UTHRIFT

Submission 06

April 2017

Team Mango Members

John Drakos Yixiang Xu Michael Feuerstein

Yi Fung Bryce Bodley-Gomes Ryan Semple

Special instructions to run:

```
$ npm install history@2.0
```

Both items and classes can be searched for in the searchbar. Items match by title or description. Classes match by title or subject.

Many bugs have been fixed since submission. Latest working version is:

```
commit df919b36caafb1189ebb0918d5667bf83d06f86a
```

Cut Features:

Submission Form: Cut out the drag and drop image upload feature. Also cut out a ton of unused features such as extra item description fields. (Weren't being pulled from database at all, so were removed). Currently the "Class Related" section on the submission page also includes features that aren't being used. These may be used later, so the HTML was left untouched. They currently do save info, but the info isn't written to the database because it is currently not needed.

Search Page: The tracking list was cut previously, so the heart icons found on the search page for each item do not perform any function (designed to add items to a tracking list).

Bugs/Issues

Classpage: No current bugs. "Schedule" and "Other Info" tabs are not in use.

Item ID's (for submitting and deleting items): At this time our application bases the next item id off the length of the items array. For this to work correctly if an item were to be deleted it would require all items AFTER it to have their id's shifted down and each ID entry relating to the shifted item in users to be downshifted. As this is a very computationally heavy task for larger datasets it clearly is not ideal. Thus our item ID should more complexly shift through and find any "missing" id's before trying to add an item. If it finds a "deleted ID" it would take that id instead of trying to assign the length of the items object to the new item.. This would allow ID's to not just go up and up forever. Instead they would fill the minimal number space. However, we have yet to implement this feature

Submitting an Item: Not a bug, just a note that the "Class Related" section of the submission page is not meant to write to the database right now because we are not pulling that information yet. We have not fully decided how we want to connect items and classes together, so this section of info currently does not write to the database. Also not a bug: currently submitting a photo still replaces the photo with the image of an iclicker. The next startup will allow writing images to the database.

HTTP Routes

Get recomendedItems/:userid
Used in page: homepage
Purpose: This route is for getting recommended items for the user.
Overview: At this stage all users have the same list of recommended items. We pass in the userID to allow future functionality (likely not to be implemented) in case we ever decide to implement customized recommended lists for individual users.

Get /profile/:userid
Used in page: profilepage
Purpose: This route is for getting user information for the user.

Get /classPage/:classID
Used in page: classpage
Purpose: This route is for getting class information.

Get /searchPage/:cat/:term
Used in page: searchresults
Purpose: This route is for getting the item information from a search query.

Get /searchPage/:term
Used in page: searchresults
Purpose: This route is for getting the class information from a search query.

Put /profile {body: UserDataSchema}
Used in page: profilepage
Purpose: This route is for updating user information for the user.

Post /submissionForm {body: ItemsSchema}
Used in page: Submission Form
Purpose: This route is for submitting new items to the database

Get ItemPage/:itemID
Used in page: Item page
Purpose: This route gets item information to display on the item page

Get ItemPage/:userID/:itemID
Used in page: Item page

Purpose: This route gets the user's information, specifically the viewingItem which is the item that needs to be displayed.

Responsibilities

Bryce Bodley-Gomes: Fixed inconsistencies in database relating to who was selling which items. Moved everything into client server sections. Removed unnecessary imports in navbar. Made homepage pull from the server side server rather than the local server. Remodeled base pages so that instead of navbar being recreated each page change it is now a constant and just the subsection is rerendered. Remodeled reset-database (so it calls server database to reset). Fixed css for homepage, and profile page. Helped write the submission document. Redesigned the search algorithm and the search page to allow rendering of more than 6 items. It also is now using the search term to filter results (rather than simply filtering on category). Helped implement class search functionality. Redesigned searchresults page to accommodate searching for items or classes.

Michael Feuerstein: Revised getClassData function to use web server. Assisted in getSearch function pathing. Created searchbar.js component and fixed client side searching issues. Assisted in minor aesthetic changes on multiple pages. Helped debug code. Helped write submission report. Assisted in redesigning the search algorithm to use filter to actually get the correct results. Helped implement class search functionality. Redesigned searchresults page to accommodate searching for items or classes.

Yixiang Xu: Revised getUserData and updateUserData function to use web server. Added userData schemas. Made profilepage work.

Yi Fung: Revised getSearch to use the web server and query for items with the corresponding category and text. Populated the database such that there are items under each of the different categories: books, tech, events, etc. Added in the componentDidUpdate and the refresh function under searchresults.js such that the page refreshes properly when making another search query. Helped write this report.

Ryan Semple: Fixed bug related to user selling list when submitting an item from last startUp. Created items.json schema. Revised the submission form page to function using the web server and revised formatting to get rid of unused features. Helped clean up database to be more consistent. Helped write submission document.

John Drakos: Revised the getUserDataItem and getItemInfo functions to use the web server instead of the mock server on the client side. This allowed the item page to be fully functional and display the correct item information when a user clicks a specific button. Helped compose submission report.