CURRICULUM WEB SERVICE RUNBOOK

SHORT DESCRIPTION

A website that is used for looking up information about course and class offerings at CSUN. The web service provides a gateway to access the information vis a REST-ful API.

REQUIRED SOFTWARE

AMAZON WEB SERVICES (AWS) ACCOUNT

- Access via provided user credentials
- Create a new account

PHYSICAL MACHINE

- Terraform
- AWS CLI
- GIT
- SSH capabilities (Windows: GIT Bash/PUTTY)

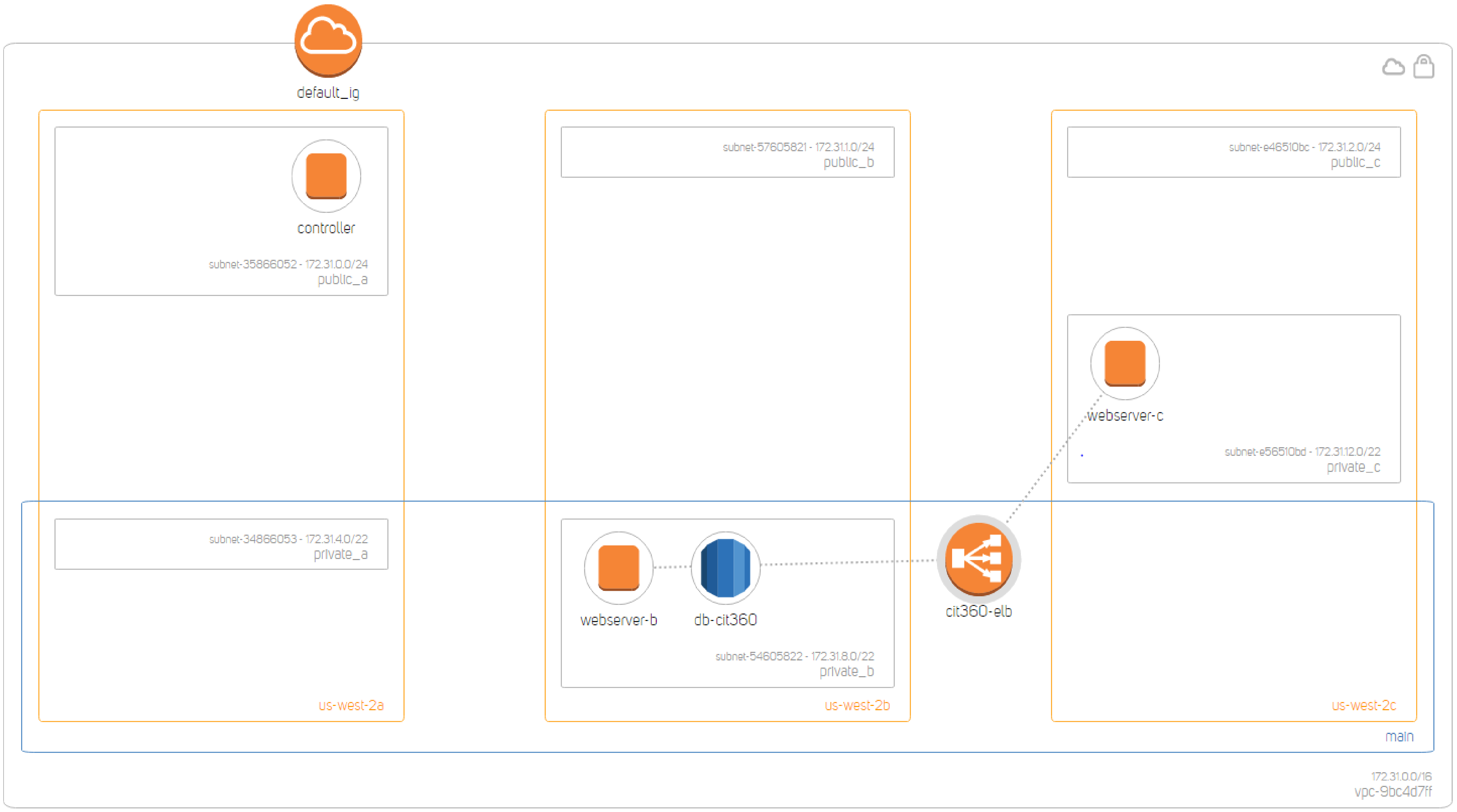
AWS CONTROL MACHINE

- Ansible
- GIT
- MariaDB-client

AWS WEBSERVERS

- nginx
- php
- php-fpm
- php-ldap
- php-mbstring
- php-mycrypt
- php-mysql
- php-phpunit-PHPUnit
- Composer

ARCHITECTURE DIAGRAM


default_ig
controller
subnet-35866052 - 172.31.0.0/24
public_a
subnet-57605821 - 172.31.1.0/24
public_b
subnet-e46510bc - 172.31.2.0/24
public_c
webserver-c
subnet-e56510bd - 172.31.12.0/22
private_c
subnet-34866053 - 172.31.4.0/22
private_a
webserver-b
db-cit360
cit360-elb
subnet-54605822 - 172.31.8.0/22
private_b
us-west-2a
us-west-2b
us-west-2c
main
172.31.0.0/16
vpc-9bc4d7ff

DEPLOYMENT

The first step is to set up the environment on your physical machine. This involves installing and configuring GIT, Terraform, and AWS CLI. The following contains information on setting up each one from scratch so feel free to skip if you already have the services installed and configured.

GIT

1. There are several ways to install Git depending on your operating system.
2. Go to https://git-scm.com/downloads
3. Select your OS and install the version that applies to you
4. Navigate to a directory you would like to use for the project and clone the project

```
$ git clone https://github.com/jordanblopez/cit-360.git
```

5. In your terminal set your Git config:

```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

TERRAFORM

1. Download Terraform from

```
http://www.terraform.io/downloads.html
```

2. Select the appropriate package for your system
3. Extract the zipped file where you want terraform to be installed.
4. Add terraform to the PATH of your OS

   For Linux/Mac:

```
$ PATH=/usr/local/terraform/bin:/home/your-user-name/terraform:$PATH
```

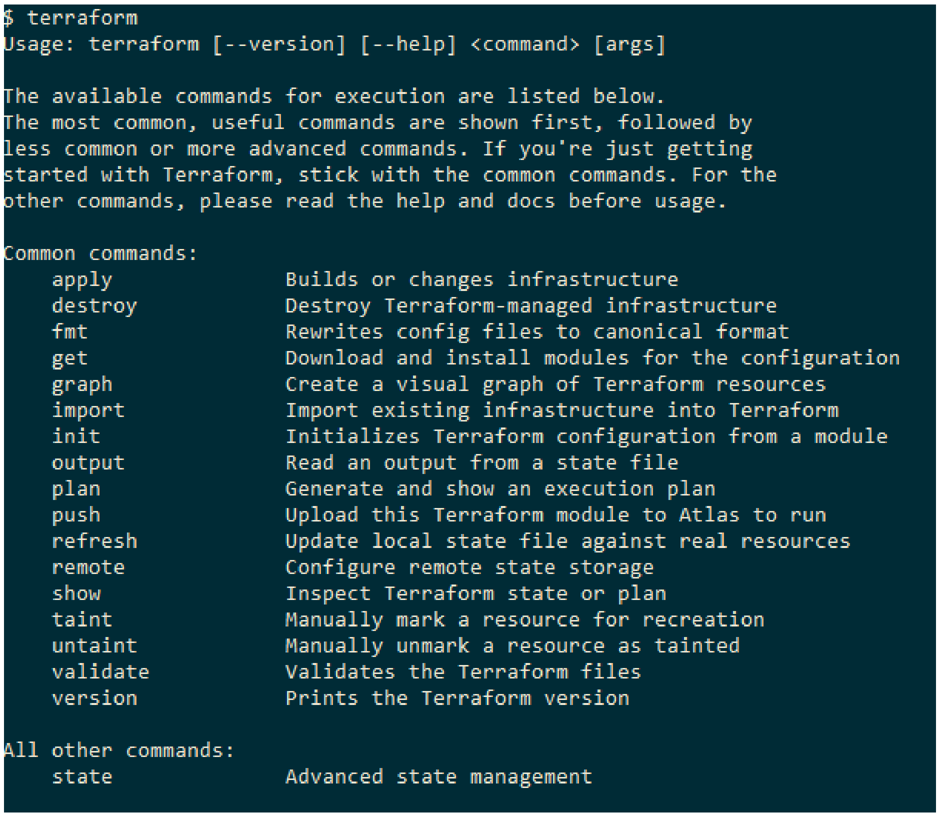
   For Windows - Powershell:

```
 [Environment]::SetEnvironmentVariable("PATH", $env:PATH +
({;C:\terraform},{C:\terraform})[$env:PATH[-1] -eq ';'], "User")
```

5. Verify the installation, open a new terminal and type:

```
$ terraform
```

- Out should look similar to this:



```
$ terraform
Usage: terraform [--version] [--help] <command> [args]

The available commands for execution are listed below.
The most common, useful commands are shown first, followed by
less common or more advanced commands. If you're just getting
started with Terraform, stick with the common commands. For the
other commands, please read the help and docs before usage.

Common commands:
    apply              Builds or changes infrastructure
    destroy            Destroy Terraform-managed infrastructure
    fmt                Rewrites config files to canonical format
    get                Download and install modules for the configuration
    graph              Create a visual graph of Terraform resources
    import             Import existing infrastructure into Terraform
    init               Initializes Terraform configuration from a module
    output             Read an output from a state file
    plan               Generate and show an execution plan
    push               Upload this Terraform module to Atlas to run
    refresh            Update local state file against real resources
    remote             Configure remote state storage
    show               Inspect Terraform state or plan
    taint              Manually mark a resource for recreation
    untaint            Manually unmark a resource as tainted
    validate           Validates the Terraform files
    version            Prints the Terraform version

All other commands:
    state              Advanced state management
```

AMAZON AWS ACCOUNT

For the purposes of this service is expected that you already have an Amazon AWS account that is already set up that has the appropriate IAM group. You will also need to generate a key pair; you can do this through the web EC2 console under Network & Security. Create a key pair with the name cit360. Make you download it and remember which directory it is located; you will need this to SSH into the instances created later.

AWS CLI

The first step in setting up AWS CLI is to get credentials from the AWS website:

```
https://jordanlopez.signin.aws.amazon.com/console
```

1. You will need to access the IAM console.
2. In the navigation pane, choose Users
3. Choose your IAM user name (not the check box)
4. Choose the Security Credentials tab and then choose Create Access Key.
5. Select Show User Security Credentials. Download the credentials

```
$ aws configure
  AWS Access Key ID [None]: Your-access-key
  AWS Secret Access Key [None]: you-secret-key
```

DEPLOY TERRAFORM

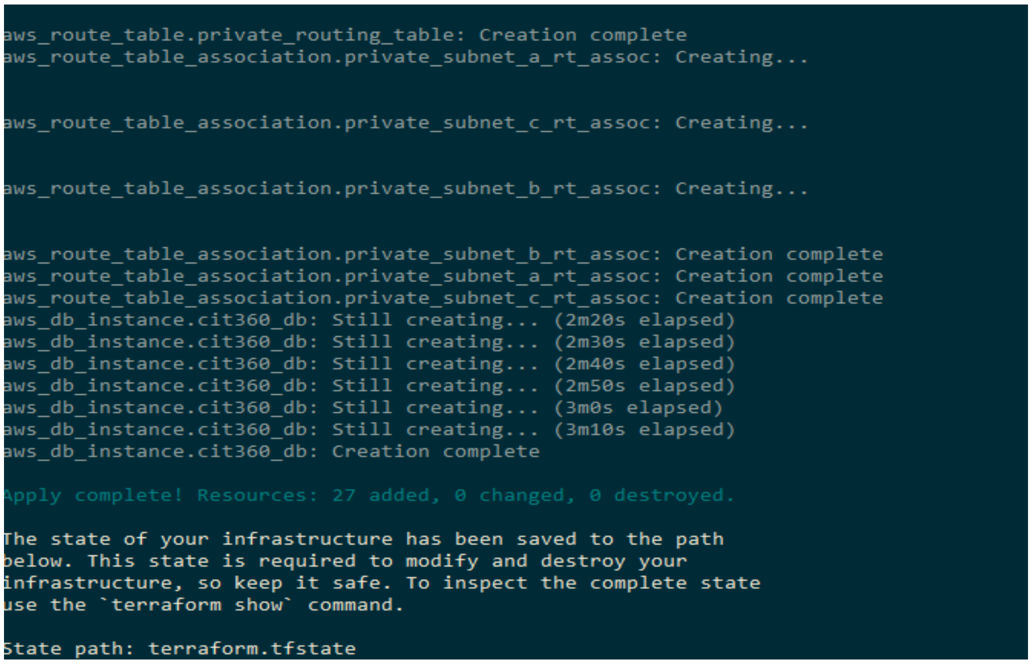
1. In the terminal navigate to the terraform directory where you cloned the Git repo.

```
$ cd path-to-file/cit-360/terraform
```

2. Run the following command to create the infrastructure for the web service

```
$ terraform apply \
   -var ‘password=your-db-password’
```

- You should see the following output



```
aws_route_table.private_routing_table: Creation complete
aws_route_table_association.private_subnet_a_rt_assoc: Creating...


aws_route_table_association.private_subnet_c_rt_assoc: Creating...


aws_route_table_association.private_subnet_b_rt_assoc: Creating...


aws_route_table_association.private_subnet_b_rt_assoc: Creation complete
aws_route_table_association.private_subnet_a_rt_assoc: Creation complete
aws_route_table_association.private_subnet_c_rt_assoc: Creation complete
aws_db_instance.cit360_db: Still creating... (2m20s elapsed)
aws_db_instance.cit360_db: Still creating... (2m30s elapsed)
aws_db_instance.cit360_db: Still creating... (2m40s elapsed)
aws_db_instance.cit360_db: Still creating... (2m50s elapsed)
aws_db_instance.cit360_db: Still creating... (3m0s elapsed)
aws_db_instance.cit360_db: Still creating... (3m10s elapsed)
aws_db_instance.cit360_db: Creation complete

Apply complete! Resources: 27 added, 0 changed, 0 destroyed.

The state of your infrastructure has been saved to the path
below. This state is required to modify and destroy your
infrastructure, so keep it safe. To inspect the complete state
use the `terraform show` command.

State path: terraform.tfstate
```

CONNECT TO THE BASTION INSTANCE AND DEPLOY ANSIBLE TO CONFIGURE THE WEB SERVICE

The software will be deployed using ansible on the control machine to install a webserver onto the EC2 instances. Ensure it is installed on the control system:

1. Using the key generated earlier, SSH into the bastion control instance

```
$ ssh -i "/path-to-pem/cit360.pem" ec2-user@ec2-ip-address-of-bastion.us-west-
2.compute.amazon.com
```

2. Install Git

```
$ sudo yum install git
```

3. Install Ansible

```
$ sudo yum install epel-release
$ sudo yum install ansible
```

4. Clone the Git redo

```
$ git clone https://github.com/jordanblopez/cit-360.git
```

5. You can sign into AWS. Go to the EC2 console to get what the IP address for each of the webservers and alter the host file accordingly.

```
$ sudo vim /etc/ansible/hosts
```

6. You may also need to change the db.yml and the web.yml to make sure that it has the correct RDS address. In the AWS online console RDS console select db-cit360 and look where it says Endpoint, this is the MariaDB address.
    a. For web.yml check

    ```
    db_host: rds-endpoint
    ```

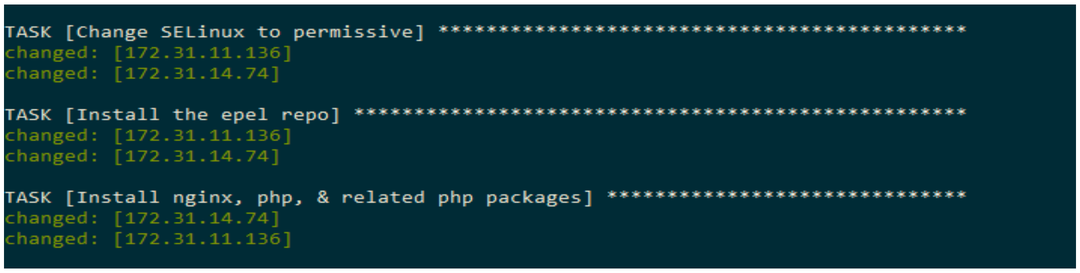
    b. For db.yml check to make sure the RDS endpoint matches

    ```
    - name: Run the script to create a db for the website
      command: ./make_databases.sh "{{ db_password }}" rds-endpoint chdir=~/db
    ```

7. Within the directory with all the configuration files in this case ~/cit-360

```
$ ansible-playbook web.yml db.yml --ask-vault-pass
```

8. This will prompt for the password to decrypt the secrets.yml file. After you enter the password ansible should indicate that it is running through the tasks needed to setup the nginx web server, and the databases. There will be a description of what task is running that looks like:

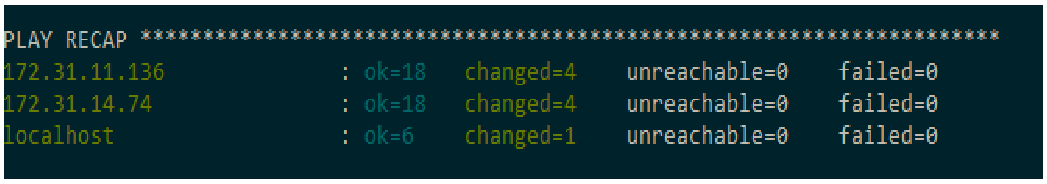


```
TASK [Change SELinux to permissive] **********************************************
changed: [172.31.11.136]
changed: [172.31.14.74]

TASK [Install the epel repo] *****************************************************
changed: [172.31.11.136]
changed: [172.31.14.74]

TASK [Install nginx, php, & related php packages] ********************************
changed: [172.31.14.74]
changed: [172.31.11.136]
```

9. When all the tasks have been run there will be a summary that looks like this:


```
PLAY RECAP *********************************************************************
172.31.11.136              : ok=18   changed=4    unreachable=0    failed=0
172.31.14.74               : ok=18   changed=4    unreachable=0    failed=0
localhost                  : ok=6    changed=1    unreachable=0    failed=0
```

USE THE WEBSERVICE

1. Find the public DNS name of the ELB. You can do this from the AWS console from the EC2 dashboard or from the command line on your local machine

```
aws elb describe-load-balancers
```

2. Copy/paste the address into a web browser. You should see a working website.

ISSUES

FAILED TO DECRYPT

- **Description:** If you run the ansible command without the correct password or without using the --ask-vault-pass you will get an error.
- **Remediation Steps:** Make sure you are using the correct password to decrypt the secrets.yml file. Contact the creator of the file for the correct password.

CONNECTION REFUSED

- **Description:** When trying to access the website it displays a 500 error code saying connection refused.
- **Remediation Steps:** It could be that the nginx service isn't running properly, restart the service using:

```
 $ sudo service nginx restart
```

UNREACHABLE

- **Description:** When trying to run the playbook it gives the error message "Failed to connect to the host via ssh".
- **Remediation Steps:** It could be that ssh isn't properly configured between the control machine and the host. On the host machine open the sshd_config file:

```
$  sudo vim /etc/ssh/sshd_config  #use vi or nano if vim isn't installed
```

- In the file:

  PublicKeyAuthentication yes
  PasswordAuthentication yes

- Restart the sshd service:

```
$ sudo service sshd restart
```

- On the control machine:

```
  $ ssh-copy-id *host IP address*
```

- Yes and put in the password to the host machine.

HOST NOT FOUND

- **Description:** If when you run the playbook you run into the issue of the host not existing
- **Remediation Steps:** Make sure that you have the correct IP addresses for webserver-b and webserver-c. You can find the IP addresses for each in the EC2 dashboard in the AWS console online or you can use the AWS CLI:

```
$ aws ec2 describe-instances
```