**Team Members :**

const team = {

"Shravan H R": "Full Stack Developer",

"Chirag R Gowda": "UI/UX Designer",

"Yadunandan K": "Frontend Developer",

"Yogeshwar R": "Web Developer",

"Thejesh": "Front-End Developer",

"Yogesh": "Developer and Scrum Master",

"Srujan U": "Banckend Developer",

};

function displayTeamDetails(team) {

console.log("Team Members and Their Roles:");

console.log("--------------------------------");

for (const name in team) {

console.log(`${name}: ${team[name]}`);

}

}

displayTeamDetails(team);

Project Document ( Docker ) :

1. Creating project directory

Create html, CSS, JavaScript for your website and a dockerfile in a single folder (OLX-Clone).

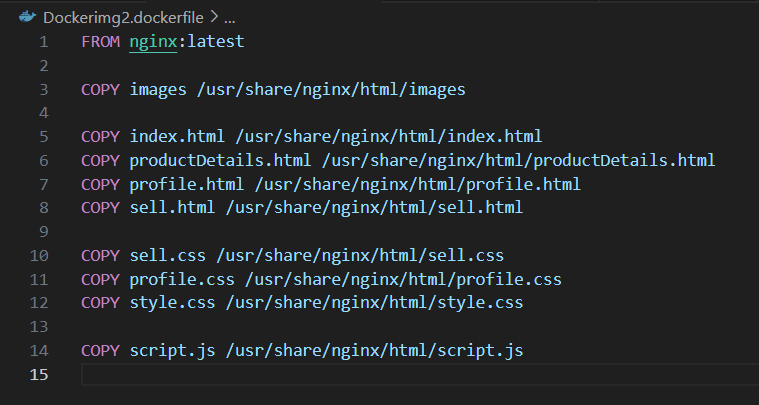


Image: Dockerimg2.dockerfile

It will import the NGINX default image.

COPY command will copy files into default image.

2. Building docker image

Open command prompt.

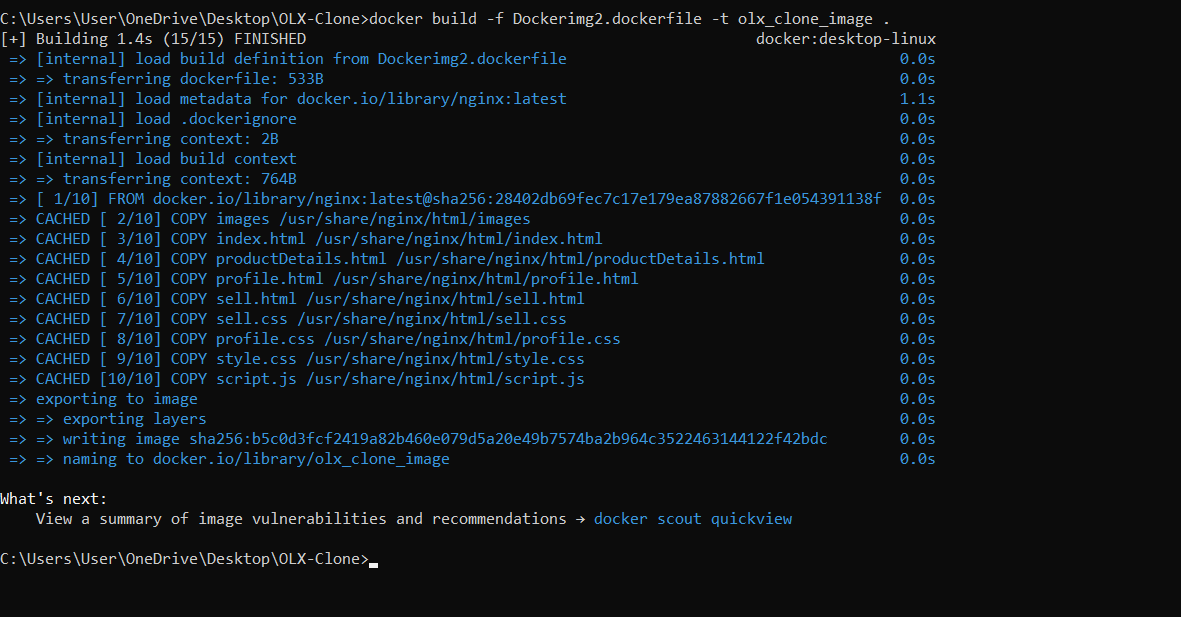
**Enter to your folder directory** by using cd.

Then give the following command:

***docker build -f Dockerimg2.dockerfile -t olx\_clone\_image .***

Dockerimg2.dockerfile : your docker file name.

olx\_clone\_image : docker image name.

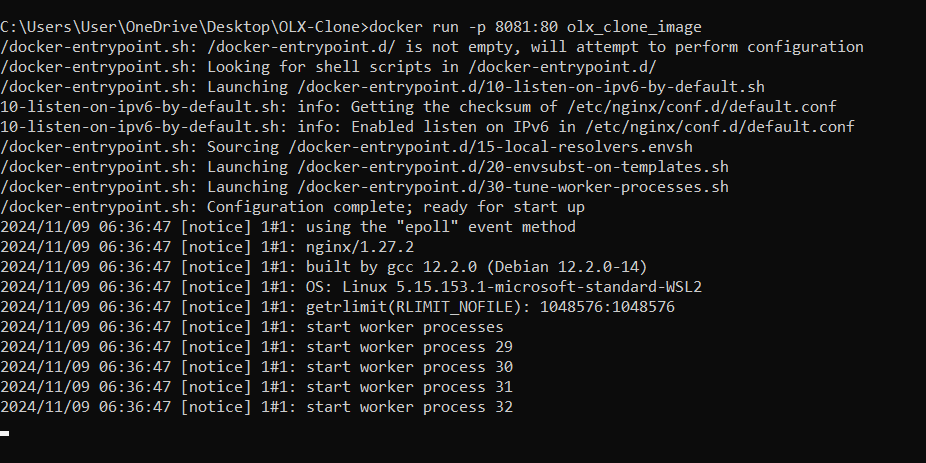


*image:* *docker build -f Dockerimg2.dockerfile -t olx\_clone\_image . command*

3. Running the docker image

After building the docker image, run the following command.

***docker run -p 8081:80 olx\_clone\_image***

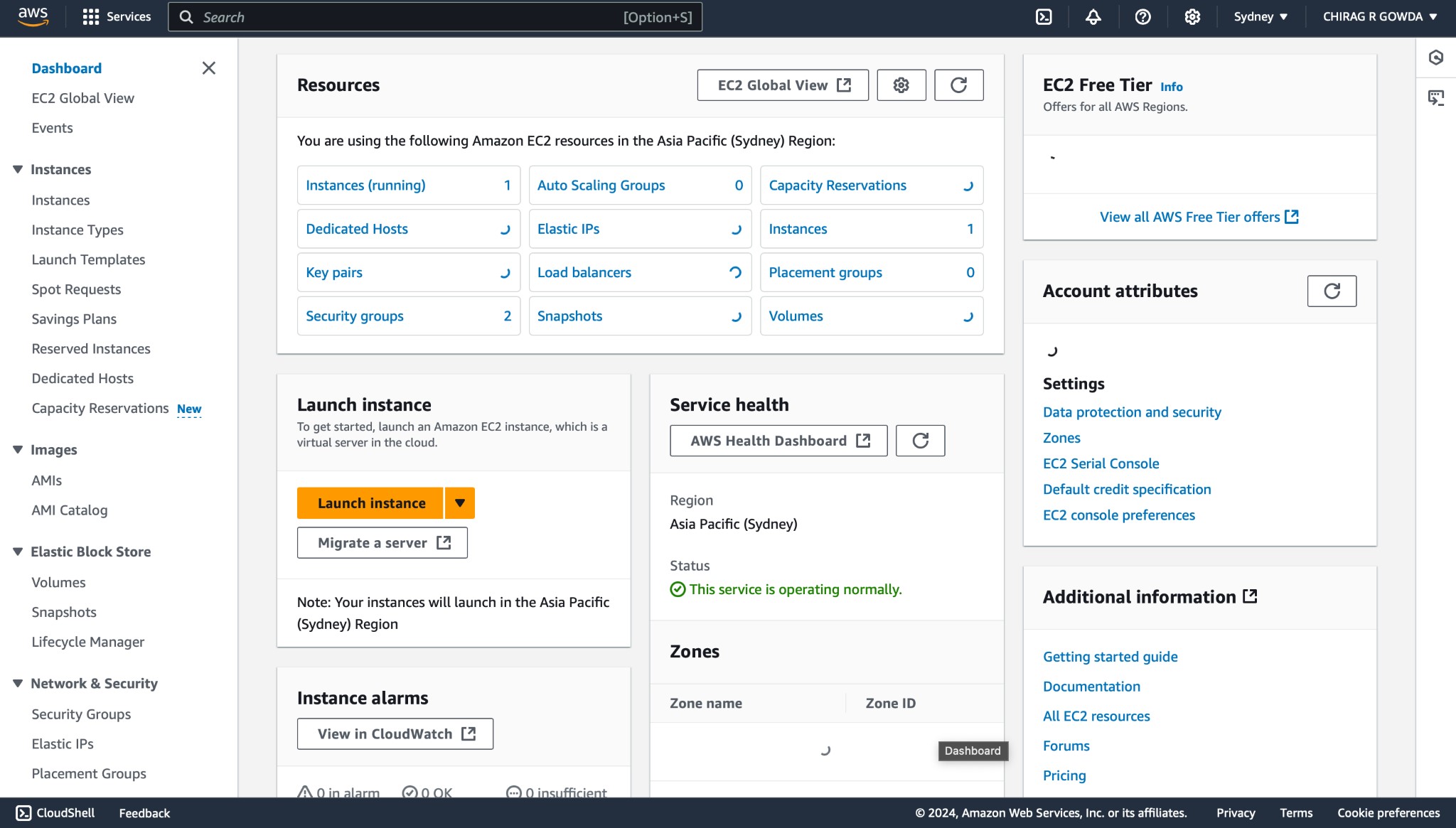
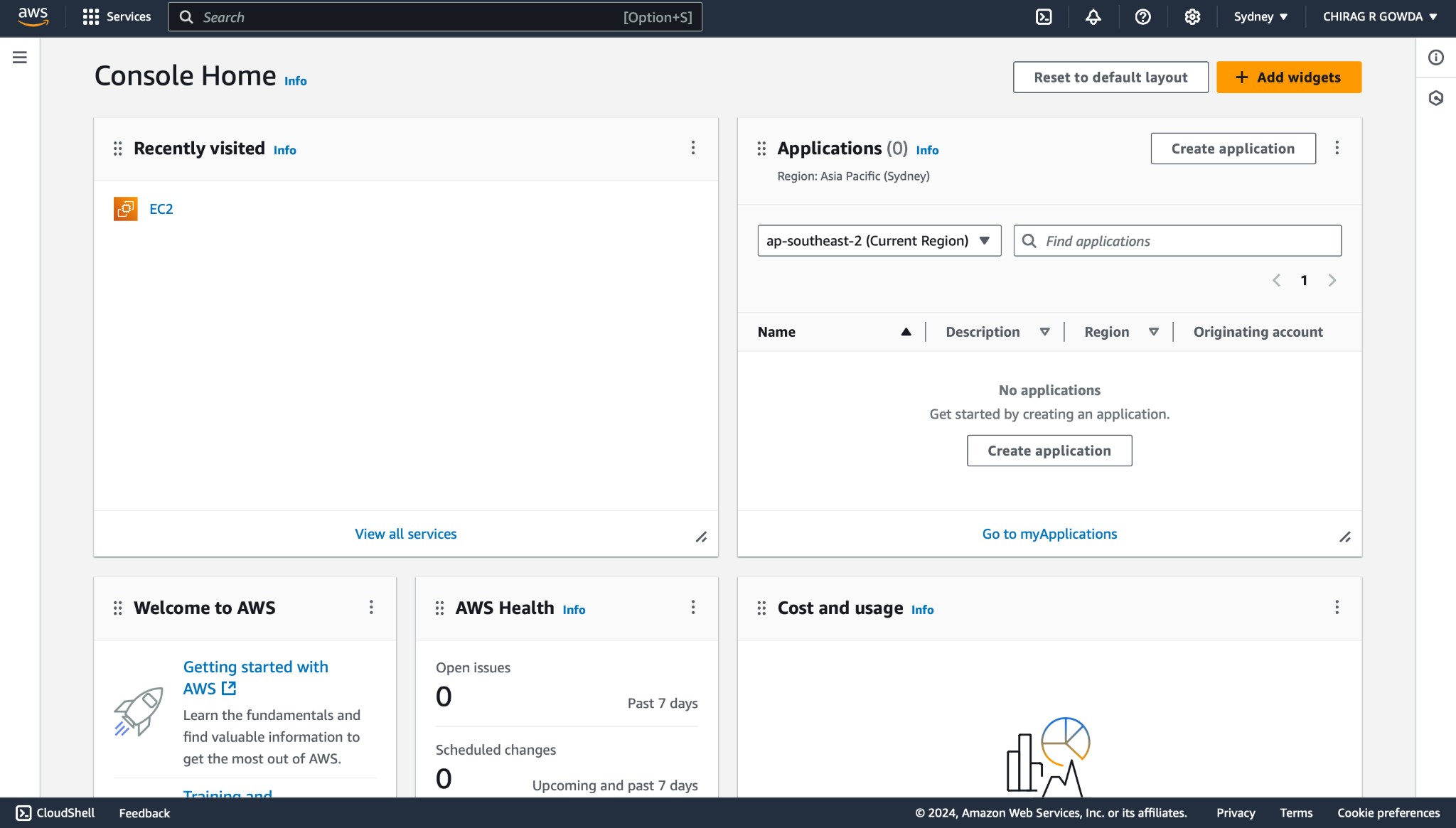


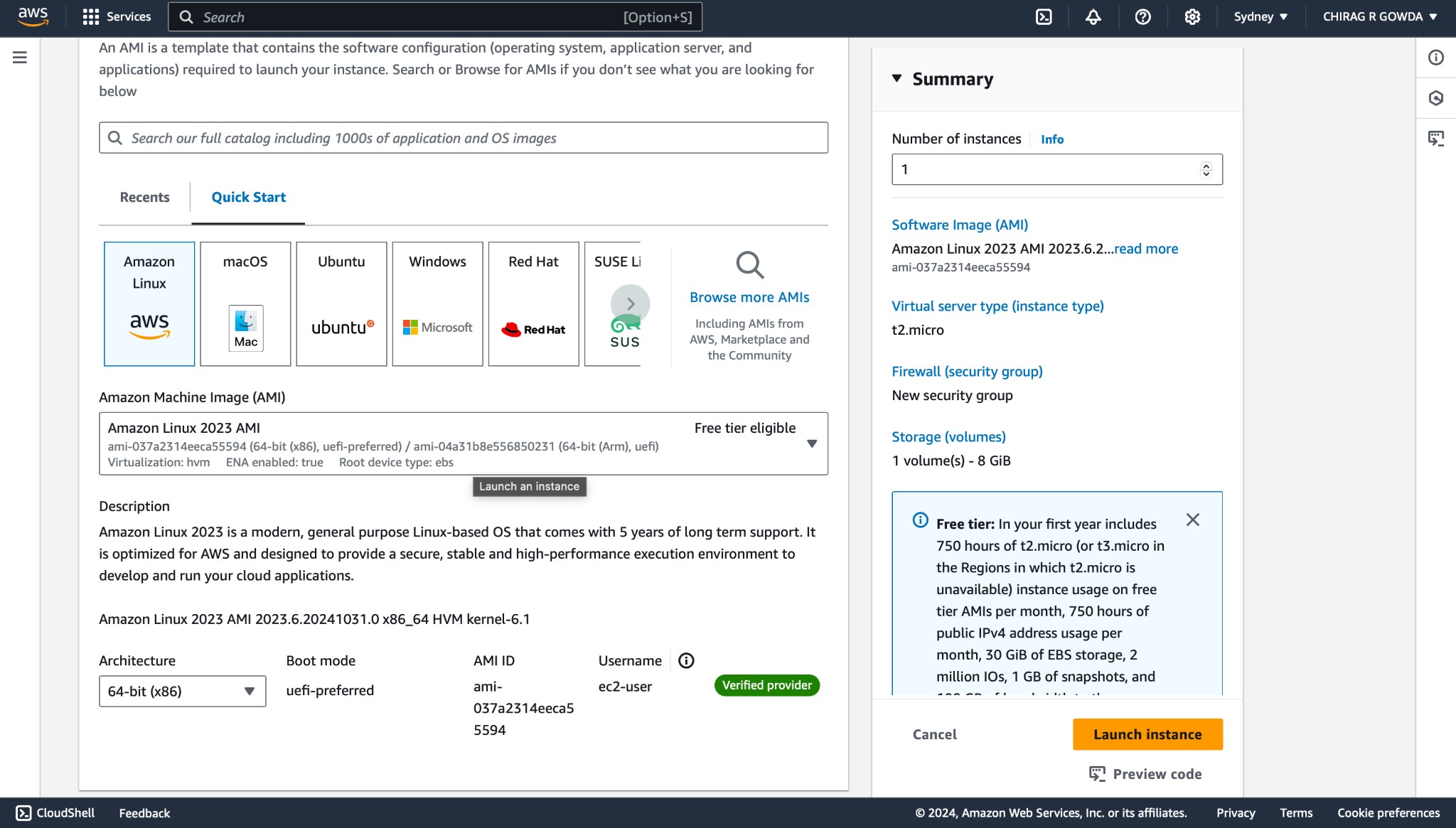
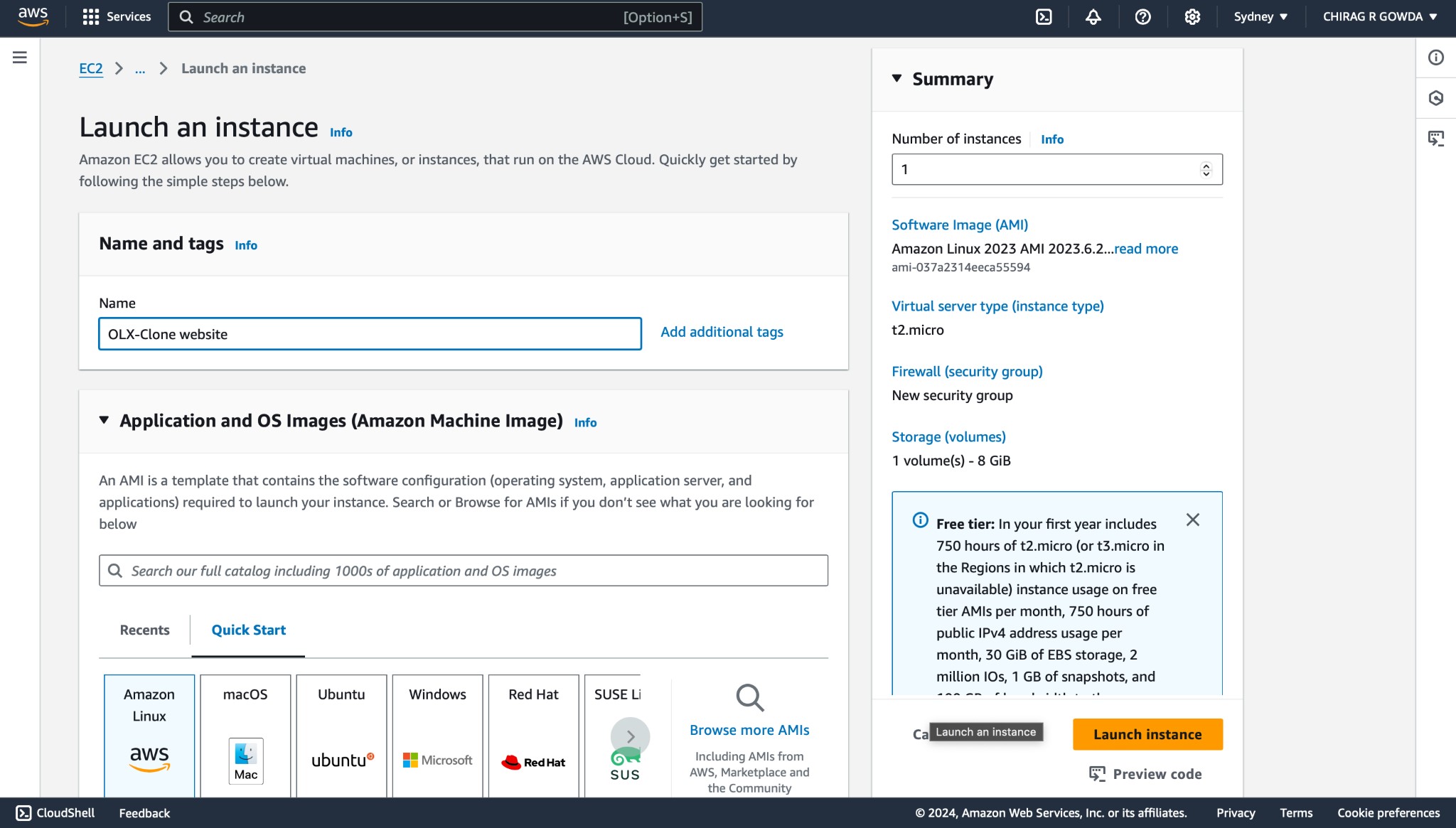
*Image:* *docker run -p 8081:80 olx\_clone\_image command*

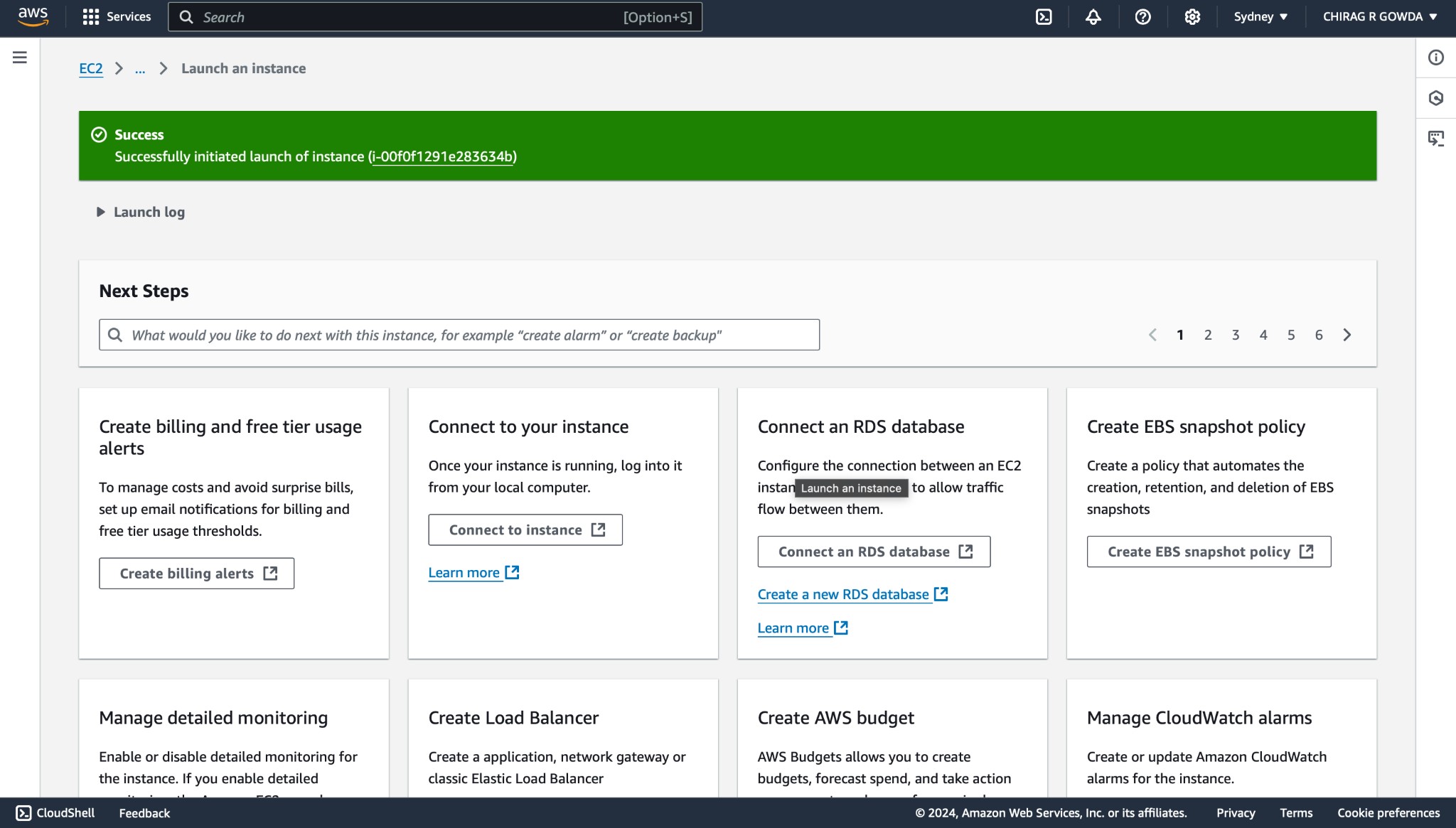
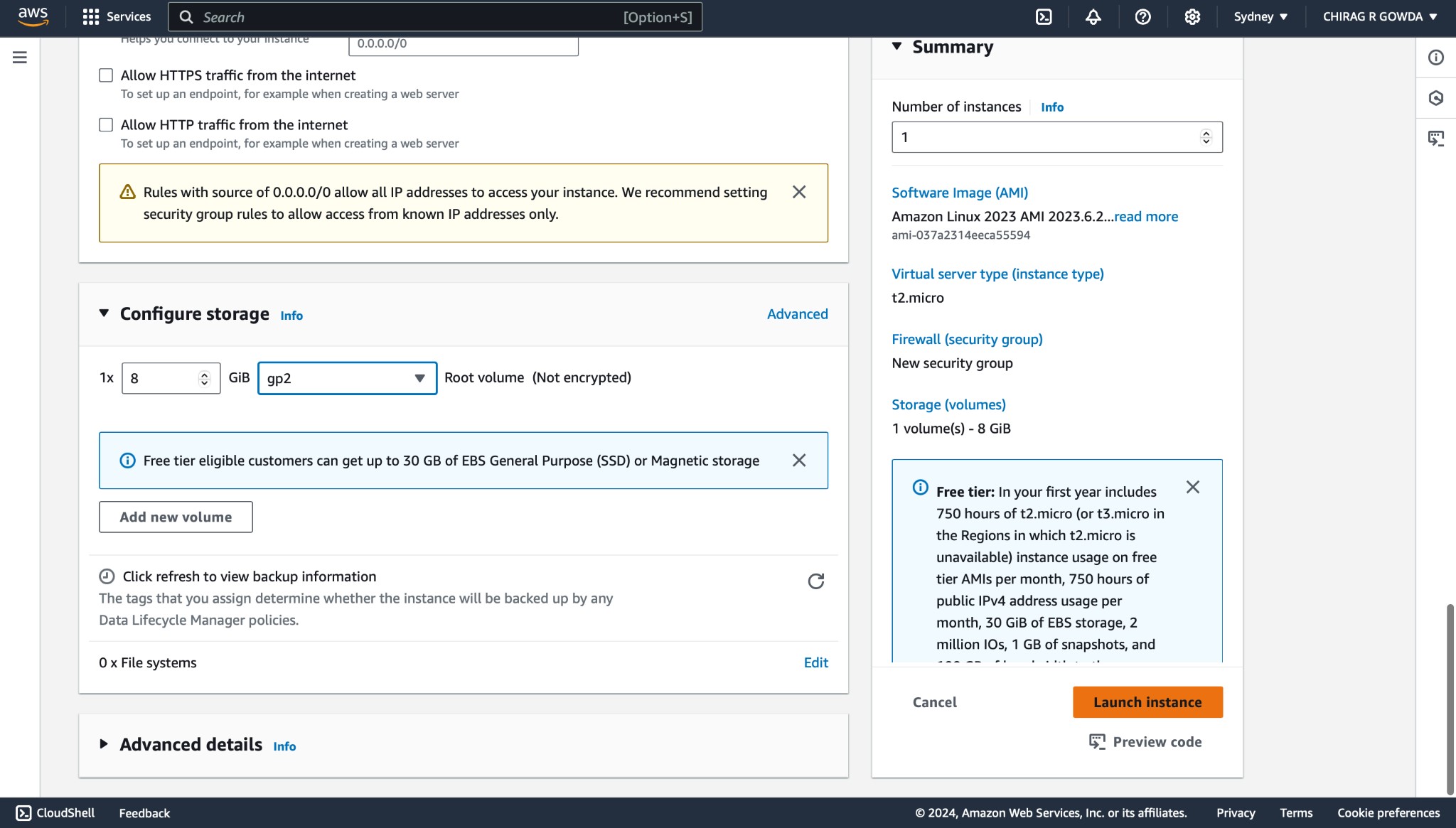
Here, we are using the port 8081 for this application to run.

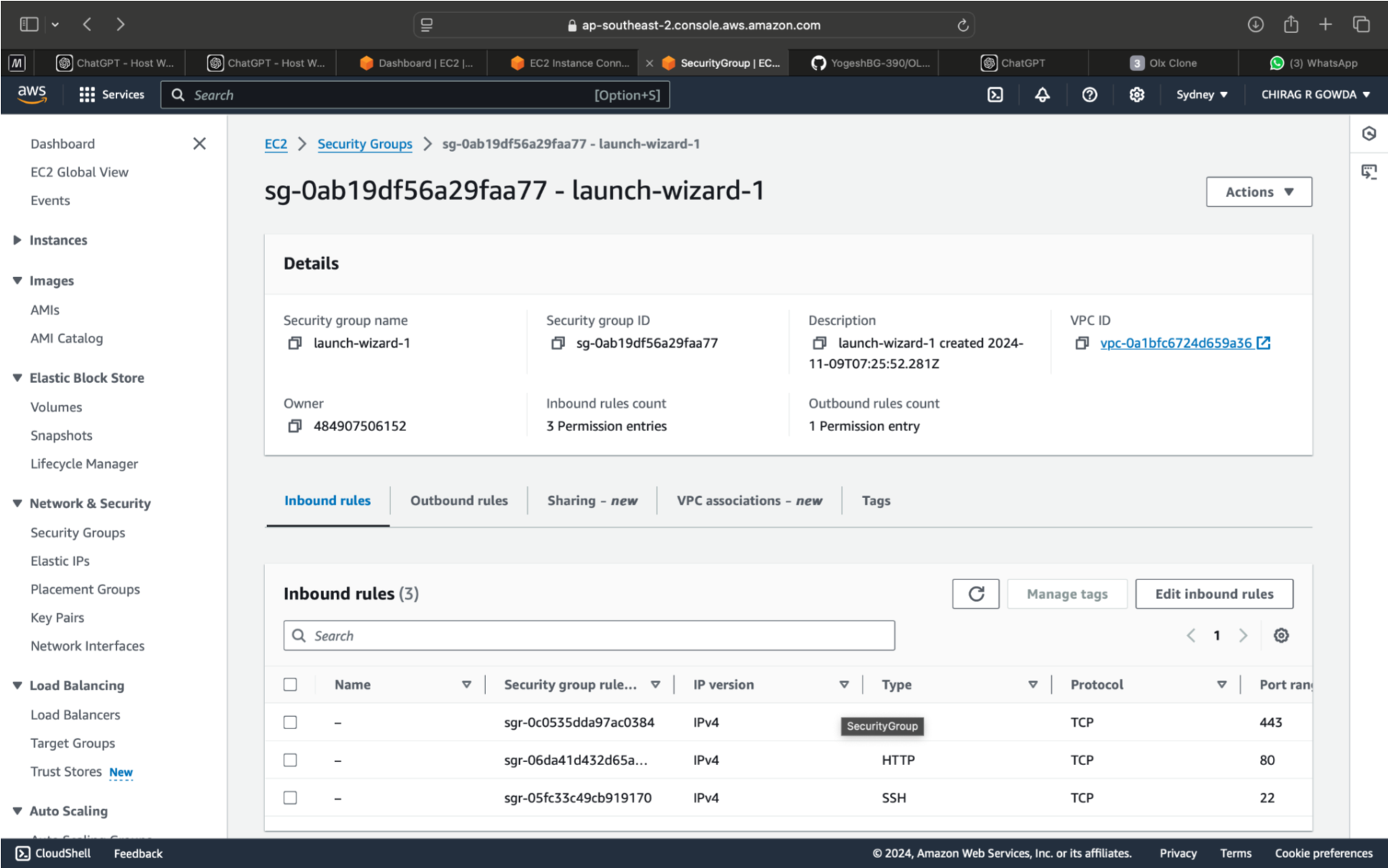
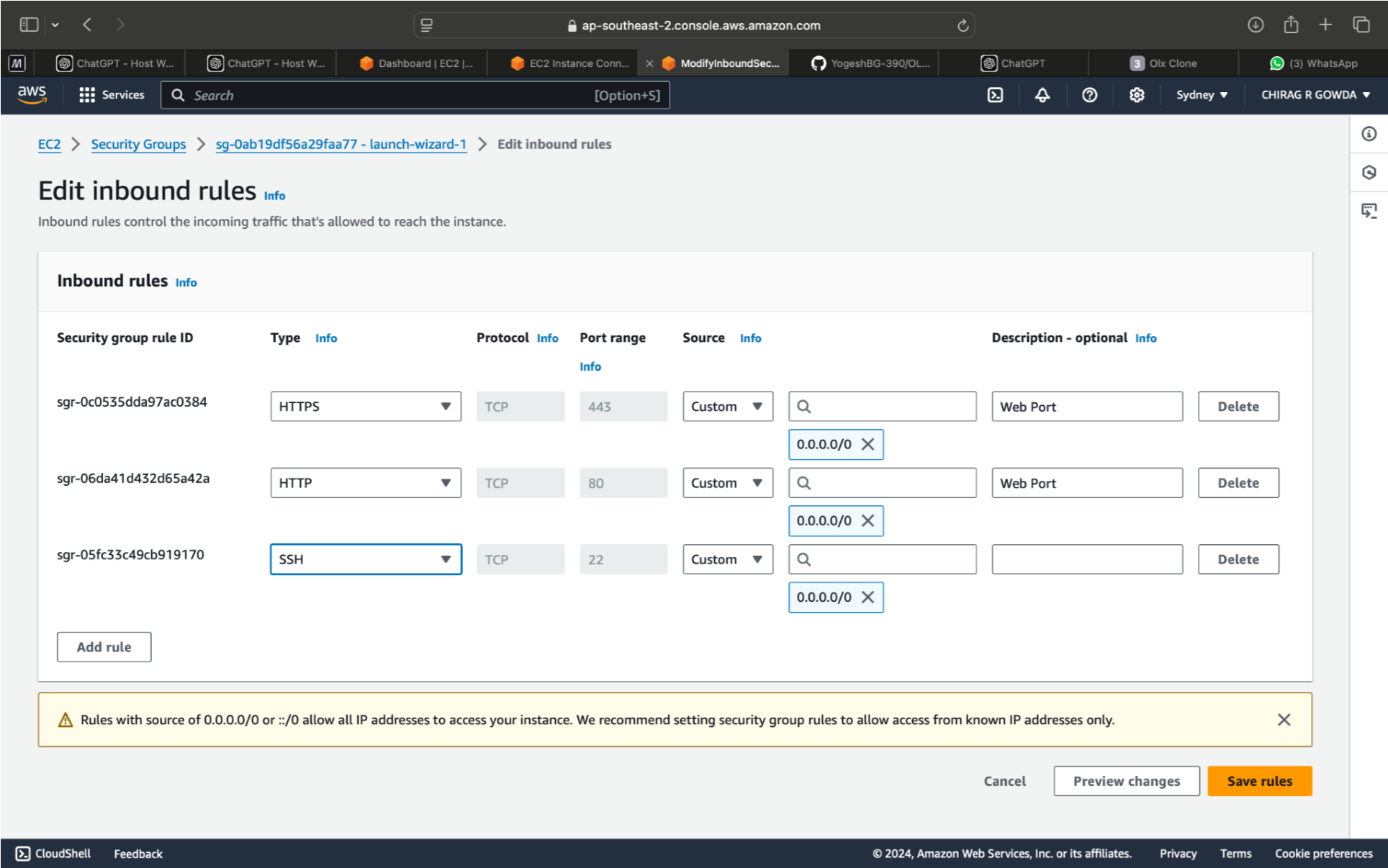
Access your web application through [**http://localhost:8081**](http://localhost:8081)

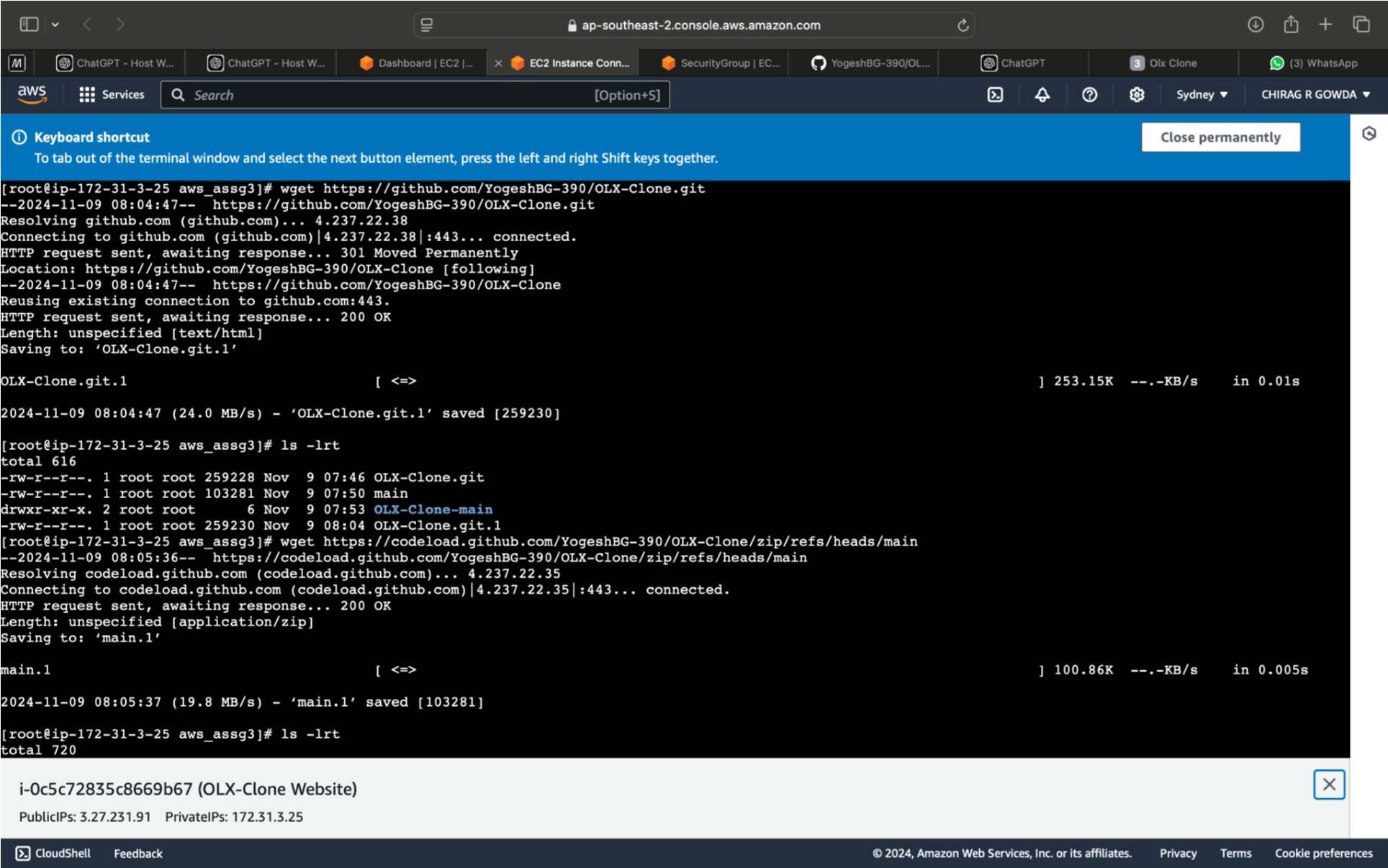
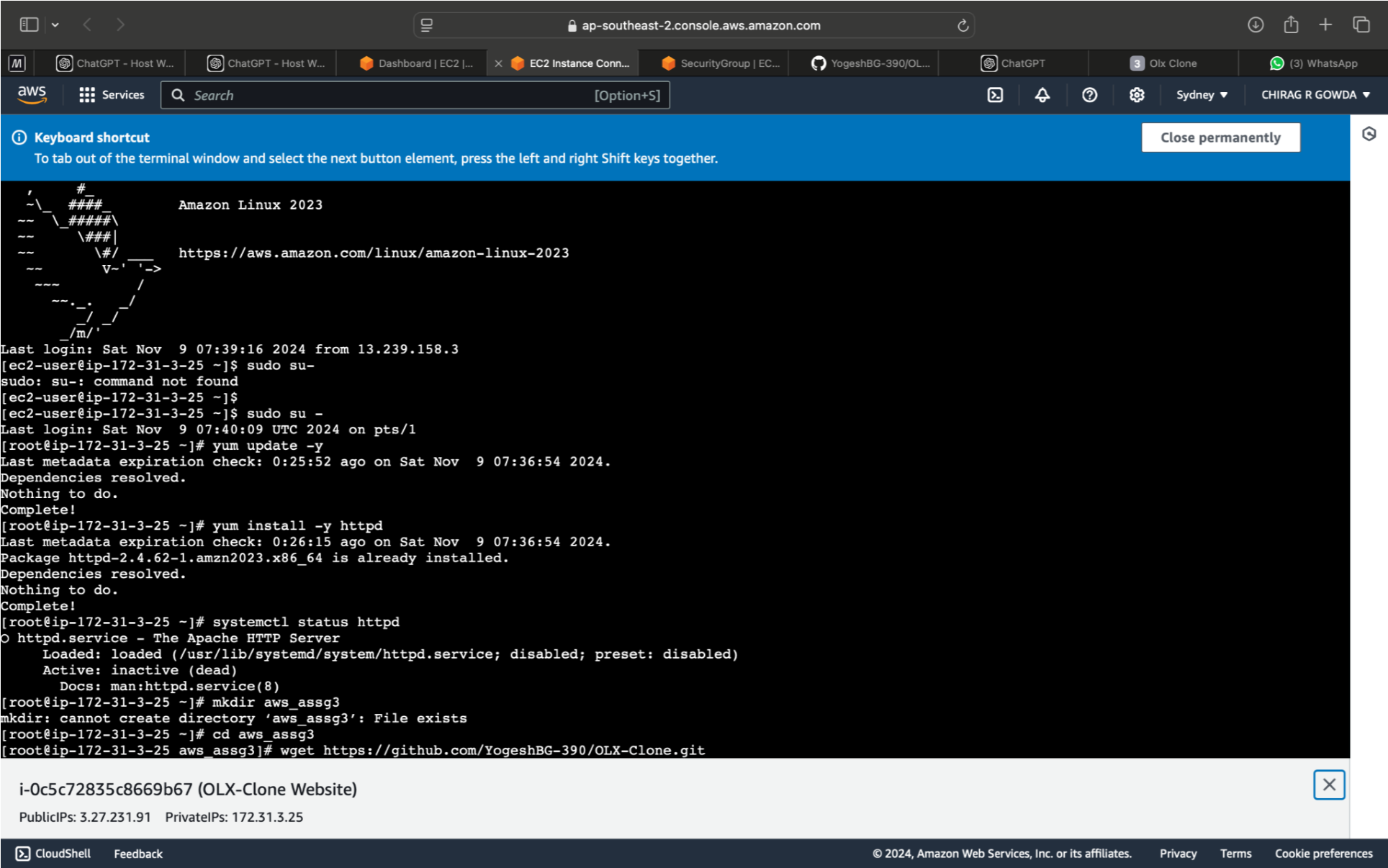
Process of Uploading OLX-Clone into AWS:

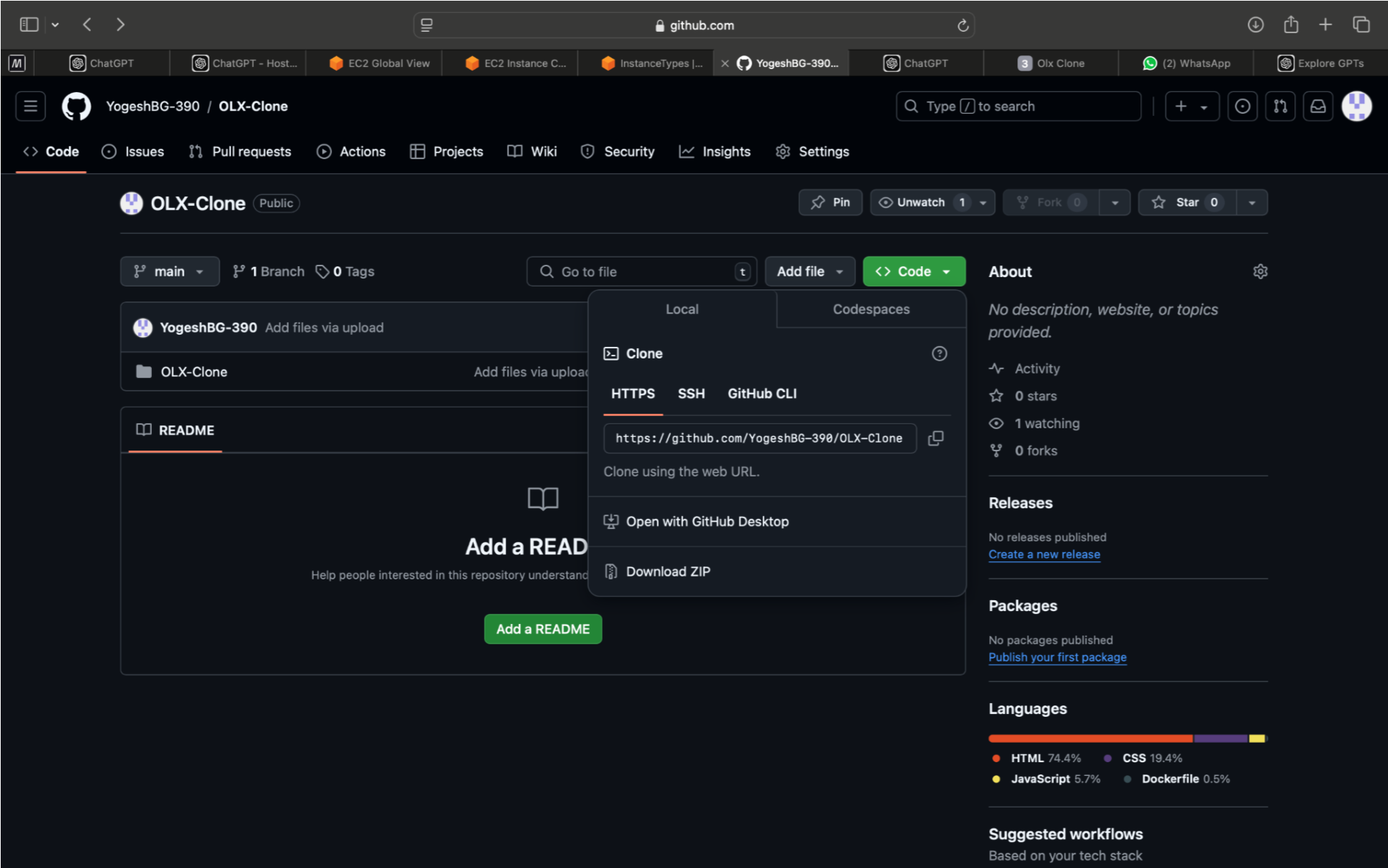
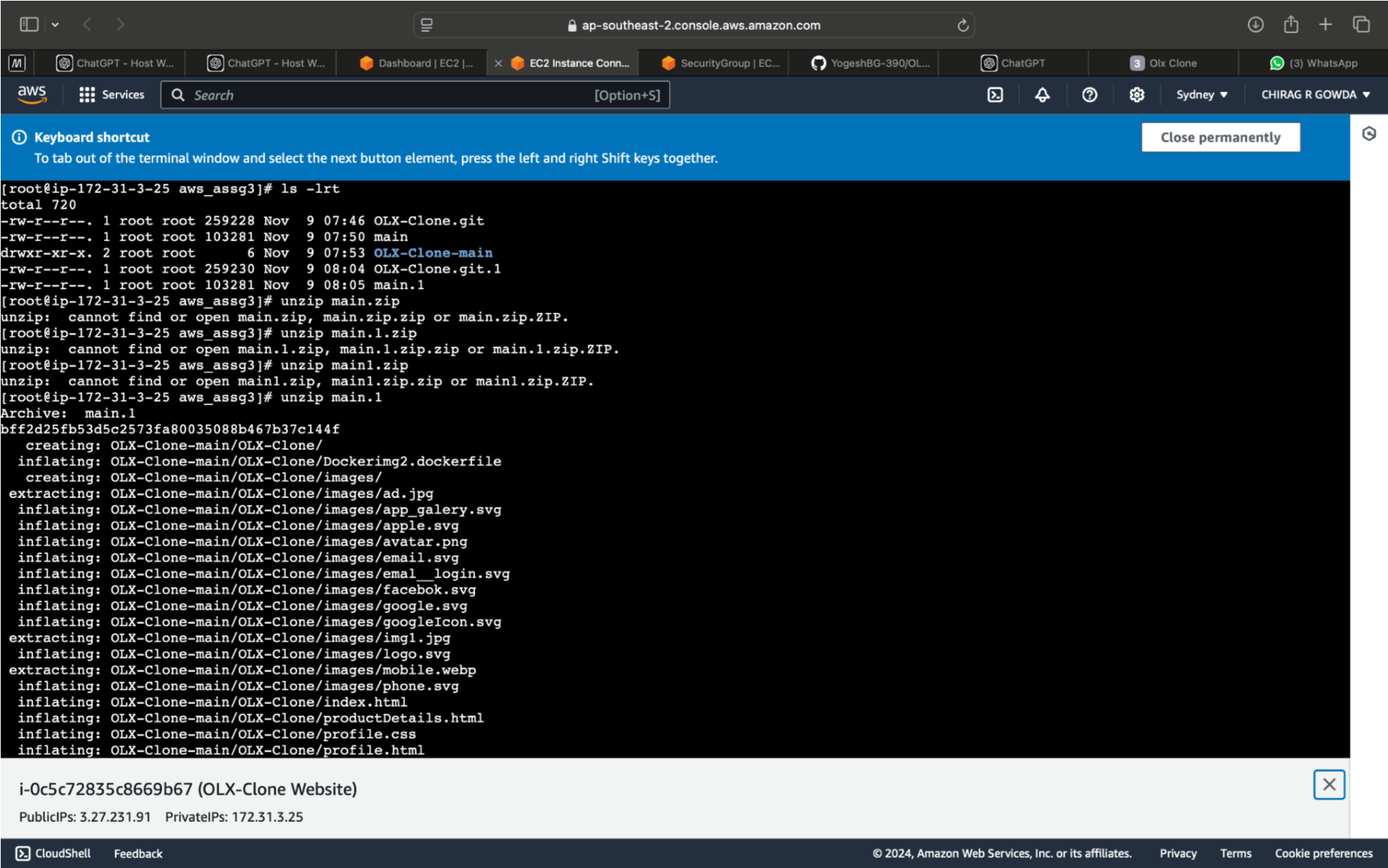


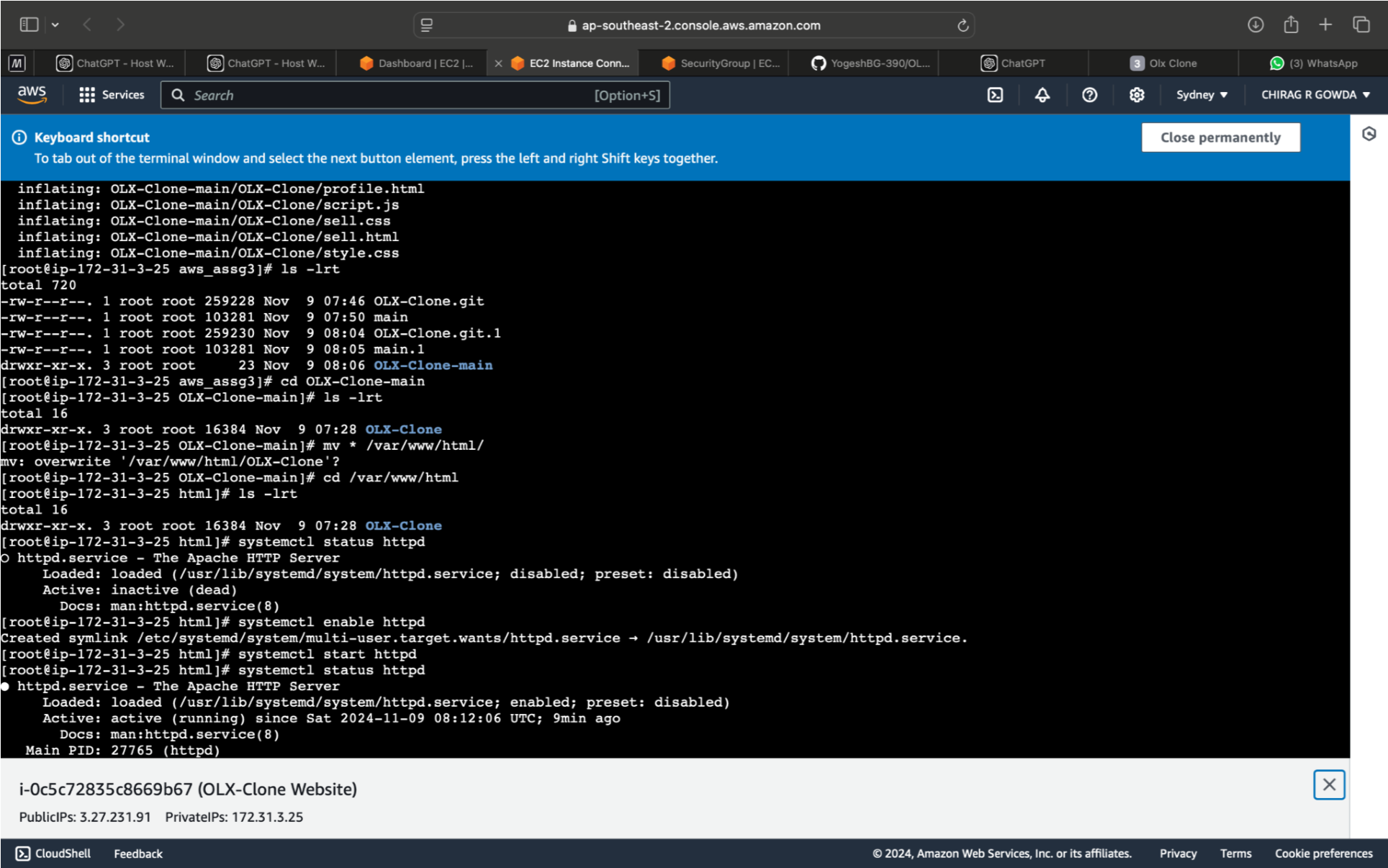


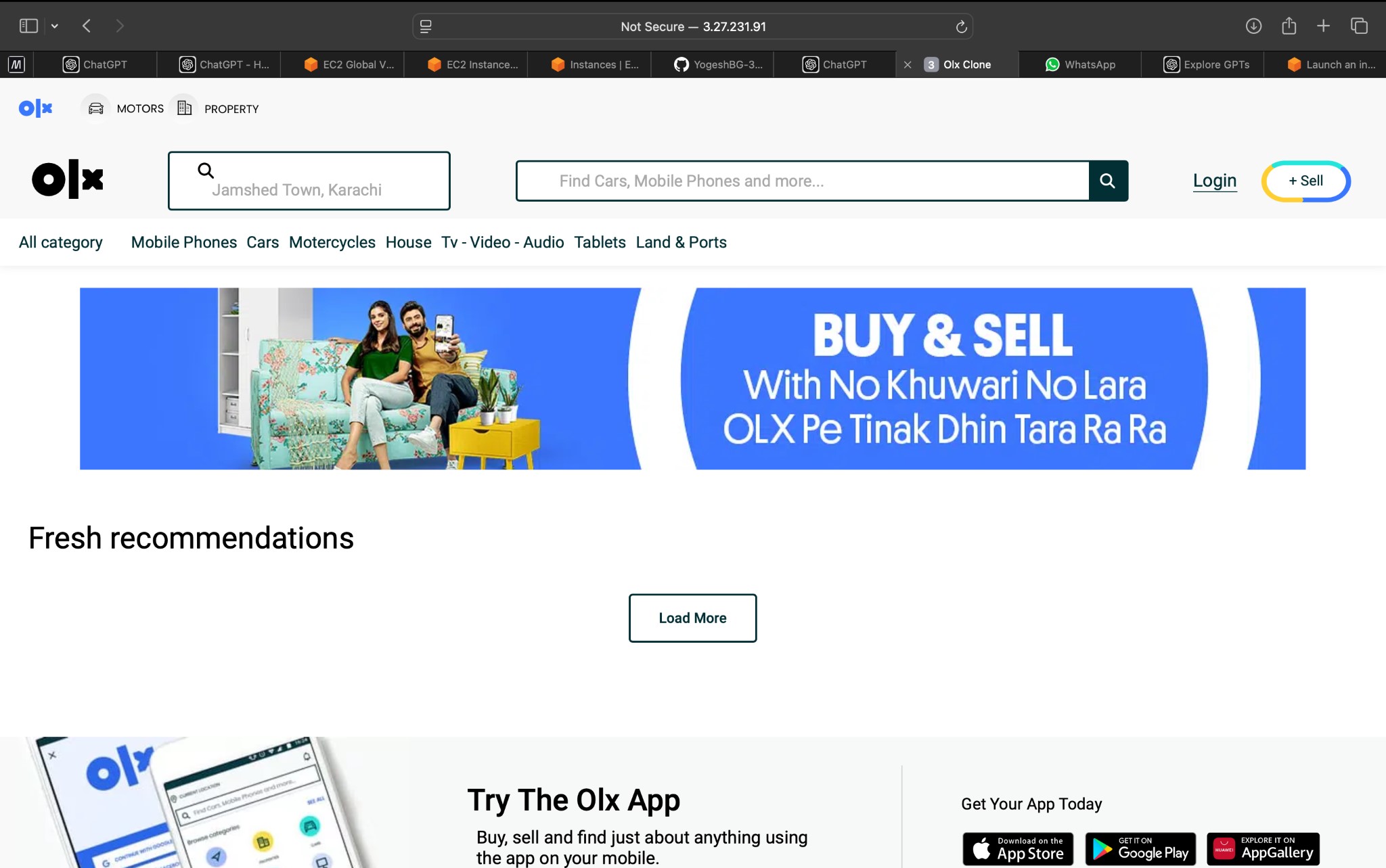
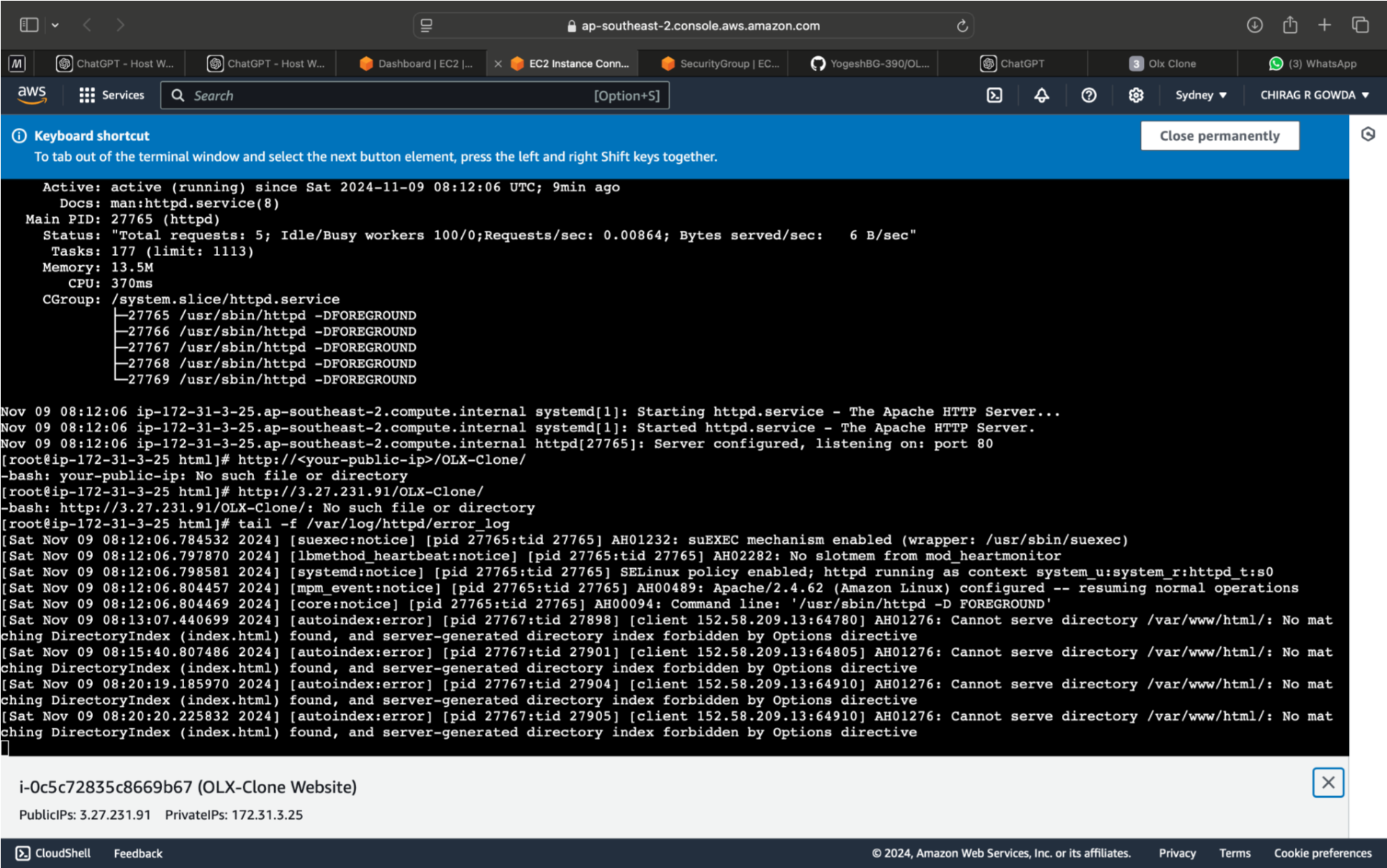












**OLX-Clone Project Deployment on Netlify**

**Project Overview**

**Project Name**: OLX-Clone  
**Project Description**: A clone of the OLX platform with core functionalities, featuring index.html as the main starting point.  
**Live Link**: [OLX-Clone on Netlify](https://flourishing-sunflower-054177.netlify.app/)

**Deployment on Netlify**

**Step 1: Create a Netlify Account**

1. Visit [Netlify](https://www.netlify.com/) and sign up or log in.
2. Connect your GitHub/GitLab/Bitbucket account if the project is stored in a repository.

**Step 2: Set Up a New Project**

1. Go to **Sites** in the Netlify dashboard.
2. Click **New site from Git**.
3. Select the Git provider where your project is hosted, and choose the repository containing the OLX-Clone project.

**Step 3: Configure Deployment Settings**

1. Select the **branch** to deploy (usually main or master).
2. **Build Settings**:
   * **Build Command**: Leave blank if your project is static and doesn’t require any build step.
   * **Publish Directory**: Set to the directory containing index.html (e.g., the root directory if index.html is in the root).

**Step 4: Deploy Site**

1. Click **Deploy Site** to start the deployment process.
2. Netlify will automatically build and deploy your site. Once completed, a confirmation message will appear.

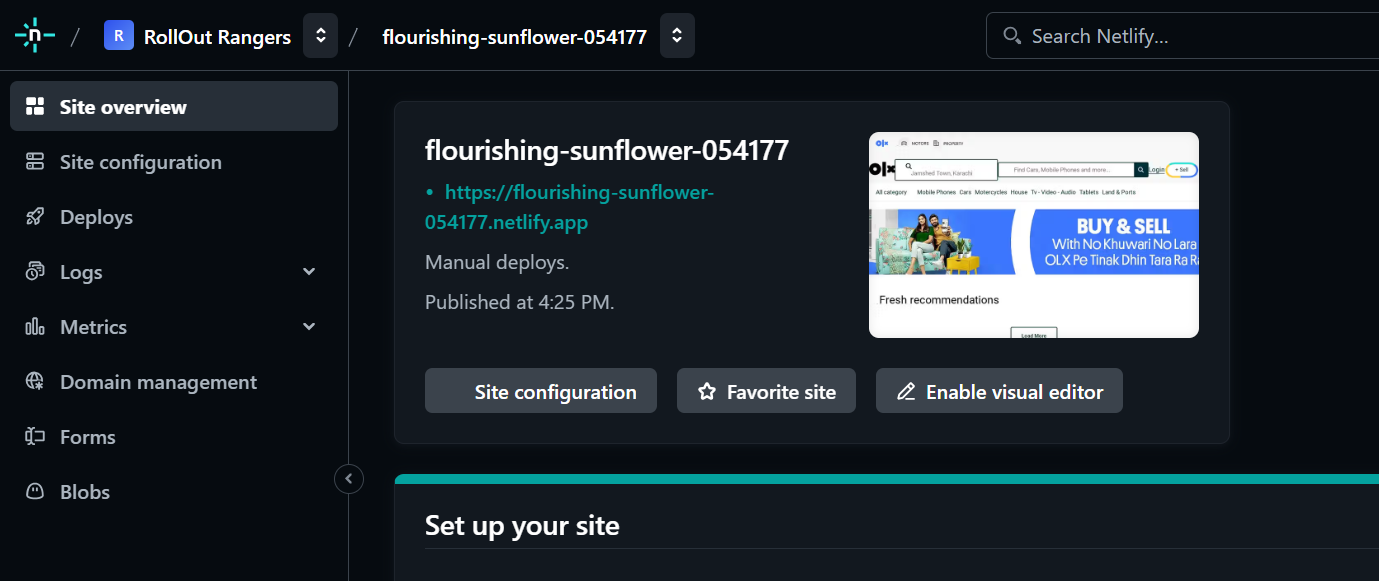
**Step 5: Customize Domain (Optional)**

1. In the **Site settings**, go to **Domain settings**.
2. You can rename the subdomain (e.g., https://your-custom-name.netlify.app/) or connect a custom domain.

**Additional Notes**

* **Automatic Deployment**: Netlify redeploys your site every time you push changes to the selected branch.
* **Environment Variables**: Add any necessary environment variables in the **Site settings** if your project requires them.

**Deployment Link**: [OLX-Clone Live Site](https://flourishing-sunflower-054177.netlify.app/)



Vercel Deployment Configuration

**1. Project Setup :**

- Make sure the `index.html` file is located at the root of the project or configured as the entry point in your settings.

**2. Create vercel.json (optional if needed for customization) :**

- If Vercel requires specific settings for static HTML projects, create a `vercel.json` file in the root directory with content like this:

{

"rewrites": [{ "source": "/", "destination": "/index.html" }]

}

**3. Deployment Steps :**

- Follow the usual steps on Vercel to import your GitHub repository, configure environment variables if any, and click **Deploy**.

This configuration should help Vercel recognize `index.html` as the main entry point. Let me know if you need more details on any specific steps!

