Sprint 1 Defect Log

Guna Kondapaneni, Lisa Campbell, Matthew Pace,
Brian Kaetzel, Joseph Khawly

Design Inspections

OAuth2 Flow Design Inspection

Defect	Description	Severity	How Corrected
1	After authorizing a user, the user is sent back to the sign in page.	1	TODO: Check if user is authorized with ping method.
2	After authorizing a user, the user is permanently signed out	3	TODO: Make sign out button and supporting method on server.

OAuth2 Flow describes the interaction of the Sign In and Authentication Modules.

Module Inspections

Sign In Module Inspection

Defect	Description	Severity	How Corrected
1	After clicking the sign in button, the, the user is displayed an invalid route	1	Correctly named and added route for sign in url
	Code Before		Code After
	`// No route handling code, assumed // plugin set up for us`		`private Filter ssoFilter() { ... // Initialize filter, filters filters.add(ssoFilter(google(), "/login/google")); filter.setFilters(filters); return filter; }`

Code Before:

```
// No route handling code, assumed
// plugin set up for us
```

Code After:

```
private Filter ssoFilter() {
  ... // Initialize filter,
filters

  filters.add(ssoFilter(google(),
"/login/google"));
  filter.setFilters(filters);

  return filter;
}
```

Main Layout Module Inspection

Defect	Description	Severity	How Corrected
1	MainLayout doesn't switch active	2	Researched correct way to

	views		pass components in React. React didn’t like that the component was named “activeComponent”.
	Code Before		Code After
	`<div>` `<Navigation />` `<this.props.activeComponent />` `</div>`		`<div>` `<Navigation />` `<this.props.activeView />` `</div>`

Authentication Module Inspection			
Defect	Description	Severity	How Corrected
1	Client ID and Secret have to sent to people via email, or in executable (can be found in files)	4	Generate a client ID and secret every time a new user is added.
	Code Before		Code After
	`class ClientResources {` `// client initialized` `// no constructor` `... // other methods` `}`		`class ClientResources {` `// client initialized` `public ClientResources() {` `client.setClientId(...)` `client.setClientSecret(...)` `}` `// clientId and secret are loaded` `// from environment variables` `... // other methods` `}`

Unit Tests

Authentication Unit Test			
Defect	Description	Severity	How Corrected
1	(MANUALLY) The Client is sent to an error screen from Google	2	Give the Server better Credentials from Google Developer’s Console. {No code}

Main Layout Unit Test (Using Jest and Enzyme)

Defect	Description	Severity	How Corrected
1	MainLayout doesn’t switch active views	2	Renamed component. Test asserts on MainLayout when rendered with two different active views.

Sign In Unit Test (Using Jest and Enzyme)			
Defect	Description	Severity	How Corrected
1	After clicking the sign in button, the, the user is displayed an invalid route	1	Correctly named and added route for sign in url
2	After checking to see if the module renders correctly, if the same unit test is run again after a change is made to the module, it will fail because the old snapshot (or render output) that it compares against is never replaced or deleted	3	Created cleanup for npm that deletes old snapshots whenever npm run cleanup is executed Added to package.json: `"cleanup": "rm -r client/__tests__/__snapshots__/",`

Unit Testing Defects

1. Yes, we are using test automation.
    a. We are using Jest.js and Enzyme on the front end, for React testing. To run all the front-end tests, simply execute *npm test* via command line. We are using JUnit and TestNG for the backend tests.
    b. Note: For a couple of the tests on the backend, a tester has to manually run, and interact with the UI. The tests that do this include:
        i. Authentication for the first time Test