTeamCreator.js

100% Statements 10/10

100% Branches 0/0

100% Functions 7/7

100% Lines 10/10

```
1 import React, {Component} from 'react';
2 import {MuiThemeProvider, Popover, Menu, MenuItem, RaisedButton} from 'material-ui';
3 import {Panel, PanelGroup} from 'react-bootstrap';
4 import {Grid, Row, Col} from 'react-bootstrap';
5 import StudentAdder from "../../components/Students/StudentAdder";
6 import TeamAdder from "../../components/Teams/TeamAdder";
7
8 class StudentTeamCreator extends Component {
9   constructor(props) {
10     3x super(props);
11
12     3x this.state = {
13       popOverOpen: false,
14       addPanel: "",
15     };
16
17     3x this.handleClick = this.handleClick.bind(this);
18     3x this.toggleAdd = this.toggleAdd.bind(this);
19   }
20
21   handleClick(event) {
22     1x event.preventDefault();
23
24     1x this.setState({
25       popOverOpen: true,
26       anchorEl: event.currentTarget
27     });
28   }
29
30   toggleAdd(type) {
31     2x this.setState({popOverOpen: false, addPanel: type});
32   }
33
34   render() {
35     6x return (
36       <MuiThemeProvider>
37         <div style={{marginLeft: '10%', marginRight: '10%', width: '80%', maxWidth: 800, textAlign:
38 'center', display: 'inline-block'}}>
39           <RaisedButton primary={true} onClick={this.handleClick} label="Create Student/Team"/>
40           <Popover
41             style={{maxWidth: '200px', width: '200px'}}
42             open={this.state.popOverOpen}
43             anchorEl={this.state.anchorEl}
44             anchorOrigin={{horizontal: 'left', vertical: 'bottom'}}
45             targetOrigin={{horizontal: 'left', vertical: 'top'}}
46           >
47             <Menu>
48               1x <MenuItem style={{textAlign: 'center', width: '200px'}} primaryText="Create
49 Student" onClick={ () => this.toggleAdd("student")} />
50               <MenuItem style={{textAlign: 'center', width: '200px'}} primaryText="Create
51 Team" onClick={ () => this.toggleAdd("team")} />
52             </Menu>
53           </Popover>
54         </div>
55       </MuiThemeProvider>
56     );
57   }
58 }
```

```

55
56                                     <PanelGroup style={{ width: '80%', marginLeft: '10%', marginRight: '10%',
57   textAlign: 'center'}}
58                                     id="add-collapsible-panel-group"
59                                     accordion
60                                     activeKey={this.state.addPanel}
61                                     onSelect={() => {}}
62                                     key="add-collapsible-panel-group"
63                                     >
64                                     <Panel id="add-user-collapsible-panel" eventKey={"student"} style=
65   {{border: 0}}>
66                                         <Panel.Body collapsible>
67                                             <StudentAdder togglePanel={this.toggleAdd} updateTable=
68   {this.props.updateTable} addNotification={this.props.addNotification}/>
69                                         </Panel.Body>
70                                     </Panel>
71                                     <Panel id="add-team-collapsible-panel" eventKey={"team"} style=
72   {{border:0}}>
73                                         <Panel.Body collapsible>
74                                             <TeamAdder togglePanel={this.toggleAdd} updateTable=
75   {this.props.updateTable} addNotification={this.props.addNotification}/>
76                                         </Panel.Body>
77                                     </Panel>
78                                     </PanelGroup>
79
                                     </div>
                                     </MuiThemeProvider>
                                     )
                                     }
                                     }

export default StudentTeamCreator;

```

All files / src/views/Teams TeamManagement.js

100% Statements 9/9

100% Branches 8/8

100% Functions 5/5

100% Lines 9/9

```
1 import React, {Component} from 'react';
2 import {MuiThemeProvider} from 'material-ui';
3
4 import TeamViewer from "../../components/Teams/TeamViewer";
5 import StudentTeamCreator from "../StudentTeamCreator";
6 import {style} from "../../variables/Variables";
7
8 import NotificationSystem from 'react-notification-system';
9
10 class TeamManagement extends Component {
11   constructor(props) {
12     3x super(props);
13
14     3x this.state = {
15       tableUpdateToggler: false,
16       _notificationSystem: null
17     }
18
19     3x this.updateTable = this.updateTable.bind(this);
20     3x this.addNotification = this.addNotification.bind(this);
21   }
22
23   updateTable() {
24     1x this.setState({
25       tableUpdateToggler: !this.state.tableUpdateToggler
26     })
27   }
28
29   addNotification(message, level, position, autoDismiss) {
30     4x if(this.state._notificationSystem) {
31       2x this.state._notificationSystem.addNotification({
32         title: (<span data-notify="icon" className="pe-7s-info"></span>),
33         message: (
34           <div>
35             {message}
36           </div>
37         ),
38         level: level ? level : 'info',
39         position: position ? position : 'tr',
40         autoDismiss: autoDismiss ? autoDismiss : 5,
41       });
42     }
43   }
44
45   componentDidMount() {
46     4x this.setState({_notificationSystem: this.refs.notificationSystem});
47   }
48   render() {
49     10x return (
50       <MuiThemeProvider>
51         <div className="content" style={{textAlign: 'center'}}>
52           <NotificationSystem ref="notificationSystem" style={style}/>
53           <StudentTeamCreator updateTable={this.updateTable} addNotification=
54 {this.addNotification}/>
```

```
55         <TeamViewer updateTable={this.updateTable} tableUpdateToggler=  
56         {this.state.tableUpdateToggler} addNotification={this.addNotification} />  
57         </div>  
58     </MuiThemeProvider>  
59     )  
60 }  
61  
62 }  
  
export default TeamManagement;
```

Code coverage generated by [istanbul](#) at Mon Apr 16 2018 22:47:35 GMT-0400 (EDT)

All files / src/views/UserProfile UserProfile.js

100% Statements 96/96 96.97% Branches 32/33 100% Functions 28/28 100% Lines 94/94

```
1 import React, { Component } from 'react';
2 import {
3   Grid, Row, Col,
4 } from 'react-bootstrap';
5
6 import {UserCard} from '../../components/Cards/UserCard.js';
7 import InputMask from 'react-input-mask';
8 import Button from '../../elements/CustomButton/CustomButton.js';
9
10 import {TextField} from "material-ui";
11 import HttpRequest from "../../adapters/httpRequest";
12 import constants from "../../utils/constants";
13 import AuthService from "../../utils/AuthService";
14 import NotificationSystem from 'react-notification-system';
15 import {style} from "../../variables/Variables";
16 import AppBar from 'material-ui/AppBar';
17 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
18 import PasswordField from 'material-ui-password-field';
19
20 class UserProfile extends Component {
21
22   constructor(props) {
23     18x super(props);
24
25     18x this.updateProfile = this.updateProfile.bind(this);
26     18x this.changePassword = this.changePassword.bind(this);
27     18x this.validPassword = this.validPassword.bind(this);
28     18x this.cleanPhone = this.cleanPhone.bind(this);
29     18x this.notify = this.notify.bind(this);
30
31     18x this.state = {
32       user : {firstName: "", lastName: "", phoneNumber: "", receiveText: false, minutesBeforeEvent:
33         "0"},
34       _notificationSystem : null,
35       password: "",
36       currentPassword : "",
37       newPassword : "",
38       confirmPassword : "",
39       imageUrl : "",
40       desc : "",
41       code: 0
42     };
43
44   }
45
46   18x componentWillMount() {
47     var _this = this;
48     //This is good shit to remember for later
49     // AuthService.getUserEmail();
50     // AuthService.getUserRole();
51     //This sends our credentials in the header
52     18x
53     HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/getUserProfile', 'GET',
54     10x constants.useCredentials(), null).then(function (result) {
```

```

55         _this.setState({
56             user: result.body,
57             desc: result.body.school ? "School: " + result.body.school.schoolName : ""
58         })
59
60     8x     }).catch(function (error) {
61         console.log(error);
62     })
63     18x    //here we will get the user type
64     const userType = AuthService.getUserRole();
65     18x
66     14x    if (userType === "ADMIN") {
67         4x        this.setState({imageUrl :
68         1x    "https://telegram.org/file/811140509/b45/dQTLewKZ9gs.22232.gif/4580677d940852f30e"}});
69         } else if (userType === "COACH") {
70             this.setState({
71                 3x        imageUrl : "https://baseballmomstuff.com/wp-content/uploads/2016/02/coach-cartoon.jpg"
72                 1x        });
73             } else if (userType === "JUDGE") {
74                 1x        this.setState({imageUrl : "https://www.how-to-draw-funny-cartoons.com/image-files/cartoon-
75                 judge-010.jpg"});
76             } else if (userType === "STUDENT") {
77                 this.setState({
78                     imageUrl : "https://classroomclipart.com/images/gallery/Clipart/Science/TN_female-student-
79                     holding-flask-and-test-tube-in-science-lab-science-clipart.jpg"
80                 });
81             19x        }
82         }
83
84     componentDidMount() {
85         8x        this.setState({_notificationSystem: this.refs.notificationSystem});
86         2x    }
87
88     notify(message, level, position, autoDismiss) {
89         if(this.state._notificationSystem) {
90             this.state._notificationSystem.addNotification({
91                 title: (<span data-notify="icon" className="pe-7s-door-lock"></span>),
92                 message: (
93                     <div>
94                         {message}
95                     </div>
96                 ),
97                 level: level ? level : 'error',
98                 position: position ? position : 'tc',
99                 autoDismiss: autoDismiss ? autoDismiss : 10,
100            });
101        }
102    }
103
104    3x    cleanPhone(cleanPhoneNumber) {
105        3x        cleanPhoneNumber = cleanPhoneNumber.replace(/\s/g, ''); // Remove spaces
106        3x        cleanPhoneNumber = cleanPhoneNumber.replace(/\(|\)/g, ''); // Remove ( and )
107        cleanPhoneNumber = cleanPhoneNumber.replace(/-/g, ""); // Remove -
108        cleanPhoneNumber = '+' + cleanPhoneNumber; // Add +
109
110        4x        return cleanPhoneNumber;
111        4x    }
112
113    4x    updateProfile(e) {
114        var body = {};

```



```

115     3x      var _this = this;
116
117     if (this.state.password !== "") {
118     3x
119         body.password = this.state.password;
120
121         //check to ensure the password matches
122     2x      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() + '/sweng500/validate',
123 'POST',
124         constants.useCredentials(), body, true).then(function (result) {
125     2x
126             //Create deep copy of user object
127     2x      body.user = JSON.parse(JSON.stringify(_this.state.user));
128
129     1x      //update the user, first clean the phone number
130      body.user.phoneNumber = _this.cleanPhone(body.user.phoneNumber);
131
132      _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
133 '/sweng500/updateUser', 'POST', constants.useCredentials(), body.user, true).then(function (result) {
134
135          _this.notify(
136     1x      "Profile Successfully Updated",
137          "success",
138          "tc",
139          5
140      );
141     1x
142     1x      _this.setState({
143          password: ""
144      })
145
146      }).catch(function (error) {
147          console.log(error);
148     1x      _this.notify(
149          "Could not update profile at this time. Please try again later.",
150          "error",
151     1x      "tc",
152     1x      5
153      );
154      _this.setState({code : 11});
155      })
156      }).catch(function (error) {
157          console.log(error);
158     1x      _this.notify(
159          "Please provide the correct password in order to update your profile.",
160          "error",
161     1x      "tc",
162          5
163      )
164      _this.setState({code : 14})
165      })
166  } else {
167     1x      _this.notify(
168          "Please enter your current password",
169          "error",
170          "tc",
171          5
172      );
173     7x      _this.setState({code : 1})
174     5x  }

```

```

175     5x    }
176     5x
177     5x    // Check if a password is valid. 8 characters, uppercase, lowercase, and numbers
178     5x    validPassword(text) {
179         if (text.length < 8) return false;
180     4x    var hasUpperCase = /[A-Z]/.test(text);
181        var hasLowerCase = /[a-z]/.test(text);
182        var hasNumbers = /\d/.test(text);
183        var hasNonAlphas = /\W/.test(text);
184     6x    if (hasUpperCase + hasLowerCase + hasNumbers + hasNonAlphas < 3) return false;
185     6x
186        return true;
187    }
188     6x
189    changePassword(e) {
190        var body = {};
191     5x    var _this = this;
192
193     4x    //check to ensure that current password is filled in
194    if (this.state.currentPassword !== "" && this.state.newPassword !== "" &&
195     3x    this.state.confirmPassword !== "") {
196
197        //check to make sure current password is correct
198     3x    if (this.state.newPassword === this.state.confirmPassword) {
199
200        if (this.validPassword(this.state.newPassword)) {
201
202     2x        body.password = this.state.currentPassword;
203
204     2x        //ensure their current password is correct
205        _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
206    '/sweng500/validate', 'POST',
207     1x        constants.useCredentials(), body).then(function (result) {
208
209        //then change the password
210        body.password = _this.state.newPassword;
211
212        _this.serverRequest = HttpRequest.httpRequest(constants.getServerUrl() +
213    '/sweng500/changePassword', 'POST',
214     1x        constants.useCredentials(), body).then(function (result) {
215
216        _this.notify(
217     1x        "Password changed successfully.",
218        "success",
219     1x        "tc",
220        5
221        );
222
223        _this.setState({currentPassword: "", newPassword: "", confirmPassword: "",
224    code: 8}))
225     1x
226        }).catch(function (error) {
227        console.log(error);
228     1x
229        _this.notify(
230        "Could not update at this time. Please try again later.",
231        "error",
232        "tc",
233        7
234        );

```

```

235     _this.setState({code : 10})
236   })
237   }).catch(function (error) {
238     _this.notify(
239       "Incorrect current password. Please provide your current password",
240       "error",
241       "tc",
242       5
243     );
244
245     _this.setState({code : 7})
246   })
247   } else {
248     //put up a notify
249     this.notify(
250       "ERROR: Your password must be 8 or more characters, contain capital letters, lower
251 case letters, and at least one number.",
252       "error",
253       "tc",
254       10
255     );
256     this.setState({code : 4})
257   }
258   } else {
259     //put up a notify
260     this.notify(
261       "New passwords do not match",
262       "error",
263       "tc",
264       5
265     );
266     this.setState({code : 3})
267   }
268 }
269 else {
270   this.notify(
271     "Please enter your current password, a new password, and confirm the new password.",
272     "error",
273     "tc",
274     7
275   );
276 }
277 }
278
279 render() {
280   return (
281     <div className="content">
282       <NotificationSystem ref="notificationSystem" style={style}/>
283       <MuiThemeProvider>
284         <Grid fluid>
285           <Row className="show-grid">
286             <Col md={4}>
287               <UserCard
288                 bgImage="https://ununsplash.imgix.net/photo-1431578500526-
289 4d9613015464?fit=crop&fm=jpg&h=300&q=75&w=400"
290                 //avatar={avatar}
291                 avatar= {this.state.imageUrl}
292                 name= {this.state.user.firstName + " " + this.state.user.lastName}
293                 userName={this.state.user.emailAddress}
294                 //maybe in description put the events they are doing? or school

```

information?

```

        description={<span> {this.state.desc} </span> }
      />
    </Col>
  </Row>
  <Row className="show-grid">
    <Col md={12}>
      <AppBar showMenuIconButton={false} title="User Profile"/>
    </Col>
  </Row>
  <Row className="show-grid">
    <Col md={4}>
      <TextField
        name="fname"
        hintText="First name"
        floatingLabelText="First name"
        onChange = {(event, newValue) => { var ryanRocks = this.state.user;
          ryanRocks.firstName = event.target.value;
          this.setState({user : ryanRocks})}}
        value={this.state.user.firstName}
      />
    </Col>
    <Col md={6}>
      <TextField
        name="lname"
        hintText="Last name"
        floatingLabelText="Last name"
        onChange = { (event, newValue) => {var ryanRocks = this.state.user;
          ryanRocks.lastName = event.target.value;
          this.setState({user : ryanRocks})}}
        value={this.state.user.lastName}
      />
    </Col>
  </Row>
  <Row className="show-grid">
    <Col md = {4}>
      <TextField
        name="phone"
        floatingLabelText="Phone number"
        onChange = {(event, newValue) => {var ryanRocks = this.state.user;
          ryanRocks.phoneNumber = event.target.value;
          this.setState({user : ryanRocks})}}
        value={this.state.user.phoneNumber}
      >
        <InputMask mask="9 (999) 999-9999" maskChar="#"
          value={this.state.user.phoneNumber}/>
      </TextField>
    </Col>
    <Col md={4}>
      <TextField
        name="timeBeforeEvent"
        hintText="Time Before Event"
        floatingLabelText="Time Before Event"
        onChange = {(event, newValue) => {var ryanRocks = this.state.user;
          ryanRocks.minutesBeforeEvent = event.target.value;
          this.setState({user : ryanRocks})}}
        value={this.state.user.minutesBeforeEvent}
      >
        <InputMask mask="99" maskChar=""
          value={this.state.user.minutesBeforeEvent}/>
      </TextField>
    </Col>
  </Row>

```

```

355     </TextField>
356   </Col>
357 </Row>
358 <Row className="show-grid">
359   <Col md={4}>
360
361   </Col>
362   <Col md={6}>
363     <PasswordField
364       name="currentPassword"
365       id="password"
366       // hintText="Current Password"
367       floatingLabelText="Current Password"
368       onChange={(event, newValue) => this.setState({password: newValue})}
369       value={this.state.password}
370     />
371   </Col>
372 </Row>
373 <Row className="show-grid">
374   <Col md={4}>
375     <input
376       name="receiveText"
377       type="checkbox"
378       checked={this.state.user.receiveText}
379       onChange={(event, newValue) => {var ryanRocks = this.state.user;
380         // console.log(event.target.value);
381         ryanRocks.receiveText = !ryanRocks.receiveText; //; ? true :
382         false);
383         this.setState({user : ryanRocks})}} />
384     <label>
385       Receive Text Messages
386     </label>
387   </Col>
388   <Col md={2}>
389     <Button
390       bsStyle="info"
391       pullRight
392       fill
393       type="submit"
394       onClick={this.updateProfile}
395     >
396       Update Profile
397     </Button>
398   </Col>
399 </Row>
400 <Row className={"show-grid"}>
401   <label>
402
403   </label>
404 </Row>
405 <Row className={"show-grid"}>
406   <AppBar showMenuIconButton={false} title="Change Password"/>
407 </Row>
408 <Row className={"show-grid"}>
409   <Col md={12}>
410     <PasswordField
411       name="currentPassword"
412       id="currentPassword"
413       // hintText="Current Password"
414       floatingLabelText="Current Password"

```

```

415         onChange={(event, newValue) => this.setState({currentPassword:
416     newValue}})}
417
418         />
419     </Col>
420 </Row>
421 <Row className={"show-grid"}>
422     <Col md={12}>
423         <PasswordField
424             name="newPassword"
425             id="newPassword"
426             // hintText="New Password"
427             floatingLabelText="New Password"
428             onChange={(event, newValue) => this.setState({newPassword: newValue})}
429             value={this.state.newPassword}
430         />
431     </Col>
432 </Row>
433 <Row className={"show-grid"}>
434     <Col md={12}>
435         <PasswordField
436             name="confirmPassword"
437             id="confirmPassword"
438             // hintText="Confirm Password"
439             floatingLabelText="Confirm Password"
440             onChange={(event, newValue) => this.setState({confirmPassword:
441     newValue}})}
442             value={this.state.confirmPassword}
443         />
444     </Col>
445 </Row>
446 <Row className={"show-grid"}>
447     <Col md={6}>
448         <Button
449             bsStyle="info"
450             pullRight
451             fill
452             type="submit"
453             onClick={this.changePassword}
454         >
455             Change Password
456         </Button>
457     </Col>
458 </Row>
459 </Grid>
460 </MuiThemeProvider>
461 </div>
462 );
463 }
464 }

```

export default UserProfile;