**Step 1: Clone the repo**

* git clone <https://github.com/aws-samples/aws-three-tier-web-architecture-workshop.git>

*IAM role creation*

* Role name: ec2-three-tier-access
* Policies to attach:
  + AmazonS3ReadOnlyAccess
  + AmazonSSMReadOnlyAccess
  + AmazonSSMManagedInstanceCore

*S3 Bucket creation*

* Bucket name: theaws3tierworkshop

**Step: 2 Networking and Security**

* VPC Name: awsdemoworkshop | CIDR: 10.0.0.0/16

*Subnets Creation:*

* Public-Web-Subnet-AZ-1: 10.0.0.0/24 | AZ: us-east-1a
* Public-Web-Subnet-AZ-2: 10.0.2.0/24 | AZ: us-east-1b
* Private-Subnet-AZ1: 10.0.4.0/24 | AZ: us-east-1a
* Private-Subnet-AZ2: 10.0.6.0/24 | AZ: us-east-1b
* Private-DB-Subnet-AZ1: 10.0.8.0/24 | AZ: us-east-1a
* Private-DB-Subnet-AZ2: 10.0.10.0/24 | AZ: us-east-1b

*Internet Gateway:*

* Name: three-tier-demo-igw
* Attach to VPC

*NAT Gateways:*

* NAT Gateway AZ1: Subnet - Public-Web-Subnet-AZ-1
* NAT Gateway AZ2: Subnet - Public-Web-Subnet-AZ-2

*Route Tables:*

* Public Route Table: Add route to IGW, associate with Public-Web-Subnet-AZ-1 and Public-Web-Subnet-AZ-2
* Private Route Table AZ1: Add route to NAT Gateway AZ1, associate with Private-Subnet-AZ1

*Security Groups:*

Internet Facing Load Balancer SG: -

* Name: internetfacing-lb-sg
* Inbound rules:-
* Type:- HTTP Protocol:- TCP Port Range:- 80 Source:- Anywhere IPv4

Web Tier SG: -

* Name: WebTier-sg
* Inbound rules:-
* Type:- HTTP Source:- MY IP
* Type:- HTTP Source:- custom add InternetFacing-lb-sg

Internal Load Balancer SG: -

* Name: internal-lb-sg
* Inbound rules:-
* Type:- HTTP Source:- custom add WebTier-sg

App Tier SG: -

* Name: Privateinstances-sg
* Inbound rules:-
* Type:- Custom TCP Port:- 4000 Source:- internal-lb-sg custom
* Type:- Custom TCP Port:- 4000 Source:- MY IP

Database SG: -

* Name: DB-sg
* Inbound rules:-
* Type:- MySQL/Aurora Source:- custom add Privateinstances-sg

**Step:- 3 Database Deployment**

*Create subnet group:*

* Name: three-tier-subnet-group
* Subnets: Private-DB-Subnet-AZ1, Private-DB-Subnet-AZ2

*RDS:*

* Engine: Aurora MySQL
* Template: Dev/Test
* Configuration options:- Aurora Standard
* Multi-AZ deployment:- (recommended for scaled availability)
* VPC: awsdemoworkshop
* SG: DB-sg
* Performance Insights: Disabled

**Step:- 4 App Tier Instance Deployment**

* Name: mywebserver1
* AMI: Amazon Linux
* Type: t2.micro
* Subnet: Private-Subnet-AZ1
* SG: privateinstance-sg
* IAM: ec2-three-tier-access

Commands:

* sudo -su ec2-user
* ping 8.8.8.8
* sudo wget https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm
* sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
* sudo rpm -ivh mysql57-community-release-el7-11.noarch.rpm
* sudo dnf clean all
* sudo dnf install mysql-community-client -y
* sudo dnf install mysql-community-server -y
* mysql --version
* sudo systemctl enable mysqld
* sudo systemctl start mysqld
* sudo systemctl status mysqld
* mysql -h CHANGE-TO-your-RDS-ENDPOINT -u USER-NAME -p
* CREATE DATABASE webappdb;
* SHOW DATABASES;
* USE webappdb;
* CREATE TABLE IF NOT EXISTS transactions (id INT NOT NULL AUTO\_INCREMENT, amount DECIMAL(10,2), description VARCHAR(100), PRIMARY KEY(id));
* SHOW TABLES;
* INSERT INTO transactions (amount, description) VALUES (200, 'grocery');
* SELECT \* FROM transactions;
* Git bash - clone the git repo
* cd application-code/app-tier
* vi .env.development.js
* Upload the app-tier to s3 bucket
* exit mysql
* curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
* source ~/.bashrc
* nvm install 16
* nvm use 16
* nvm install -g pm2
* cd ~/
* aws s3 cp s3://BUCKET-NAME/app-tier --recursive
* cd ~/app-tier
* npm install
* pm2 start index.js
* pm2 list
* pm2 logs
* pm2 startup (run the command that shows after running this)
* pm2 save
* sudo curl http://localhost:4000/health
* curl http://localhost:4000/transaction

**Step:- 5 Internal Load Balancing and Auto Scaling**

* Select the instance we created go to Actions>Image and Templates>Create Image
* Name:- AppTierImage

*Target Group: -*

* Name: AppTierTargetGroup
* Protocol: HTTP Port: 4000
* Health: /health
* Advance health check settings:- Healthy threshold:- 2

*Load Balancer:*

* Name: App-Tier-Internal-lb
* Type: Internal
* Availability Zones: us-east-1a, us-east-1b
* Subnets:
  + Private-Subnet-AZ1 (10.0.4.0/24)
  + Private-Subnet-AZ2 (10.0.6.0/24)
* Security Group: internal-lb-sg
* Add to VPC

*Launch Template: -*

* Name: AppTier-LT
* MY AMIS:- Owned by me
* Type: t2.micro
* SG: Privateinstance-sg
* IAM: ec2-three-tier-access

*Auto Scaling Group: -*

* Name: AppTier-ASG
* Template: AppTier-LT
* Availability zones:- Private-Subnet-AZ1, Private-Subnet-AZ2
* Attach to Load Balancer
* desired capacity:- 2
* maximum capacity:- 2
* minimum capacity:- 2

**Step:- 6 Web Tier Instance Deployment**

Edit nginx.conf with internal load balancer DNS Upload nginx.conf + web-tier to S3

*Web Instance Deployment:*

* Name: Web Tier
* AMI: Amazon Linux
* Type: t2.micro
* Subnet: Public-Web-Subnet-AZ-1
* Public IP: Enabled
* SG: WebTier-sg
* IAM: ec2-three-tier-access-role
* Enable DNS hostname

Assign IAM Role to EC2 instance (Actions > Security > Modify IAM Role)

Commands:

* sudo -su ec2-user
* ping 8.8.8.8
* curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
* source ~/.bashrc
* nvm install 16
* nvm use 16
* cd ~/
* aws s3 cp s3://theaws3tierworkshop/web-tier/web-tier –recursive
* cd ~/web-tier
* npm install
* npm run build
* sudo yum install nginx -y
* cd /etc/nginx
* sudo rm nginx.conf
* sudo aws s3 cp s3://theaws3tierworkshop/nginx.conf .
* sudo vi nginx.conf → (Replace internal load balancer DNS)
* sudo systemctl restart nginx
* sudo systemctl status nginx
* chmod -R 755 /home/ec2-user
* sudo chkconfig nginx on

Check App: - Use App Tier public IP with :80