**MultiCloud, DevOps & AI Challenge - Day 1 - Experienced**

<aside> 💡

Remember to use the class replay as a reference.

</aside>

**Streamlined Guide: Using Claude as AI Assistant to Terraform**

**Step 1: Use Claude to Generate Terraform Code**

1. Start a conversation with Claude.
2. Ask Claude to create Terraform code for an S3 bucket. Use a prompt like: "Please provide Terraform code to create an S3 bucket in AWS with a unique name."
3. Claude should generate code similar to this:
4. provider "aws" {
5. region = "us-west-2" # Replace with your desired region
6. }
7. resource "random\_id" "bucket\_suffix" {
8. byte\_length = 8
9. }
10. resource "aws\_s3\_bucket" "my\_bucket" {
11. bucket = "my-unique-bucket-name-${random\_id.bucket\_suffix.hex}"
12. tags = {
13. Name = "My bucket"
14. Environment = "Dev"
15. }
16. }
17. resource "aws\_s3\_bucket\_acl" "my\_bucket\_acl" {
18. bucket = aws\_s3\_bucket.my\_bucket.id
19. acl = "private"
20. }
21. Save this code for use in Step 5.

**Step 2: Create IAM Role for EC2**

1. Log in to the AWS Management Console.
2. Navigate to the IAM dashboard.
3. Click "Roles" in the left sidebar, then "Create role".
4. Choose "AWS service" as the trusted entity type and "EC2" as the use case.
5. Search for and attach the "AdministratorAccess" policy. Note: In a production environment, use a more restricted policy.
6. Name the role "EC2Admin" and provide a description.
7. Review and create the role.

**Step 3: Launch EC2 Instance**

1. Go to the EC2 dashboard in the AWS Management Console.
2. Click "Launch Instance".
3. Choose an Amazon Linux 2 AMI.
4. Select a t2.micro instance type.
5. Configure instance details:
   * Network: Default VPC
   * Subnet: Any available
   * Auto-assign Public IP: Enable
   * IAM role: Select "EC2Admin"
6. Keep default storage settings.
7. Add a tag: Key="Name", Value="workstation".
8. Create a security group allowing SSH access from EC2 Connect IP.
9. Review and launch, selecting or creating a key pair.

**Step 4: Connect to EC2 Instance and Install Terraform**

1. From the EC2 dashboard, select your "workstation" instance.
2. Click "Connect" and use the "EC2 Instance Connect" method.
3. In the browser-based SSH session, update system packages:
4. sudo yum update -y
5. Install yum-utils:
6. sudo yum install -y yum-utils
7. Add HashiCorp repository:
8. sudo yum-config-manager --add-repo <https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo>
9. Install Terraform:
10. sudo yum -y install terraform
11. Verify installation:
12. terraform version

**Step 5: Apply Terraform Configuration**

1. Create a new directory and navigate to it:
2. mkdir terraform-project && cd terraform-project
3. Create and open [main.tf](http://main.tf/):
4. nano main.tf
5. Paste the Terraform code generated by Claude in Step 1.
6. Save and exit the editor (in nano, press Ctrl+X, then Y, then Enter).
7. Initialize Terraform:
8. terraform init
9. Review the plan:
10. terraform plan
11. Apply the configuration:
12. terraform apply
13. Type "yes" when prompted to create the resources.

**Step 6: Verify S3 Bucket Creation**

1. Use AWS CLI to list buckets:
2. aws s3 ls
3. Verify that your new bucket is in the list.

**Step 7: Create the Cloud DynamoDB tables**

Remove the S3 lines and add the lines below to create the DynamoDB tables used by CloudMart

provider "aws" {

region = "us-east-1"

}

# Tables DynamoDB

resource "aws\_dynamodb\_table" "cloudmart\_products" {

name = "cloudmart-products"

billing\_mode = "PAY\_PER\_REQUEST"

hash\_key = "id"

attribute {

name = "id"

type = "S"

}

}

resource "aws\_dynamodb\_table" "cloudmart\_orders" {

name = "cloudmart-orders"

billing\_mode = "PAY\_PER\_REQUEST"

hash\_key = "id"

attribute {

name = "id"

type = "S"

}

}

resource "aws\_dynamodb\_table" "cloudmart\_tickets" {

name = "cloudmart-tickets"

billing\_mode = "PAY\_PER\_REQUEST"

hash\_key = "id"

attribute {

name = "id"

type = "S"

}

}

1. Apply the configuration:
2. terraform apply
3. Type "yes" when prompted to create the resources.

Congratulations! You've successfully used Claude to generate Terraform code, set up an EC2 workstation, installed Terraform, and created an S3 bucket. This completes Day 1 of the MultiCloud DevOps & AI Challenge.