Assessment || DevOps Engineer || Shivam Yadav

Task 1 : Setup a Linux server in Custom VPC server with use of terraform and ansible and install and configure MySQL, tomcat on 8080 port Memcached Redis and deploy a sample war file.

- First we will create a custom VPC and instance using terraform i.e. **main.tf** file.
- Then we will make sure that we will use correct ami (as per region).
- Then we will intialize terraform using cmd : **terraform init**.
- Then we will plan terraform using cmd : **terraform plan**.
- Finally we will apply the changes using cmd : **terraform apply**.
- Now we will make sure inventory is created by terraform for ansible.
- After that we will create **playbook.yml** , we can also use role based approach.
- **Note** : Make sure we are using correct packages for MySql , Redis which are compatible with ami.
- Finally , To install Mysql,redis memcached,tomcat and war file we will use **ansible-playbook -i inventory playbook.yml**

To Initialze terraform using cmd :

```
$ terraform init
```

To plan terraform using cmd :

```
$ terraform plan
```

To apply terraform using cmd :

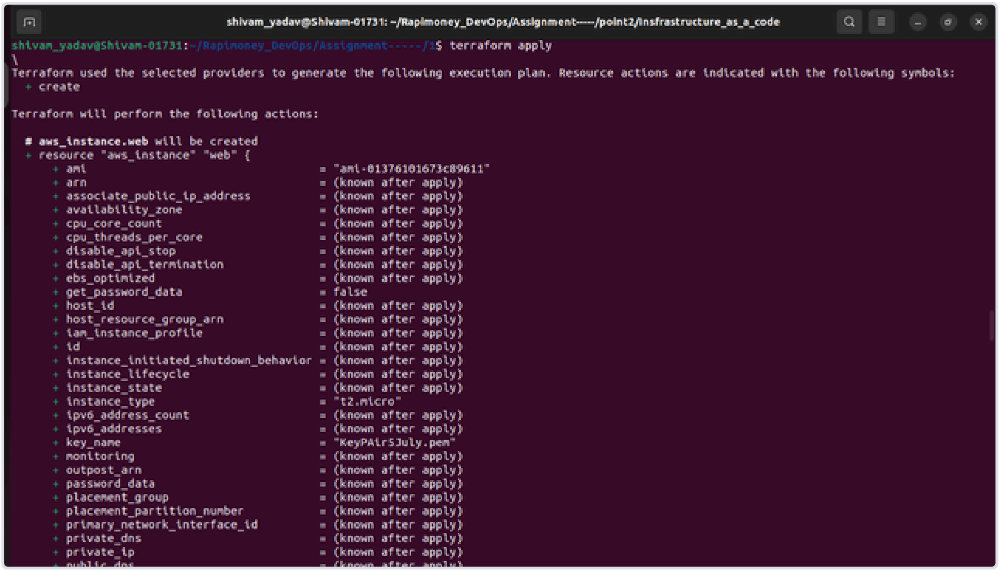
```
$ terraform apply
```

Then finally we will run the cmd on terminal :

```
$ ansible-playbook -i inventory playbook.yml
```

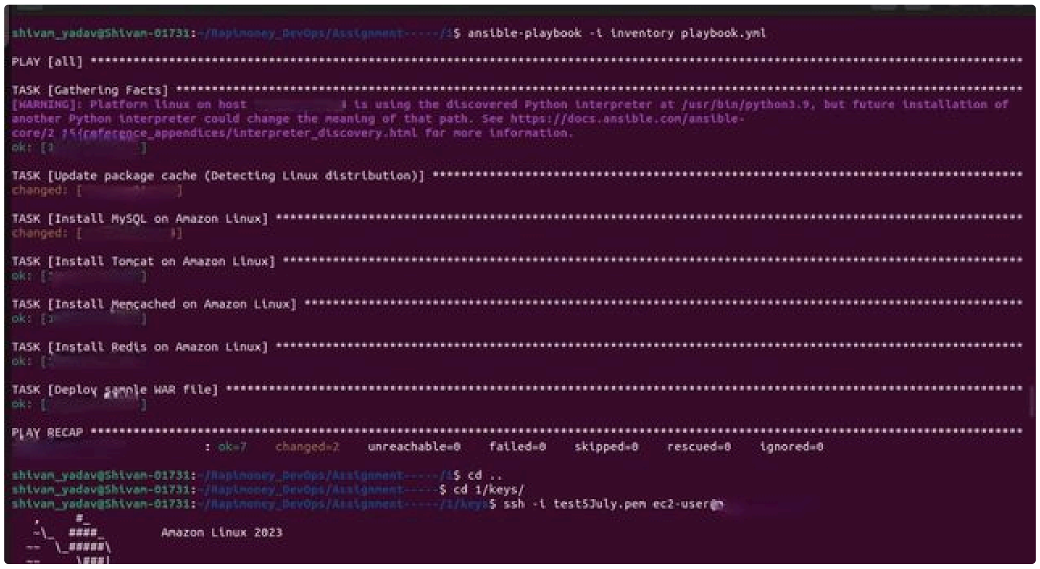
Task 1 Output :

Terraform :



```
shivam_yadav@Shivam-01731: ~/Rapimoney_DevOps/Assignment-----/point2/insfrastructure_as_a_code
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/1$ terraform apply
\
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.web will be created
  + resource "aws_instance" "web" {
      + ami                                  = "ami-01376101673c89611"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "KeyPAir5July.pem"
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
```

Ansible Playbook Output :



```
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/1$ ansible-playbook -i inventory playbook.yml

PLAY [all] **************************************************************************************************

TASK [Gathering Facts] **************************************************************************************
[WARNING]: Platform linux on host [redacted] is using the discovered Python interpreter at /usr/bin/python3.9, but future installation of
another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [[redacted]]

TASK [Update package cache (Detecting Linux distribution)] **************************************************
changed: [[redacted]]

TASK [Install MySQL on Amazon Linux] ************************************************************************
changed: [[redacted]]

TASK [Install Tomcat on Amazon Linux] ***********************************************************************
ok: [[redacted]]

TASK [Install Memcached on Amazon Linux] ********************************************************************
ok: [[redacted]]

TASK [Install Redis on Amazon Linux] ************************************************************************
ok: [[redacted]]

TASK [Deploy sample WAR file] *******************************************************************************
ok: [[redacted]]

PLAY RECAP **************************************************************************************************
[redacted]                 : ok=7    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/1$ cd ..
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----$ cd 1/keys/
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/1/keys$ ssh -i test5July.pem ec2-user@[redacted]
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
```

Task 2: **Create ALB Using Terraform and add machine and configure health check on 80 port and share URL.**

- For this task we are using modular approach of Terraform.
- Directory structure :

  -
    ```
    ├── main.tf
    ├── modules
    │   ├── alb
    │   │   ├── main.tf
    │   │   ├── outputs.tf
    │   │   └── variables.tf
    │   ├── ec2
    │   │   ├── main.tf
    │   │   ├── outputs.tf
    │   │   └── variables.tf
    │   ├── security_group
    │   │   ├── main.tf
    │   │   ├── outputs.tf
    │   │   └── variables.tf
    │   └── vpc
    │       ├── main.tf
    │       ├── outputs.tf
    │       └── variables.tf
    ├── outputs.tf
    └── variables.tf
    ```

To Initialze terraform using cmd :

```
$ terraform init
```

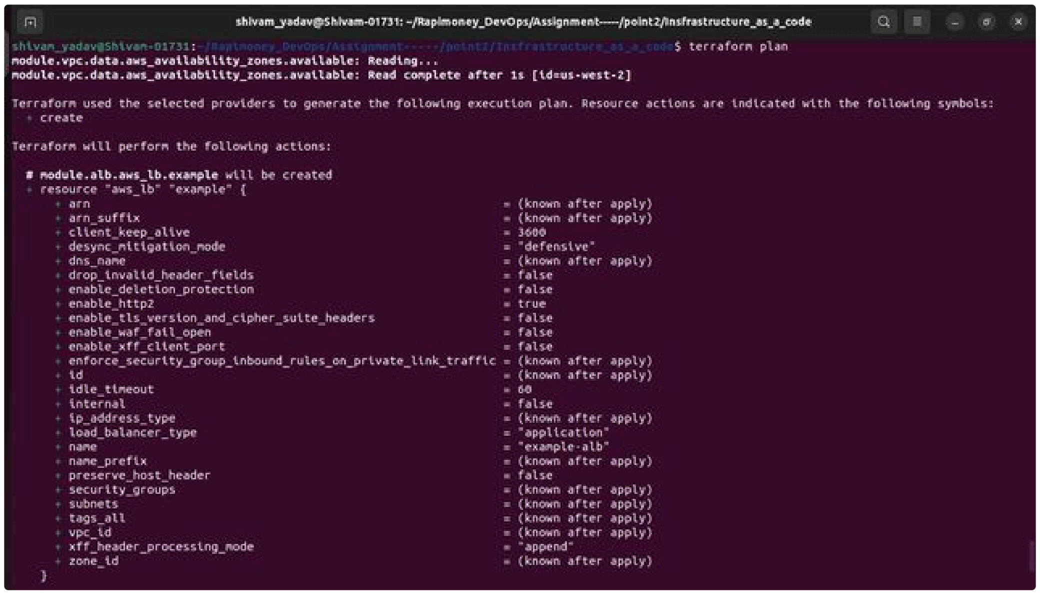
To plan terraform using cmd :

```
$ terraform plan
```

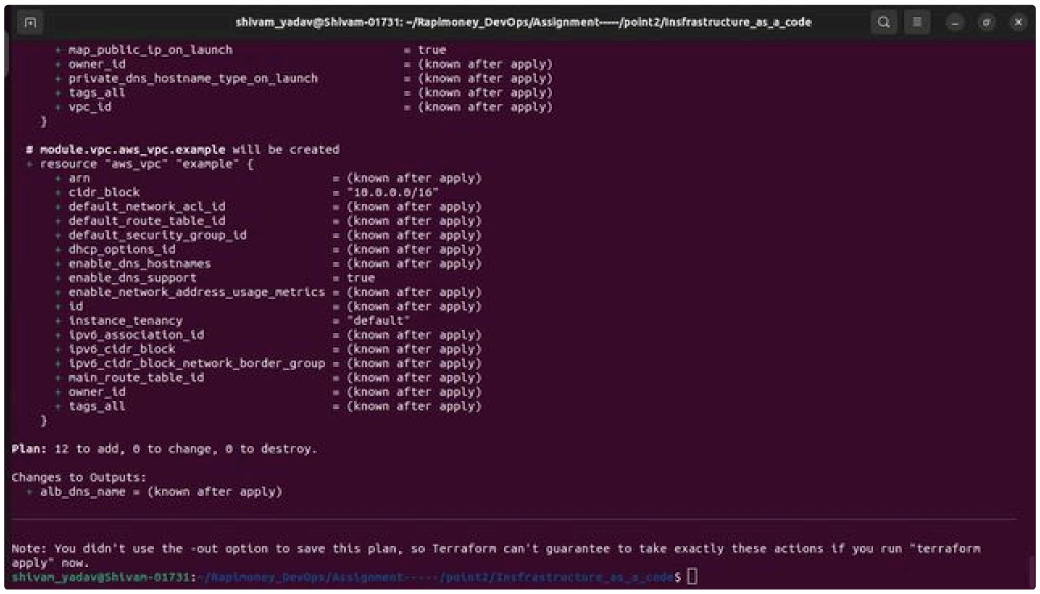
To apply terraform using cmd :

```
$ terraform apply
```

Task 2 : Output



```
shivam_yadav@Shivam-01731: ~/Rapimoney_DevOps/Assignment-----/point2/Insfrastructure_as_a_code
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/point2/Insfrastructure_as_a_code$ terraform plan
module.vpc.data.aws_availability_zones.available: Reading...
module.vpc.data.aws_availability_zones.available: Read complete after 1s [id=us-west-2]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # module.alb.aws_lb.example will be created
  + resource "aws_lb" "example" {
      + arn                                                          = (known after apply)
      + arn_suffix                                                   = (known after apply)
      + client_keep_alive                                            = 3600
      + desync_mitigation_mode                                       = "defensive"
      + dns_name                                                     = (known after apply)
      + drop_invalid_header_fields                                   = false
      + enable_deletion_protection                                   = false
      + enable_http2                                                 = true
      + enable_tls_version_and_cipher_suite_headers                  = false
      + enable_waf_fail_open                                         = false
      + enable_xff_client_port                                       = false
      + enforce_security_group_inbound_rules_on_private_link_traffic = (known after apply)
      + id                                                           = (known after apply)
      + idle_timeout                                                 = 60
      + internal                                                     = false
      + ip_address_type                                              = (known after apply)
      + load_balancer_type                                           = "application"
      + name                                                         = "example-alb"
      + name_prefix                                                  = (known after apply)
      + preserve_host_header                                         = false
      + security_groups                                              = (known after apply)
      + subnets                                                      = (known after apply)
      + tags_all                                                     = (known after apply)
      + vpc_id                                                       = (known after apply)
      + xff_header_processing_mode                                   = "append"
      + zone_id                                                      = (known after apply)
    }
```



```
shivam_yadav@Shivam-01731: ~/Rapimoney_DevOps/Assignment-----/point2/Insfrastructure_as_a_code
      + map_public_ip_on_launch                        = true
      + owner_id                                       = (known after apply)
      + private_dns_hostname_type_on_launch            = (known after apply)
      + tags_all                                       = (known after apply)
      + vpc_id                                         = (known after apply)
    }

  # module.vpc.aws_vpc.example will be created
  + resource "aws_vpc" "example" {
      + arn                                  = (known after apply)
      + cidr_block                           = "10.0.0.0/16"
      + default_network_acl_id               = (known after apply)
      + default_route_table_id               = (known after apply)
      + default_security_group_id            = (known after apply)
      + dhcp_options_id                      = (known after apply)
      + enable_dns_hostnames                 = (known after apply)
      + enable_dns_support                   = true
      + enable_network_address_usage_metrics = (known after apply)
      + id                                   = (known after apply)
      + instance_tenancy                     = "default"
      + ipv6_association_id                  = (known after apply)
      + ipv6_cidr_block                      = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                  = (known after apply)
      + owner_id                             = (known after apply)
      + tags_all                             = (known after apply)
    }

Plan: 12 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + alb_dns_name = (known after apply)

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform
apply" now.
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/point2/Insfrastructure_as_a_code$ 
```

Task 3: **Create an EC2 instance with a Linux based OS that is accessible over the internet via SSH.**

- We will create an ec2 instance and in security inbound rule we will allow SSH.
- For that we are creating the terraform main.tf file .
- While applying terraform , we have to pass input variables like subnetID , region , vpcID ,key name.
- Directory structure :
  - ├── main.tf
    ├── modules

```
│   └── ec2
│       ├── main.tf
│       ├── outputs.tf
│       └── variables.tf
├── outputs.tf
├── provider.tf
└── variables.tf
```

- The will give output of publicIP address and instanceID.

To Initialze terraform using cmd :

```
$ terraform init
```

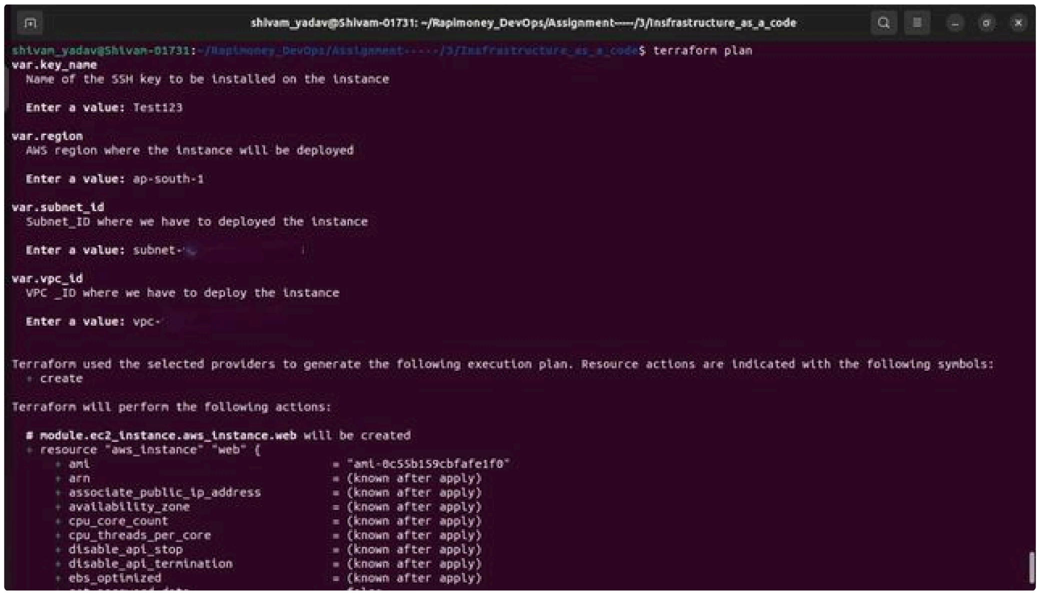
To plan terraform using cmd :

```
$ terraform plan
```

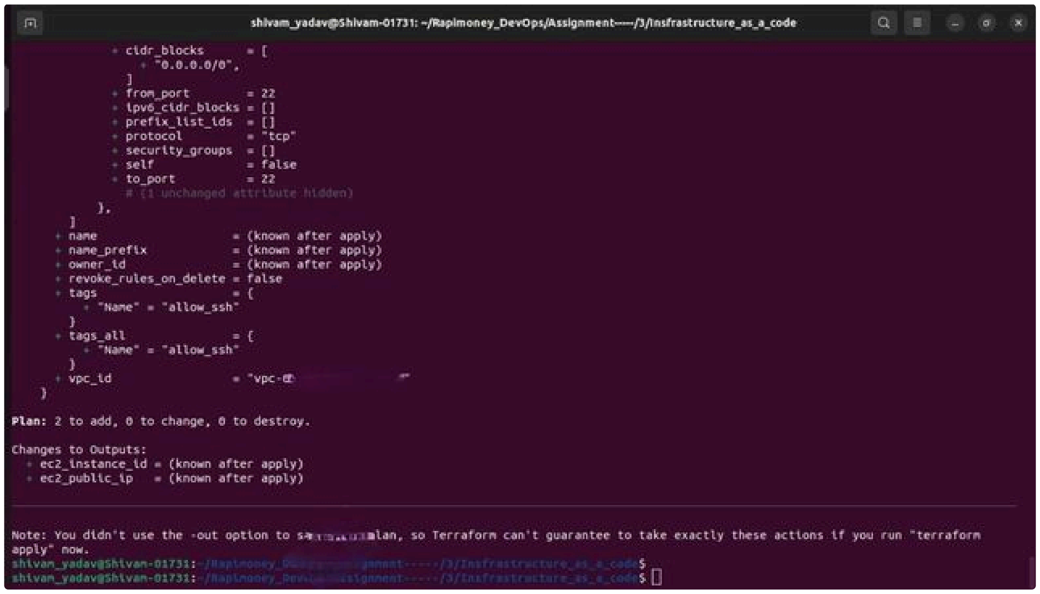
To apply terraform using cmd :

```
$ terraform apply
```

Task 3 : Output



```
shivam_yadav@Shivam-01731: ~/Rapimoney_DevOps/Assignment-----/3/Insfrastructure_as_a_code
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/3/Insfrastructure_as_a_code$ terraform plan
var.key_name
  Name of the SSH key to be installed on the instance

  Enter a value: Test123

var.region
  AWS region where the instance will be deployed

  Enter a value: ap-south-1

var.subnet_id
  Subnet_ID where we have to deployed the instance

  Enter a value: subnet-[redacted]

var.vpc_id
  VPC _ID where we have to deploy the instance

  Enter a value: vpc-[redacted]


Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # module.ec2_instance.aws_instance.web will be created
  + resource "aws_instance" "web" {
      + ami                                  = "ami-0c55b159cbfafe1f0"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
```



```
shivam_yadav@Shivam-01731: ~/Rapimoney_DevOps/Assignment-----/3/Insfrastructure_as_a_code
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + from_port        = 22
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 22
                # (1 unchanged attribute hidden)
            },
        ]
      + name                   = (known after apply)
      + name_prefix            = (known after apply)
      + owner_id               = (known after apply)
      + revoke_rules_on_delete = false
      + tags                   = {
          + "Name" = "allow_ssh"
        }
      + tags_all               = {
          + "Name" = "allow_ssh"
        }
      + vpc_id                 = "vpc-0[redacted]"
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + ec2_instance_id = (known after apply)
  + ec2_public_ip   = (known after apply)

─────────────────────────────────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to sa[redacted]lan, so Terraform can't guarantee to take exactly these actions if you run "terraform
apply" now.
shivam_yadav@Shivam-01731:~/Rapimoney_D[redacted]gnment-----/3/Insfrastructure_as_a_code$
shivam_yadav@Shivam-01731:~/Rapimoney_Dev[redacted]ssignment-----/3/Insfrastructure_as_a_code$ 
```

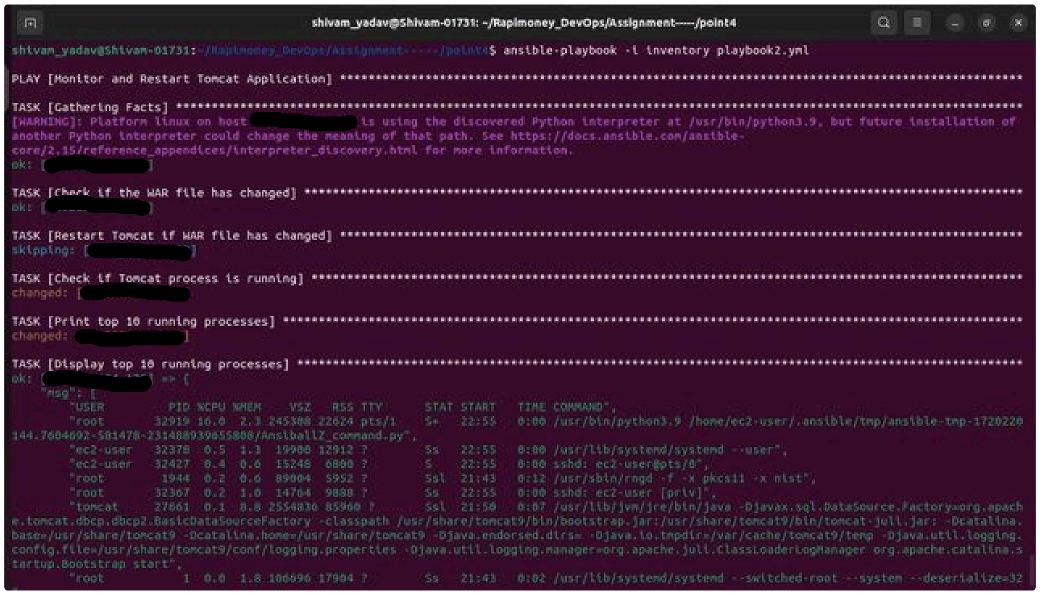
Task 4: Create Ansible Playbook to restart tomact application if new if any change war file.How to check process up and running using ansible.Also print top 10 running process.

For this task ,

- We are using time based approach method (we can also use checksum based method) for achieving the same but every method is having there pro's and con's. For TimeBased , It's simplicity and efficiency.
- We have created playbook.yml for the same.
- we will run the cmd on terminal :

```
$ ansible-playbook -i inventory playbook.yml
```

Task 4 : Output



```
shivam_yadav@Shivam-01731: ~/Rapimoney_DevOps/Assignment-----/point4
shivam_yadav@Shivam-01731:~/Rapimoney_DevOps/Assignment-----/point4$ ansible-playbook -i inventory playbook2.yml

PLAY [Monitor and Restart Tomcat Application] *********************************************************************************

TASK [Gathering Facts] ********************************************************************************************************
[WARNING]: Platform linux on host [redacted] is using the discovered Python interpreter at /usr/bin/python3.9, but future installation of
another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [[redacted]]

TASK [Check if the WAR file has changed] **************************************************************************************
ok: [[redacted]]

TASK [Restart Tomcat if WAR file has changed] *********************************************************************************
skipping: [[redacted]]

TASK [Check if Tomcat process is running] *************************************************************************************
changed: [[redacted]]

TASK [Print top 10 running processes] *****************************************************************************************
changed: [[redacted]]

TASK [Display top 10 running processes] ***************************************************************************************
ok: [[redacted]] => {
    "msg": [
        "USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND",
        "root       32919 16.0  2.3 245308 22624 pts/1    S+   22:55   0:00 /usr/bin/python3.9 /home/ec2-user/.ansible/tmp/ansible-tmp-1720220
144.7604692-501478-231488939655808/AnsiballZ_command.py",
        "ec2-user   32378  0.5  1.3  19900 12912 ?        Ss   22:55   0:00 /usr/lib/systemd/systemd --user",
        "ec2-user   32427  0.4  0.6  15248  6800 ?        S    22:55   0:00 sshd: ec2-user@pts/0",
        "root        1944  0.2  0.6  89004  5952 ?        Ssl  21:43   0:12 /usr/sbin/rngd -f -x pkcs11 -x nist",
        "root       32367  0.2  1.0  14764  9888 ?        Ss   22:55   0:00 sshd: ec2-user [priv]",
        "tomcat     27661  0.1  8.8 2554836 85960 ?       Ssl  21:50   0:07 /usr/lib/jvm/jre/bin/java -Djavax.sql.DataSource.Factory=org.apach
e.tomcat.dbcp.dbcp2.BasicDataSourceFactory -classpath /usr/share/tomcat9/bin/bootstrap.jar:/usr/share/tomcat9/bin/tomcat-juli.jar: -Dcatalina.
base=/usr/share/tomcat9 -Dcatalina.home=/usr/share/tomcat9 -Djava.endorsed.dirs= -Djava.io.tmpdir=/var/cache/tomcat9/temp -Djava.util.logging.
config.file=/usr/share/tomcat9/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager org.apache.catalina.s
tartup.Bootstrap start",
        "root           1  0.0  1.8 106696 17904 ?        Ss   21:43   0:02 /usr/lib/systemd/systemd --switched-root --system --deserialize=32
```