CNG 495

Fall – 2022

Term Project Progress Report

I

Ibrahim Ozkan 2456275

Adil B. Kebapcioglu 2455954

1. Milestones Achieved

1.1. Week 1 - October 31 - November 6

After getting approval for the project, I started making research on the cloud technologies that I will use to deploy our project. I made a research on setting up an instance for AWS EC2 server with desired Linux distro which is Ubuntu 20.04. Also after making research on setting up an instance, I had to search for Nginx installation to Ubuntu with PHP 7.4 version since we are planning to host our RestAPI in the EC2 instance. Lastly, Composer needs to be installed in order to run Laravel on the machine. According to my research I listed each installation steps for AWS EC2 cloud utilization below:

Creating an EC2 Instance

1. First you need to create an account on AWS with your credentials. You also need to provide your credit card details before registering. AWS will not charge anything to your credit card, it will just only verify your credit card only by charging 1$ (it will be given back to you after registration). [You can visit registration page by clicking this text.] The Figure 1 demonstrates initial registration screen of the AWS.

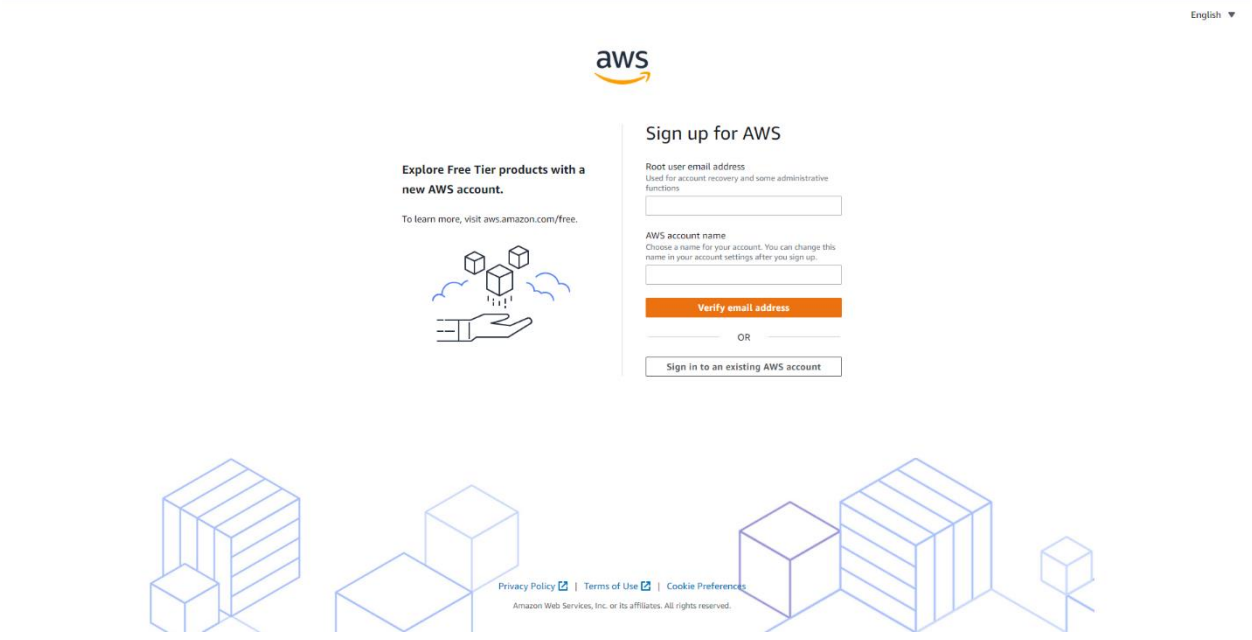

Figure 1 AWS Registration Screen

2. After registration, your account needs to be approved by the AWS which can take up to 24 hours. Mine took only 10 minutes.
3. When approval to your account is done you will be directed to your console page. Here type “EC2” to the search box on the page and click on “EC2” which is also shown in the Figure 2 and you will be directed to Dashboard of EC2 management.

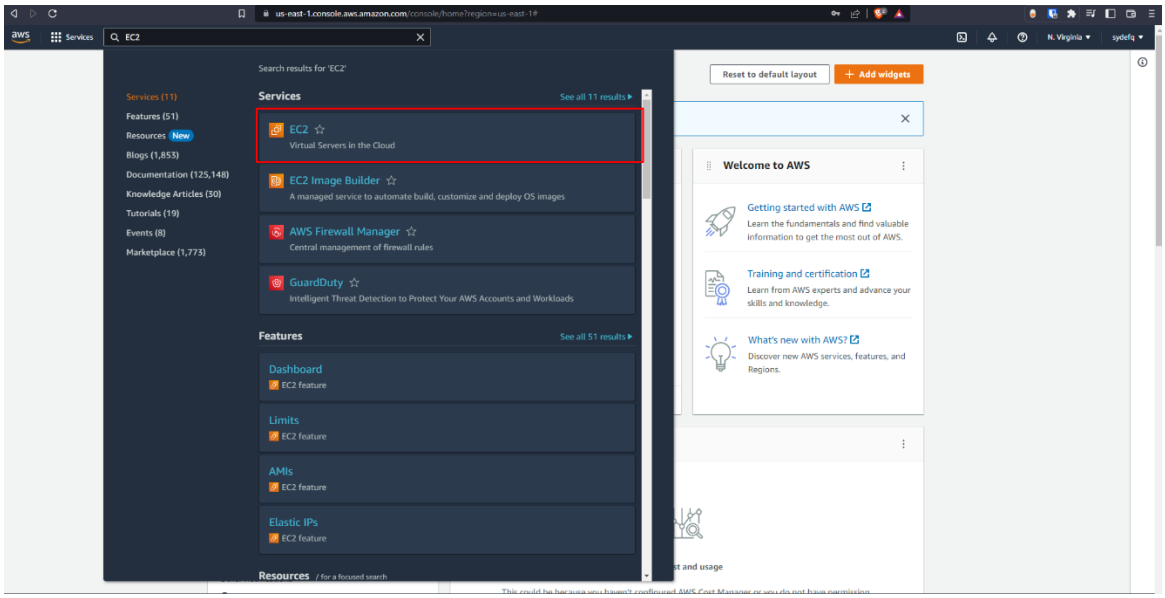

Figure 2 AWS Console Page

4. Here inside Dashboard page, to create a new instance click on Launch Instance button as shown in the Figure 3.

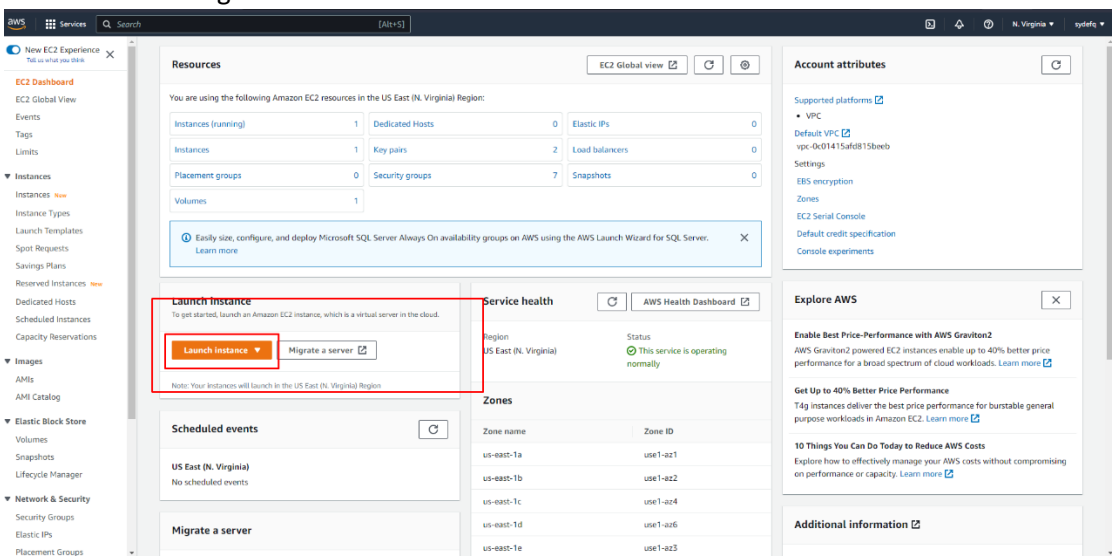

Figure 3 EC2 Dashboard

5. In this page you need to fill all necessary fields to create an instance.
    - “Name and tags” field needs to be filled which will be the identifier of your server. Lets name it as “CNG 491”

- “Application and OS Images” part is to decide which OS will run on your system. We will select Ubuntu for desired OS and select “Ubuntu Server 20.04” as distro version. Architecture will be selected as “64-bit (x86)”
- For instance type we will be selecting any of the free-tiers to use free version of the EC2. If you want greater resources for your EC2, you can select paid instance types. Mine is “t2.micro” with 1gb memory, 1 vCPU.
- Key pair field is necessary for SSH connection you can create your key pair and name it whatever you wish. Lets name it as “CNG491”.
- Network settings are necessary for setting up some rules for incoming and outgoing traffic for EC2. To host a web server, select Allow SSH traffic, Allow HTTPS traffic and HTTP traffic.
- Finally, we need to configure our storage. AWS provides free storage up to 30gb. Lets use them all and configure our storage as 30gb.

After you fill all fields page should look like Figure 4, 5 and 6.

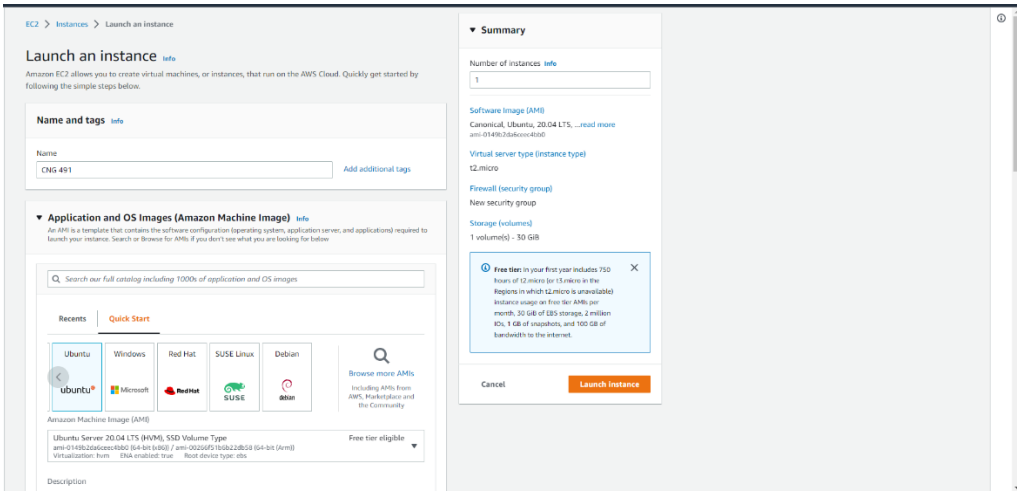

Figure 4 Instance Launch Page 1

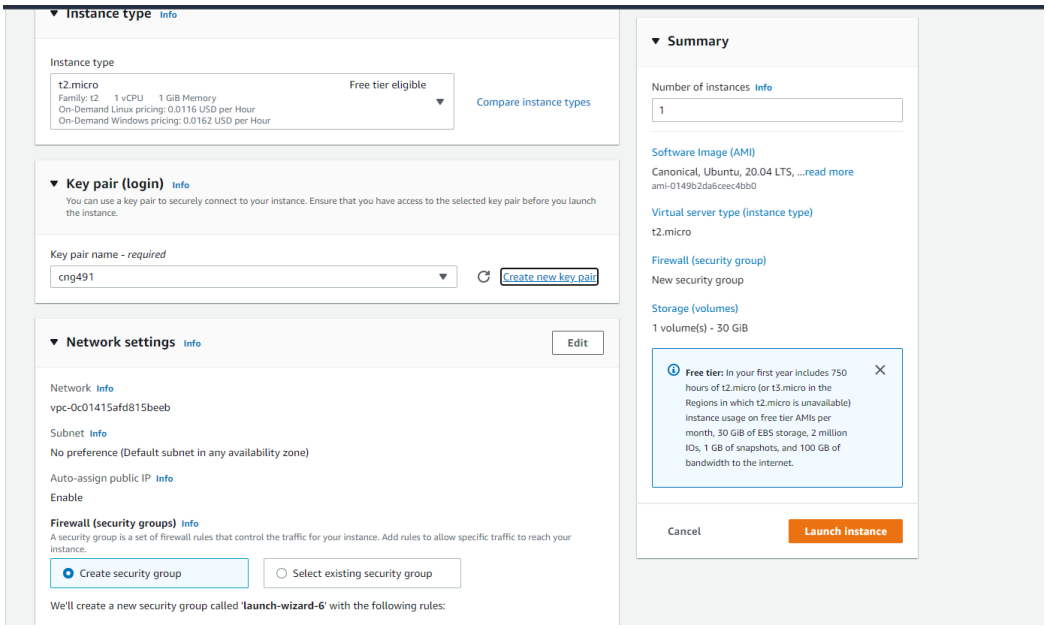

Figure 5 Instance Launch Page 2

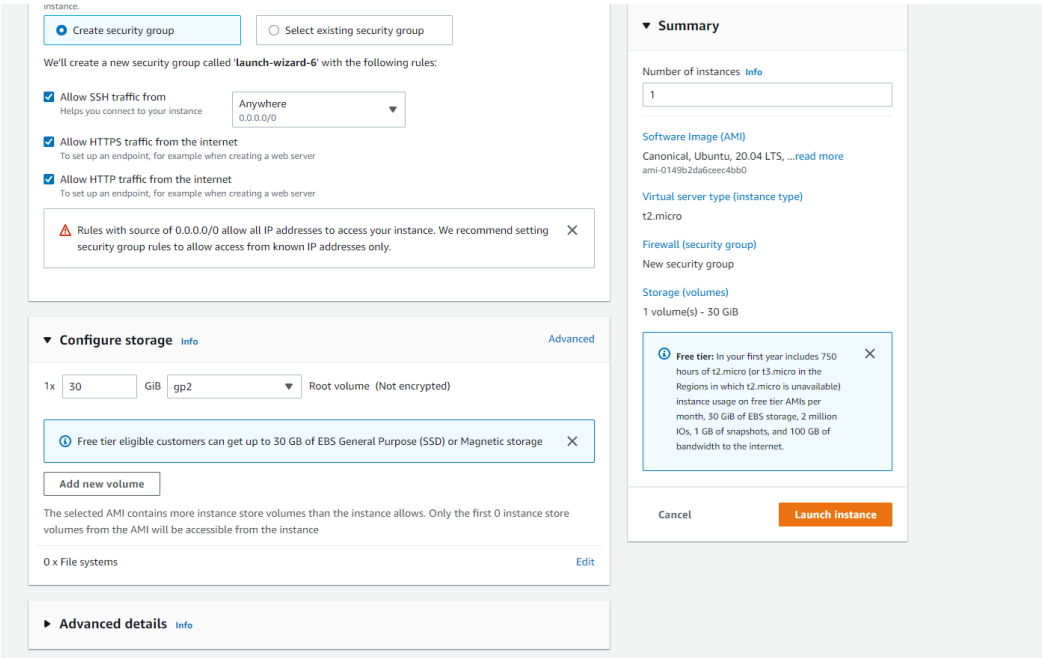

Figure 6 Instance Launch Page 3

6. Click on Launch instance to create a new instance.

1.2. Week 2 - November 7 - November 13

After making research on week 1, I followed the steps that I provided for creating a EC2 instance. After the creation, I connected to terminal of the EC2 instance to setup a Nginx server with PHP and Composer.

To connect terminal of EC2 Instance follow these steps:

1. When you are in your EC2 Dashboard, click on “Running Instances” which will direct you to a page where all your running instances are listed as in Figure 7

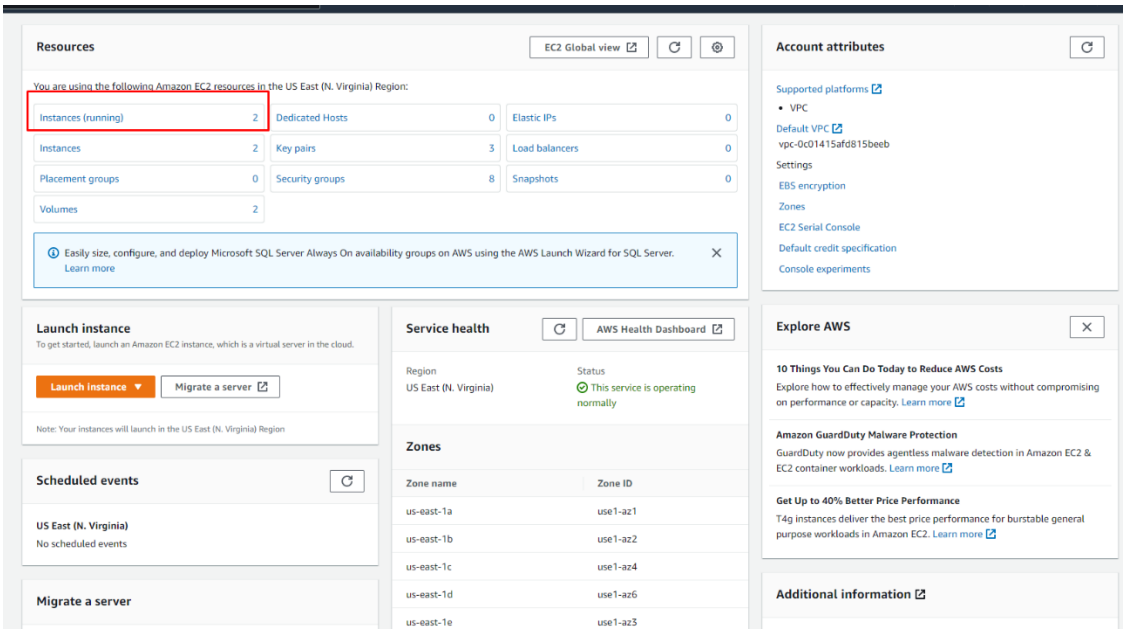

Figure 7 EC2 Dashboard

2. Then, mark the instance that you want to connect and click on the “Connect” button to be directed to the connecting to EC2 page like in Figure 8.

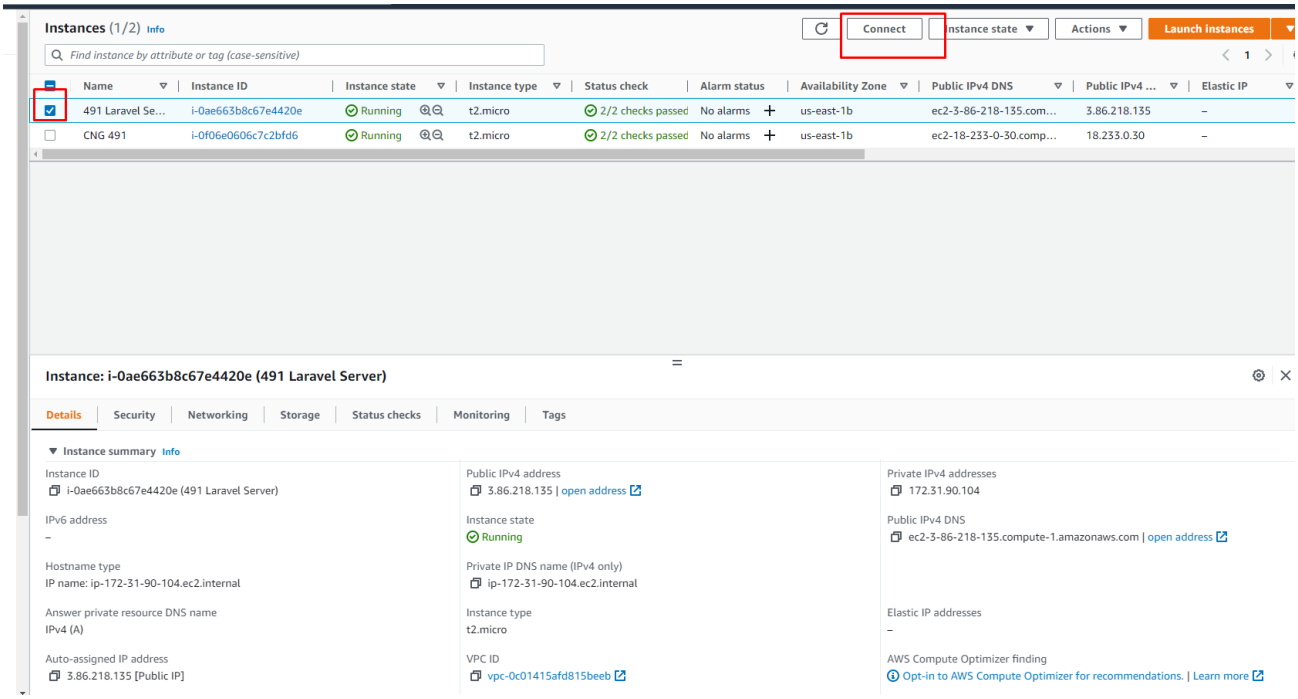

Figure 8 EC2 Instances Page

3. After that, select “EC2 Instance Connect” and press connect to connect to the terminal of EC2.

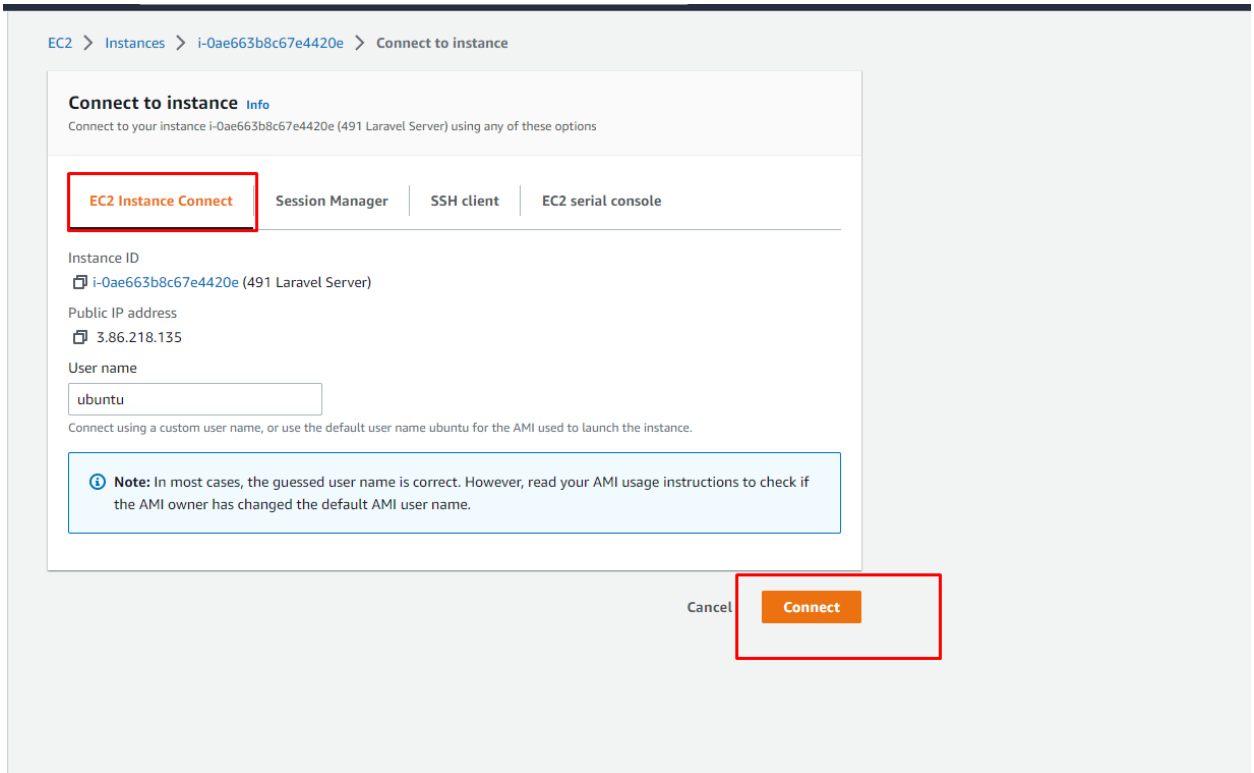

Figure 9 EC2 Instance Connect Page

For installation of Nginx, PHP and Composer, I followed the tutorial provided by the Rahul Gupta which you can visit the tutorial web page by clicking here (Gupta, 2020). I had difficulties in installation of Composer since it required to have different PHP versions for each Laravel project. For instance, if Laravel project is built with PHP version 8, Composer will give an error for any PHP version under 8 which ours was 7.4. We solved this issue by changing the “composer.json” file inside Laravel project to decrease PHP version requirement.

1.3. Week 3 - November 14 - November 20

Last week of our sprint was about deployment of the project to the AWS via Github. Also, I developed a simple demo application which involved showing current population of a in-door area with increment and decrementing functions of the population as well which more detailed explanation of mobile application will be done in later paragraphs.

Cloning our project to the Github was quite a challenge since Github disabled authentication with username and password for SSH connections. Therefore we needed to create an SSH RSA key for Github inside our server and configure the authentication.

To clone our private Github repository and SSH authentication with RSA key, I followed these steps:

1. First go to directoy of "/.ssh" by executing the following command.
   `cd ~/.ssh`
2. Then execute the following command with providing your Github email to create an SSH key. Press enter for all asked fields.
   `ssh-keygen -t ed25519 -C "your github email"`
3. After creating your key you need to execute the following command to get your SSH key which will be used in your Github profile settings.
   `cat id_rsa.pub`

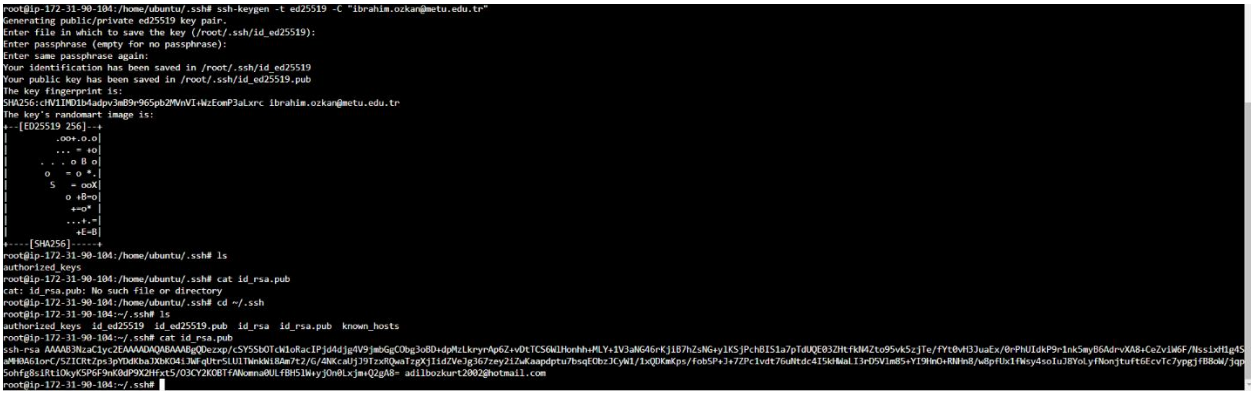

Figure 10 Ubuntu Terminal with Steps for Github

4. Copy the following key and open your Github profile settings, click on “SSH and GPG Keys”.

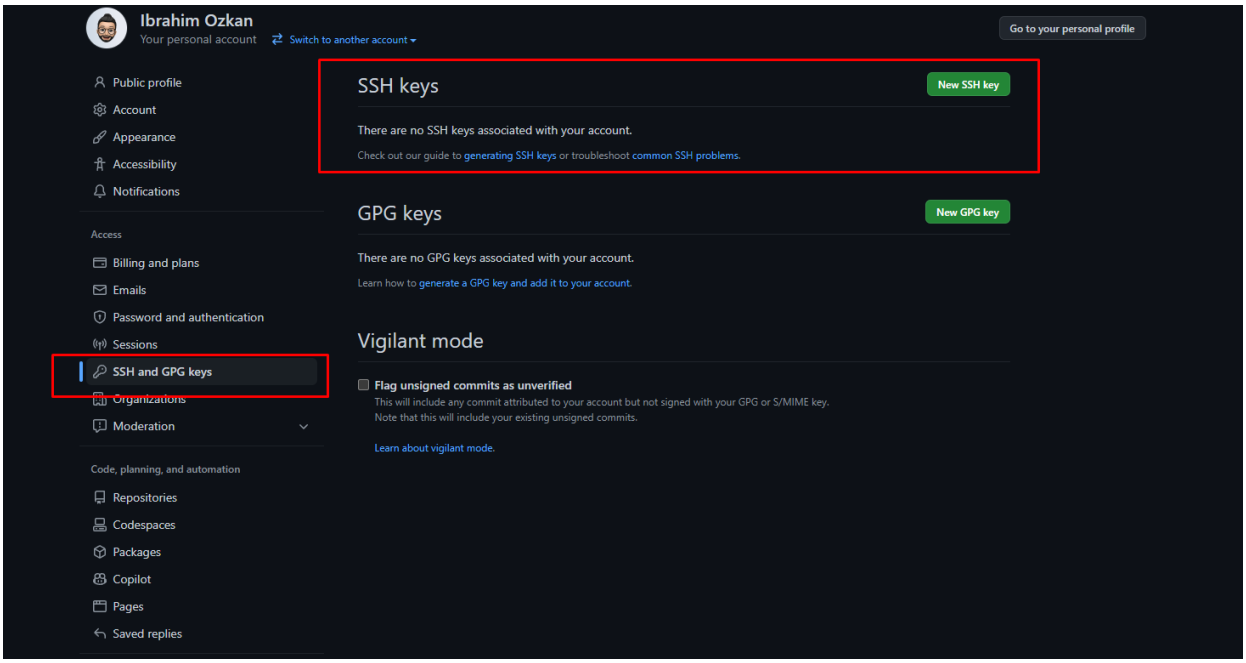

Figure 11 Github Profile Settings

5. Click on the “New SSH Key” and paste your SSH key like in Figure 12

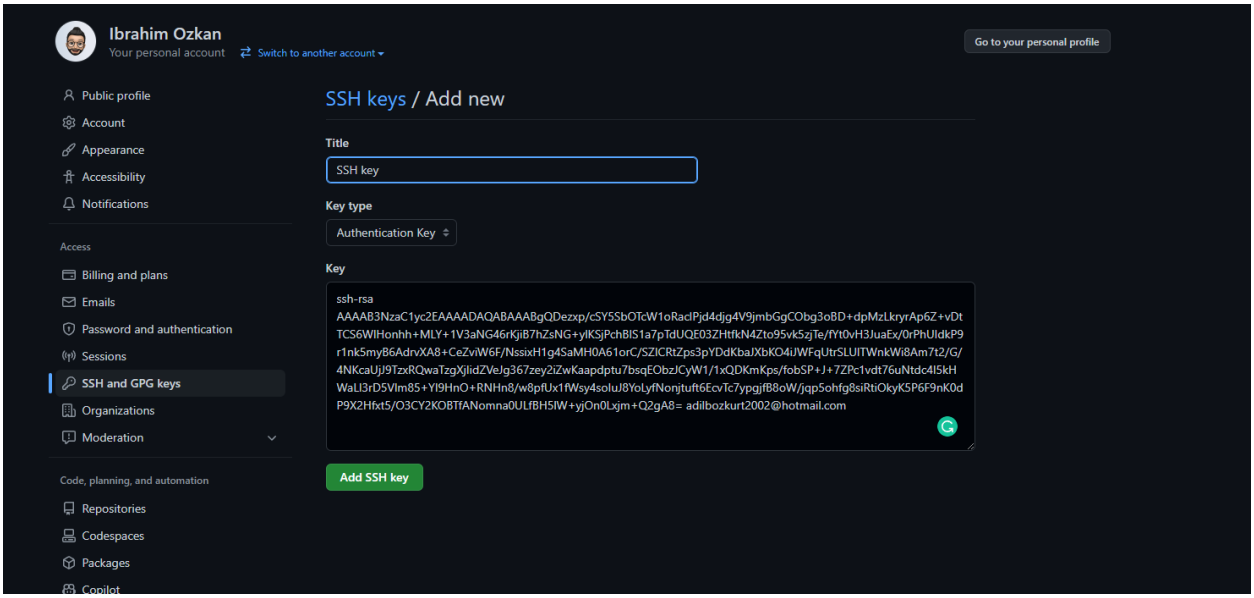

Figure 12 Github Create SSH Key Page

6. You successfully authenticated your Github account with your EC2 instance. Now to clone our repository go to directory by the following command:
   cd /var/www/html
7. Here, execute the following command to clone your repository successfully.
   git clone git@github.com:ibrahimozkn/CNG495-F22-CloudComputing.git

After cloning our repo, which can be visited by clicking on this text, I developed a mobile application for demonstration. Application is just a single screen with few buttons which can be seen in Figure 13.

Figure 13 Mobile Application Main Page

Here, current population inside a business premisses is shown in numbers. Since we did not still implemented a Arduino UNO to carry out entrance and departure, I added two buttons to increment and decrement the population via API calls. For API calls, I created a class called DioClient which is using the Dio package of Flutter. You can visit the Github page to check the implemented version of DioClient class.

DioClient class Github Page

Cloning URL of our repository:

https://github.com/ibrahimozkn/CNG495-F22-CloudComputing.git

For API calls there are 3 calls that I use which are:

- Get the population info
- Increment the population
- Decrement the population

When these calls are made, returned JSON data will be used to update the population information. At each increment or decrement API call, Population information call is done to get the up-to-date population count.

2. Milestones Remained

Milestone No	Week	Description	Responsible Student(s)
1	November 28 - December 4	Arduino UNO Configuration & Installation	Ibrahim Ozkan, Adil B. Kebapcioglu
2	December 5 - December 11	Fully implementation of API Database	Adil B. Kebapcioglu
3	December 12 - December 18	Fully implementation of API	Adil B. Kebapcioglu
4	December 19 - December 25	Design UI of the mobile application for all functionalities	Ibrahim Ozkan
5	December 26 - January 1	Setup Google Maps API for map view in mobile application	Ibrahim Ozkan
6	January 2 - January 8	Make mobile application fully functioning	Ibrahim Ozkan

7	January 9 - January 15	Connect all components (API, Arduino, Mobile Applications) together and make system ready to use	Ibrahim Ozkan, Adil B. Kebapcioglu
8	January 16 - January 22		

2.1. Delivery List

Materials that will be delivered when project finishes are listed below:

- Code for API (Laravel)
- Code for Mobile Application (Flutter)
- Instructions for setting up EC2 server, Google API and Laravel Deployment
- APK file for Mobile Application
- Database file of demo
- Arduino code
- Arduino UNO setup instructions