Antonio Rios

Deployment 2

The purpose of this deployment is to demonstrate the process of building, testing and deploying an application on AWS using Jenkins as the CI/CD pipeline.

To achieve the deployment successfully we need to set up the environment as follows:

1. Jenkins installation.
2. Set up Jenkins as a user.
3. Create a Jenkins user on AWS.
4. Install AWS CLI on the EC2.
5. Install EB CLI(Elastic Beanstalk)
6. Connect GitHub to Jenkins server.
7. Create a multibranch build on Jenkins.
8. Deploy the application on elastic beanstalk.
9. Update deployment repository
10. Add changes to the application

Jenkins installation

First to set up Jenkins it's necessary to log in to an aws account and create an ubuntu EC2 with the correct security groups enabled.

EC2's settings and configuration:

- Select an ubuntu image.
- Most settings can be left on default.
- Security groups: allow access to ports 22, 80 and 8080.
- Configure RSA authentication for accessing the computer through SSH if necessary.
- To install Jenkins add this script in the user data section in advanced settings.

Set up Jenkins as a user

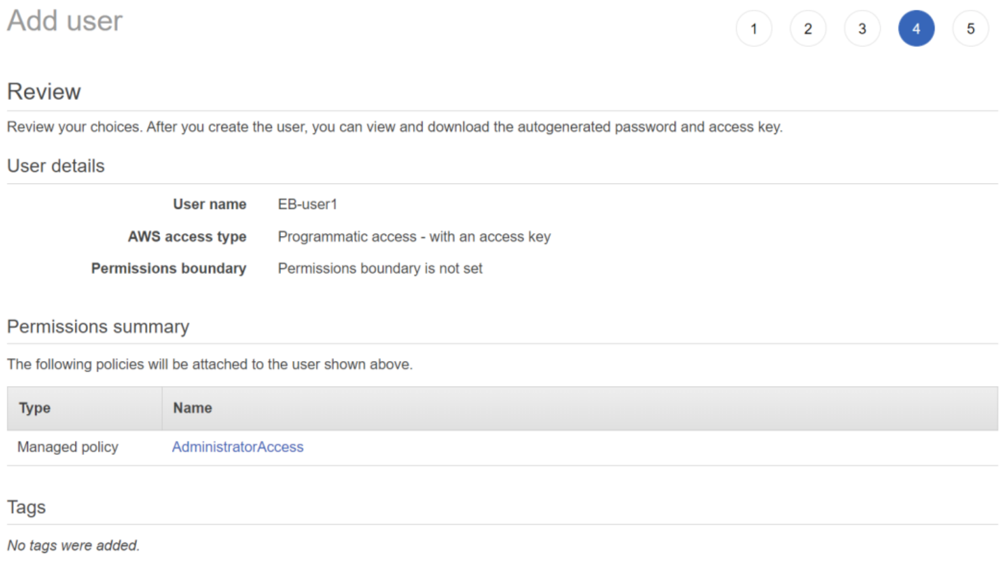
To set up Jenkins as a user within the EC2:

- Assign the "Jenkins" user a password.
- Assign the "Jenkins" user a shell, in this case is the /bin/bash shell.

Create Jenkins user on AWS

To create the user, head to **IAM** on the AWS console and under access management look for users.

- Add a new user
- Select programmatic access with access key



Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	EB-user1
AWS access type	Programmatic access - with an access key
Permissions boundary	Permissions boundary is not set

Permissions summary

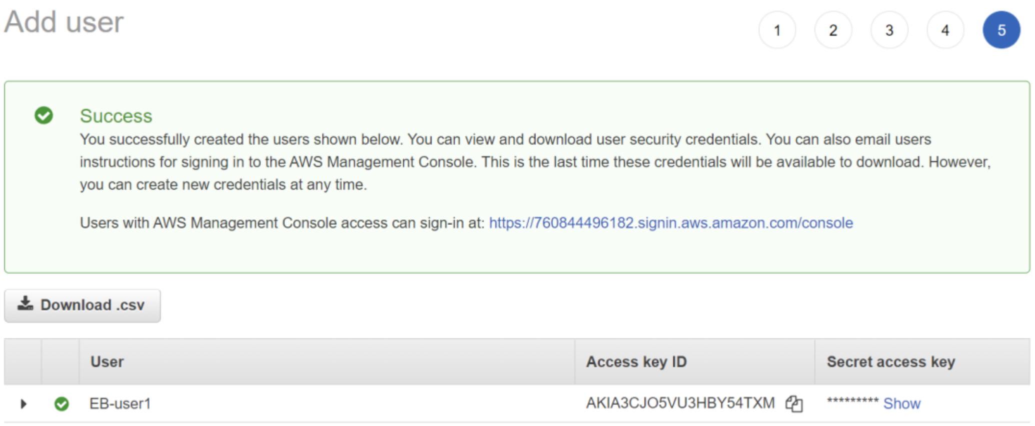
The following policies will be attached to the user shown above.

Type	Name
Managed policy	AdministratorAccess

Tags

No tags were added.

- Policies:
  - Select existing policies
    - Administrator access: The account needs administrator access to work properly
- Make sure to save security the **Access key ID and Secret Access key.**



Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://760844496182.signin.aws.amazon.com/console

	User	Access key ID	Secret access key
✓	EB-user1	AKIA3CJO5VU3HBY54TXM	********* Show

Install AWS CLI

To perform the installation, make sure to run the commands on root:

- Curl the repositories for aws cli
- Unzip the contents of the file on the EC2
- Install the application
- To verify that aws is installed run the command "aws –version
- AWS configuration:
  - Here we use the **Access key ID and Secret Access key**
  - Set region to us-east-1
  - And format to Json

EB CLI installation (Elastic beanstalk)

To run the commands, it's important to switch back to root.

The installation is managed by pip by using:

```
pip install awsebcli –upgrade –user
```

To verify successful installation of the program you can run

```
eb --version
```

Warning:

Important information:

It's important to add the path for EB to work properly using the following commands:

```
export PATH=~/.local/bin/:$PATH
```

```
source ~/jenkins
```

This fixes the issue if eb is not able to run in the Jenkins profile.

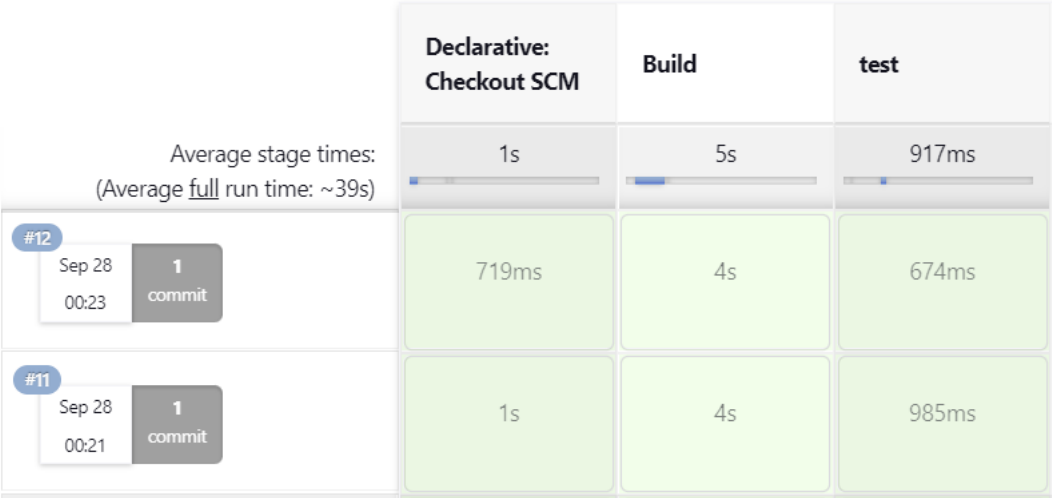
Connect GitHub to Jenkins Server

- Log in to GitHub
- Create an access token on the developer settings
  - Allow access to repo and admin:repo_hook

Create a multibranch build on Jenkins

- Create a new multibranch pipeline
- Add GitHub as the source
- Connect Jenkins to the repository using the token
- Validate the credentials of GitHub.
- **Make sure the mode is Jenkinsfile**

After the steps are successful. Jenkins will start building and run the test.



	Declarative: Checkout SCM	Build	test
Average stage times: (Average full run time: ~39s)	1s	5s	917ms
#12 Sep 28 00:23 1 commit	719ms	4s	674ms
#11 Sep 28 00:21 1 commit	1s	4s	985ms

Troubleshooting information:

When I finished setting up Jenkins it failed to build and test. It was still missing the dependencies for running the virtual environment. To fix the issue run the command below:

```
apt install python3.10-venv
```

This will add the missing virtual environment necessary to complete the building and testing.

Deploy the application on Elastic Beanstalk

- Switch to the Jenkins user
- Make sure you are in the workspace directory of the Jenkins profile
- Initialize elastic beanstalk with "eb init"
  - Select all default for the configuration except for code commit(N for code commit)

- Create the elastic beanstalk environment (“eb create”)
- Select the default configurations except for spot fleet (no for spot fleet)
- The environment will start creating after the selections.

The application and environment will be created on a different EC2 running elastic beanstalk.

Update deployment repository

The Jenkinsfile inside GitHub Must be updated to reflect the deployment stage.

- Add the path to elastic beanstalk followed by the deployment of the application.
- Update and commit the changes from github
- In my case I added a webhook to the repository to update the changes automatically.

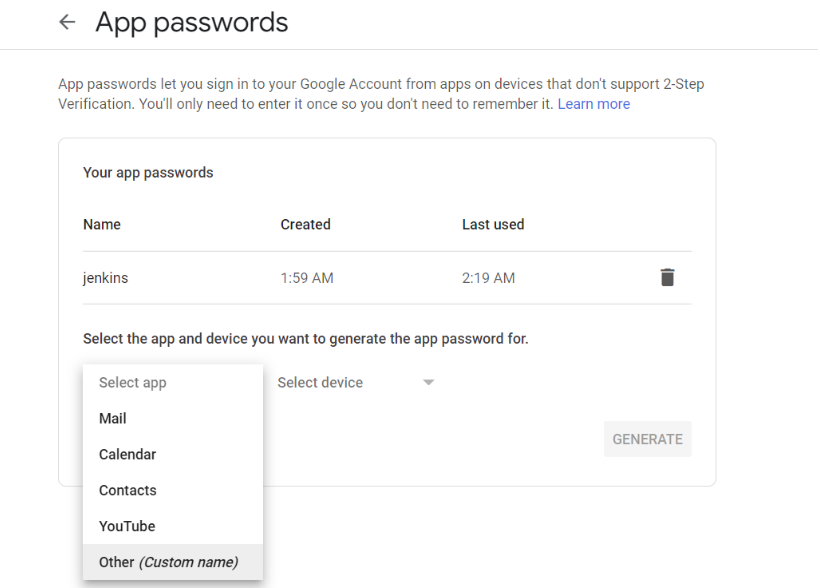
Add changes to the application

- I added a different color to the application front end by adding a new line for color in the html files

Add email notifications for builds and changes

This method applies for gmail accounts:

- Go to your google account
- Select security
- Select app passwords



← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. Learn more

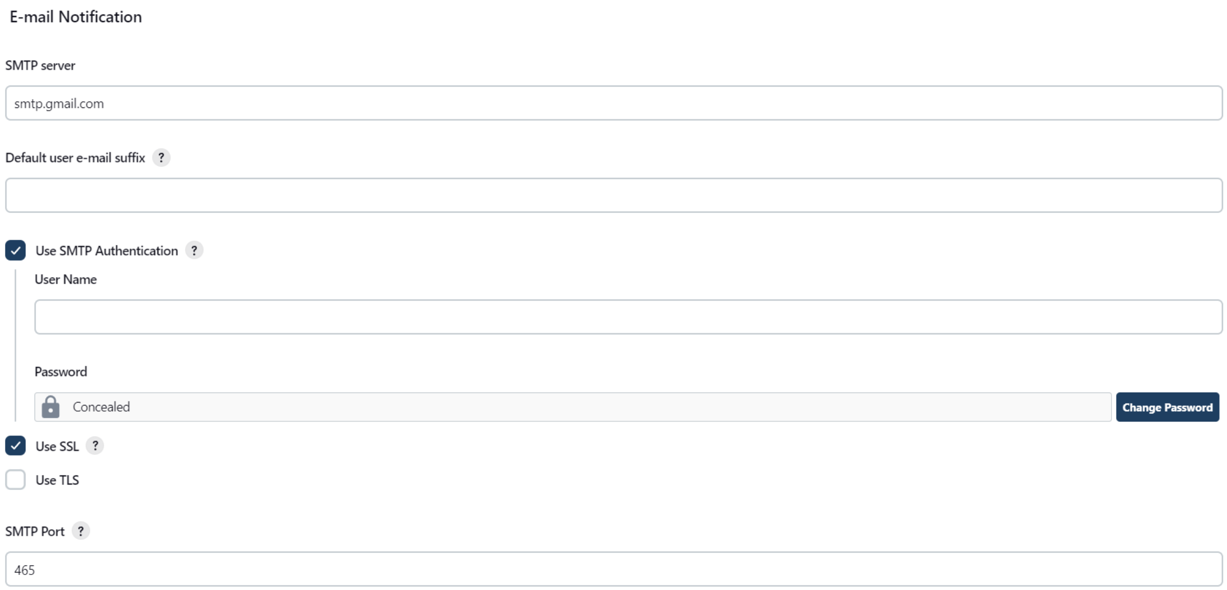
Your app passwords

Name	Created	Last used
jenkins	1:59 AM	2:19 AM

Select the app and device you want to generate the app password for.

- Select mail and other for the device

- Save the password as you won’t be able to see it again after.
- On Jenkins access the dashboard > manage Jenkins > Manage system
- Look for email notification



E-mail Notification

SMTP server

smtp.gmail.com

Default user e-mail suffix ?

☑ Use SMTP Authentication ?

User Name

Password

Concealed

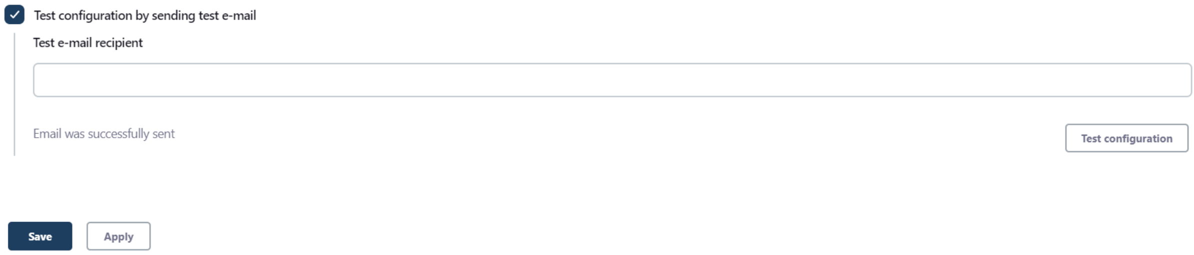
☑ Use SSL ?

□ Use TLS

SMTP Port ?

465

- Add the following:
  - Smtp server: smtp.gmail.com
  - Username: your email username
  - Password: the app password recently created
  - Check SSL and port 465
- Test email configuration



☑ Test configuration by sending test e-mail

Test e-mail recipient

Email was successfully sent

- Add a test email to check the configuration.

After the configuration is complete, every time there are changes in the repository, an email will trigger to notify of those changes and where it happened.

Improvements:

One way in which the deployment could have been improved is by adding more necessary steps into the script that builds Jenkins initially as well as adding as many of the dependencies that were missing later on. The more steps that are included in the script will facilitate the replication of this deployment. Automating the process can speed up the process of the whole deployment.

As far as notification they could be tailored to specific task using different applications like data dog and cloudwatch to monitor the system in case something fails. Right now it only notifies of changes in the repository and new builds but would not be able to notify that the application is down and needs to be turn back on.