GitHub Link: <https://github.com/GarMoore18/HomeInventoryApp>.

**Week 1**

At the start of this week, I began to learn to use Android Studio. I also created a rough flowchart of what I want the application to look like at the end of development. There are a few ideas that I will need to iron out as I learn to better utilize Android Studio.

**Week 2**

At the start of this week, I created the home navigation bar, added icons to the navigation bar, and a temporary theme. The way I alter the theme needs to be placed into a styles.xml file. I also created the fragments for each of the navigation bar tabs. I will do the java side the next time I do some work.

I worked on moving to a new page when a destination is selected in the navigation bar. After some trouble, I found that I should have used activities instead of fragments. I now know the difference between the two. Each of the icons load an instance of the corresponding activity. I also worked on formatting the bottom navigation better and correcting the issues that I had with the coloring of the bar. The bar now has the overall theme and I also was able to explore some of the alterations I can make to the theme.

I learned some about the layouts that are available to format the activity screens. I used the constraint layout to add some of the text inputs on the add screen. There is probably a better way to have multiple inputs, but it works for now. I will add the title, submit, and camera options to the screen next.

**Week 3**

At the start of this week, I worked on the layout for the add and remove screens. I was able to utilize the information that I learned about layouts last week. The screens now have constraint layouts and a navigation bar. I initially wanted a floating navigation icon because a bottom navigation bar could get accidently pushed when confirming. I move the navigation bar to the top and removed the title to combat the issue. Next, I will work on finishing the layout for the search screen and the settings. After that I will do the database design.

I continued to work on the layout and set the search and settings screen layouts up. I realized that I was repeating a lot of code for the navigation bar. In week 1 I found that I had an easier time using activities over fragments, but now realize that fragments are better for my needs. There are only five pages and the navigation bar controls them. The overall responsiveness of the application was also quicker when using fragments.

I added an active color to the navigation icons. I did some light reading on what I want to do for a database. The biggest thing I did today was start with the camera implementation for the barcode scanning. The documentation for the zxing scanner was very helpful and I believe that I have that coded correctly. The issue that I have is that I am looking for the button in the main activity, but the image button is in fragments.

**Week 4**

I finished reading about which type of database I should use with an Android application. I think that I will use a MySQL server and store the data on the device with SQLite. This will allow all users to alter the database when connected to the network. The device database will be used when the user is not connected to the network but will still be able to search items. The plan is to have the application grab a copy of the database when the application is opened and/or closed to keep the device data updated.

I started with resolving the issue that I had with the camera implementation. I initial was using an activity to control the camera. After some further research to learn how to implement the camera with fragments, I am now using IntentIntegrator. This implementation allows me to only open the camera once the user navigates to the fragment. I then added the onClickListener to ensure the camera only opened when the user clicked the scan button. The final thing I did to finish the camera was change the orientation to portrait mode. In addition to the camera, I also created a structure for my database. The plan is to have four tables; items, locations, users, and one to store item/location/user key and the time/date. I think doing it this way will reduce redundancy and allow displaying recent items on the home screen to be easier.

**Week 5**

I had an emulator issue that was fixed by cold booting the device. Did not help progress the project but is good to know. I set up a login screen that I will use later and is currently not in use for testing purposes.

The biggest part that I did this week was setting up the computer that I will be using for the server. I downloaded MySQL onto the computer and set up the remote permissions, so I am able to access it from my computer. In addition, I installed the necessary MySQL items. There were so many issues with the first installation for an unknown reason and I had to reinstall. The second time installed correctly, and I was able to set up a server to ensure it was running. I also did some refreshers on how the MySQL syntax works and.

**Week 6**

Using the database design that I planned out, I created the database and the tables. Setting up the tables independently went smoothly except I had some issues with foreign keys. It ended up being a stupid mistake. The other two things that were a struggle was correctly implementing the TIMESTAMP and the blob. I had some settings wrong for the column that was defaulting both to int values.

After finishing the tables, I tested inserting data to the tables. The only one that I had to alter was the combined table because it was allowing me to add foreign ids that were not defined. I checked and had forgot to add one as a foreign key.

I still need to work on storing the passwords securely and how to access the database from the android application.

**Week 7**

I attempted to connect to the database from the application using JDBC, but I needed an intermediate service. After researching, it would be best to utilize a web intermediate. This is done with apache and PHP, which I did not have setup on the server computer. I switched to download XAMPP and phpMyAdmin because I have used it before and it comes with the items that I need to connect to the database through the application.

Setting up the database in phpMyAdmin was quick and did result in any issues. Being more familiar with XAMPP, I was able to alter user privileges and do not need to remote in to see the database anymore.

**Week 8**

Spring Break

**Week 9**

I was able to create a PHP file that will allow me to connect from the application to select all form a table. I initially was using the HTTPClient, but had one item that was deprecated. After reading through countless tutorials, I would not be able to use the HTTPClient because support stopped at 23.

I have used volley in Phone Development, so I attempted to use that. I first started by using POST to try to get a certain barcoded item from the item table. It worked, so I put that on the on click and wanted to autofill certain fields if something was returned. I am reworking the add/remove/search fragment to have just the barcode field appear. The user must enter a valid EAN13 barcode and then hit the fill button. Based on the response, the fragment will change to new item or update item.

I got the fragments set up and can pass data between fragments, but there is an issue because of the asynchronous nature of volley. The fragments are supposed to change based on if an item is found or not. I am doing this with a try/catch block. If I can create a JSONObject then the screen would change to the update, else the screen changes to the new item. The issue is that the JSONObject tries to be created before the response is returned, thus always resulting in the new item screen.

I started to investigate synchronous volley, but it appears that I will run into crashes because of the stress placed on the main thread. Putting stress on the main thread causes the UI to freeze and eventually the application crashes.

Using async will work in my case. I added more loggers to catch more of the errors and it turns out I had not declared a POST variable in my PHP file. Simply defining the barcode POST variable fixed the issue and I can now correctly add items to the item\_info table. I attempted to start another request but was unsure where to request next with volley.

To change pace from the volley requests, I worked on ensuring that the user will have to fill in all required fields. I am doing this with the setError for textView. When the button is clicked, the text fields are checked if they are and empty string or not. A boolean variable is used and if any required field is not met, the boolen is set to true and the click event will be ended. The locations field is not a text field and needed to have preset locations so the database does not have instances of similar names (e.g. Downstairs vs downstairs). I created a spinner to do this and then created a custom spinner adapter to allow the spinner to have the locations pulled from the database. This allows me to not hard code the locations into an array since locations could be added later. A temporary error check for the spinner since there is not a setError(), was to add a ‘Select Location’ to the array before adding the adapted database items. This allows me to check if the user has changed the location to a valid database location.

After finishing the required fields, I moved back to the volley requests. I knew I needed to have three requests for a new item. These are a POST to the item\_info to add the new item, a POST to the item\_info to get that item from item\_info, and then a POST to the combined\_info table. I initially had these listed on the same level, but this did not work because the third response would finish before the second response. Instead I nested the requests in the onResponse() of the previous request. This causes each request to wait until the previous request had responded.

A small issue that I need to fix is the fragment stack when returning to the previous add page results in overlapping text. This should be a simple fix since I have done it on the previous page. The bigger issue is that there is an issue with running the application on a hardware device. The volley error is that there is noConnection to the server which is strange since I allow access from all and the emulator works. Most solutions state that changing the ip from localhost to the server ip will work, but it does not in my case. I also added a HttpTrustManger class which is a better way to accept the data from the emulator.

**Week 10**

To start the week, I needed to fix how I was querying with PHP because I was not using prepared statements. After reworking all the files, I also moved the files to separate folders within where they are stored so each file is easier to find.

Next, I had issues with how I had set up the database. I ended up adding another user that only has certain privileges in the inventory database. After fixing that, I added the list view to the home screen. I started to work on the remove screen and nearly finished the function, but there are a few issues that I need to fix, one of which is getting the max quantity that a user can remove based on the location that they choose.

To fix the issue of location quantities, I added a custom request and then store the information in a item object. Depending on the location then, the quantity field will display an error message with the max quantity allowed and will only allow numbers less than or equal to the max. There are also error messages that pop up if the fields are not filled.

**Week 11**

All of the remove screen functionality now works, and I have moved the fragment transition to a public method in the main class since I use it in every class. I also altered the add fragment to more closely match how the remove fragment works with volley.

Next, I did the search fragment buttons which took some time to figure out how to use view flipper within a fragment. I also moved the spinner adapter to a public method since I use the same spinner at every spinner instance. When auto searching with a barcode, the list view shows. The manual is set up but has no function.

At the end of this week I wrapped up the issues with the PHP prepared statements and found some errors with the add screen that were resolved by moving some of the code into smaller functions. The manual search screen works except for the autofill text.

**Week 12/13**

Not much has changed. I worked on adding the ability to show images on all the screens, so the user can see what item they are adding or removing. There is an issue with some of the images being sideways.

I fixed an issue with the login screen for week 13 and the rotated image issue is fixed. SQLite is now implemented and will be used for autocompleting and loading the correct spinners items. I also saw some issues with the layout on the hardware device that I will fix. The camera sometimes still rotates images but appears to be an issue with the android version compatibility, not the code. The passwords are now hashed instead of simply having a string password stored in the database.

**Week 14**

This week was dedicated to cleaning up the code that I have and looking for errors. I tested the app myself and let others play with the app. The main thing I wanted to change with the cleaning was to have less in the on create method and have dedicated methods for specific functions. This then made the code easier to comment and understand what I was doing where.

Some of the issues that were found and fixed were more duplicate removing, spamming the confirm button (volley issues), and the camera crashing when the back button was hit. All of these were addressed and should no longer be issues.

The issue that is the most complex and still appears to have issues sometimes is the rotated images. I tried to use the new interface for rotating images, and it works on the emulator and my phone. The other older android phone that I am testing on cannot use that interface because of the version. Depending on how the camera is held causes weird rotations for the old device.

**Week 15**

The final week I tried to implement some other features. There is now a register screen that works. I also wanted to change the theme of the app in settings with a switch. I had trouble getting it to work since I am using fragments. I then tried to get it to simply change the color of the navigation bar. It works, but once the fragment is changed the color resets. The switch currently has no function since I do not want it to cause issues elsewhere. The user can also logout from the settings tab and there is a tutorial that the user can visit in the readme.