StationX
Deployment/Installation Instructions

Lily Trissel, Lavani Somesan, TerriLynn Hale, & Nash Henry

Capstone Project Fall 2021

The University of West Florida

12/3/2021

CIS4595 Capstone Project

Dr. Bernd Owsnicki-Klewe

Table of Contents

1 Project Setup

1.1 Environment

The environment that we used to create and run this application is the latest Ubuntu Linux OS. Though, any Linux environment should be able to run this application. If you don't have a Linux OS, you can either download VMware or VirtualBox which will allow you to create a virtual machine of a Linux OS. There are multiple guides online on how to set up a virtual machine if you need help doing so. For this project, we used VMware to host our Ubuntu Linux OS depicted in Figure 1.

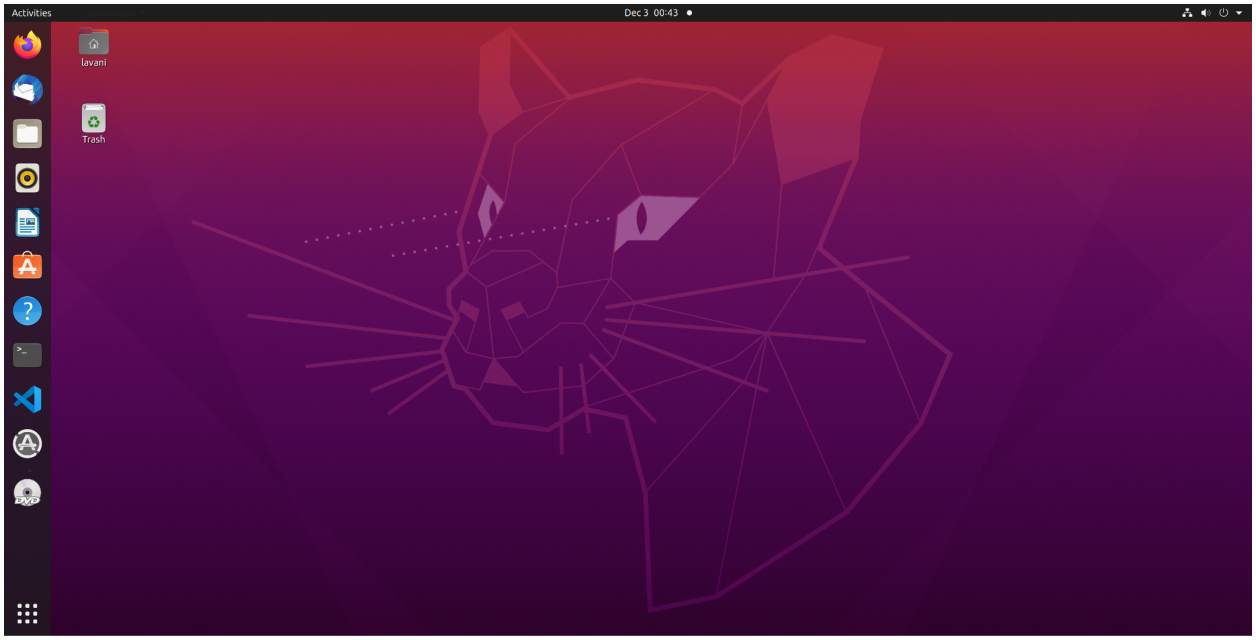

Figure 1. Screenshot of Ubuntu Linux OS hosted on VMware

○ 1.2 Code Editor

For this project, you will need a code editor to be able to view, edit, and run the code. We chose Visual Studio Code (VScode) because we are familiar with it and NodeJS works well with VScode. If you want to use other code editors you would need to install the correct dependencies to be able to run it, therefore, we highly recommend you use Vscode as the code editor for this application. An example of the VSCode environment is depicted in Figure 2.

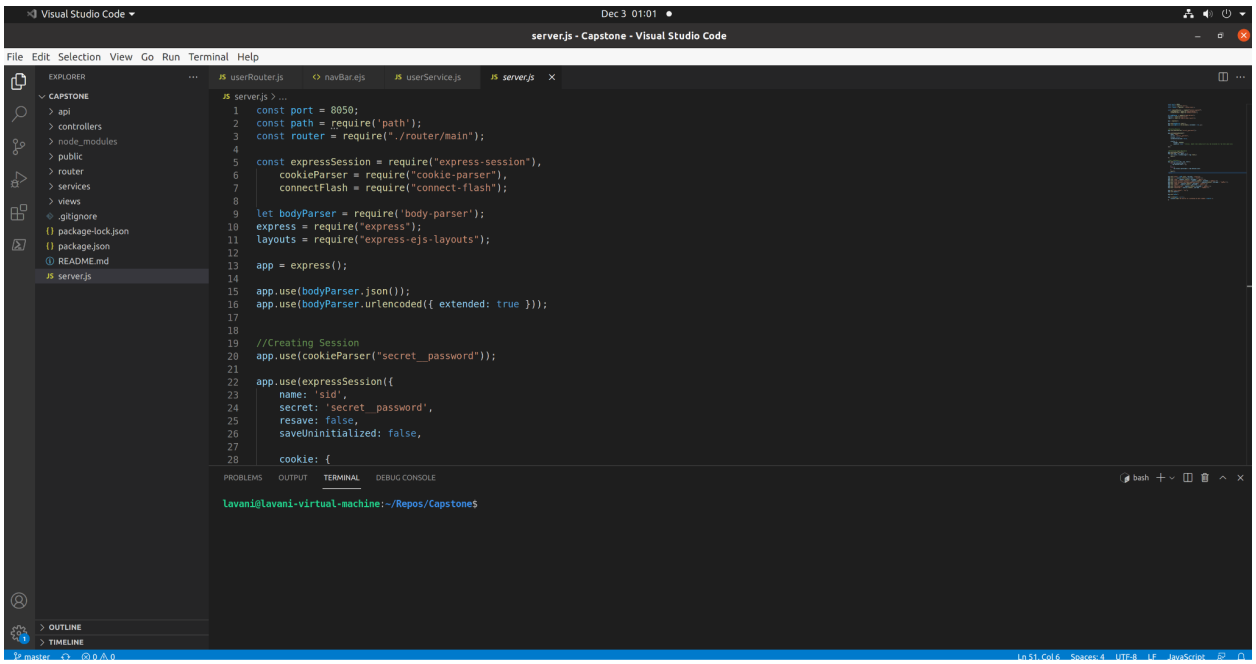

Figure 2. Screenshot of Visual Studio Code Editor

○ 1.3 Dependencies

Our application depends on NodeJS and Node Package Manager (npm). You will need to have these both installed through your Linux Terminal. There are various guides online on how to download and install them. The guide that we

used to first implement them can be found at:

https://www.geeksforgeeks.org/installation-of-node-js-on-linux/

○ 1.4 MongoDB

For this project, we used MongoDB as our database which stores our users, inventory, and newsfeed. When you run the web application, you will be connected to the database so there is no fancy setup in trying to get inventory, login, etc.. Though, if you want to have access to our database you will first need to have a MongoDB account, and second, request access to our software lead, Lavani Somesan, either via email or discord. She is the database creator so only she can add or remove people with regards to access to our application's database. An example of the MongoDB dashboard is shown in Figure 3.

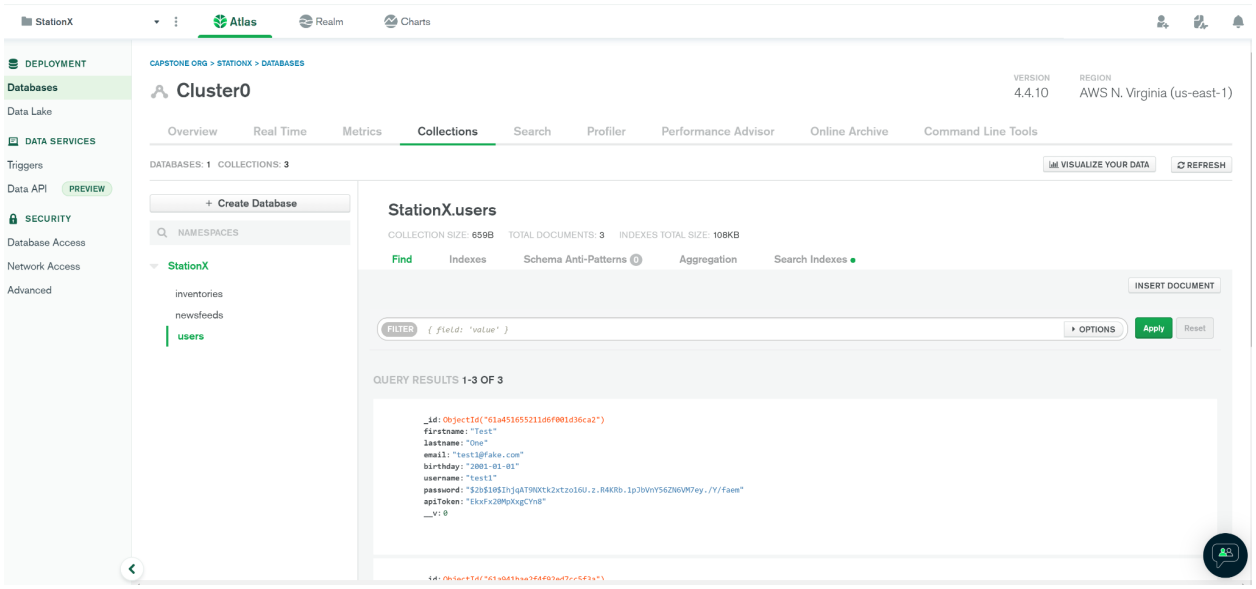

Figure 3. Screenshot of StationX Database on MongoDB

2 Project Installation

2.1 GitHub Account

You will need to have a GitHub account that has access to our project repository. If you have access, you should be able to clone the repository by hitting the Green Code Dropdown list button and copying the HTTPS link to your clipboard. This process is shown in Figure 4.

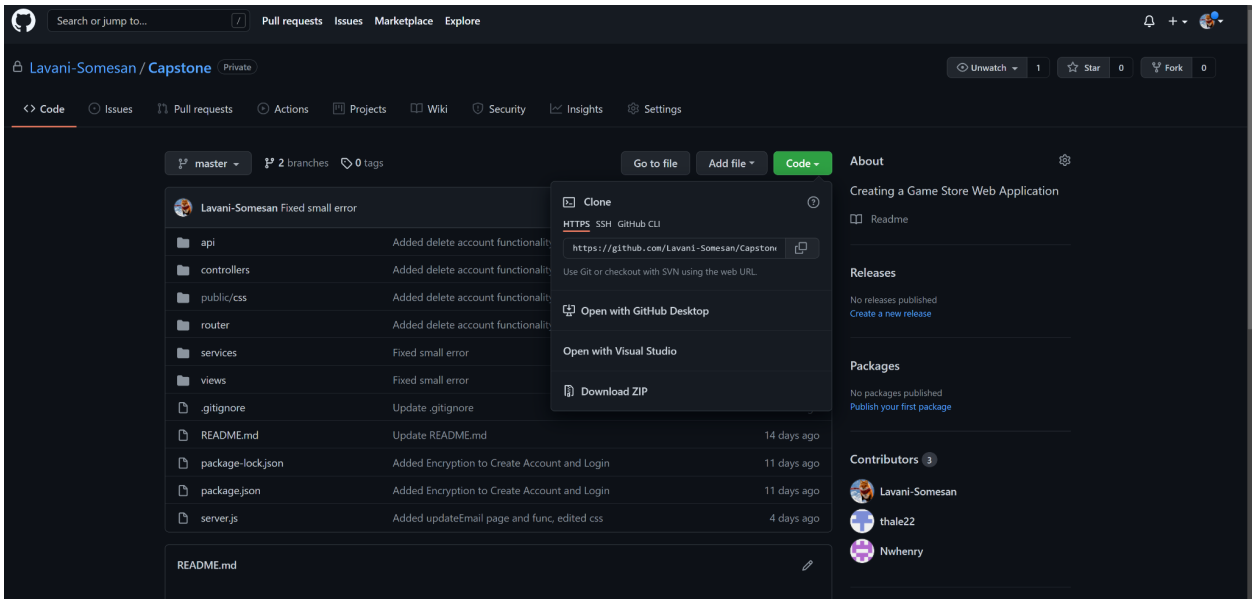

Figure 4. Screenshot of GitHub repository

2.2 Cloning Repository to Code Editor (VScode)

As said in our code editor section, we chose to use VScode for this project. If you want to run this application we highly recommend using VScode. First, you need to click the Source Control button on the left panel (highlighted in white in the image below). You will then select Clone Repository and copy the repository link into the textbox at the top. VScode will then prompt you to choose a location

for the repository. Once you have made your selection, VScode will clone the repository and load the project into a new VScode window. An example of the VScode screen where you can clone repositories is shown in Figure 5.

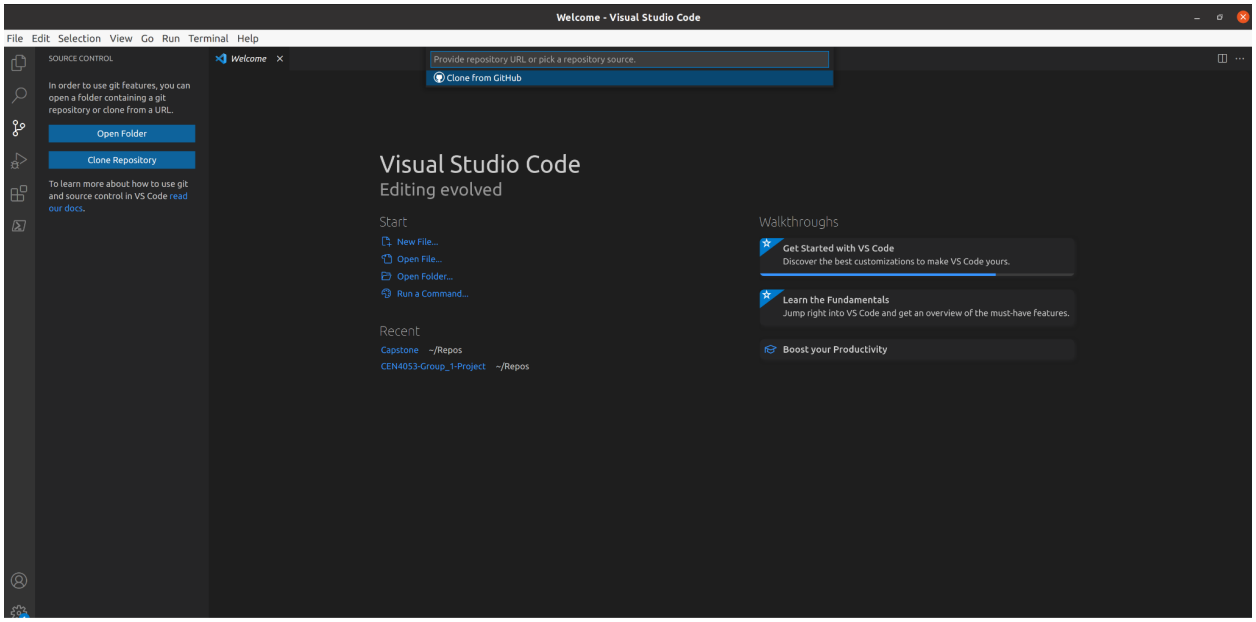

Figure 5. Screenshot of Cloning Repository

- **2.3 Final Touches**

Before running the application, you will first need to do a “git pull” to make sure everything is up to date (If git pull fails for some reason, do git pull origin branchName same rule applies to push, git push origin branchName). Then, you need to enter in the following command, either “sudo npm install” or “npm install”. This will pull all of the project dependencies that are being used for this project into your project. I recommend doing an npm install whenever you do a git pull otherwise you might have issues running the application since it will let you know it cannot find a certain dependency.

3 Project Deployment

3.1 Running Application

For this application, you can either run the server, API, and services (user, inventory, and newsfeed) separately or together. Again, remember to do a "sudo npm install" or "npm install" after doing a git pull. Once you've run the application either together or separately, you can navigate to "http://localhost:8050/" in the web browser to access the web application. From there, you can use our web application to your heart's content. Down below is how to run the application either concurrently or separately.

3.1.1 Running Components Concurrently

To run components concurrently i.e. server, API, and services (user, inventory, and newsfeed), you must type the command "npm run dev" in the VScode terminal which will then run all the components together. Note, this will not work if you have done an "npm install" since we are using the npm dependency, concurrently, to run everything together.

3.1.2 Running Components Separately

To run components separately i.e. server, API, and services (user, inventory, and newsfeed), you must first open up a separate terminal for each component. So, you need one for the server, one for the API, and one for each service.

Terminals:

- Terminal 1: Do command “npm start” - Starts the main server
- Terminal 2: Do command “cd api” and then do command “node main.js” - Starts API server
- Terminal 3: Do command “cd services/userService” and then do command “node main.js” - Starts User Service Server
- Terminal 4: Do command “cd services/inventoryService” and then do command “node main.js” - Starts Inventory Service Server
- Terminal 5: Do command “cd services/newsService” and then do command “node main.js” - Starts NewsFeed Service Server

Technically, you can run our application with only the main server being run but you won’t be able to use any user functionality such as login/create an account, etc, or view any inventory unless you have the API server and service(s) running.

3.2.1 TextReference Error & Fix

If you get an error “Reference Error: TextEncoder is not defined” or “.../Capstone/node_modules/whatwg-url/dist/encoding.js:2 [3] const utf8Encoder = new TextEncoder();” then this is an error with older versions of node. There is a simple fix for it.

- Navigate to the folder node_modules
- Inside node_modules navigate to the folder whatwg-url
- Inside whatwg-url navigate to the folder dist
- Inside dist click on the file encoding.js

- At the top of encoding.js you will see:

  “ **const utf8Encoder = new TextEncoder();**

  const utf8Decoder = new TextDecoder(“utf-8”, {

  ignoreBOM } : true); ”

- You will add this line above the previous code:

  “ **var util= require('util');** “

- Then you will edit the code like so:

  “ **const utf8Encoder = new util.TextEncoder();**

  const utf8Decoder = new util.TextDecoder(“utf-8”,

  { ignoreBOM } : true); ”

This should fix the problem of the TextEditor not being defined.

- **3.2.2 Database Error & Fix**

  If you are able to run the application but are unable to do anything with the database this means that the database is paused and needs to be unpaused. You will need to contact the owner of the application to do this.