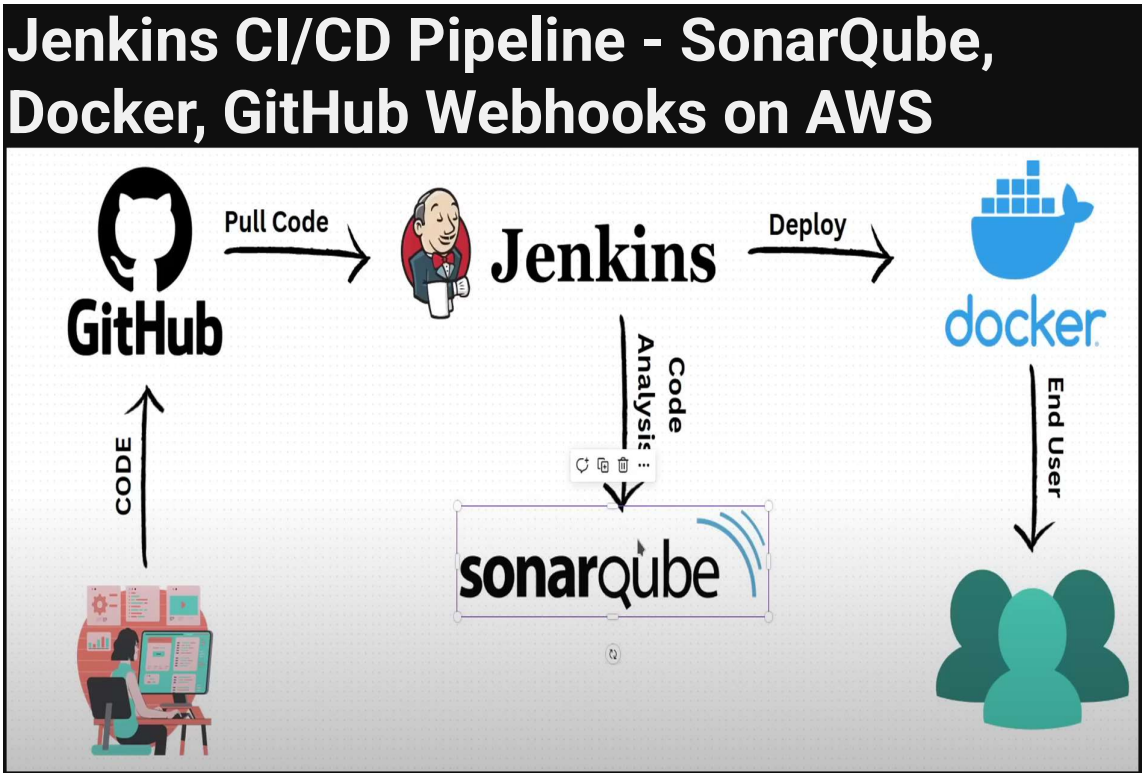

Demonstrates the process of creating a CI/CD pipeline using Jenkins, SonarQube, Docker, and GitHub webhooks on AWS. The pipeline involves pushing code to GitHub, pulling it through Jenkins, testing it on SonarQube, and deploying it on a Docker container. The presenter also shows how to access the deployed website on a browser.

- Creating a CI/CD Pipeline with GitHub, Jenkins, Sonar Cube, and Docker
- Setting Up Jenkins on a Virtual Machine
- Setting up a Jenkins Pipeline and SonarQube Server
- Setting up SonarQube on Linux and Configuring a Project
- Setting up SonarQube Scanner and Jenkins Integration
- Setting up Docker on EC2 Instance and SSH Configuration
- Configuring Jenkins to Execute Remote Commands on Docker Server
- Setting up Docker and Running a Container - Step by Step Guide

We will be creating a ci/cd pipeline in which we will push the code to the GitHub and from the GitHub we will pull the code through Jenkins and after pulling the code from the GitHub we will test the code on Sonar Cube which is a static code analysis tool on which we can scan the bugs and vulnerabilities of

the code and also we can add the rules regarding the scanning and then after scanning the code if our code pass we will deploy it on docker .

Everything will be done on ec2 instance so you will need three ec2 instances one for the Jenkins one for the SonarQube and one for the docker deployment so then after deploying on Docker we will access it on our browser just to verify if it's working or not so I have a website here which I've downloaded. It is the basic HTML template and if you want to download template then you can search for free CSS and here's the website that you can download the templates from.

You can just download any template that you want so after downloading that template just extract it anywhere on a desktop after accepting the template what we have to do is we have to log into the GitHub and after logging in the GitHub just create a new Repository so I've created a repository called Jenkins sonar Cube docker and in which I've also uploaded the code and after pushing the code to the repository what we have to do is we have to create three ec2 to install one for the Jenkins.

Let's just create an ec2 click on launch instance and just give it any name that you want let me just give it Jenkins and from here just select the operating system that you want to use so I will be using Ubuntu. So, I will click on Ubuntu and after selecting the operating system just select the type of the instance so I'm going to select T2 medium which has two virtual CPU's and 4GB memory and after selecting the instance type, we have to create a key pair so we can SSH into.

The virtual machine so give it any name that you want let's just give it SSH-Jenkins give it any name you want and then click on create key pair and as you can see the key is downloaded in the downloads folder and after selecting the key pair just scroll down and click on launch instance and it will take only a few minutes only a few seconds to launch and let's just create another instance while it's in pending state to click on launch instance again and then give it a name SonarQube

and then select the operating system also you have to keep in mind that now you consume a lot of memory so make sure you have at least two or four GB of memory just for the SonarQube then select the instance type and from here we're going to select the key that we have created earlier so we're not going to create another key we're going to just use the older one and after selecting the

key just click on launch instance so this is also creating and once it's created just let's just create another one for the docker

give it the name docker server and from here select the image that you want like the instance type for the docker I will be using T2 medium also and then select the key pair and then click on launch instance and once the instance is created as you can see the SonarQube is in running State and the Jenkins is in also running state so let's just SSH into the Jenkins instance just click on the instance ID that you want to access it and just copy its public IP and after copying the IP just open your terminal and from terminal just go to the downloads folder by typing

pass the key that we want to use so what was the name of the key that we created earlier it was SSH key Jenkins so SSH key Jenkins and after the key we have to type the username so whenever we installed the Ubuntu on ec2 it creates a user named Ubuntu by default so we're going to use the default user and then pass the IP address and here we will get an error as you can see we got a permission denied error it says bad permissions for the key so it says the permission 0664 for the SSH key are to open it is required that your private key files are not accessible by others so what we have to do is we have to give it the permissions so it can only be accessible by the current user so to change the permissions you have to type CHMOD 4400 and then the name of the key and now we will be able to access it into the machine as you can see we are in the virtual machine now so after getting into the virtual machine you have to type sudo apt update so we can update it and then we're going to install the Jenkins on it

and once it's updated, we're going to install jdk11 open jre11 Java runtime environment which is required for the Jenkins solution so while it's updating let's just open jenkins.io on our browser and after opening the jenkins.io website click on documentation and click on installing Jenkins and after clicking on the installing the engines select your desired operating system in my case it's Linux so I will click on Linux and from here it says Debian/Ubuntu so click on Ubuntu you want to inform here just copy the whole command and before pasting the command we have to install Java runtime environment go to install the Java runtime environment

the open jdk is installed successfully and now we're going to install the Jenkins just paste the Jenkins command and hit enter and while the Jenkins is installing let's just allow the port 8080 so we can access it on our browser just go to the

ec2 instance and from here just scroll down a little bit and you will see security groups in the security section and just click on the security groups and from here just scroll down and click on edit inbound tools so we're going to allow the port 8080 on this specific virtual machine just click on ADD rule from here select custom TCP and here give it the port that you want to open in my case it's 8080 which is the default port for the Jenkins and click on Save rules and we have successfully added the security group rule and now let's just verify the Jenkins is installed or not type system CTL status Jenkins and here you can see it's running and from here you will see

something like this what you have to do is copy the secret key here just copy it using the Ctrl shift C command and then go to the instance and copy its public IP so just copy the Jenkins public IP and paste it here and write the port 8080 and as you can see we can access the Jenkins web page it's loading it will take less than a few seconds so while it's loading let's just copy our token it's taking a while to load let's just refresh it again and it's still loading

so here so if you forgot to copy the secret key and it's not visible in the system CTL status Jenkins command then what you can do is you can go to the certain path which is mentioned here and what you have to do is just copy the path and just type cat and then the path so it says permission then I just write sudo before the CAT command and here you can see the keys also the same key which we copied from the system CTL command just paste the key and click on continue click on install suggested plugins

now the plugins are installing in the background and while the plugins are installing let's just look what we have done till now so we have created a server for the Jenkins and we have also pushed the code to the GitHub now what we are going to do we're going to pull the code that we have pushed to the GitHub repositor in the Jenkins server and then we're going to set up the SonarQube server

and from the SonarQube we will test the code if it's okay to deploy and if the code is passed then we will deploy it on the next instance which will be on docker so here the plugins are installed just create a user give it a username and give it any password that you want and give it the full name just give it any email address that you want then click on Save and continue and click on finish now we can start using Jenkins so first what we have to do is we have to create a pipeline and to create a pipeline just click on new item and from the new item just give it any name that

just give it automated pipeline and select the project type so in my case I will be creating a freestyle project and you can also look at the other projects but it's not in the scope right now so I will be just going with the freestyle project click on OK and once the project is created what we have to do is scroll down a little bit and from here you will see source code management you can also see it from here if you click here it will just bring you here and from the source code management we will be using git so we will click on get and here it's asking for a repository URL just go to the repository and click on code and then copy the URL from the https tab and then just paste the URL here and give it a branch so our branch is MAIN so I will just write Main and after adding the branch just click on Save and we also have to do one more thing here so just go to the configure again and just from here you will have to click on GitHub hook trigger for SCM Cooling so what this will do is it will just trigger

the pipeline automatically whenever we make changes to the Repository so just enable this option and go to the Jenkins by GitHub repository and from the repository just click on the settings from here we will be adding a GitHub webbook so whenever we make changes to the code it will trigger the pipeline automatically and from here just look for web hooks here just delete the older one so it will ask me for the authentication let me just verify the access from my mobile phone

so I've approved it from my phone and now it should login within a second yes as you can see we have logged in now click on ADD web book and what we're going to do here we're going to copy the Jenkins URL which is the IP and the port just paste it and after pasting it type slash GitHub hyphen webhook slash and then scroll down and from here click on let me select individual events and from the events we have to allow the pull requests and the pushes and then click on ADD web book so now let's just verify the webbook if it's working or not .

So let's just build our pipeline without the web first verify if it's working or not click on build now and as you can see the status is success so the pipeline is running successfully so now we're going to verify it using the webbooks just go to the repository and make any changes that you want first let me just verify and show you something so when you're in the pipeline just click on workspace here you can see we only have three folders two folders and two files which is one is HTML file and one is the config. file and two are the folders one for the

kit and one for the HTML website so let's just create a file here click on create file and just give it any name test Dot txt and just write something this is a test to verify the GitHub and just click on Commit new file and it should trigger the pipeline let's just go to the Jenkins again and verify as you can see the pipeline is in pending state and it is successfully let's just verify as you can see the pipeline is working fine so what we have done we have automated this process till here so whenever the developer changes the code or adds some file or

anything in the code or in the repository it will trigger the Jenkins automatically and Jenkins will pull the code from the GitHub we have done till Here and Now what we're going to do we're going to create a server for the SonarQube so let's just go and copy its public IP then go back to your terminal just open a new tab and from here you have to do the same step SSH hyphen I and the name of the key which was SSH key Jenkins and then the username and then the IP of the instance.

So we are into the SonarQube machine now let's just change its hostname updating the repository we are going to install sonar Cube and for the sonar Cube we need Java 11 so to install the Java runtime environment 11 while the Java is installing let's just go to the Sonar Cube website and after searching for the SonarQube just click on the download link and from here we're going to download the Community Edition so to download the Community Edition just right click on the download for free and copy the link

and after copying the link we have to paste it here and download it using the wget command so let's just paste the URL here and hit enter and as you can see the SonarQube is downloading right now and let's just see so here we have a zip to unzip it we have to install a utility called unzip so to install it type sudo apt install unzip and it's installed right now and to unzip it just type unzip and the name of the zip and it's I'm compressing it right now and as you can

SonarQube folder and from here just go to Linux so I will go to the Linux directory and from here we have a test file called a batch file called sonar.sh so we're going to execute it using the dot slash sonar dot sh while the sonar cube is starting let's just allow its port on the ec2 instance so let's just go back to the ec2 instance and from here go to security and click on Security Group and from here we're going to add an inbound rule which will be 9000 port the SonarQube and from here we're going to select the source which will be 0.0.0.0

it means it will accept the traffic from all IP's and give it a name SonarQube and then save so as you can see the SonarQube is started and let's just verify the SonarQube by going to the ec2 instance and copy the IP of the SonarQube paste the IP and write the port 9000 so the SonarQube is working fine as you can see the page is opening so the default login username and password for the SonarQube is admin and the password is also admin and now just write the old password which is admin and replace it with the new one

so I will also set the new password to the admin and then click on update so I think we have to write some other password so I will just change the password so now we have logged in into the SonarQube and from here just click on the project type is manual so we will click on manually and give it a name so I will give it the name abcd website and then the branch also we have to give the project display key website scan and it should just give it the same project key here and now click on setup

and after clicking on setup we have to use the CI method that we're going to use so we will be using Jenkins click on Jenkins here and from here choose the devops platform which is GitHub and just click on configure analysis and then it's telling us the steps that we have to use just skip it and click on continue and also click on continue here and from here click on other so we are using the HTML project so we will click on the other and if you're using the Maven or dot net project then you can select it so I'm selecting the other and from

here we have to copy two things so we have to open a text editor and copy this project key and paste it for the later use so we will be using it later in once you've copied the copied this just click on finish the tutorial and from here after click on clicking on finishing go to the admin user on the top right corner and from here click on my account and after going to my account just click on security and from here we have to create a token so I will just give it a name Jenkins token

or we can also give it the name SonarQube token and then select the token type so if you select the project analysis token it will only be specific to the specific project for example we have to select the project also so this token will only be authenticated for this project but if you select the global analysis token it can scan all projects that are in your SonarQube solution so here you have to choose the expiry let's just select it 30 days and click on generate token and from here just copy this token and also paste it in the Notepad

and now we have created a token and what we have to do we have to go back to the Jenkins and from the Jenkins we have to install a plugin go to the manage Jenkins section and from here click on global tool configuration just go to the plugins not the global tool configuration we have to install the plugin first so just click on manage Jenkins manage plugins and from here click on available plugins here search for sonar scanner SonarQube scanner click on install without restart and we also have

to install another plugin as you can see the SonarQube is installing so while the sonar plugin is installing there let's just install the other plugin which is SSH to easy so our plugins are installing what we have to do is we have to go back to the Jenkins section

and from here we have to go to the global tool configuration and from the global tool configuration just scroll down and you will see sonar scanner click on ADD SonarQube scanner and give it any name that you want and check the install automatically and then save it so after saving it what we have to do we have to go to the configure system now and from the configure system just scroll down and you should see SonarQube servers so click on ADD SonarQube give it any name that you want

let me just give it so now our server and from here we have to give it the URL of the sonar Cube just copy this and paste it here and then save it now just go to the pipeline that we have created earlier and go to the configure and what we're going to do we're going to add a build step here which will scan execute the SonarQube scanner and here just ignore everything and paste the token here that not the token but the key also just paste the key here and save it and now we have to go back to the Jenkins and configure system because we also have to add the SonarQube token here which we haven't edited yet and from here we have to click on ADD and from here click on Jenkins and the token type will be the secret type and it will be secret text and here we have to paste the token that we have copied from the SonarQube and give it any ID so I will get give it sonar token and then save it then select a token from here and click on Save now go back to the pipeline go to configure and let's just see if we have any other thing to add so we don't have to add anything else now let's just build our pipeline to verify if it's working or not so as you can see it's currently running it now the SonarQube is scanning our code the scan is in progress as you can see from the Jenkins we also got the status success so let's just refresh the page and as you can see it scanned our code and our code

passed so once our code passed we will deploy it on the docker server and that's just go to our third ec2 instance in which we will be installing docker from here just click on the instance ID and copy its public IP then open the new tab for the terminal and then SSH-i and the name of the key username and IP so we are into the docker instance now let's just change its hostname type sudo just post name CTL set hyphen hostname docker then type in slash as you can see the host name of the server changed and now just clear the screen and run the apt update command and while it's updating let's just install docker go to install Docker let's just

go to the docker website or just search for Docker install Ubuntu and from here we have to start from here setting up the repository for the docker so we're going to be installing the tools that are required now as you can see we are into the Docker instance and now just paste the command that we have copied from the docker website and just type yes and let's just copy the other command to add the keys just paste the keys and now what we're going to do we're going to run the third command

then we're going to run the update command then we're going to install the docker so what we have done till now is we have pushed the code to the GitHub then we pulled it through the Jenkins then we scanned the code through SonarQube and after scanning the code we will be deploying it on the docker and then we will access it on our browser so now we're going to deploy the code from Jenkins to the docker server then we will access it now what we're going to do here let's just verify if we have the keys installed

so, we have to do two things on both of the servers the Jenkins and the docker server to first go to the Jenkins server and switch to the Jenkins user so now we are as Jenkins user so whenever we solve the Jenkins it creates a user named Jenkins let's just SSH the other server from here and verify if we can SSH it

we have to go to the docker server and from here just switch to the root user and from the root user we have to edit the ssh config files this is set the config and change the password authentication to yes then save it and restart the sshd service system CTL restart sshd now let's just verify it again and as you can see it's asking for the Ubuntu password so we have to set a password for the Ubuntu user which we haven't already so to change the password of the Ubuntu user type

the password is updated successfully and now if we type the password here, we are into the Ubuntu machine so exit it and then we're going to copy we're going to generate a key SSH key gen to generate a public and private key hit enter and we have generated a key now we have to type SSH ID and the name of the server so the keys are added now let's just verify it again

so as you can see we can get into the docker server now let's just go to the Jenkins and from the Jenkins we have to go to the managing section and from the Jenkins section we have to go to the configure system and here we will add the docker server here so it can execute the remote commands on the docker server and here we have to scroll down and look for server group Centre let's just add a group name which is Docker servers and the port and the username so the username is Ubuntu

and the password is whatever password you wrote in the possibility command then save it and go back to the Jenkins and configure system and now scroll down and from here what we're going to do we're going to add here we edit server group and now we have to add the server so let's just select the group select the group is Docker servers and the server name is Docker one and just give it the IP what was the IP of the docker server so just copy it from the ec2 and paste IP then save it

and now just go back to the pipeline and go to the configure tab and from the configure tab just scroll down and let's just verify it if we can execute the remote comm and from here just look for remote command here you can see remote shell just select the remote shell and select the target servers which is this and let's just run a command called touch test Dot txt let's just build the pipeline again to verify if it's working or not so it's seems to be working so now the code is being scanned by the SonarQube

and once it's scanned then it will run the remote command on the remote server so we got the success message let's just verify just type ls and as you can see we got that test.txt file here so now what we're going to do we're going to go to the repository and from here we're going to add a file click on create new file and here it will be a Docker file so just type Docker file in the name the D is capital and there's no extension of the docker file and from here we're going to use the image nginx and then we're going to copy the current contents of the directory to the user let's just go to the nginx Docker Hub website and from here we will see what's the default path for the nginx so the default path should be user share nginx HTML let's just copy the path from here and go back

to the GitHub and from here just paste the path and now we're going to save it commit new file and it should trigger the pipeline automatically so let's just verify it if it it's working or not so as you can see the status isn't pending

and it will execute the pipeline in a second so our pipeline is executing now and once the pipeline is executed what we're going to do we're going to go back to the configure tab and from here we have to run a command we have to let's just remove this remote shell and click on execute shell and from execute shells just type SCP so basically what we're doing we're copying the current contents to the remote server to secure copy command and then the directory which is dot slash static so we're going to we're going to be copying all the contents of the current directory to the remote server which is Ubuntu at the rate and then the IP of the Docker server then type call in and Slash uh where we we're going to be pasting the file so let's just create a folder here exit uh make a folder here called website or any folder that you want so now we have a folder called website here and what is the path of the folder let's just get its path which is home Ubuntu website

website and then just click on Save and let's just build it again so we got an error here called failed let's just verify it what caused our website to fail so this failed because we added a Docker file and it's finding the bugs in the docker file so let's just verify it again so as you can see it's failing because of the docker file so there's nothing to worry about and we should not put the docker file in the Repository but just to show you we are going to put the docker file in the repository but it's not recommended to put it in the same repository you can also create a different repository for the docker file and then you can move it to the remote server using the SCP command so we got a failure what failure we got so we have to add hyphen R after the SCP command so it will also copy the directory and the content inside the directory just go back to the pipeline click on configure and just scroll down and from here just type hyphen r and then save it then click on build now.

so as you can see we got the success message here our pipeline is successful and let's just verify here you can see the contents that we created are in the docker server now and now what we're going to do we're going to build the image and then run the container from it so just go back to the pipeline and go to the configure tab and from here just go to the execute just add a remote shell execute a remote challenge in which we will be just going to the Ubuntu website folder

and there we will run the docker build command so first we have to verify if we can run the docker command from here or not as you can see we're getting an error it says permission denied so if we run the same command with the sudo it will work but with the current User it's not working so to fix this issue what we have to do is we have to type a command called sudo user mode hyphen a small and the g is capital so we're going to be adding the current user to the docker group and

here's the name of the group and the username is Ubuntu then hit enter and then type new group docker and now let's just type the docker PS command as you can see it's working now so now what we're going to do run a command called Docker build the name of the image that we're going to give it and then we're going to run a container from this image so Docker run in detachable mode and so the port will be let's just give it 8085 and 80.

So this is the port of the container and this is the port of the system in which it will be exposed on the port 8085 and the container Port is port 80 and then give it a name and then the name of the image which is my website let's just save it and build it now so you can see the pipeline is executing right now building the docker image and now we got the success message let's just verify if it's our container is working fine or not so as you can see our container is working fine on the port 8085 but we cannot access it right now because we haven't allowed the port 8085 so to allow the port just go back to the ec2 instance and here go to the security and from the security groups just add an inbound rule which will be custom TCP and port 8085 and the IP address the IDR block which will be zero and the description will be your next website or you can give it any name that you want now save it and let's just verify it go to the instances copy the public IP the space the IP and write the port 8085 and as you can see the website is working.