





readme

▼ My Information

Hussain Alramadan

201853540

▼ COE Dept. Cloud Computing CX Project

To Professor: Yahya Osais

▼ Watch This Video for Easier Understanding (9 min)



Run it at 1.5x speed :)

<https://youtu.be/6PUj8q7s2Mw>

▼ Workload Samples



For Driver Lambda

```
{
  "ID": 1,
  "Name": "Hussain",
  "Kind": "D",
  "Src": "DMM",
  "Dest": "RYD"
}
```



For Rider Lambda

```
{
  "ID": 2,
  "Name": "Abdullah",
  "Kind": "R",
  "Src": "DMM",
  "Dest": "RYD"
}
```

▼ Terraform Setup



Put AWS credential in variables.tf of the root `main.tf`



Run `terraform init` and `terraform apply` in the **AWS Project** directory.

AWS Project contains the following:

- Terraform files
- 2 Ansible Scripts
 - `go_setup.yml`
 - `hosts.yml`
- 4 GoLang Scripts
 - `api.go`
 - `deleteTableItem.go`
 - `getTableItem.go`
 - `logic.go`

Note that Lambda function Python Scripts are inside the Lambda terraform file.



After `terraform apply` three outputs will generated

- Public EC2 ip address
- AWS API Link without paths to lambda functions
- Private and public EC2 `keypair`
 - this will be generated and ready to be used in **AWS Project** file for SSH connection to both public and private EC2 instances.

▼ Everything is Automated Except the Following:

Follow them sequentially



Execute the following commands in the public EC2

```
chmod 400 ec2_key_pair
ssh-agent bash
cp ec2_key_pair ~/.ssh/
ssh-add ~/.ssh/ec2_key_pair
export ANSIBLE_HOST_KEY_CHECKING=False
```



Execute `go_setup.yml` script in master host (i.e., public EC2) to be executed in the target hosts in `hosts.yml` script (i.e., private EC2).

```
ansible-playbook -i hosts.yml go_setup.yml
```



SSH to private EC2 (its IP is fixed to be 10.0.1.50)

```
ssh -i ec2_key_pair ec2-user@10.0.1.50
```



Configure AWS SDK Credentials in the private EC2 to give the ability to reach the dynamoDB

```
aws configure  
  
<access_key>  
<private_key>  
<region>
```



Execute Go scripts in the private EC2

```
go run logic.go &  
go run putTableItem.go &  
go run getTableItem.go &  
go run deleteTableItem.go &  
// & is used to execute the script in the background to be able to run multiple scripts
```

Before those scripts get started executing by the EC2, GoLang runtime will download packages for data dependencies.

Give the GoLang runtime some time to complete downloading.

You can check by seeing the .go script name when writing this command `ps`

to show the currently running processes in the EC2.



Exist from private EC2 and execute `api.go` in public EC2

```
exit  
go run api.go &
```



Go to lambda functions in AWS console and add the Public IP to Python Code there for **all functions** (which are only two) as shown in this picture.

```
r = requests.post("http://3.80.84.146:9999/api", jsonData, headers)  
r.connection.close()
```



Get API link and add routes

```
/lambda/DriverLambda
```

```
/lambda/RiderLambda
```