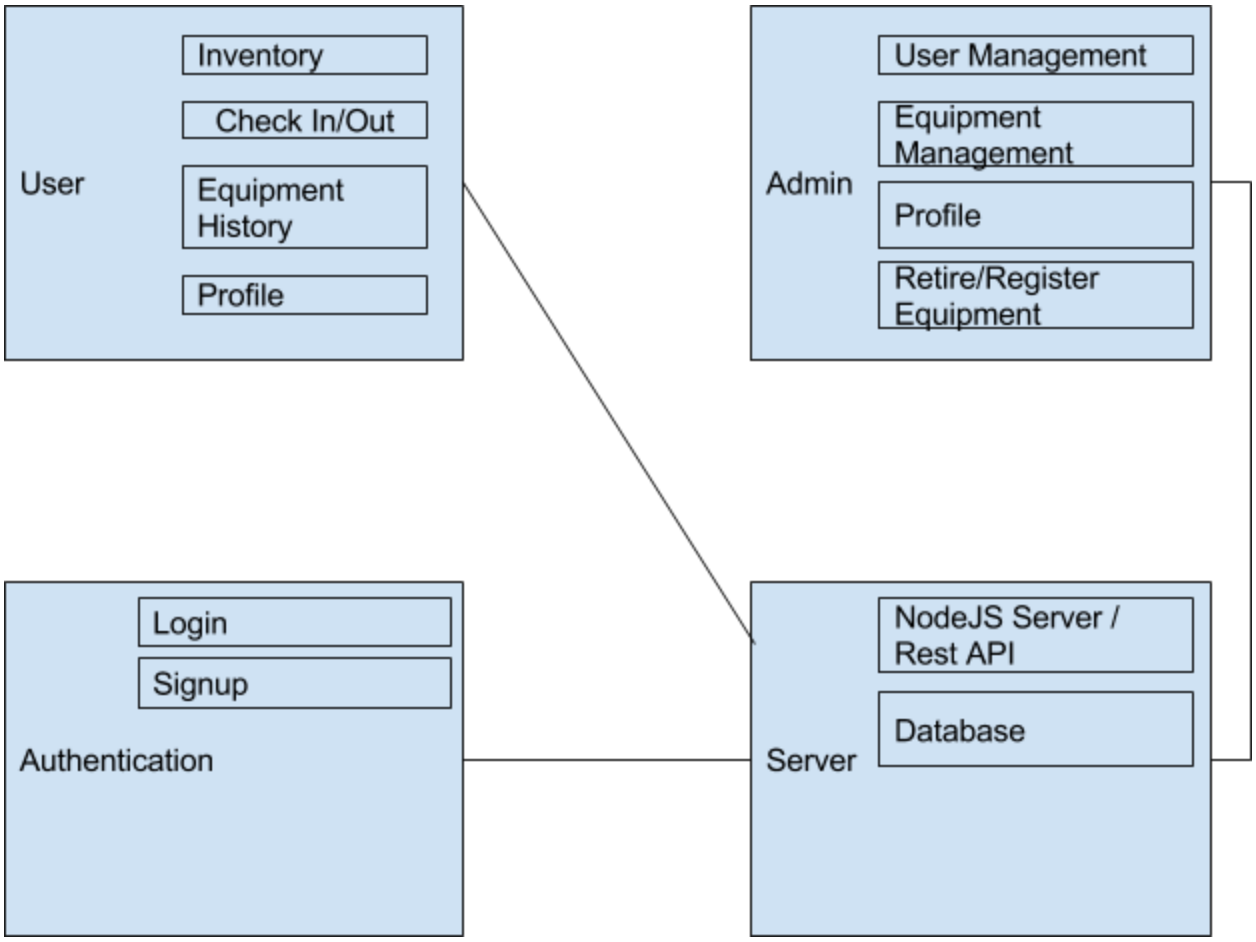
Sprint 2: Incremental and Regression Test Log

Team 17: Brian Chan, Nick Saggese, Xu Han, Do Bor Shiuan Chen, Hongda Zeng

1. Classification of Components

1.1 Component Definitions



Module Authentication:

First time users will have to create an account by providing them with basic information: first name, last name, phone number, email and password. After pressing create account, it will send a RESTful API query to the server creating the user on the database. If there are any incorrect inputs, it would be caught here. The page will then be redirected to the login page.

The login module will be the first page that a user will see. This is where the user would authentic their user information against the database. It sends a RESTful API query

containing the email and password. If the email and password is incorrect, it would send back an error. The login page would then display a error message. If the login was successful, the login page will be redirected to user.html.

It largely takes click input from the user or in some cases form data. It outputs data both as JSON payloads to the server and display to the screen.

Module User:

The user module will be the second page that a user may see. It will display all the items in the inventory. A user can filter checked in item and checked out item. A user can also view the details of all the item. A user will be able to check their own profile detail such as first and last name, phone number, and email, if they wish to edit their details, they can update their info. An admin can give user admin privileges. All the data for the user is contained in the server module. All inputs are driven through Server API responses and user button clicks. Output is sent via API payload to the server and display to the screen.

Module Admin:

The admin module will require special permission to access. Some users will be elevated to this privilege level through the database. This module will be used to take care of administrative work such as checking user/item history. It largely takes click input from the user or in some cases item information for register/retire actions. It outputs data both as JSON payloads to the server and display to the screen.

Module Server :

The server module contains all REST API routes and the database for storing all user information. Anytime a client connects to the checkout website and views or edits information, the server routes are called and information are exchanged between the client and server. Inputs are JSON payloads sent via HTTP from the other modules. Outputs are JSON payloads sent via HTTP to the other modules.

1.2 Incremental testing Method

We used a top-down method of testing in this sprint as we finally had a working front end in order to test the full stack all at once. We tested front end modules like the Admin, User, and Authentication first then proceeded to lower level modules like the Server to test back end functionality.

2. Incremental and Regression Testing

Module: Server

Incremental Testing

Defect #	Description	Severity	How to Correct
1	Server crash on incorrect password	1	Check for incorrect password on server side before query.

2	Server does not deny requests if cookies are expired	1	Fix if statements to automatically return if token is expired
3	Server crashes on bad input	1	Use if statements to check for bad data

Regression Testing

Defect #	Description	Severity	How to Correct
1	Login does not login at all.	1	Fixed if condition for incorrect password which was always executed

Module:Admin

Incremental Testing

Defect #	Description	Severity	How to Correct
1	Click listener for viewing user details in admin automatically “clicked” every time on page load.	1	Assign function for onclicklistener in js to a closure. Did this universally across the code base and it fixed many other bugs.
2	Get Device Log was not showing output. Triggered through view details in admin inventory.	2	Nothing was displayed as there was no history. Instead created an if condition to check if there is no history and output an explanatory message to the end user.

Regression Testing

Defect #	Description	Severity	How to Correct
1	Viewing an individual item’s log from admin. Global Navigation would not work anymore.	1	Item display was attaching to pageContent instead of pageBody. This overwrote some of

			the global navigation actions.

Module: User

Incremental Testing

Defect #	Description	Severity	How to Correct
1	When trying to load inventory as a user, there would be no items.	1	The cookies were not being set correctly due to CORS settings. Had to set CORS to the domain name.
2	When trying to load inventory with correct token in cookie, it was still empty.	2	Cookies were expiring too fast. Changed cookie expiration date to 1000 days
3	Adding logged in, the user would be added to the master list; however, user couldn't access any features	1	Token was being generated twice. The token in master list and the token returned was different. Returned the token in master list instead.

Regression Testing

Defect #	Description	Severity	How to Correct
1	After setting CORS settings, CORS warning would appear preventing the page to load.	3	CORS setting was set to the domain name; however, when tested locally, the domain is null which wasn't set in CORS setting. Since this issue wasn't only working locally, decided to work on server instead.
2	Setted cookies TTL to sufficient period to observe expiration. Cookie expiration does not occur on server side	3	This is an issue because a man-in-the-middle attack could occur resulting in security issue. Server wasn't recycling expired cookies properly

Module: Authentication

Incremental Testing

Defect #	Description	Severity	How to Correct
1	Login page redirects to index.html, but login page should be index.html and redirect to user.html	3	Modify server to host different file orders
2	Returning to the login page from sign up gave a 404 error because login.html does not exist anymore.	2	Redirected to the right page, which is index.html

Regression Testing

Defect #	Description	Severity	How to Correct
1	Cancel changes made to a person's profile threw a 404 because login.html no longer existed.	2	Change the cancel button to direct to user.html or admin.html depending on the user.
2	Login page directed to user.html whether the signed in user was an admin or not.	2	Redirected to the right page for the right user. If is a normal user, go to user.html. If is an admin user, go to admin.html