Design Inspection

Code Inspection

Unit Testing

Team 7 uCal

Andrew Bostock, Ben Collier, Matt Kramer, Eli Scherrer, Maneesh Tewani, Hector Trevino

1. Design Inspection Defects

Product	uCal		
Date	3 March 2018		
Author	Team 7		
Moderator			
Inspectors	Andrew Bostock, Ben Collier, Matt Kramer, Eli Scherrer, Maneesh Tewani, Hector Trevino		
Recorder			
Defect #	Description	Severity	How Corrected
1	The events schema as for the database is excessively complex and messy	2	Reworked the schema and endpoints to simplify the code.
2	Creating a group should not also create invite. This should be handled by the invite function to allow for the more specific response codes	3	Removed the ability to invite users in the group creation endpoint

2. Code Inspection Defects

Product	uCal		
Date	3 March 2018		
Author	Development Team		
Defect #	Description	Severity	How Corrected
1	Accurate error message displayed when creating a calendar by user	2	Was more specific in error messaging in createcalendar route for users
2	Accurate error message displayed when creating a calendar by group	2	Was more specific in error messaging in createcalendar route for groups
3	Groups schema has a property defined as “creator” where we were writing tests for it as “owner”	2	Changed the schema to reflect the use of “owner” instead of “creator”
4	Time is not being displayed properly on the calendar event	2	Updated calendarcomponent addition
5	Button to delete events not properly showing up	2	Added the action necessary to delete an event when mapping on ngOnInit()
6	Google Login wouldn’t work because password was required	2	Removed password requirement from schema
7	Inviting members to groups does not provide descriptive return information in the case of some failures	2	Tracked what invites failed and which were successful.
8	Cannot rsvp to events	2	Added routes for accepting and declining events
9	The route for removing users from a group’s return messages were not specific/helpful and would return the wrong information	2	Rewrite sections of the function to assure the right return value was used and they contained more information.

3. Unit Testing Defects

Specify whether you automated the unit testing. If yes, give a brief description of your automation framework/strategy.

Our system contains the following modules:
Backend:

- User Endpoints
- Group Endpoints
- Friend Endpoints
- Calendar Endpoints
- Event Endpoints

Frontend:

- AppModule
- CalendarComponent
- AppRoutingModule
- AuthService
- AuthGuardService
- GroupComponent
- LoginComponent
- NavBarComponent
- LoginSuccessComponent
- SettingsComponent
- TokenHandlerService

For our Front-End, we used a karma task runner with the language jasmine to test our angular code.
For Back-end we used a karma-runner with the language mocha to test our express server endpoints.

User Endpoints
The User Endpoint module is the set of all the routes that directly involve users and their account, such as account creation. The front end will use these routes to access and update user data in the database.

Product	uCal		
Module	User Endpoints		
Date			
Author			
Defect #	Description	Severity	How Corrected
1	Everytime a specific user was requested the server would return	2	Changed the name of the get current user

	the current user		route
2	When adding a user to a group, the group ID was being added to the user’s data but the user ID was not being added to the group’s data	2	Added code that updates the group’s data

Group Endpoints

The Group Endpoint module is the set of all the routes that directly involve groups like their creation, deletion, and modification. The front end will use these routes to access and update group data in the database.

Product	uCal		
Module	Group Endpoints		
Date	04 March 2018		
Author	Andrew Bostock		
Defect #	Description	Severity	How Corrected
1	When inviting users to groups it does not check that the users are not already in the group before inviting	2	Added check to see if users were already in the group before inviting them.
2	When inviting members the endpoint did not wait for the promise to resolve causing invalid output	2	Waited for promises to resolve before continuing and returning the appropriate value.
3	The remove user from group function compared invalid arguments causing the appearance of success when it failed	2	Changed to correct variables to assure that the correct operations were taken

Friend Endpoints

The friend endpoints are the set of routes that allow for friends to be requested, declined, and accepted in the database. The routes will be called by the frontend to create requests to the database.

Product	uCal

Module	Friend Endpoints		
Date	04 March 2018		
Author	Andrew Bostock		
Defect #	Description	Severity	How Corrected
1	Users were being added to their own friends list rather than others.	1	Corrected statement to use proper variable
2	Users that are already friends can send friend requests to each other adding each other multiple times.	2	Added checking to the send friend request route to see if the users are friends.

Calendar Endpoints

The Calendar Endpoint module is the set of all the routes that directly involve calendars and their creation, deletion, and modification as well as routes for adding users/ groups/ events to calendars. The front end will use these routes to access and update calendar data in the database.

Product	uCal		
Module	Calendar Endpoints		
Date	3/4/2018		
Author	Matthew Kramer		
Defect #	Description	Severity	How Corrected
1	When updating a calendar, response 200 would be sent back even if user was trying to change values that were not name/description	1	Added guards to make sure only name and description were being updated, errors if other properties were trying to be changed
2	Deleting a calendar does not remove it from the users list of calendars	1	Removed it from the users list of calendars
3	Deleting events doesn’t actually delete events	3	Added functionality to remove events

Event Endpoints

The Event Endpoint module is the set of all the routes that directly involve events like their creation, deletion, and modification. The front end will use these routes to access and update event data in the database.

Product	uCal		
Module	Event Endpoints		
Date			
Author			
Defect #	Description	Severity	How Corrected
1	Remove endpoint wouldn’t remove from calendar	1	Used findByIdAndRemove to remove from both event collection and calendar collection
2	Created events did not follow the proper database schema	2	Added the missing fields before saving the event
3	The create event endpoint was not making sure the calendar existed before adding it to the event	2	Added a check to validate the Calendar ID
4	Getting events would sometimes return successful even if the requested event didn’t exist	2	Added a check to see if the mongodb returned null instead of an event
5	Adding an event to another calendar was adding event IDs even if they didn’t correspond to real events	2	Added a check to make sure the event ID was valid

AuthGuardService*

“Guards” routes and checks whether user can access that page with/without the token.

CalendarComponent*

Module that displays calendar and event information on the calendar. Deals with adding and removing event information to the calendar.

LoginComponent*

Handles validation and logging in of users through email/password and Google OAuth Login.

SettingsComponent
Manages settings of user profile and group information.

Product	uCal		
Module	SettingsComponent		
Date	03/04/18		
Author	Maneesh Tewani		
Defect #	Description	Severity	How Corrected
1	Test wasn’t using http mock	2	Configured testbed to mock the data sent back from backend

GroupsComponent
Manages settings of user profile and group information.

Product	uCal		
Module	GroupsComponent		
Date	03/04/18		
Author	Maneesh Tewani		
Defect #	Description	Severity	How Corrected
1	Test wasn’t using http mock	2	Configured testbed to mock the data sent back from backend

AppModule*
Container for other components to make sure parts of the application are separated. Crucial part of the application that bootstraps other components.

AppRoutingModule*
Encompasses logic for client-side routing of the angular application.

AuthService*
Checks if the user is authenticated by checking JWT and if it has expired.

GroupComponent*
Displays group information.

LoginSuccessComponent*
Intercepts the token credentials and stores them in localStorage.

NavBarComponent*
Displays navbar and handles navigating to different client-side routes.

TokenHandlerService*
Sets and clears tokens for JWT authentication and interception of HTTP Requests.

‘*’ Indicates that unit tests were written, but no defects were found after running the unit tests.