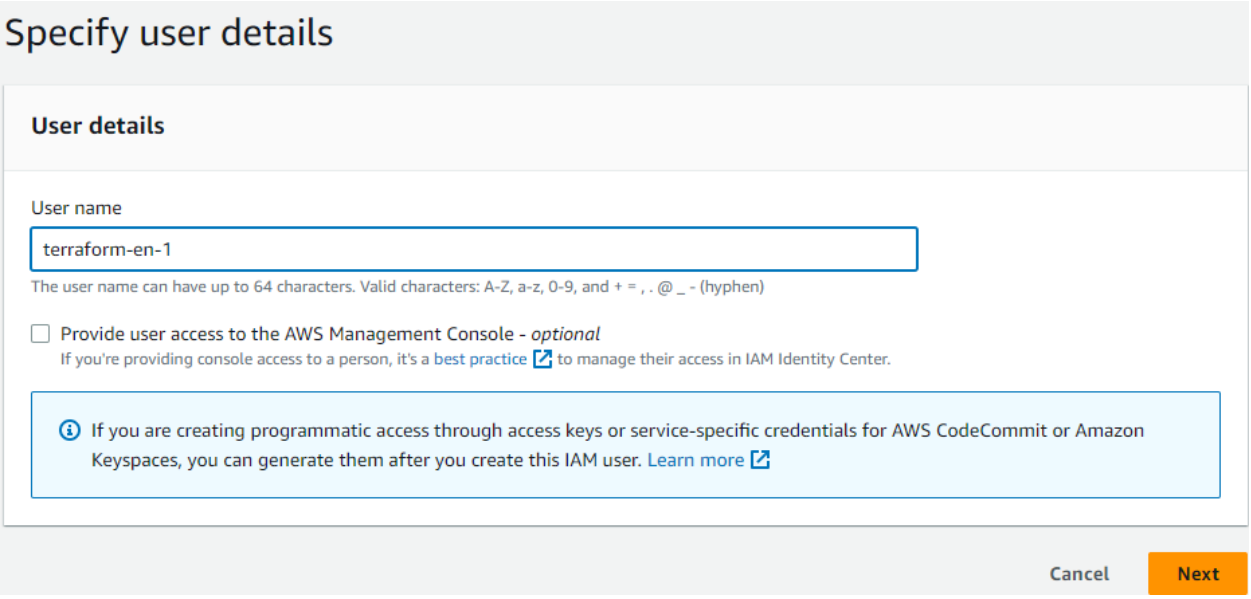
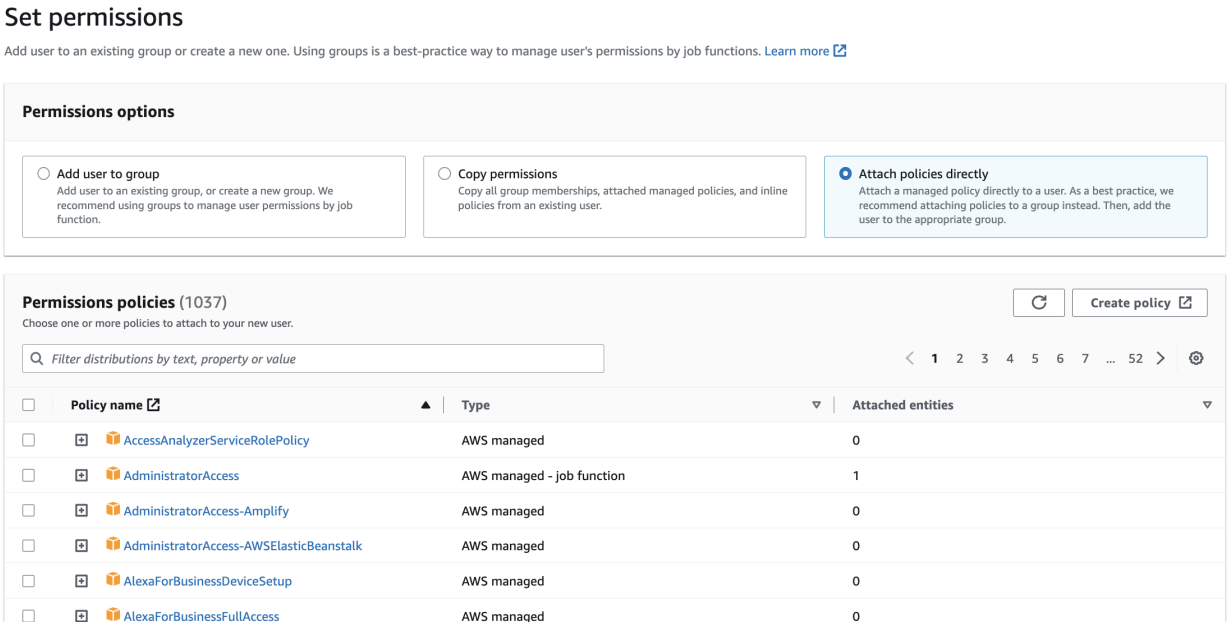
# Steps to implement Hands-on Project - Mission 1

## Amazon Web Services

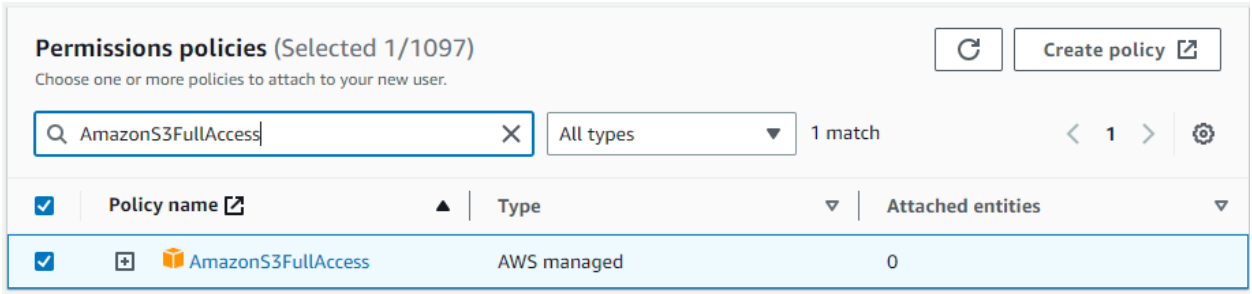
- Access AWS console and go to IAM service
- Under Access management, Click in "Users", then "Add users". Insert the User name **terraform-en-1** and click in **Next** to create a programmatic user.



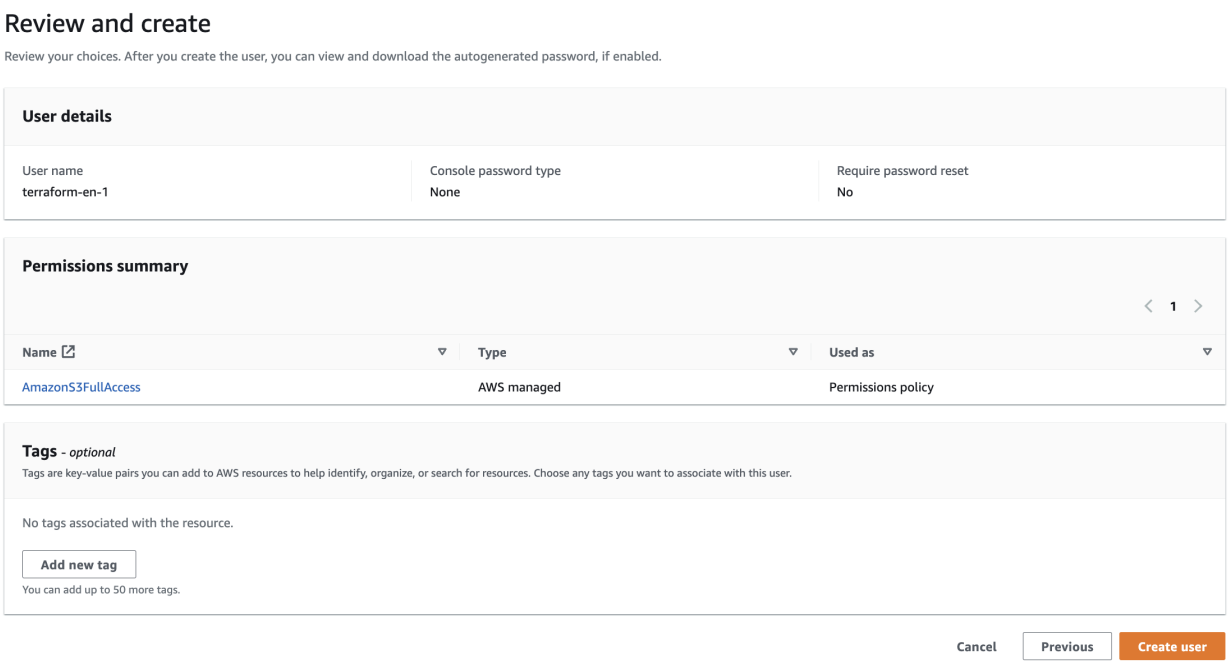

- On Set permissions, Permissions options, click in "Attach policies directly" button.




- Type **AmazonS3FullAccess** in **Search**.
- Select **AmazonS3FullAccess**

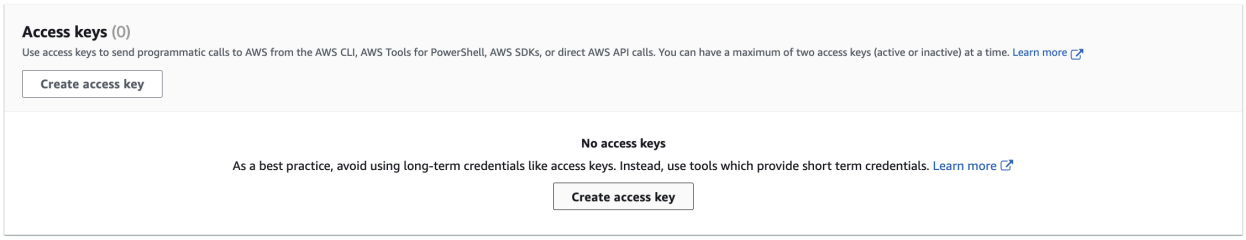



- Click in **Next**
- Review all details, then click **Create user.**




**[NEW] AWS has recently changed the way to download the key. Follow the new steps:**

- Click on the user you have created.
- After this, click on **Security credentials** tab.
- Scroll down and go to Access keys section.
- Click on **Create access key**




- Select **Command Line Interface (CLI)** and **I understand the above recommendation and want to proceed to create an access key** checkbox.
- Click **Next**.
- Click on **Create access key**

- Click on **Download .csv file**
- After download is finished, click on **Done**

### ▼ [TIP] Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

- After download, click **Done**.
- Now, rename .csv file downloaded to **accessKeys.csv**

# Google Cloud Platform (GCP)

- **CLICK HERE to download the mission1.zip hands-on files**.
- Access GCP Console and open Cloud Shell
- Upload **accessKeys.csv** and **mission1.zip** hands-on file to GCP Cloud Shell
- Check if upload has been successfully completed using the command **ls -la**
- Hands-on files preparation

```
mkdir mission1_en
mv mission1.zip mission1_en
cd mission1_en
unzip mission1.zip
mv ~/accessKeys.csv mission1/en
cd mission1/en
chmod +x *.sh
```

- Run the following commands to prepare AWS and GCP environment. Authorize when asked.

```
./aws_set_credentials.sh accessKeys.csv
```

```
gcloud config set project <project_id>
```

- Execute the command below

```
./gcp_set_project.sh
```

- Enable the Container Registry API, Kubernetes Engine API and the Cloud SQL API

```
gcloud services enable containerregistry.googleapis.com
gcloud services enable container.googleapis.com
gcloud services enable sqladmin.googleapis.com
```

**IMPORTANT (DO NOT SKIP):**

- **Before executing the Terraform commands, open the Google Editor and update the file tcb_aws_storage.tf replacing the bucket name with an unique name (AWS requires unique bucket names).**
  - Open the **tcb_aws_storage.tf** using Google Editor
  - On **line 4** of the file **tcb_aws_storage.tf**:
    - Replace **xxxx** with your name initials, using **5 letters** plus **5 random numbers**:
      Example: **luxxy-covid-testing-system-pdf-en-jerod29292**

- Run the following commands to finish provision infrastructure steps

```
cd ~/mission1_en/mission1/en/terraform/

terraform init
terraform plan
terraform apply
  Type Yes and go ahead.
```

**Attention**: The Cloud SQL database may take **15 to 25 minutes** to create, always check the **CloudShell** and click **Reconnect** when the session expires (the session expires after **5 minutes** of inactivity by default)

- The **warning** message at the end of **terraform apply** command execution is not a problem, please go ahead:

```
Warning: Argument is deprecated

  with aws_s3_bucket.b,
  on tcb_aws_storage.tf line 5, in resource "aws_s3_bucket" "b":
   5:   acl = "private"

Use the aws_s3_bucket_acl resource instead

(and one more similar warning elsewhere)
```

After finished, access the link to Compare GKE Autopilot and Standard.

## SQL Network Configuration

https://prod-files-secure.s3.us-west-2.amazonaws.com/9a82f1a4-c086-4469-a0ee-c028781a5e80/6ab20977-4163-4519-8d2d-87fbfe5d5f8e/SQL_Networking_Configuration_EDITED.mov

- Once the Cloud SQL instance is provisioned, access the Cloud SQL service
- Click on your Cloud SQL instance.
- On the left side, under Primary Instance, click on **Connections**.
- Go to **Networking** tab.
- Under **Instance IP assignment**, select Private IP to enable.
    - Under **Associated networking**, select "Default"

    - Click **Set up Connection**
    - Click on **Enable API**, to enable Service Networking API (if asked).
    - Select **Use an automatically allocated IP range in your network**.
    - Click **Continue**
    - Click **Create Connection** and **wait a minutes until conclude.** You will see the message: "*Private services access connection for network default has been successfully created.*"
- Under **Authorized Networks**, click "Add Network".
- Under **New Network**, enter the following information:
    - **Name:** Public Access (For testing purposes only)
    - **Network**: 0.0.0.0/0
    - Click **Done**.
    - Click **Save** and wait to finish the update.
      This update may take from **10 to 20 minutes** to finish

PS: For production environments, it is recommended to use only the Private Network for database access.
⚠️ Never grant public network access (0.0.0.0/0) to production databases.

> 💡 Download Visual Studio Code used by Jean during the Training HERE