Product	Lendr Design Inspection		
Date	September 15th, 2017		
Author	Rahul Pai, Devaunsh Sambhav		
Moderator	Aditya Dhingra, Shreyaansh Baasi, Yash Shiroya, Rahul Pai, Devaunsh Sambhav		
Inspectors	Aditya Dhingra, Yash Shiroya, Shreyaansh Baasi		
Recorder	Rahul Pai, Devaunsh Sambhav		
Defect #	**Description**	**Severity**	**How Corrected**
1	Status of Item for displaying (lent/bought or available)	1	Added key to items table for checking status
2	Association of items to users so that buyers can contact the sellers directly using their emails	2	Added foreign key to items table for users email
3	Item Description storage storing in database, too complicated to store it together and then split it into components afterwards	1	Decided to split description into table keys and combine in UI
4	Item Posting on separate page too much re-rendering of the page for a single page application	1	Placed button for new post on same page and made app single page application
5	Posting Items for rent vs posting items for sale is currently not distinguished	3	Created status for items to show if they are for rent, for sale, or when they will be available
6	Item waitlist will be ineffective because it should just be first to rent when it becomes available	3	Instead of having a waitlist we will have the date when it is available again and send push notifications when it is

			available
7	Contacting/paying sellers for items when the items have not been designated to a user	1	Each user will have an itemlist where all the itemsIDs of items they post will be. This is so that they can be matched to the users venmo and email
8	The file that holds the redux modules connection settings (all dependencies) should be in a separate directory/folder and not in the file where the setup of the application is	2	The npm tools and other dependencies were put in a folder called reducers.
9	Database in SQL for storing the user password and username was not effective because we would have to be secure in storage of the password.	1	Firebase allows for easy authentication through things like Google and Facebook and allows for easier local sign in
10	Files being bundled were being added every git push which was causing problems when pushing to github or when some files were not to be pushed	3	Added a bundler and the bundle.js file to the .gitignore so to not push files everytime but still be able to properly test
11	The file containing system specific commands and functionality should be in a separate directory to avoid unnecessary exposure, create maintainable design structure and to avoid confusion	1	The system functionality containing files were put in a new directory called actions.
12	Project structure for react cannot have all of the js finals be in the general space. They are otherwise not read properly and cannot be imported	1	Broke down into folders such as actions,

	correctly.		components, reducers, containers, etc.

Code Inspection Defect Log

Product	Lendr Code Inspection Defect Inspection		
Date	September 18th, 2017		
Author	Rahul Pai, Devaunsh Sambhav		
Moderator	Aditya Dhingra, Shreyaansh Baasi, Yash Shiroya, Rahul Pai, Devaunsh Sambhav		
Inspectors	Aditya Dhingra, Yash Shiroya, Shreyaansh Baasi		
Recorder	Rahul Pai, Devaunsh Sambhav		
Defect #	**Description**	**Severity**	**How Corrected**
1	When the user is trying request a service that is not yet supported by the application the error received for the intent was null.	2	The error of null intent was gracefully handled by not re-rendering the page and giving no response.
2	Incorrect resizing of components based on window size of browser. Components were cut off and not re-rendering in the correct size.	1	Made the sizes based of the different components in the windows relative to the window size itself rather than a set size (moved to percentages in divs)
3	Incorrect utilization of props throughout the code which caused problems when fetching data from one module to the next when an action occurred.	2	Made sure to send props correctly in the render() function of each js file and in the different divs that we created for each module to ensure data was received.
4	Due to the asynchronous nature of React, problems occur when a setState is used before a query is finished that will assign a value to that variable or variable would be changed again before it could be used	2	Promises were used to represent the eventual result of an asynchronous operation. The object was used as a placeholder in which the successful result

			value or reason for failure occurs.
5	Tried to hold state in too many places caused problems with the interaction between modules. Caused re-rendering too often.	1	Only set state on updating the component and set state for just that component rather than continuously resetting state for the whole application.
6	Too much nesting of divs made it hard to test some of the components later on due to many props being used between components and hard to test divs individually.	2	Changed to using lists and row items in order to not nest divs frequently. In other places split divs up to make the code more readable and easier to test.
7	Description, price, name, of items being posted were not being required so items could be posted without certain attributes causing errors when retrieving these values as they would be null.	2	Set the default value to null and specified required values of items being posted
8	Item type or status could be specified as any value without an error, causing problems when fetching the item data to show the price. If the item was for rent the price should include the rate, otherwise the price should show as is.	2	Every item must have an item_type and that type must be specified as either rent or sale
9	After logging in, screen did not show any noticeable changes and it was hard to see if the user was actually logged in	3	After User had been authenticated, changed to logout and added name next to logout button
10	When posting the new item, pop-up box would stay on screen even when the "Item Submitted!" badge would flash, indicating the new item had been submitted. This also allowed for a new item to be submitted at the same time without clearing fields.	2	Cleared fields of the pop-up form for an item and allowed submit badge to show up, allowing the user to submit another item or exit the form and continue using the application.
11	Values were not being restricted when they were inputted into the pop-up box for posting items. Also the non-null	3	Switched to a form with restrictions of certain values and forcing other

	values were not being upheld.		values (description, name, etc.) to be non-null.

Product	Lendr Unit Testing
Date	09/17/17
Author	Rahul Pai

Defect#	Description	Severity	How Corrected
1	When not logged in, users are unable to see the available items until signing in	2	Instead of having entire app render upon authentication, available items now renders when the user enters the site, but post items is not active until the user authenticates
2	After logging out, if the user was in the post item form then it remains active and users are able to post items without the item requiring their userID in firebase DB “item_name”:”vacuum cleaner” “userID”:null	1	When user logs out, they are automatically shown the available items form and are then unable to access the post item form. Made sure database requires USERID on input of new item otherwise page would again redirect to available item form. *Correct:* DB “item_name”:”vacuum cleaner” “userID”:{email}
3	After logging in, user does not see any information from post items page or available items page until clicking one of the buttons. User may be confused if they are actually signed in.	2	Once user has been authenticated, they see the available items page in it’s entirety with all of the current posting on there.
4	If the user posts an item for rent, the button does not change whether the item is for rent or for sale. It just says buy for both items for rent and items for sale.	2	Made sure to check the item_type when re-rendering the available items page and have the button say BUY for items that are for sale and RENT for items that are to be rented.
5	Cards for items were not showing	3	Extended the card space for each

	seller's information for contact even though their userIDs were showing up in the DB due to improper allocation of space on the cards.		card to account for more pieces of the description so that price, description, name, and seller info could be seen on the card.
6	Items were showing up in a list format and were not interactive.	1	Created cards for each individual item to allow space for all of the components that were being fetched from the DB and being displayed for each item.
7	Fetching name and email from user once logged in	2	Did not implement a local authentication, instead have a Google authentication so all of their information is provided through Firebase API. This will also help for contacting the seller using their gmail account.
8	Tedious process to host Facebook services on Heroku cloud-platform for local testing	1	Decided to not use Heroku for middleware testing because multiplatform tunnelling is possible using other servers
9	Unable to connect to a public endpoint (i.e. internet) using a locally running network service	1	Used ngrok to host a server on which we run Lendr
10	User unable to verify if logged in and what account is logged in	2	Added users name and picture to the top right corner next to logout getting information from Google account
11	Items for rent did not contain a rate at which they are to be rented Vacuum Cleaner Price: $5	1	Created "item_type" for every item which will give the status of every item. Required a "rate" for the items to be rented. *Correction*: Vacuum Cleaner Price: $5/hour DB Rate:"hourly" "item_type":"rent"

12	Input fields of the form that a user fills to post an item were not mandatory	2	The code of the form was fixed to make sure that certain fields are mandatory.
13	When an item was posted to rent/buy it was not being simultaneously updated on the page of available items	2	Corrected the code to fetch the data from firebase and display it on the available items page right after a post was made.
14	User could enter anything in the field of Price on the Post item page.	2	Re-designed the code of the form to make sure that the price user enters has a proper format.
15	Price was being shown as just a value without the $ because the price was saved in the database as just the number.	3	Manually added the “$” to the price when fetched to the database before displaying it on the card.