To download from git and run:   
npm install and npm run dev

Created an instance with ubuntu image t2.micro, allowed ssh traffic and https and http traffic

ssh -i "pvtkey.pem" [ubuntu@ec2-18-223-22-210.us-east-2.compute.amazonaws.com](mailto:ubuntu@ec2-18-223-22-210.us-east-2.compute.amazonaws.com)

installed Jenkins: <https://www.jenkins.io/doc/book/installing/linux/>

sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \

https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

/etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update

sudo apt-get install Jenkins

Step -3 Start jenkins

sudo systemctl enable jenkins

sudo systemctl start jenkins

sudo systemctl status jenkins

then open up port 8080 in security groups

A screenshot of a computer

Description automatically generated

password is saved in **ubuntu@ip-172-31-43-74**:**~**$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword

A screenshot of a login page

Description automatically generated

admin

To commence the integration of Jenkins with GitHub, follow these technical steps:

1. Install Jenkins and access its dashboard by logging in.

2. Navigate to create a new project and select "Freestyle project" as the project type. Provide a suitable project description.

3. For configuring source code management, choose Git as the version control system.

4. Generate a key pair using the **SSH-keygen** tool. This will create a public key and a private key. cd to .ssh

5. Add the generated public key to your GitHub account. This can be done by navigating to your GitHub user settings, then SSH and GPG keys section, and adding the public key there.

6. Provide Jenkins with the private key. This typically involves adding the private key to the Jenkins credentials system.

7. Specify the branches that Jenkins should build. In this case, set it to build the "main" branch.

8. Save the configuration.

9. Upon saving, Jenkins will automatically clone the repository from GitHub to the Jenkins server, bringing all the necessary files for the project.

By following these steps, you have successfully integrated Jenkins with GitHub, enabling automated builds triggered by changes pushed to the configured branches in the GitHub repository.

Challenge I encounterd:

The node version was 12 I had to use 21  
I solved is by

sudo npm cache clean -f

sudo npm install -g n

sudo n stable

sudo n latest

sudo node --version

sudo npm install

sudo npm run dev

You encountered a version mismatch requiring Node.js 21, while your system had version 12. To resolve:

1. Clear npm cache: `sudo npm cache clean -f`.

2. Install 'n': `sudo npm install -g n`.

3. Switch to stable Node.js version: `sudo n stable`.

4. Upgrade to latest Node.js version: `sudo n latest`.

5. Verify Node.js version: `sudo node --version`.

6. Install dependencies: `sudo npm install`.

7. Run development server: `sudo npm run dev`.

Installing the project  
sudo apt install npm

https://askubuntu.com/questions/426750/how-can-i-update-my-nodejs-to-the-latest-version