\*whatever you do don’t work with bruce

\*To do

-(boys)mailbox feature will be done by the boys

-(boys)favorite items stoped working(might be the class name in the empty container in the ejs file)

-(only if it looks bad and only after presentation) fix the section of the website you are going to present

\*Presentation

**-Use earphones**

**-use stop watch**

**-unmute**

-**welcome to Buyer,** here you can buy and sell anything you want, just keep it family friendly please.

-**Home page**, register

-I’m going to pass it on to Mike now to show you some other great features

\*Stretch

+README file(inform in readme file that the db server needs to be started, and talk about password etc)

+mouseover for buttons

+ Clean up (delete and lint)

+show a page with all photos of featured item

+Page appending for: user already exists, empty Field(s), incorrect password, user not registered

+we should have options to delete and add photos after listing has been created

\*database

midterm

\*repo

<https://github.com/Desyn6/LHL-Jan23-Midterm>

\*Use: “npm run db:reset” run this each time there is a change to the database schema or seeds.

npx kill-port 8080; npm run local

\*Folder and file brakedown:

-bin: has a js file that resets the database, it uses fs module to read schema and seed files the console log the correct code to reset the database using those files. In package.jason there is a script that enablers you to type “npm run dB:reset” to trigger this program

-db folder holds schema, seeds and query(js functions to query stuff) folders. there is also a file named connection.js, this file starts a database connection using the pool module

-lib:has a sass.js file this is some kind of framework for css (ask denny)

-Public

\*\*+scripts: This folder handels event listeners and appends(injects HTML using ajax), based on the events, however the rout for this comes from the folder routs file users-api.js. So this code doesn’t have the app.get, the users-api runs modules from this file inside it’s own router.get. The routs can be traced back in srever.js under the .use modules where the rout is started, the the sub rout is handeled in users-api.js then the listener function and the appening is handeled in the public folder in the file users.js: this handles event listeners not related to the message app

App.js: this will Handel Ajax and jquery and appending for the Messenger app

+styles: CSS stuff there is a .gitkeep (ask denny)

+vendor: what is this??

-routs: (routs should Handle .ejs)

users-api.js: this file contains the routs that involve listeners and uses Ajax and jquery and HTML injections coming from public/scripts/users.js

users.js: this file handels routs that request info from the database regarding the user

widjests-api.js: this file handels apps small apps that run in the site like the Messenger app (this interacts with the file in public/scripts/app.js)

-styles:

This contains .scss files (ask denny)

-views: this folder was created by denny it should be in public

\*Merging:

\*\*git branch shows what branch you’re in

git checkout dan

git add .

git commit -m ‘’

git push origin dan

git checkout master

git pull origin master

\*\*git branch <create new branch>

git checkout dan

git merge master

solve conflicts(in the console you’ll see the files that you need to open in vscode)

git ad .

git commit -m ‘’

git push origin dan

\*\*you can try again just in case(if you do repeat previous steps): git merge master

in the main website click pull request

then wait for admin to accept

\*For finals

\*One way to divide work is to in back end and front end. The issue here is that if a student is shit either back end will never be complete or front end will look like crap

\*another way of dividing work, it might take longer, also more conflicts, is to create db and a shitty front end then do back end together on top of the shitty website, then do the front end together.