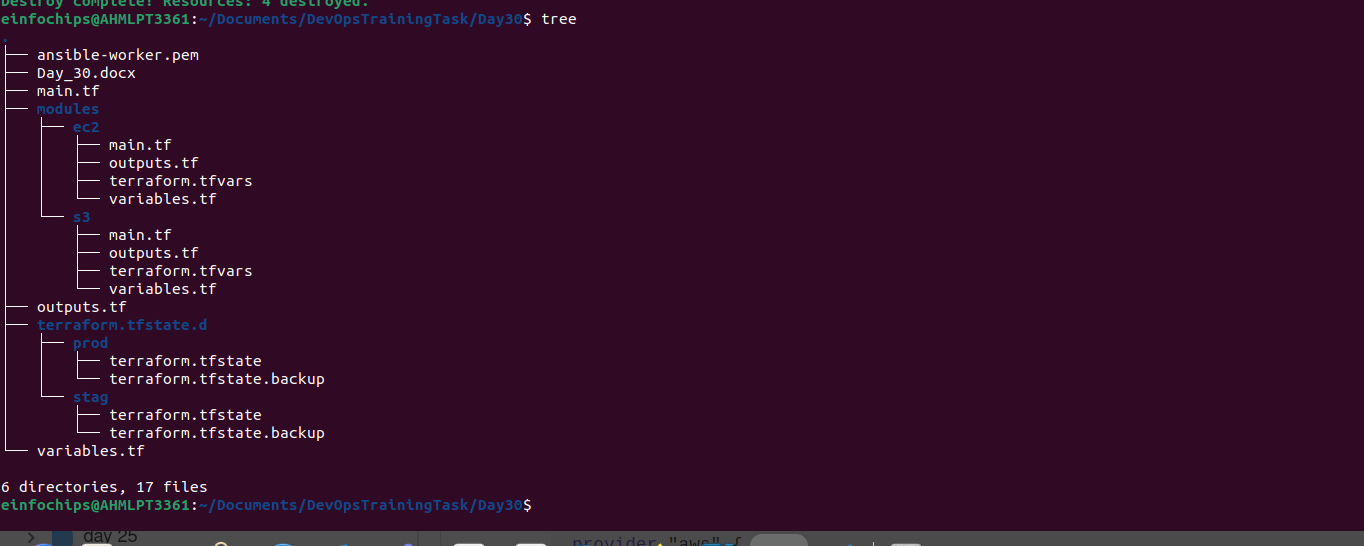
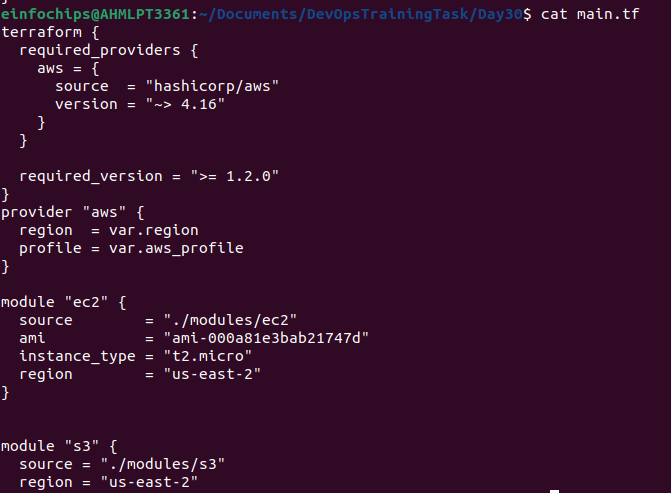
1. **Module Development**:
   * **Module Setup**: Create a directory for the module (e.g., modules/aws\_infrastructure).
   * **Resource Definitions**: Define the resources for an EC2 instance and an S3 bucket within the module.
   * **Variable Inputs**: Define input variables for instance type, AMI ID, key pair name, and S3 bucket name.
   * **Outputs**: Define outputs for the EC2 instance's public IP and the S3 bucket's ARN.

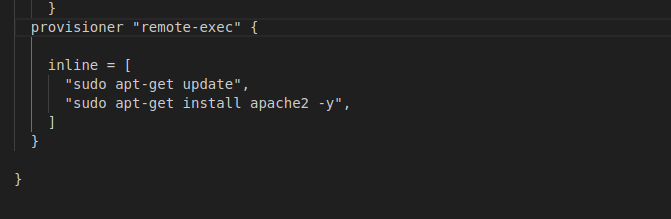




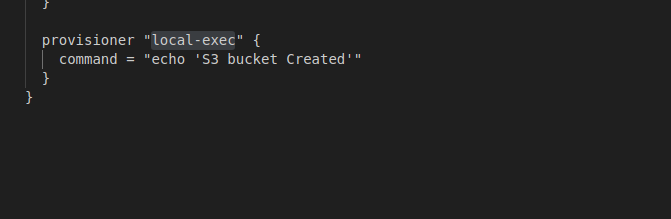
1. **Main Terraform Configuration**:
   * **Main Config Setup**: In the root directory, create a Terraform configuration that calls the custom module.
   * **Backend Configuration**: Configure Terraform to use local state storage for simplicity (optional for Free Tier compliance).



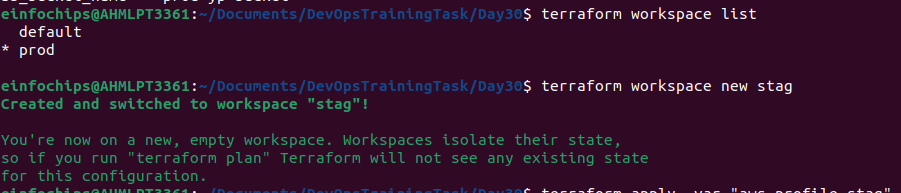
1. **Provisioner Implementation**:
   * **Remote Execution**: Use the remote-exec provisioner to SSH into the EC2 instance and execute a script that installs Apache.



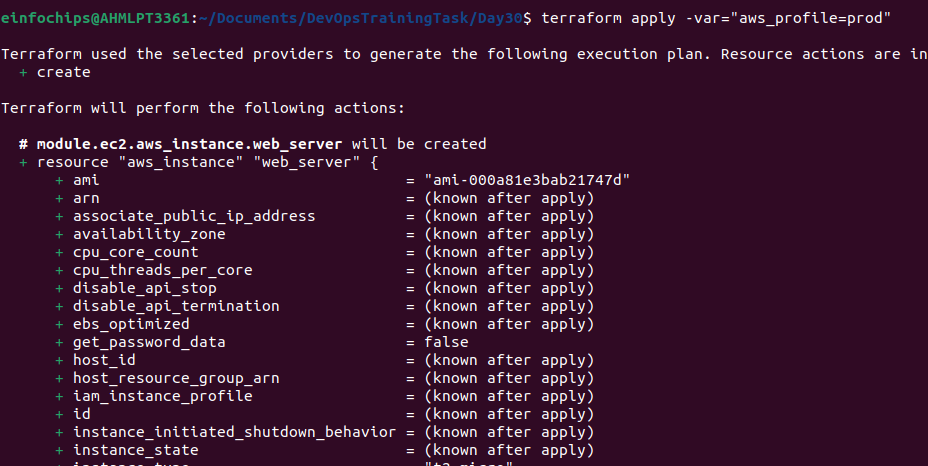
* + **Local Execution**: Use the local-exec provisioner to print a confirmation message on the local machine after successful deployment.

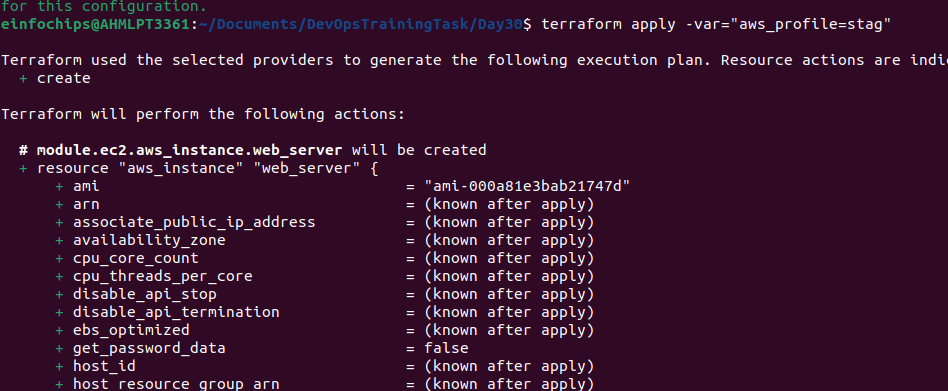


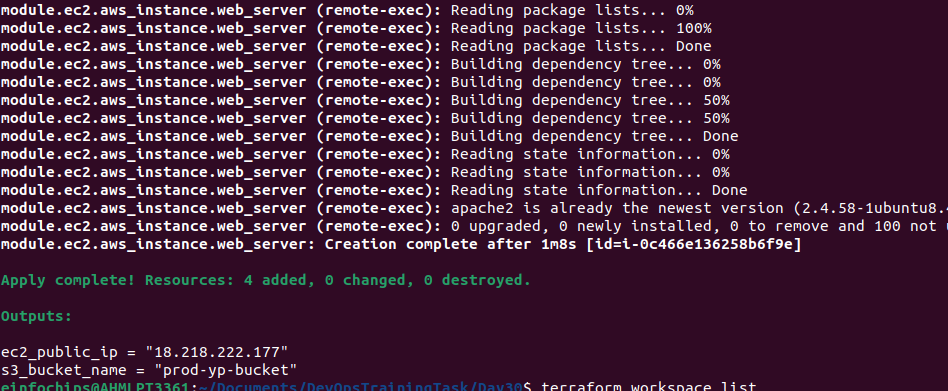
1. **Workspace Management**:
   * **Workspace Creation**: Create Terraform workspaces for dev and prod.
   * **Environment-Specific Configurations**: Customize the EC2 instance tags and S3 bucket names for each workspace to differentiate between environments.
   * **Workspace Deployment**: Deploy the infrastructure separately in the dev and prod workspaces.

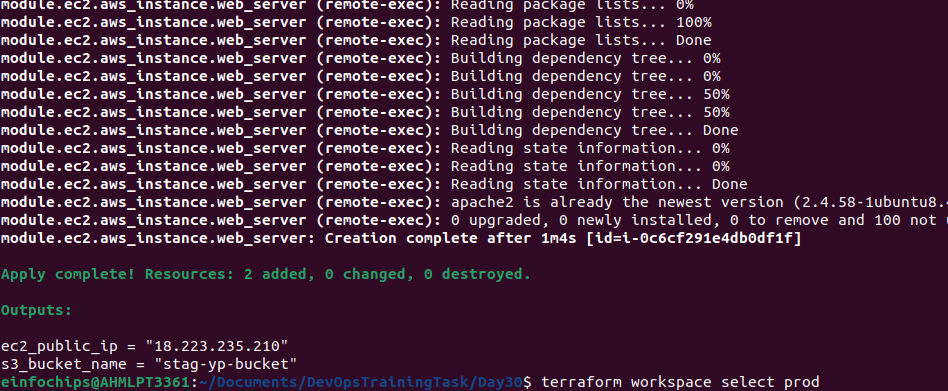


1. **Validation and Testing**:
   * **Apache Installation Verification**: After the deployment, verify that Apache is installed and running on the EC2 instance by accessing the public IP address in a web browser.
   * **Workspace Separation**: Confirm that each workspace has its own isolated infrastructure and state files.
   * **Provisioner Logs**: Review the output from the local-exec provisioner to ensure it indicates successful deployment.









1. **Resource Cleanup**:
   * **Destroy Resources**: Use terraform destroy to remove the resources in both workspaces.
   * **Workspace Management**: Confirm that the resources are destroyed separately in each workspace and that the state files are updated accordingly.

