**BUILD A REACTJS APPLICATION AND DEPLOY IT IN TO WEB BY USING SIMPLE CALCULATOR**

**What Is React?**

React is a framework that employs Webpack to automatically compile React, JSX, and ES6 code while handling CSS file prefixes. React is a JavaScript-based UI development library. Although React is a library rather than a language, it is widely used in web development. The library first appeared in May 2013 and is now one of the most commonly used front-end libraries for web development.

React offers extensions for entire application architectural support, such as Flux and React Native, beyond mere UI.

**Why React?**

React is popularity today has eclipsed that of all other front-end development frameworks.

1) Easy creation of dynamic applications

2) Improved performance

3) Reusable components

4) Unidirectional data flow

5) Small learning curve

6) It can be used for the development of both web and mobile apps

7) Dedicated tools for easy debugging

**The Significance of AWS EC2:**

Amazon Elastic Compute Cloud (EC2) is a scalable and flexible cloud computing service provided by Amazon Web Services (AWS). It allows users, including DevOps engineers like yourself, to run virtual servers in the cloud. EC2 instances offer a wide range of configurations to meet various computing needs, making it a pivotal choice for deploying and managing applications.

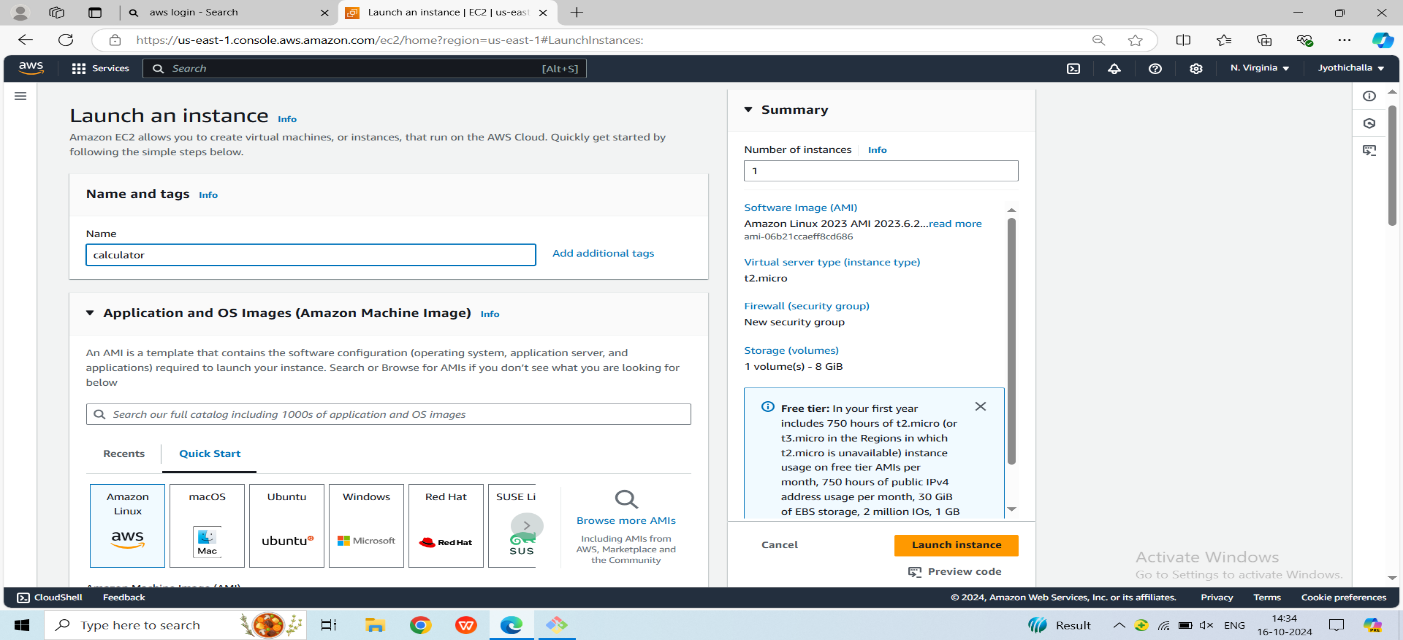
**Prerequisites for Deploying ReactJS on AWS EC2**

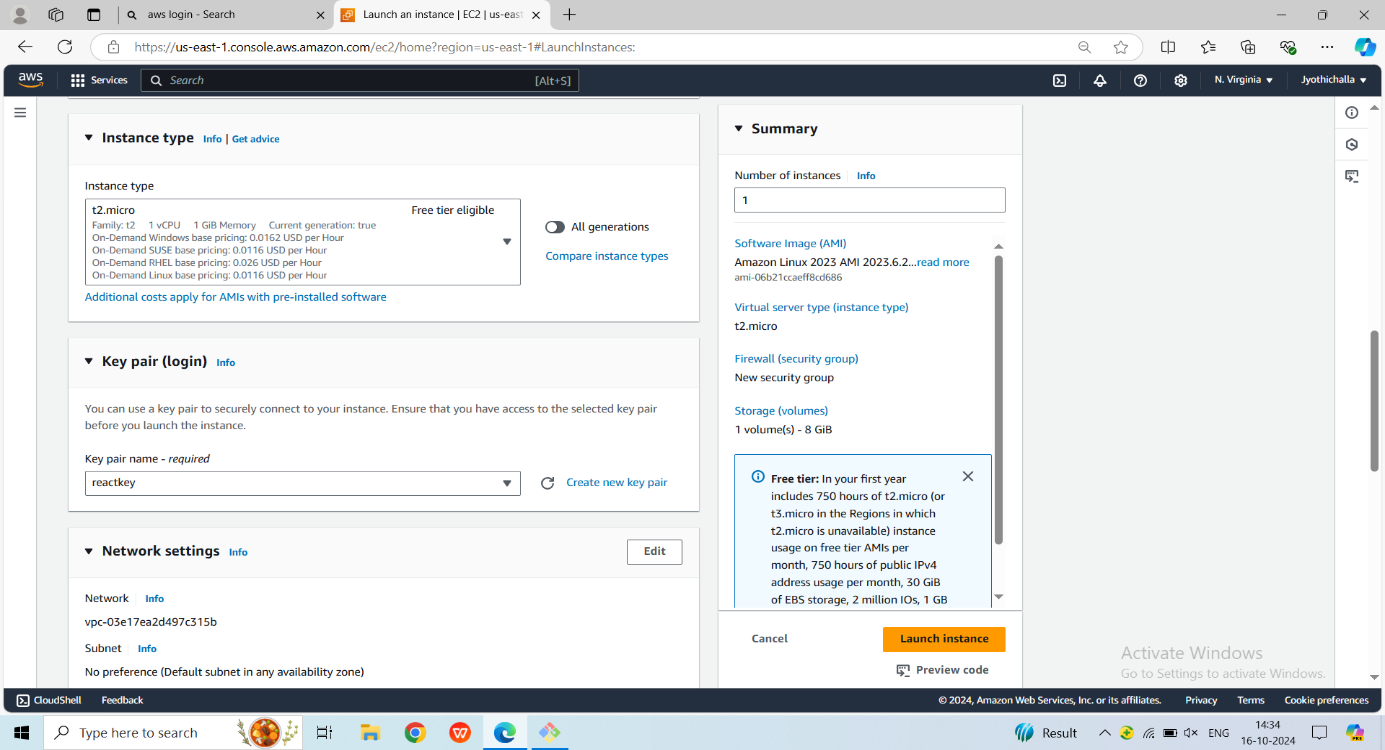
Before diving into the deployment process, ensure you have the following prerequisites in place:

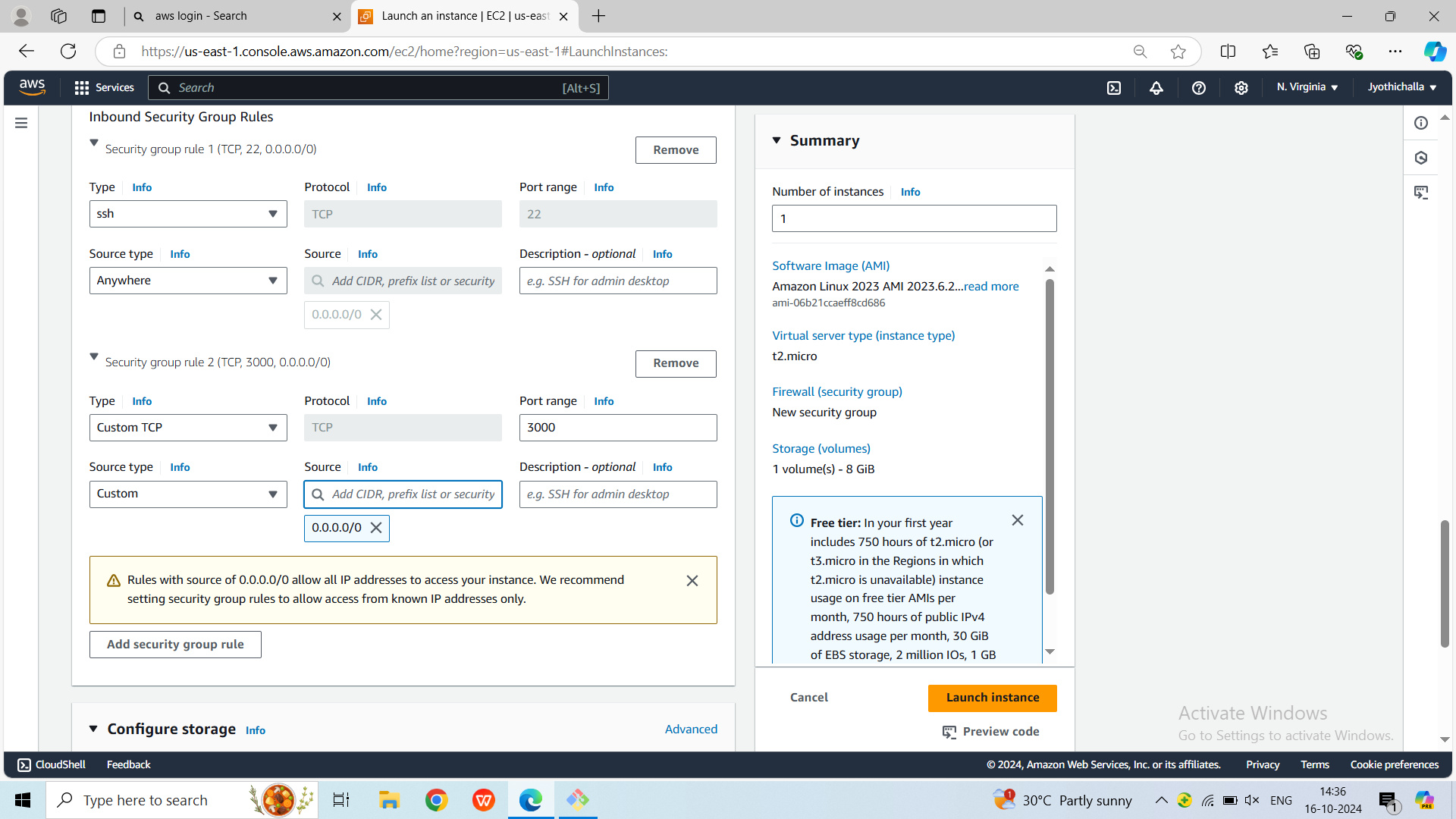
1. An AWS account with EC2 access.
2. Basic knowledge of ReactJS and its application structure.

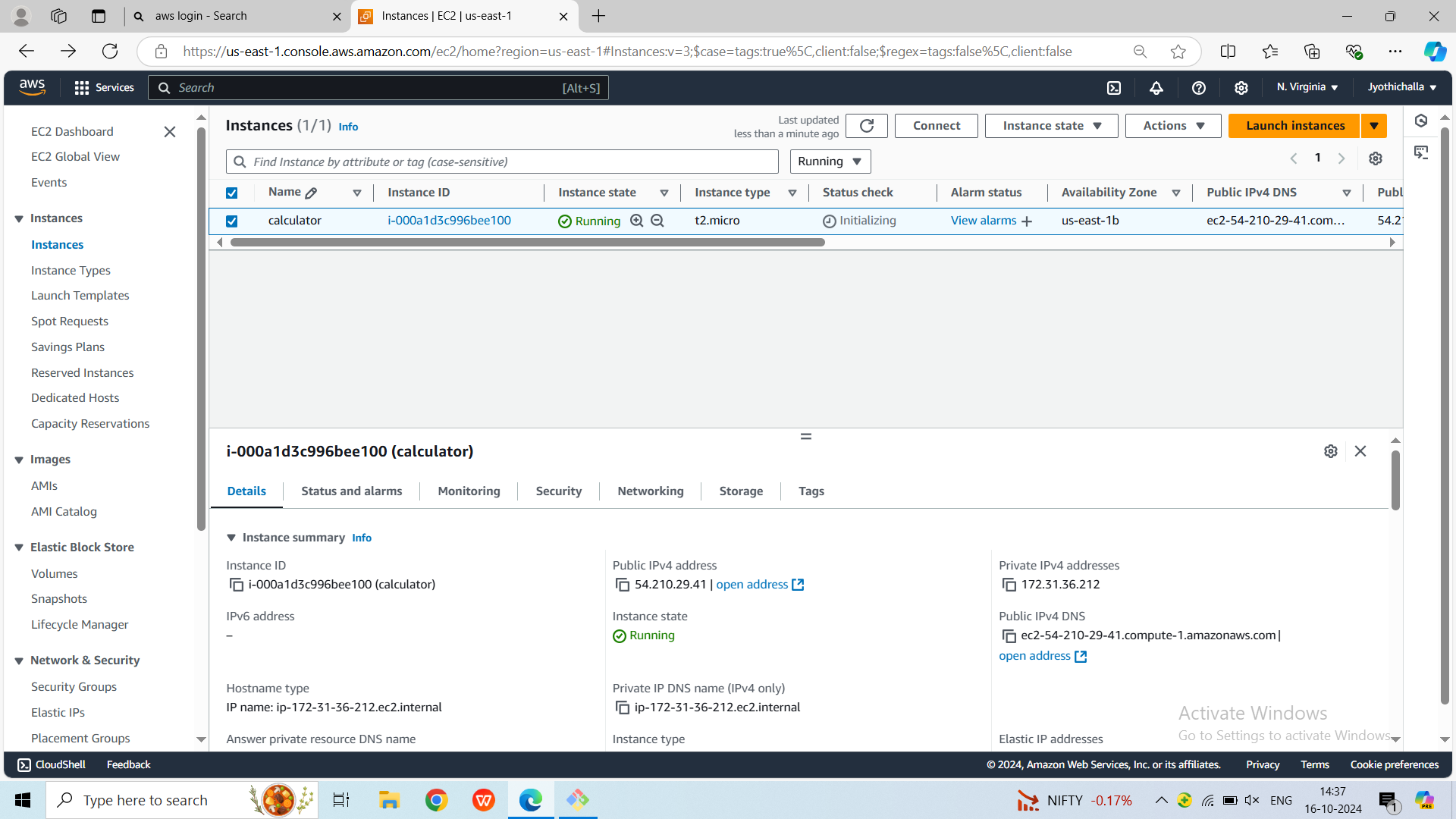
***Step 1: Launching an EC2 Instance***

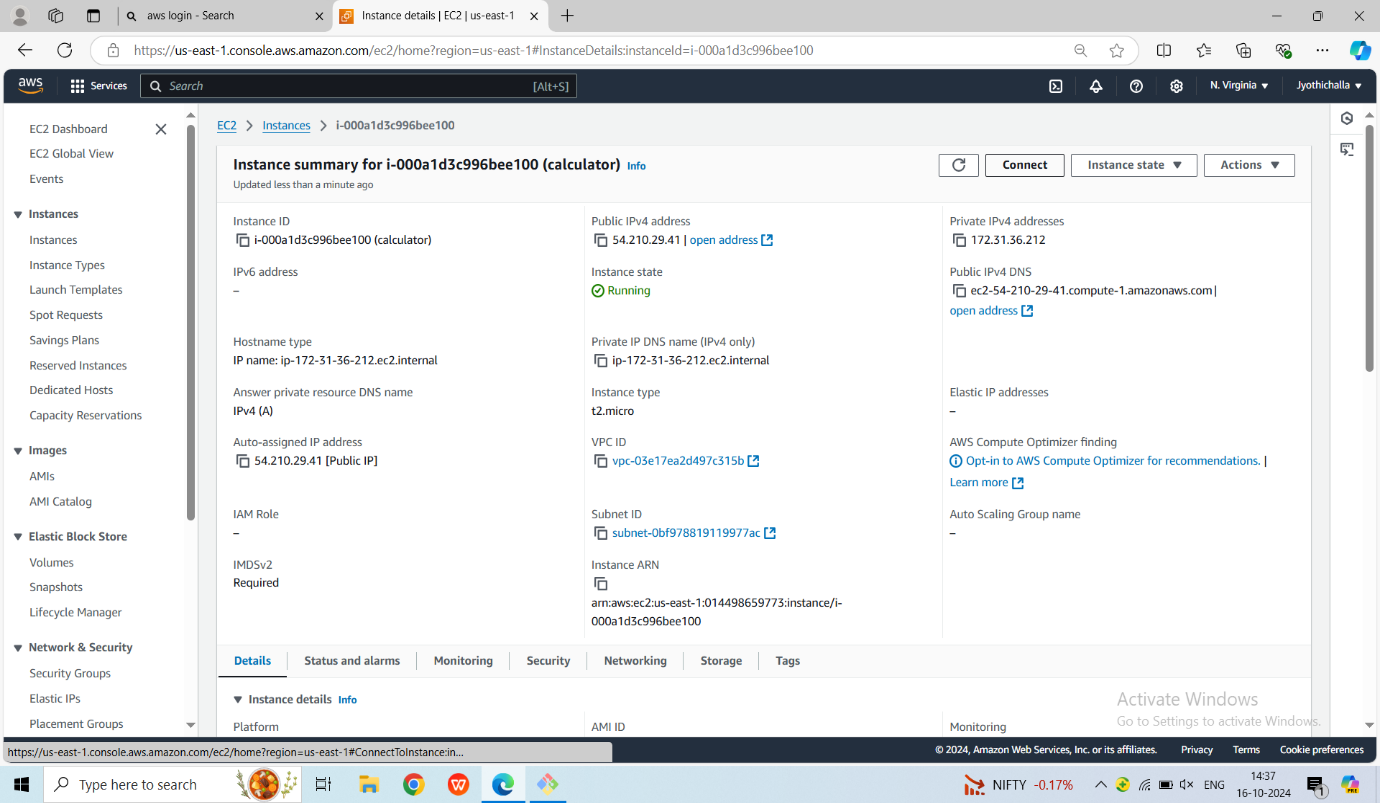
* Log in to the AWS Management Console.
* Navigate to the EC2 Dashboard.
* Click on “Launch Instance” and choose an Amazon Machine Image (AMI) with your preferred OS.

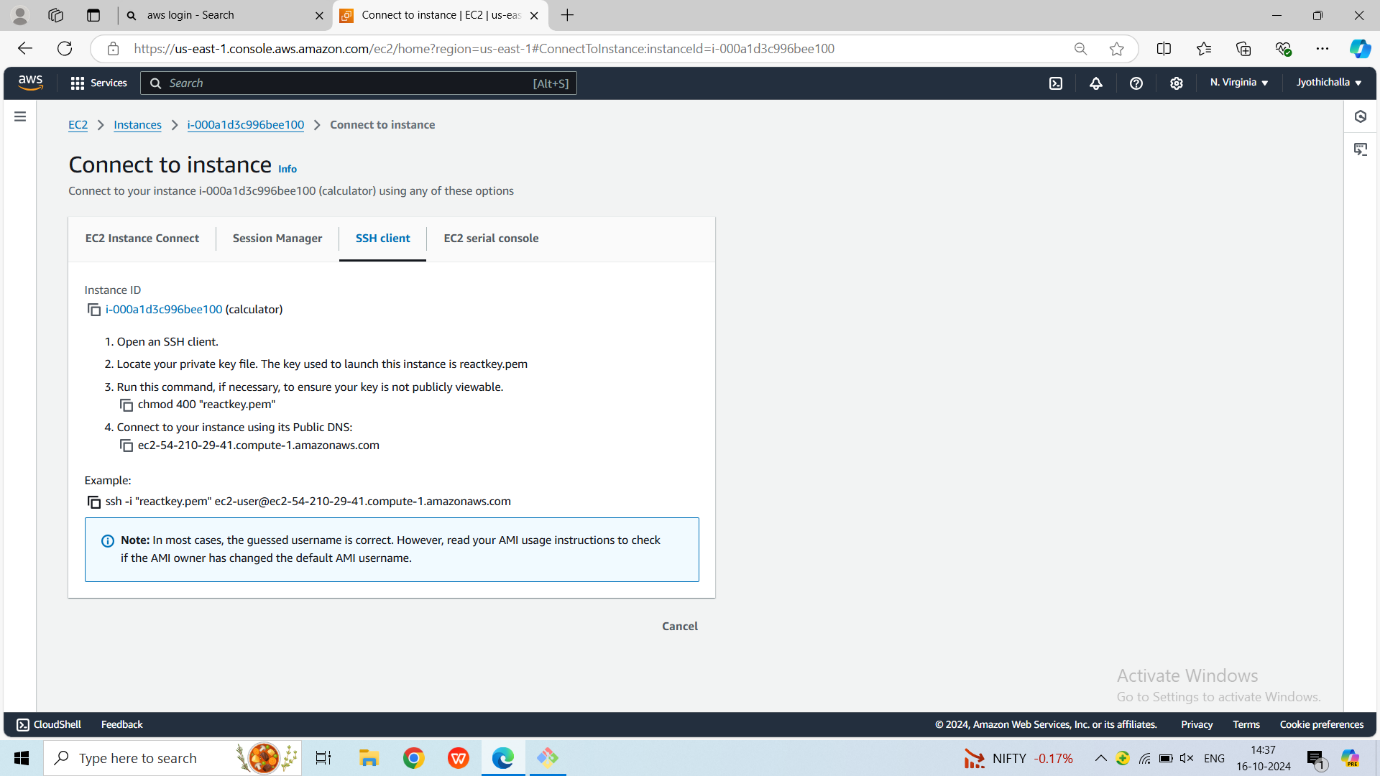








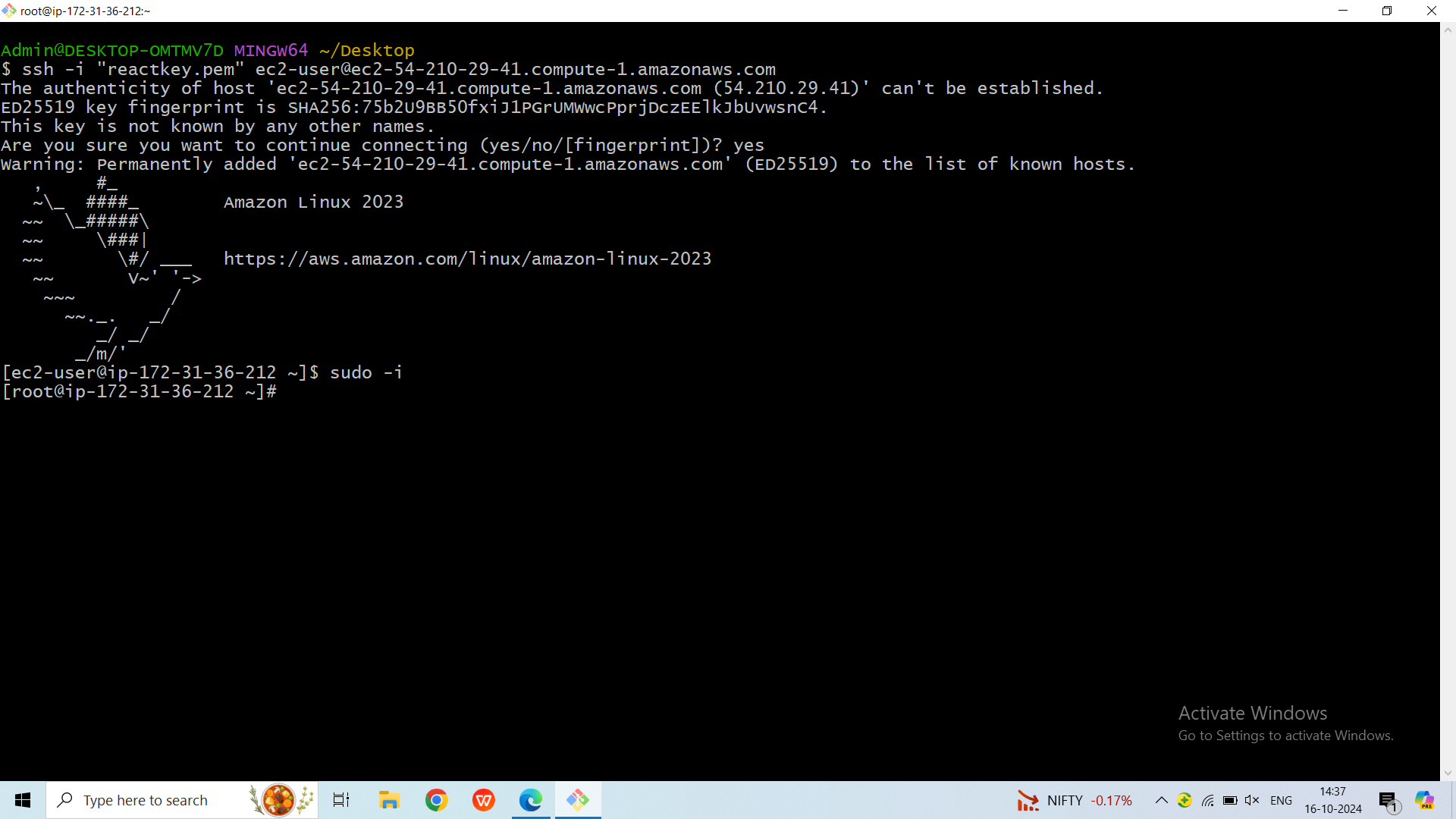




**Step 2: Connecting to the EC2 Instance**

Utilize your preferred SSH terminal to connect to the EC2 instance.

Normal user to root user we use **sudo -i**



To updating the package list : **yum update –y**

****

**Step 3: Install Node and NPM using nvm**

Install node version manager (nvm) by typing the following at the command line.

**curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh | bash**

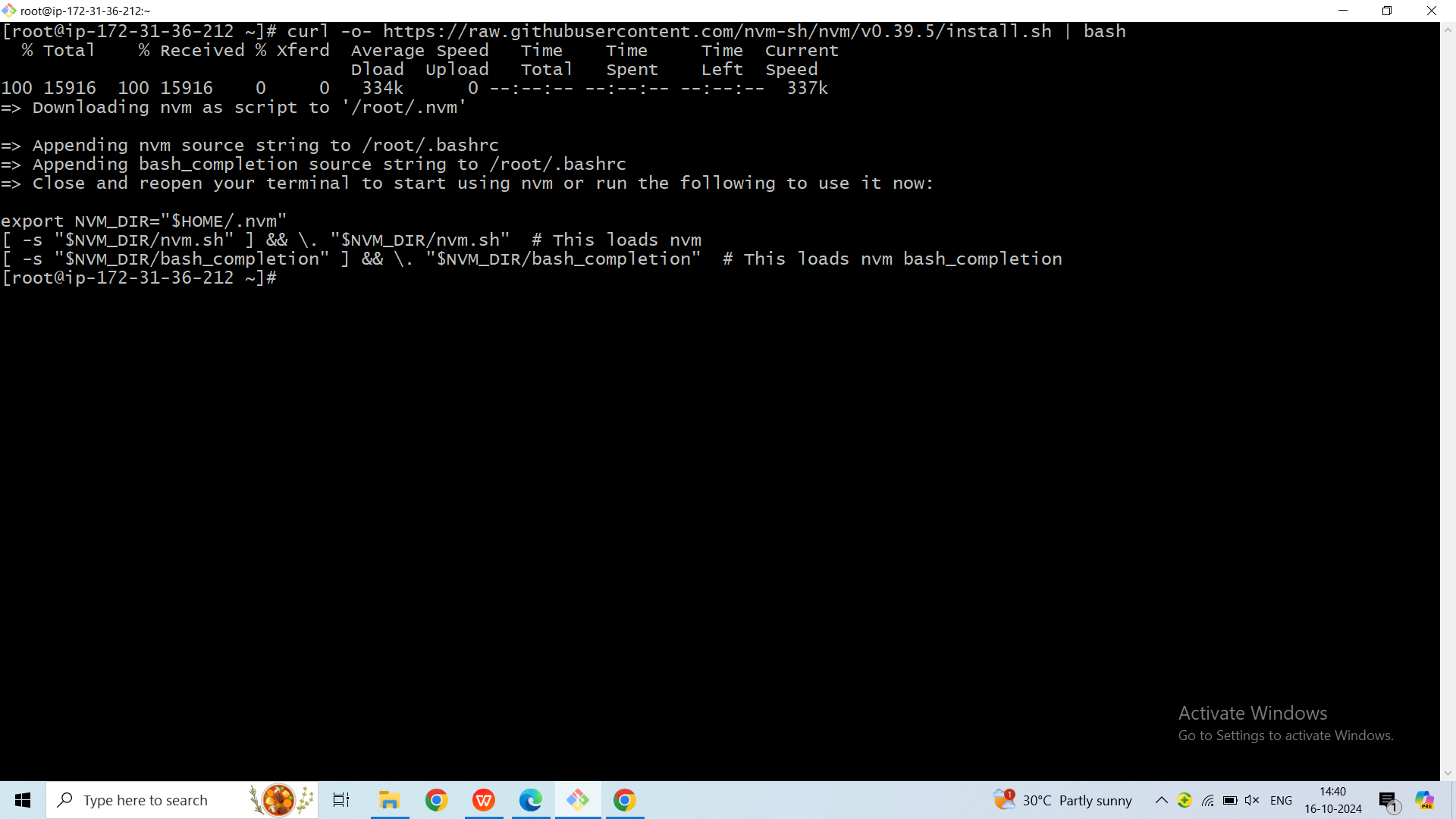
We will use nvm to install Node.js because nvm can install multiple versions of Node.js and allow you to switch between them.

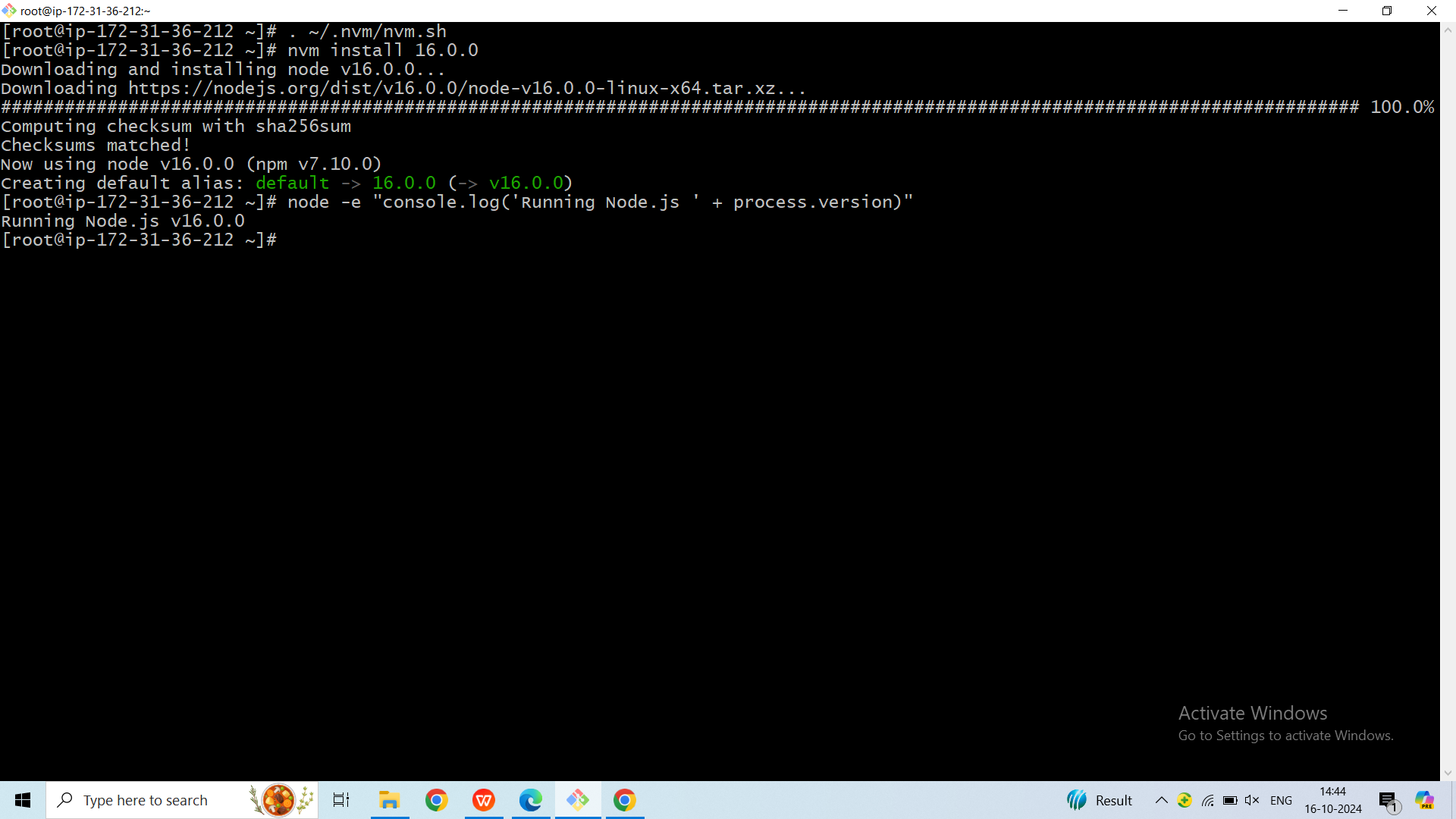
To activate nvm : **. ~/.nvm/nvm.sh**

Use nvm 16.0.0 which is a supported and stable version : **nvm install 16.0.0**

Installing Node.js also installs the Node Package Manager (npm), so you can install additional modules as needed.

Test that Node.js is installed and running correctly by typing the following at the command line: **node -e "console.log('Running Node.js ' + process.version)"**

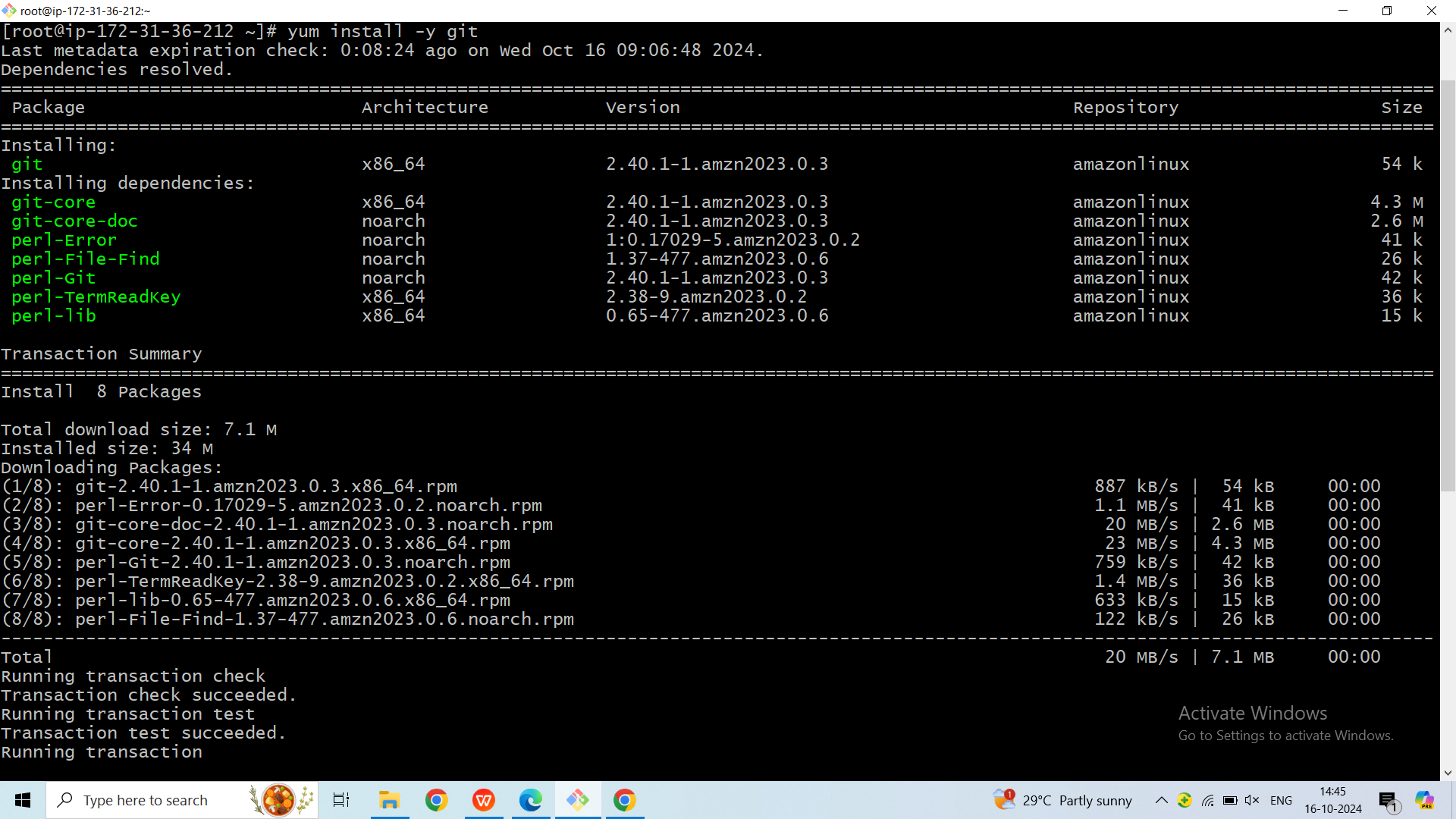
****

****

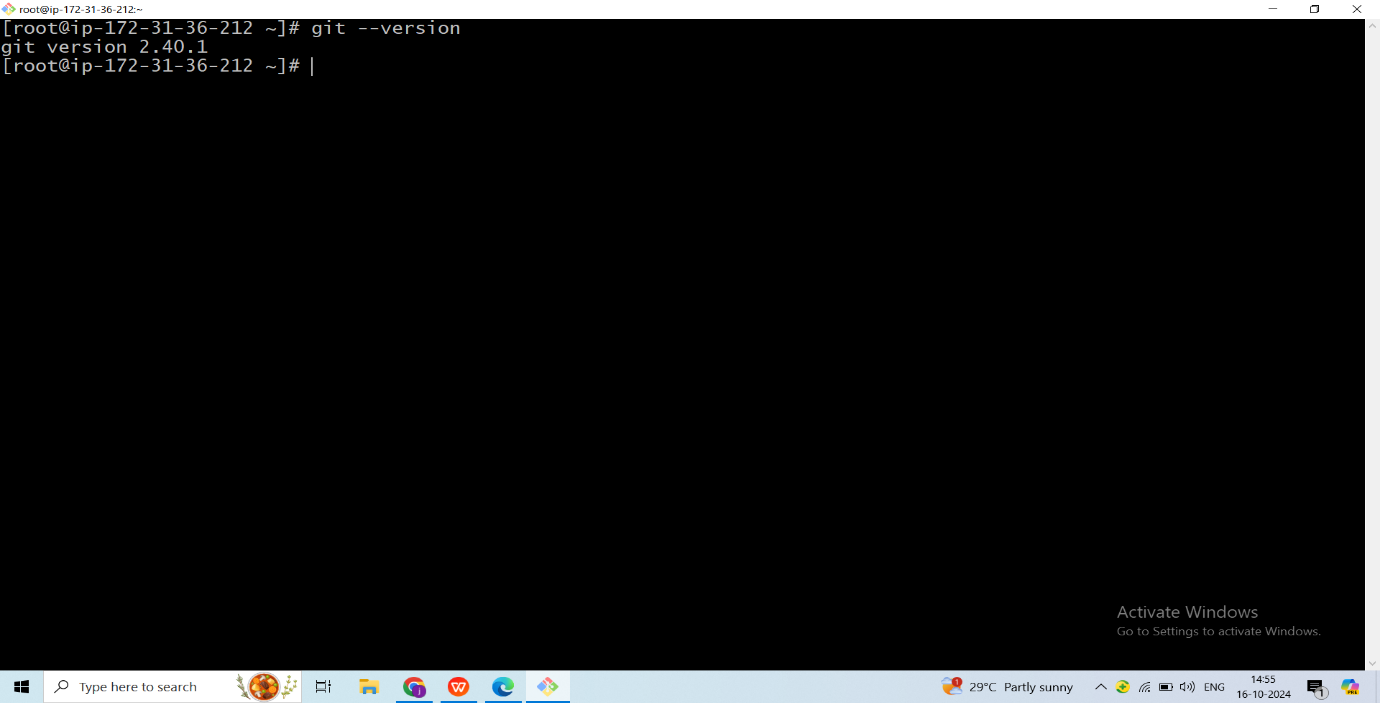
**Step 4: Cloning ReactJS App to EC2**

Before proceeding with the deployment, ensure you have Git installed on your EC2 instance. If not, install it using

**Yum install git -y**

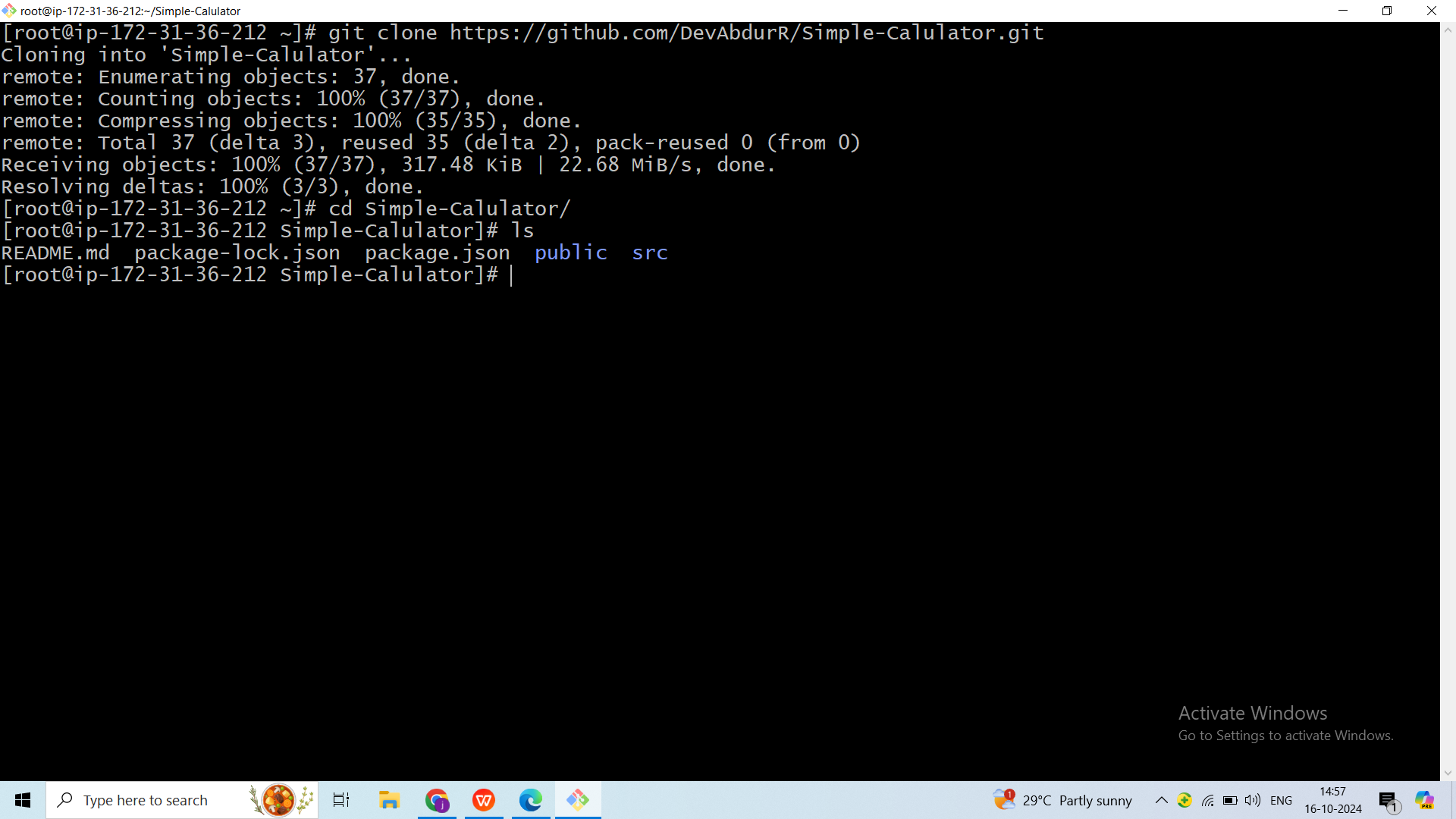
****

To check the git version we use **git --version**

Clone the ReactJS app repository (replace the GitHub URL with your actual repository). Here we have taken the open source react app from the GitHub

**git clone https://github.com/DevAbdurR/Simple-Calulator.git**

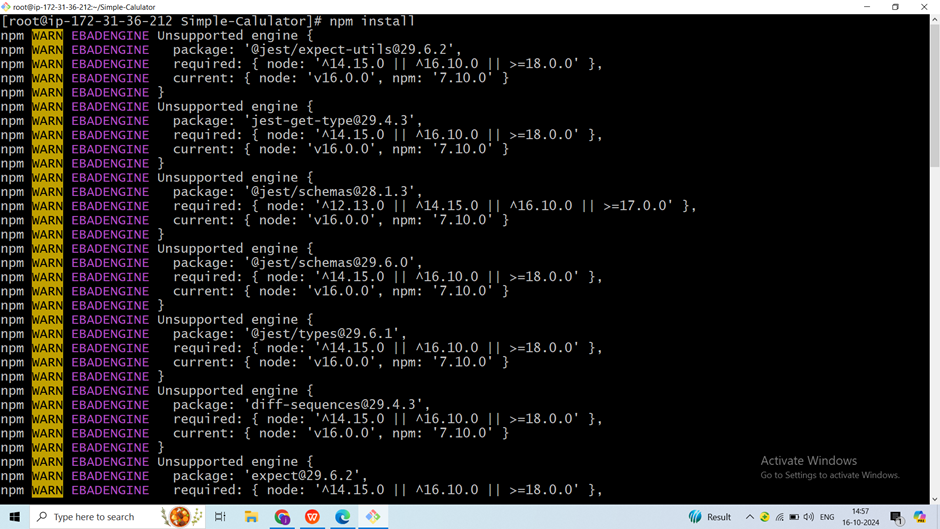
Move into the cloned directory :  **cd simple-calulator/**

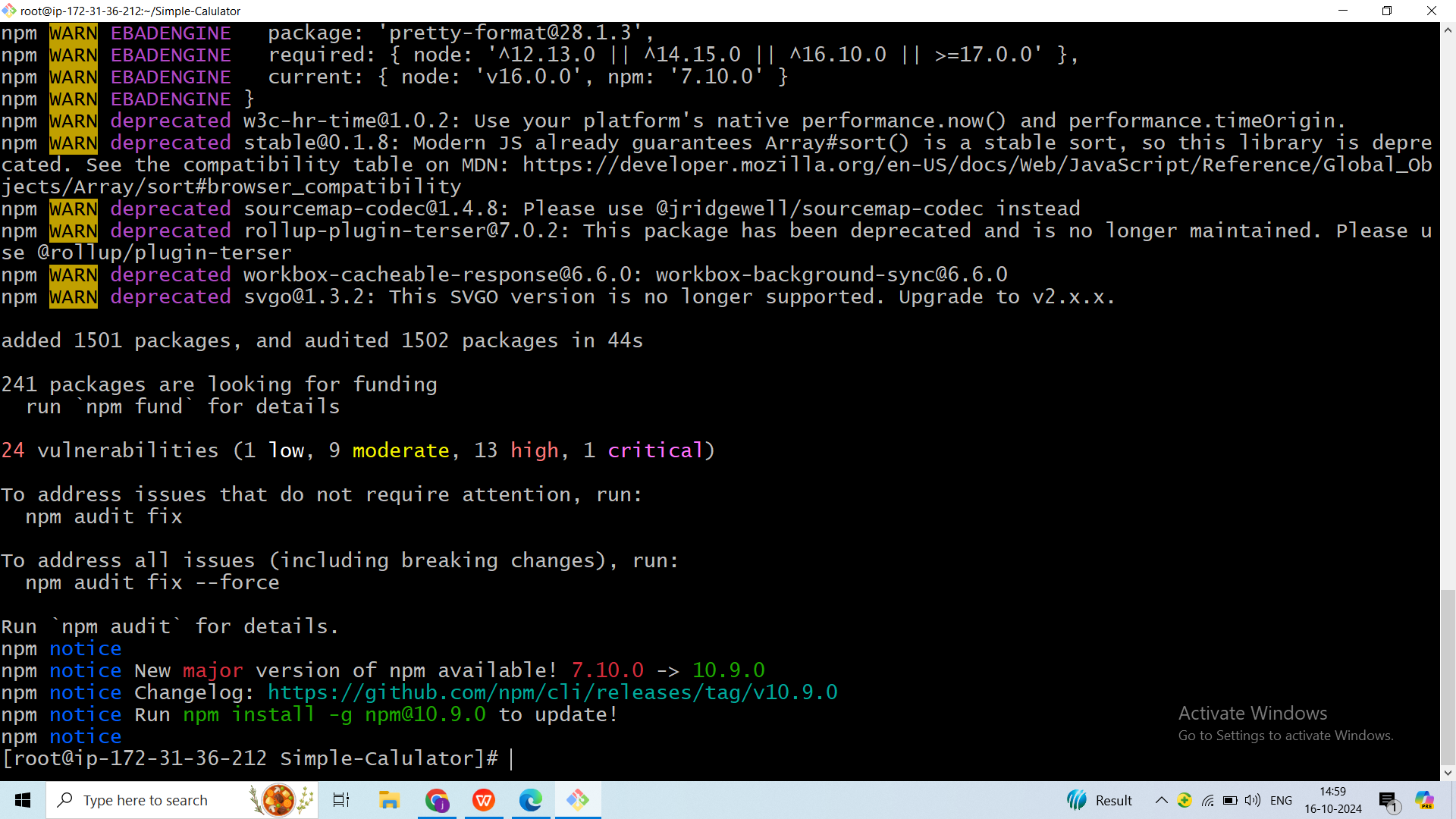


This ensures you have the latest version of your ReactJS app on your EC2 instance. Now, proceed to build and deploy the app.

**Step 5: Install all the required dependencies**

To install the dependencies we use the command : **npm install**

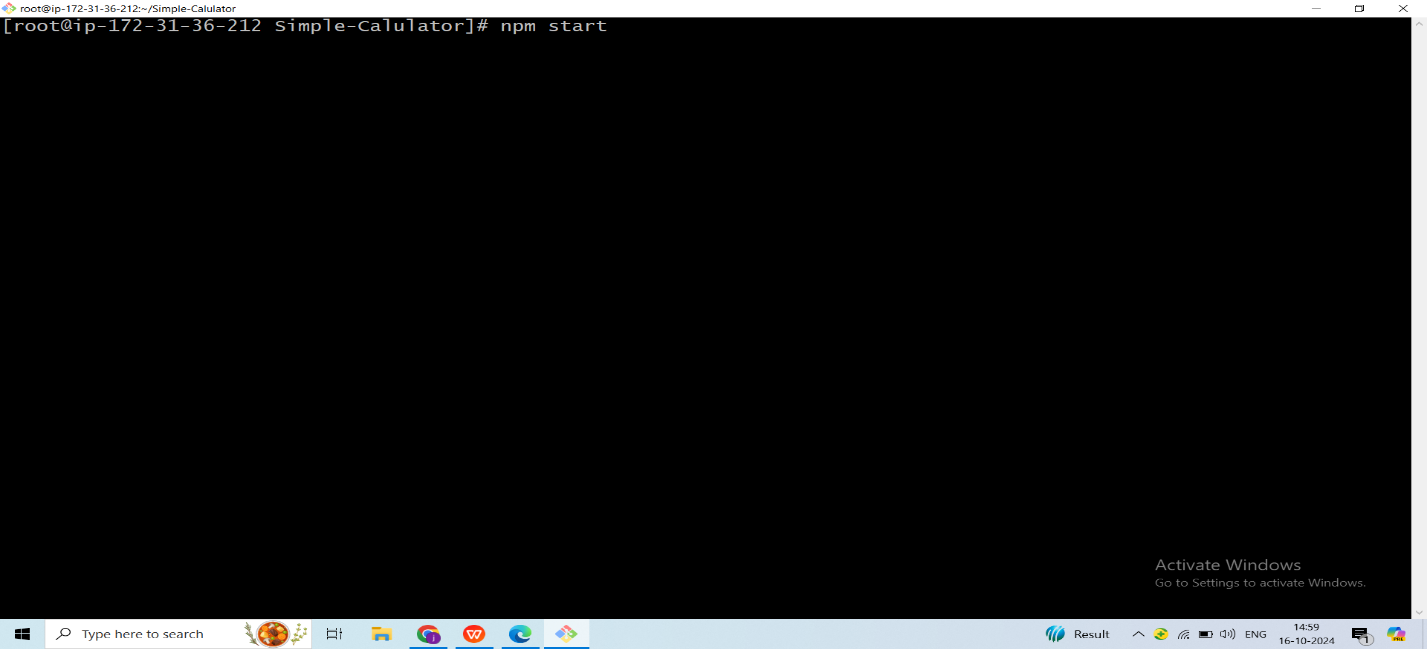


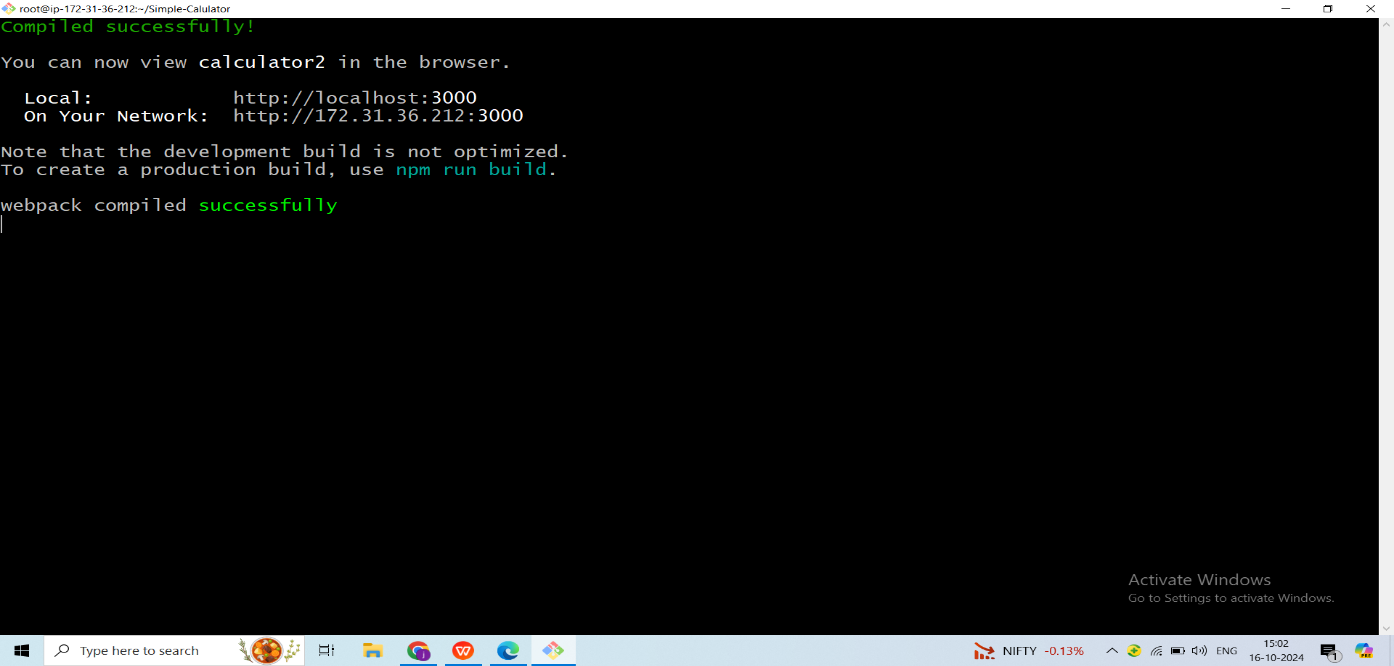


From the above output, we’ve successfully installed the npm packages for your project.

**Step 6: Run the application**

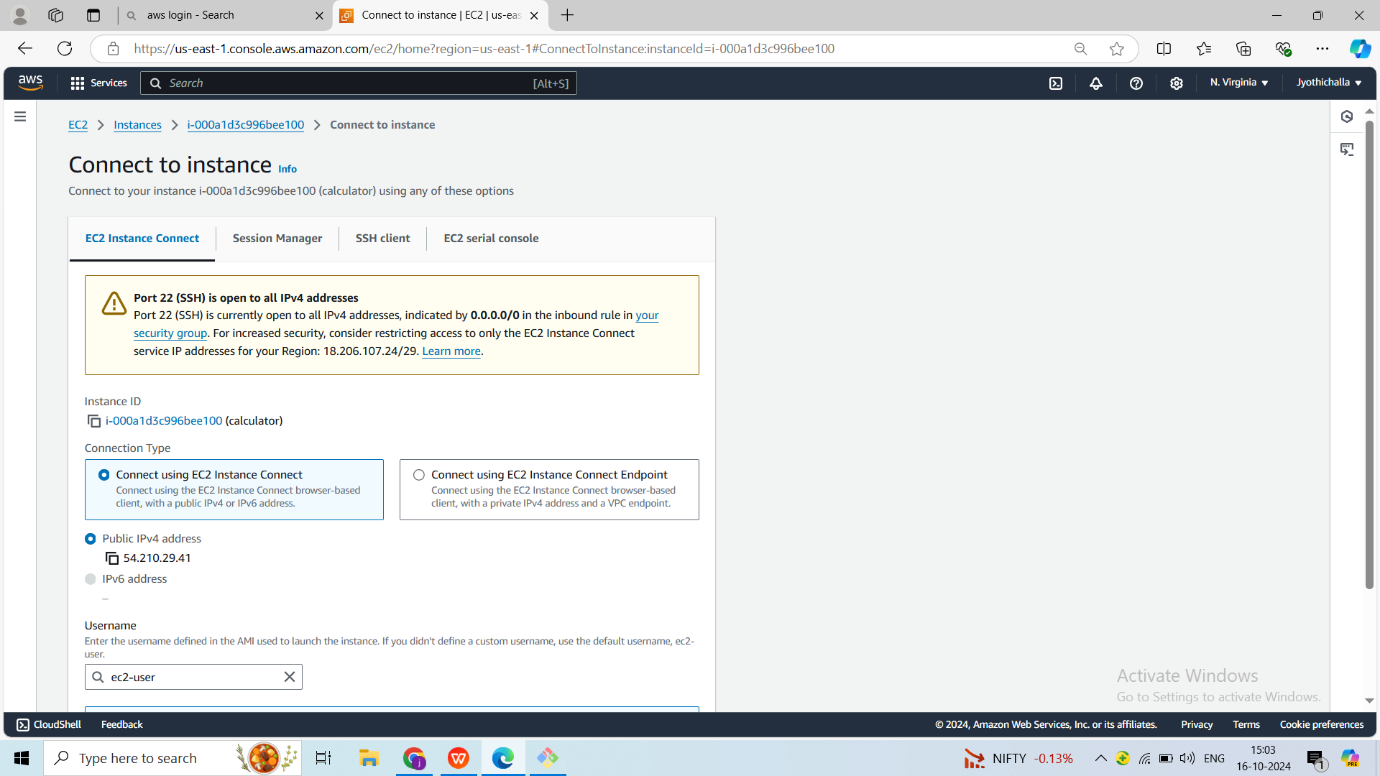
To run the application put the below command : **npm start**

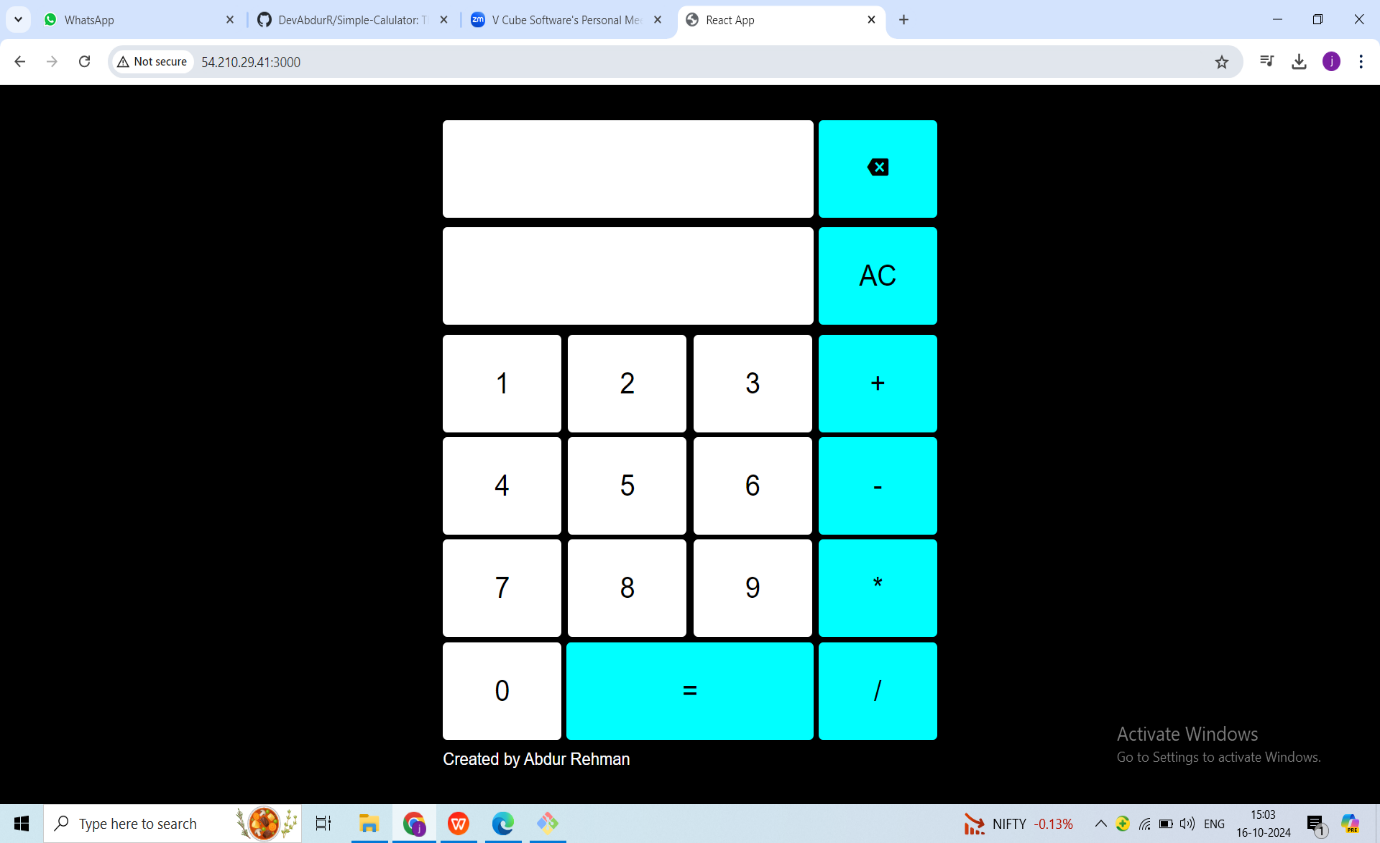
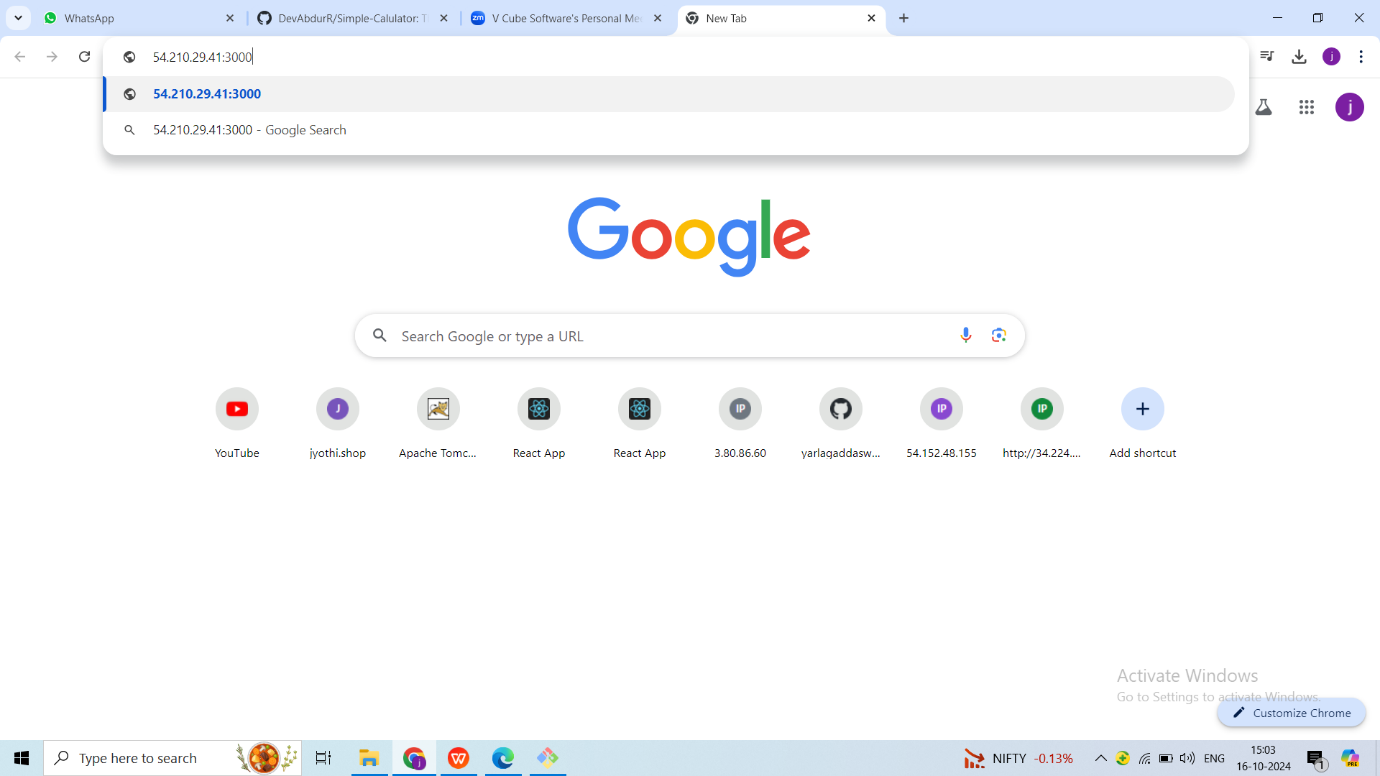




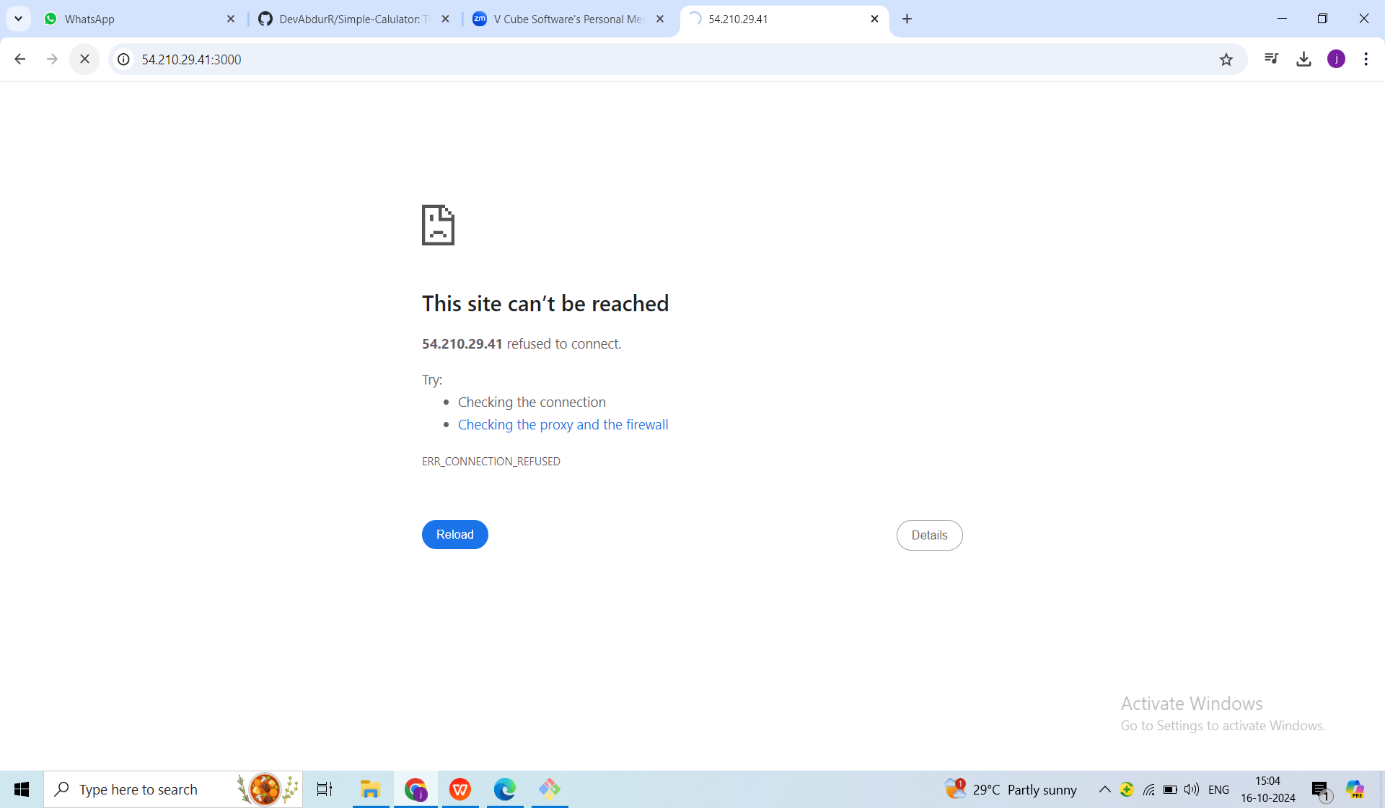
After we have to copy the public IP and paste it any browser to run our application.

We use **port number : 3000**





When we stop the server our application will not run.



When we install and run PM2 commands. We can run our application without our server also.

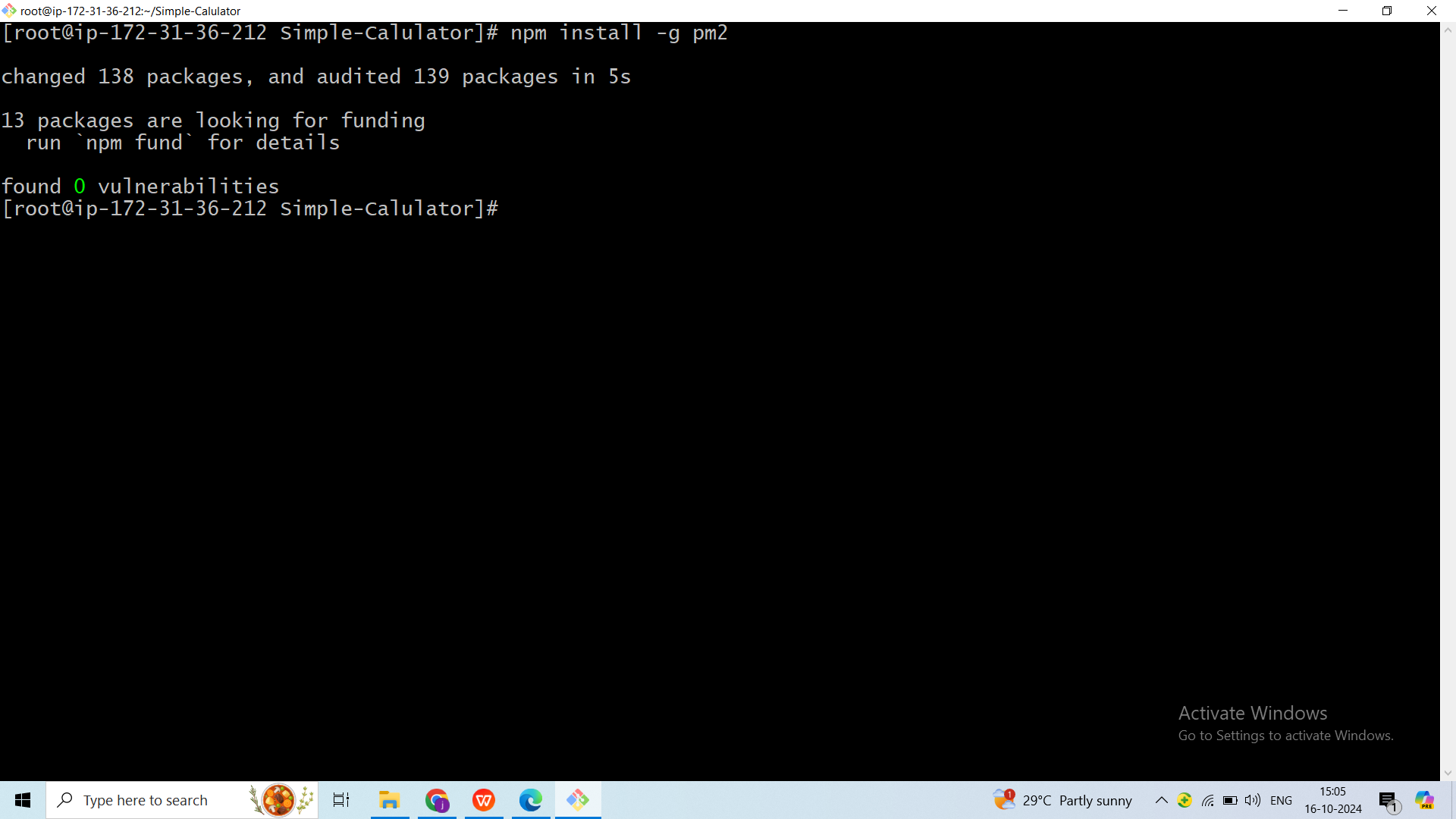
**Step 7: Using PM2 to Run and Manage React Apps**

Running a React app using pm2 offers several advantages, especially in a production environment. PM2 acts as a process manager for React.js applications, ensuring they run persistently, can be easily monitored, and automatically restart in case of failures. Here's a step-by-step guide on why and how to use PM2 with a React app:

Install PM2 Globally:

Open a terminal and install PM2 globally using the following command:

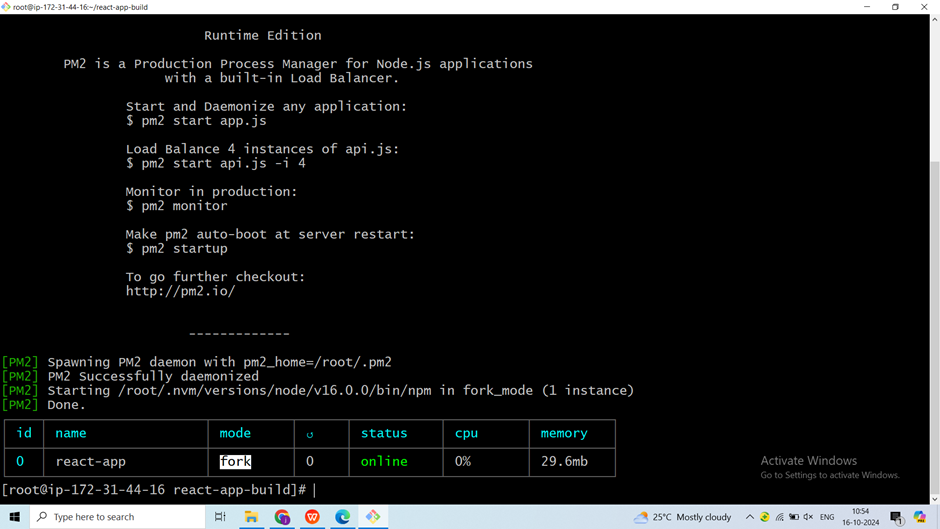
**npm install -g pm2**

****

**Start Your React App with PM2:**

Use the following command to start your React app using PM2:

**pm2 start npm --name "react-app" – start**

****