BoilerX

Incremental Testing and Regression Testing Log

Team Members: Sean Chew, Zuyuan Fan, Qi Meng, Ling Zhang

Classification of Components

Module Description

Query Module

Query module is the backend module responsible for getting items according to search keywords and filters from the database.

Input: A string representing the searched item that the user wants to retrieve.

Output: The array of item objects that matches the string input.

Interface Components

Interface components include buttons, input manager and other components in the web interface.

Input: Frontend interface labels and callback function.

Output: Callback function is called when button is pressed.

Item Display Interface

The item interface is the frontend user interface that handles displaying all fetched items on the search result page as well as handles displaying detailed item page.

Input: A string inputted into the search bar representing the desired item.

Output: The item objects that matches the string input.

Authentication Module

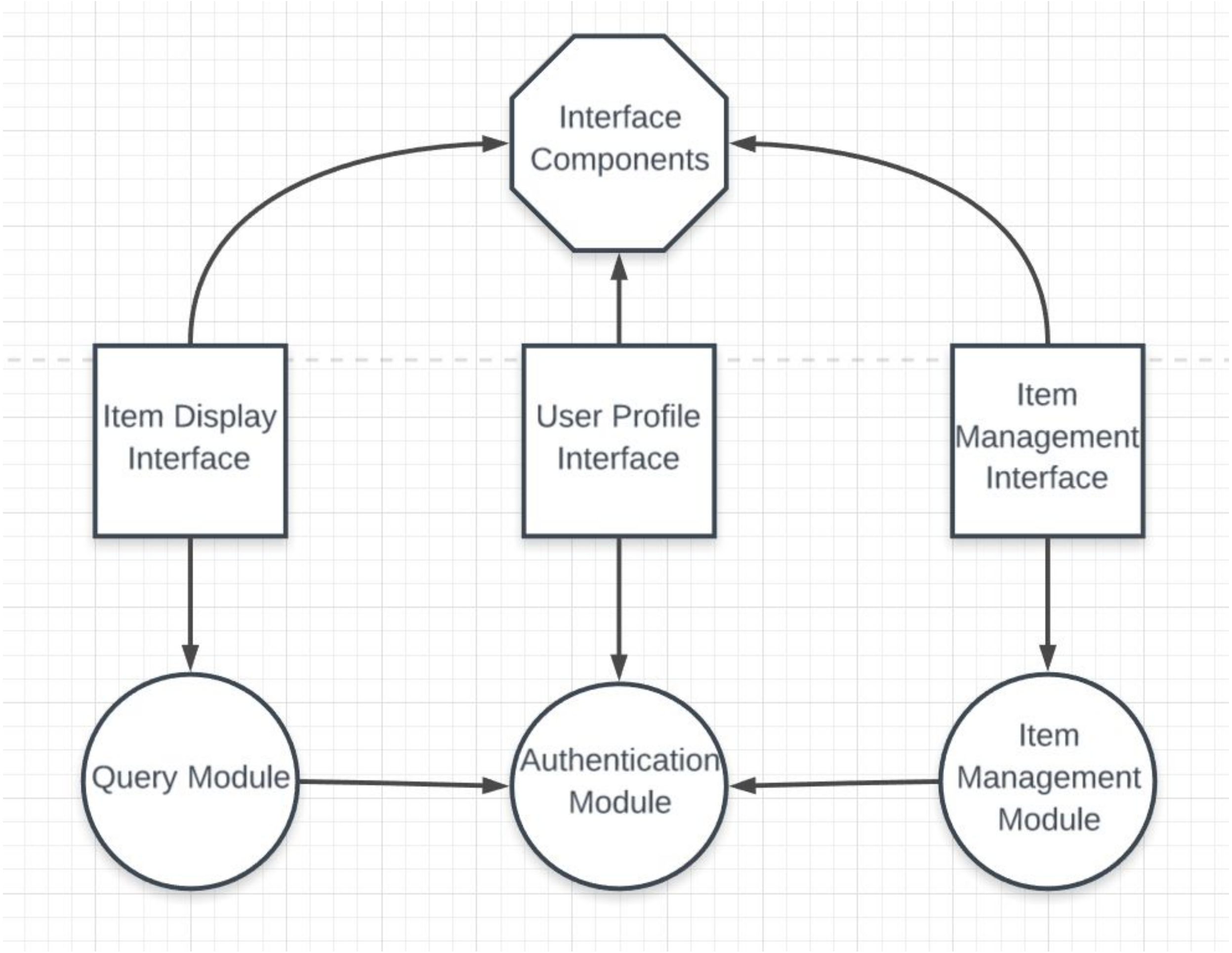
The authentication manager is the integrated module that handles user signup, login, and authentication as well as a two-step verification process verifying that the user has a valid purdue email account.

Input: User signup/login credentials such as Purdue email address, account password, and confirmation code.

Output: A confirmation of whether or not the signup/login was successful.

Incremental Testing Technique

We incrementally tested our application using a bottom-up approach. The modules at the bottom were implemented and tested first and we would fix those bugs. Next, we would integrate the modules together with the interfaces at the top. Then, testing would have to be done again until the application fixes all the bugs.



From a bottom-up incremental testing approach, we first used drivers to test the backend modules at the bottom to ensure that we got rid of the majority of the bugs located within the AWS modules. Afterwards, we would integrate the lower level modules with the upper level interfaces one step at a time and tested progressively. At each stage, the integrated modules and interfaces would have to pass a series of unit test cases before we would move on with more integration which is tested.

Incremental and Regression Testing

Automation

Query Module

Module testing changed from manual command line testing with valid or invalid mock events to automated Linux shell script tests. This component is also tested with sets of regression tests. Each time a bug is fixed, this component is tested on inputs that previously caused the bug, the inputs that may relate to this bug, and random inputs. Thus, we make sure that the bug is correctly fixed and no new bugs are introduced.

Interface Components

Interface components are tested using the Jest and Enzyme React testing framework with automated test runs. Tests are done to examine all boundary cases like the visibility of interface component, which includes a variety of inputs and different combination of components.

Item Display Interface

Item display interface testing uses WebdriverIO automation testing framework. Tests are designed to cover edge cases and a variety of inputs under different user status. Item display interface acts differently when users are logged in and logged out.

Authentication Module

Module testing changed from manual command line testing with valid or invalid mock events to automated Linux shell script tests. This component is also tested with sets of regression tests. Each time a bug is fixed, this component is tested on inputs that previously caused the bug, the inputs that may relate to this bug, and random inputs. Thus, we make sure that the bug is correctly fixed and no new bugs are introduced.

Defect Log

Module	Query Module

Incremental test

Defect No.	Description	Severity	Solution
1	The items received on the interface for popular items are not sorted by popularity.	2	Add a callback function to sort the items according to price or popularity in the Lambda functions.
2	When the frontend was calling the API function, the backend failed to execute GET request.	1	Instead of parsing filter attributes in event.body, we now parse filter attributes in event.query. Since default API gateway endpoints do

			not support body in GET requests.
3	When the frontend was using the API GET request to fetch the items by keywords using “get-item” Lambda function, the backend failed to execute GET request.	1	Fix in progress.The response header should be configured for pre-flight cors requests.
4	When the frontend was using the API to get most popular items for the homepage, all items in the database are returned	2	Add a “Limit: 5” attribute in scan request of dynamoDB. This way, only the items having top 5 popularity will be returned to the client.

Regression test

Defect No.	Description	Severity	Solution
1	The items returned by query is less than the expected number in get_items_popularity function.	2	Remove the global secondary index field in dynamoDB scan parameters.

Module	Interface Components

Incremental test

Defect No.	Description	Severity	Solution
1	The search term entered in the search bar has been completely cleared on the search result page.	1	Removed componentWillReceiveProps function for the search bar.
2	After login, the profile button with a profile picture on the top navigation bar cannot be displayed.	2	Fix in progress. Remove the logout button and create the profile button with profile picture and transfer the logout button to the profile button

Module	Item Display Interface

Incremental test

Defect No.	Description	Severity	Solution
1	The item containers are overlapped when the window is resizing.	2	Change the class name for the columns from “col-lg-3” to “col-md-4”.

Module	Authentication Module

Incremental test

Defect No.	Description	Severity	Solution
1	Cookie crashed and displayed error message: “NotAuthorizedException: Logins don't match. Please include at least one valid login for this identity or identity pool.”	1	Fix in progress. Possible solution is to display another message and recommend the user to clean up the cookie.

Updated Product Backlog

Requirements

- Functional

Backlog ID	Functional Requirements	Hours	Status
1	As a user, I would like to be able to sign up using my Purdue email	2	Completed in Sprint 1

2	As a user, I would like to be able to verify my email address	4	Completed in Sprint 1
3	As a user, I would like to be able to set a password and change password any time I log in.	3	In-progress: moved to Sprint 2
4	As a user, I would like to be able to change username	2	Planned for Sprint 2
5	As a user, I would like to be able to set a profile picture	2	Planned for Sprint 2
6	As a user, I would like to be able to login to my account with my Purdue email and password	5	Completed in Sprint 1
7	As a user, I would like to be able to logout of my account	3	Completed in Sprint 1
8	As a user, I would like to be able to search for items by keywords	7	In-progress: moved to Sprint 2
9	As a user, I would like to be able to filter items by category	3	In-progress: moved to Sprint 2
10	As a user, I would like to be able to sort items by price or popularity	4	In-progress: moved to Sprint 2

11	As a user, I would like to be able to see the most popular items on the homepage	5	Completed in Sprint 1
12	As a user, I would like to be able to view my account information, which includes my personal profile and item listings	4	Planned for Sprint 2
13	As a user (seller), I would like to be able to post an item for sale	3	Planned for Sprint 2
14	As a user (seller), I would like to be able to add title and/or description for the item that I want to sell	2	Planned for Sprint 2
15	As a user (seller), I would like to be able to upload pictures of the item that I want to sell	3	Planned for Sprint 2
16	As a user (seller), I would like to be able to create tags for the item that I want to sell	2	Planned for Sprint 2
17	As a user (seller), I would like to be able to set a price for the item that I want to sell	2	Planned for Sprint 2
18	As a user (seller), I would like to be able to edit my item information after I post them for sale	2	Planned for Sprint 2
19	As a user (buyer), I would like to be able to buy an item by clicking the “buy button” which will send an system email notification to the seller	5	Planned for Sprint 2
20	As a user (seller), I would like to be able to receive a system email stating the buyer’s email and the item information once a sale is made	3	Planned for Sprint 2
21	As a user (seller), I would like to be able to take down the listed item after it is sold	5	Planned for Sprint 2

22	As a developer, I would like to inspect my code, unit inspect my code, and inspect the design of my code	100	In-progress: moved to Sprint 2
	Total Hours:	*171*	

- Non-functional

Backlog ID	Non-functional Requirements	Hours	Status
23	As a developer, I would like to learn how to use ReactJS	10	Completed in Sprint 1
24	As a developer, I would like to learn how to use serverless framework to configure AWS services	10	Completed in Sprint 1
25	As a developer, I would like to learn to build a custom API hosted on AWS API Gateway	20	Completed in Sprint 1
26	As a developer, I would like to learn how to configure DynamoDB and S3 cloud storage for our web app	10	Completed in Sprint 1
	Total Hours:	*50*	