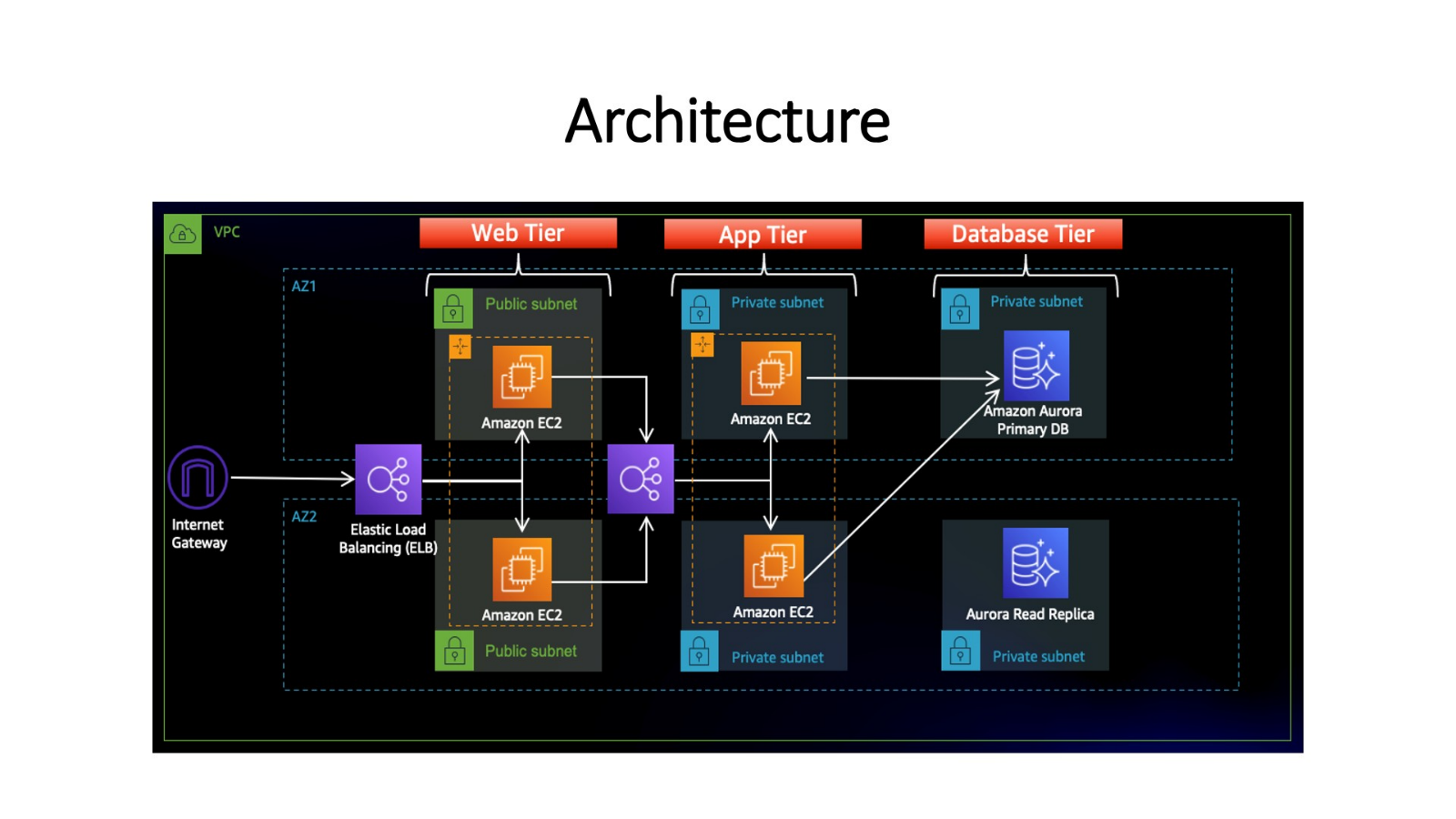
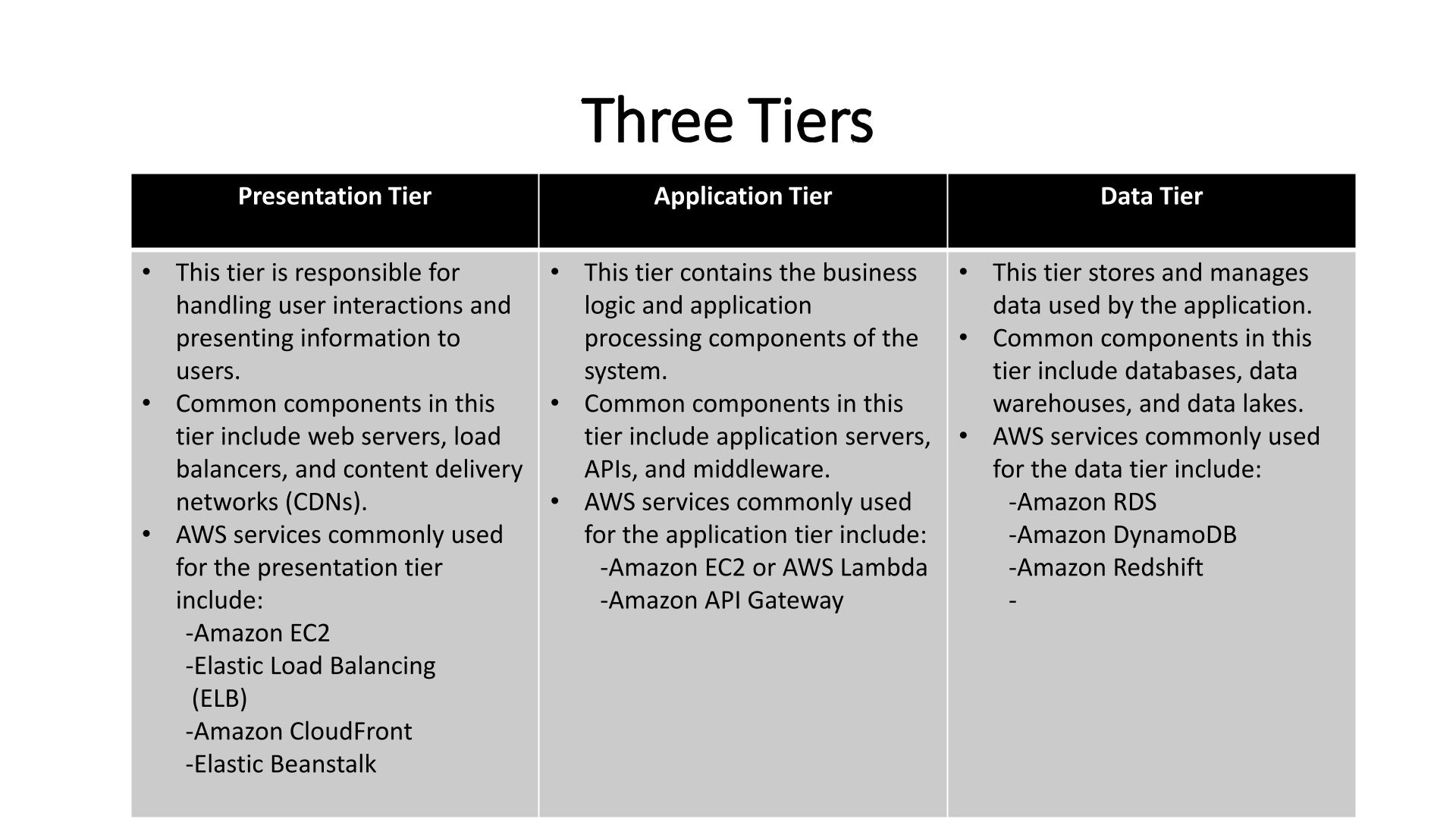
**Kubernetes Three-Tier Application**

****

****

**1] Now we create Backend Directory in which we create Dockerfile, deployment.yaml, service.yaml & init.sql**

**Backend:-**

**Dockerfile:**

FROM mysql

ENV MYSQL\_ROOT\_PASSWORD="1234"

ENV MYSQL\_DATABASE="studentapp"

COPY init.sql /docker-entrypoint-initdb.d/

**deployment.yaml:**

apiVersion: apps/v1

kind: Deployment

metadata:

name: backend-app

spec:

replicas: 1

selector:

matchLabels:

app: backend-app

strategy:

type: RollingUpdate

template:

metadata:

labels:

app: backend-app

spec:

containers:

- name: backend-app

image: shan20000/k8s\_3teir\_project:backend

ports:

- name: java

containerPort: 8080

protocol: TCP

**init.sql:**

CREATE DATABASE IF NOT EXISTS studentapp;

USE studentapp;

CREATE TABLE IF NOT EXISTS students (

student\_id INT NOT NULL AUTO\_INCREMENT,

student\_name VARCHAR(100) NOT NULL,

student\_addr VARCHAR(100) NOT NULL,

student\_age VARCHAR(3) NOT NULL,

student\_qual VARCHAR(20) NOT NULL,

student\_percent VARCHAR(10) NOT NULL,

student\_year\_passed VARCHAR(10) NOT NULL,

PRIMARY KEY (student\_id)

);

**service.yaml:**

apiVersion: v1

kind: Service

metadata:

name: backend-service

spec:

selector:

app: backend-app

ports:

- name: http

targetPort: 3306

port: 3306

type: ClusterIP

**2] Here we create Frontend Directory in which we create Dockerfile, deployment.yaml, service.yaml & config**

**Frontend:-**

**Dockerfile:**

FROM tomcat:9.0-slim

WORKDIR /opt

ENV APP\_HOME=/usr/local/tomcat

ENV PORT=8080

ADD https://s3-us-west-2.amazonaws.com/studentapi-cit/student.war $APP\_HOME/webapps/

ADD https://s3-us-west-2.amazonaws.com/studentapi-cit/mysql-connector.jar $APP\_HOME/lib

COPY config /opt

RUN sed -i '20r /opt/config' /usr/local/tomcat/conf/context.xml

EXPOSE $PORT

CMD ["catalina.sh", "run"]

**Service.yaml:**

apiVersion: v1

kind: Service

metadata:

name: frontend-service

spec:

selector:

app: frontend-app

ports:

- name: http

targetPort: 8080

port: 80

type: LoadBalancer

**config:**

<Resource name="jdbc/TestDB" auth="Container" type="javax.sql.DataSource"

maxTotal="100" maxIdle="30" maxWaitMillis="10000" username="root"

password="1234" driverClassName="com.mysql.jdbc.Driver"

url="jdbc:mysql://backend-service:3306/studentapp"/>

**deployment.yaml:**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: backend-app**

**spec:**

**replicas: 1**

**selector:**

**matchLabels:**

**app: backend-app**

**strategy:**

**type: RollingUpdate**

**template:**

**metadata:**

**labels:**

**app: backend-app**

**spec:**

**containers:**

**- name: backend-app**

**image: shan20000/k8s\_3teir\_project:backend**

**ports:**

**- name: java**

**containerPort: 8080**

**protocol: TCPapiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: frontend-app**

**spec:**

**replicas: 1**

**selector:**

**matchLabels:**

**app: frontend-app**

**strategy:**

**type: RollingUpdate**

**template:**

**metadata:**

**labels:**

**app: frontend-app**

**spec:**

**containers:**

**- name: frontend-app**

**image: shan20000/k8s\_3teir\_project:frontend**

**ports:**

**- name: tomcat**

**containerPort: 8080**

**protocol: TCP**

**3] Here we create Frontend Directory in which we create Dockerfile, deployment.yaml, service.yaml & nginx.conf**

**Proxy:-**

**Dokerfile:**

FROM nginx:latest

# Copy the custom configuration file to the NGINX directory

COPY nginx.conf /etc/nginx/nginx.conf

# Remove the default.conf file

RUN rm /etc/nginx/conf.d/default.conf

# Expose the necessary port (e.g., 80 for HTTP)

EXPOSE 80

**service.yaml:**

apiVersion: v1

kind: Service

metadata:

name: proxy-service

spec:

selector:

app: proxy-app

ports:

- name: http

targetPort: 80

port: 80

type: LoadBalancer

**nginx.config:**

user nginx;

worker\_processes 1;

error\_log /var/log/nginx/error.log warn;

pid /var/run/nginx.pid;

events {

worker\_connections 1024;

}

http {

include /etc/nginx/mime.types;

default\_type application/octet-stream;

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

sendfile on;

tcp\_nopush on;

tcp\_nodelay on;

keepalive\_timeout 65;

types\_hash\_max\_size 2048;

include /etc/nginx/conf.d/\*.conf;

server {

listen 80;

server\_name localhost;

location / {

proxy\_pass http://frontend-service/student/;

}

}

}

**deployment.yaml**:

apiVersion: apps/v1

kind: Deployment

metadata:

name: proxy-app

spec:

replicas: 1

selector:

matchLabels:

app: proxy-app

strategy:

type: RollingUpdate

template:

metadata:

labels:

app: proxy-app

spec:

containers:

- name: proxy-app

image: shan20000/k8s\_3teir\_project:proxy

ports:

- name: proxy

containerPort: 80

protocol: TCP

**step 1:-**

Clone dockerfile of frontend, backend and proxy and Build it. [give your dockerhub repository name]

git init

git clone https://github.com/shantanu20000/k8s.git

cd k8s/

cd backend/

docker build -t shan20000/k8s\_3teir\_project:backend .

cd ../frontend

docker build -t shan20000/k8s\_3teir\_project:frontend .

cd ../proxy

docker build -t shan20000/k8s\_3teir\_project:proxy .

docker login

Give login and tocken of github

Push images to dockerhub [ give your dockerhub repository path ]

docker push shan20000/k8s\_3teir\_project:backend

docker push shan20000/k8s\_3teir\_project:frontend

docker push shan20000/k8s\_3teir\_project:proxy

**step 2:**

Apply deployment.yml and service.yaml file on Cloud Shell. go in every directory [backend, frontend, proxy]

kubectl apply -f deployment.yaml

kubeclt apply -f service.yaml

**step 3:**

View all services copy proxy loadbalancer DNS and paste it on url bar of Crome

kubeclt get svc

You will see Student form fill up It and check data is store or not .

