Online Shopping Application (Sprint 2)

1) Docker compose:

Steps to create a Docker file:

- The first line has to start with the **FROM** keyword. It tells docker, from which base image you want to base your image from. In our case, we are creating an image from the **openjdk:16**.
- The **EXPOSE** instruction informs Docker that the container listens on the specified network ports at runtime.
- The **ADD** instruction copies new files, directories or remote file URLs from source and adds them to the file system of the image at the path destination.
- The **ENTRYPOINT** instruction makes your container run as an executable. The executable command for java is: ["java", "-jar", "jar-filename.jar"].

```
FROM openjdk:16-alpine3.13
LABEL maintainer="pranaysingireddy24@gmail.com"
EXPOSE 8080
ADD target/Online-Shopping.jar app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

Installation steps for docker:

- Download Docker:
  https://docs.docker.com/desktop/windows/install/
- Double –click Install Docker.
- Follow the install wizard: accept the license, authorize the installer, and proceed with the installation.
- Click finish to launch Docker.
- Docker starts automatically.

Steps to create a docker image:

- Open a terminal and go to the directory with the Docker file.
- Now build the container image using the **docker build** command:

```
$ docker build –t <image-name> .
```

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppingAppilcationSprintOne>docker images
REPOSITORY                                  TAG       IMAGE ID       CREATED        SIZE
online-shopping                             latest    98d4261141f6   44 hours ago   362MB
pranaysingireddy1/online-shopping           latest    98d4261141f6   44 hours ago   362MB
employee-service-postgres                   latest    b1fef619f0a6   2 days ago     366MB
pranaysingireddy1/employee-service-postgres 0.0.2     b1fef619f0a6   2 days ago     366MB
pranaysingireddy1/employee-service          0.0.2     cda44d03e482   2 days ago     366MB
employee-service                            latest    f0621f3fbb05   3 days ago     366MB
employee-service-h2                         latest    0a08eb75a4fa   4 days ago     365MB
pranaysingireddy1/employee-service-h2       latest    0a08eb75a4fa   4 days ago     365MB
ecommerce-rest-api                          latest    699fa6fcf410   10 days ago    903MB
```

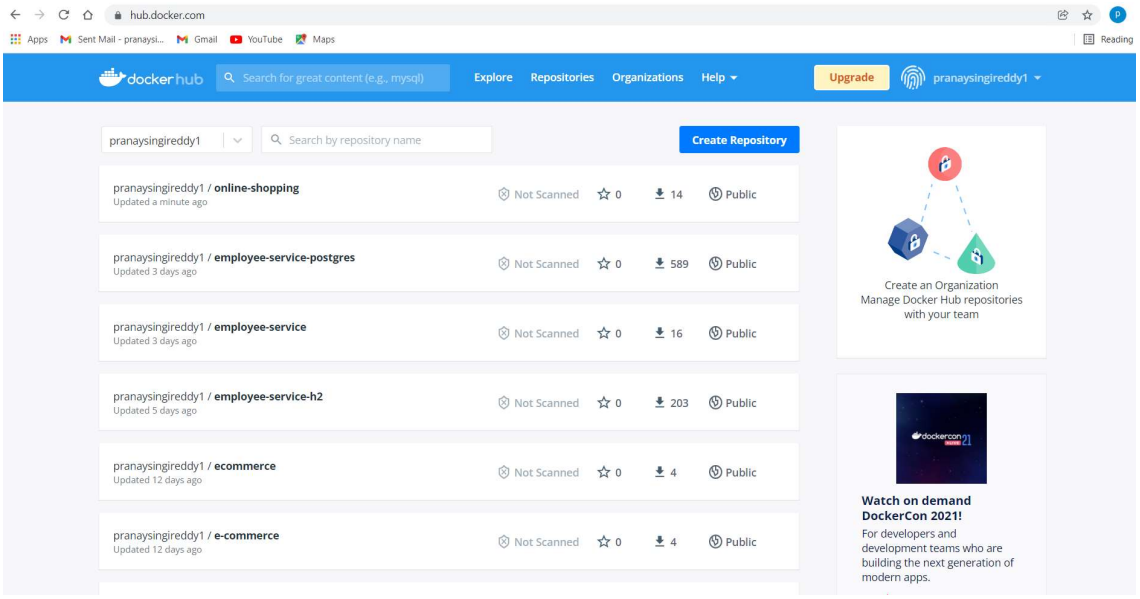
Steps to push the image onto the docker hub:

- Login to the docker hub with the username.
- Tag the image using the **docker tag** command:

```
$ docker tag <image-name> username/image-name
```

- Push the image into the docker hub using the **docker push** command:

```
$ docker push username/image-name
```



Steps to create docker-compose file:

- At the root of the app project, create a file named **docker-compose.yml**.

- In the compose file, we'll start off by defining the schema version.

```
version: "3.7"
```

- Next, we'll define the list of services (or containers) we want to run as part of our application.

```
version: "3.7"

services:
```

And now, we'll start migrating a service at a time into the compose file.
This Compose file defines two services: app and postgresql

- First, let's define the service entry and the image for the container.
- Migrate the -p 8080:8080 part of the command by defining the ports for the service.
- We will first define the new service and name it postgresql.
- Finally, we only need to specify the environment variables.

```
version: '3.7'
services:
  app:
    container_name: online-shopping
    image: online-shopping
    ports:
      - 8080:8080
    depends_on:
      - postgresqldb
    links:
      - postgresqldb:postgres
  postgresqldb:
    image: "postgres:latest"
    ports:
      - 5432:5432
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: postgres
```

Application Properties:

```
spring.datasource.url=jdbc:postgresql://postgresqldb:5432/postgres
spring.datasource.username=postgres
spring.datasource.password=postgres
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
spring.datasource.driverClassName=org.postgresql.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.generate-ddl=true
```

- Start up the application stack using the **docker-compose up** command.

```
$ docker-compose up
```

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppirgAppilcationSprintOne>docker-compose up
[+] Running 2/0
 - Container onilineshoppingappilcationsprintone_postgresqldb_1  Created                                    0.0s
 - Container online-shopping                                     Created                                    0.0s
Attaching to postgresqldb_1, online-shopping
postgresqldb_1   |
postgresqldb_1   | PostgreSQL Database directory appears to contain a database; Skipping initialization
postgresqldb_1   |
postgresqldb_1   | 2021-12-20 03:39:55.856 UTC [1] LOG:  starting PostgreSQL 14.1 (Debian 14.1-1.pgdg110+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 10.2.1-6) 10.2.1 20210110, 64-bit
postgresqldb_1   | 2021-12-20 03:39:55.856 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
postgresqldb_1   | 2021-12-20 03:39:55.856 UTC [1] LOG:  listening on IPv6 address "::", port 5432
postgresqldb_1   | 2021-12-20 03:39:55.861 UTC [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
postgresqldb_1   | 2021-12-20 03:39:55.865 UTC [27] LOG:  database system was shut down at 2021-12-18 11:43:24 UTC
postgresqldb_1   | 2021-12-20 03:39:55.870 UTC [1] LOG:  database system is ready to accept connections
online-shopping  |
online-shopping  |   .   ____          _            __ _ _
online-shopping  |  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
online-shopping  | ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
online-shopping  |  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
online-shopping  |   '  |____| .__|_| |_|_| |_\__, | / / / /
online-shopping  |  =========|_|==============|___/=/_/_/_/
online-shopping  |  :: Spring Boot ::                (v2.6.0)
online-shopping  |
online-shopping  | 2021-12-20 03:39:59.689  INFO 1 --- [           main] c.c.springboot.MyProjectDemoApplication  : Starting MyProjectDemoApplication v0.0.1-SNAPSHOT using Java 16-ea on 0773e30ff1fa with PID 1 (
/app.jar started by root in /)
online-shopping  | 2021-12-20 03:39:59.691  INFO 1 --- [           main] c.c.springboot.MyProjectDemoApplication  : No active profile set, falling back to default profiles: default
online-shopping  | 2021-12-20 03:40:00.951  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
online-shopping  | 2021-12-20 03:40:01.045  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 82 ms. Found 7 JPA repository interfaces.
online-shopping  | 2021-12-20 03:40:01.652  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
online-shopping  | 2021-12-20 03:40:01.665  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
online-shopping  | 2021-12-20 03:40:01.665  INFO 1 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.55]
online-shopping  | 2021-12-20 03:40:01.730  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
online-shopping  | 2021-12-20 03:40:01.730  INFO 1 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1990 ms
online-shopping  | 2021-12-20 03:40:01.915  INFO 1 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
online-shopping  | 2021-12-20 03:40:01.966  INFO 1 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.6.1.Final
online-shopping  | 2021-12-20 03:40:02.113  INFO 1 --- [           main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
online-shopping  | 2021-12-20 03:40:02.198  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
online-shopping  | 2021-12-20 03:40:02.399  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
online-shopping  | 2021-12-20 03:40:02.427  INFO 1 --- [           main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.PostgreSQLDialect
online-shopping  | 2021-12-20 03:40:03.219  INFO 1 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.int
ernal.NoJtaPlatform]
online-shopping  | 2021-12-20 03:40:03.227  INFO 1 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
online-shopping  | 2021-12-20 03:40:04.154  WARN 1 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed dur
ing view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
online-shopping  | 2021-12-20 03:40:04.807  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
online-shopping  | 2021-12-20 03:40:04.820  INFO 1 --- [           main] c.c.springboot.MyProjectDemoApplication  : Started MyProjectDemoApplication in 5.564 seconds (JVM running for 5.969)
online-shopping  | Server Started
```

Deploying the application in Kubernetes environment:

Application Properties:

```
spring.datasource.driverClassName=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://${DB_HOST}:5432/${DB_NAME}
spring.datasource.username=${POSTGRES_USER}
spring.datasource.password=${POSTGRES_PASSWORD}
spring.jpa.hibernate.ddl-auto=update
```

Step-2 Creating manifest files:

Defining a service:

- The specification creates a new Service object named "**online-shopping**", which targets TCP port 9001 on any Pod with the **app=online-shopping** label.
- The default protocol for Services is TCP.
- Kubernetes assigns this Service an IP address which is used by the service proxies.
- The controller for the Service selector continuously scans for Pods that match its selector, and then posts any updates to an Endpoint object also named **"online-shopping "**.

Defining a deployment:

- It creates a ReplicaSet to bring up three **plant-nursery-postgres** Pods.
- A deployment named **plant-nursery-postgres** is created, indicated by the **.metadata.name** field.
- The deployment creates three replicated Pods, indicated by the **.spec.replicas** field.
- The **.spec.selector** field defines how the Deployment finds which Pods to manage. In this case, you select a label that is defined in the Pod template (app: plant-nursery-postgres).
- The template field contains the following sub-fields:
    - The Pods are labelled app: plant-nursery-postgres using the .**metadata.labe**ls field.
    - The Pod template's specification, or .**template.spec field**, indicates that the Pods run one container, plant-nursery-postgres, which runs the plant- nursery-postgres DocHub image.

Creating a ConfigMap:

The ConfigMap configures the container(s) in Pod based on the data in the ConfigMap.

Creating a secret file:

- A Secret is an object that contains a small amount of sensitive data such as a password, a token, or a key.
- When creating a `Pod`, Kubernetes automatically creates a service account Secret and automatically modifies your Pod to use this Secret.
- When using this Secret type, the data field of the Secret must contain one of the following two keys:
    - username: the user name for authentication.
    - password: the password or token for authentication.

Step-3 Installation of minikube:

- Download the latest release of minikube from: https://minikube.sigs.k8s.io/docs/start/
- From a terminal with administrator access (but not logged in as root), run:

```
$ minikube start
```

```
C:\Users\Lavanya>minikube start
* minikube v1.24.0 on Microsoft Windows 10 Pro 10.0.19042 Build 19042
* Using the docker driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
! Executing "docker container inspect minikube --format={{.State.Status}}" took an unusually long time: 3.5029163s
* Restarting the docker service may improve performance.
* Restarting existing docker container for "minikube" ...
* Preparing Kubernetes v1.22.3 on Docker 20.10.8 ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

- Docker engine already provides kubectl pre-installed, so in order to check whether it is installed check for the version by using the following command:

```
$ kubectl version
```

```
C:\Users\Lenovo>kubectl version
Client Version: version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.4", GitCommit:"b695d79d4f967c403a96986f1750a35eb75e75f1", GitTreeState:"clean", BuildDate:"2021-11
-17T15:48:33Z", GoVersion:"go1.16.10", Compiler:"gc", Platform:"windows/amd64"}
Server Version: version.Info{Major:"1", Minor:"21+", GitVersion:"v1.21.2-eks-06eac09", GitCommit:"5f6d83fe4cb7febb5f4f4e39b3b2b64ebbbe3e97", GitTreeState:"clean", Build
Date:"2021-09-13T14:20:15Z", GoVersion:"go1.16.5", Compiler:"gc", Platform:"linux/amd64"}
```

Step-4 Build and push the image to docker hub:

Build and push the image into the docker hub using the above mentioned commands.

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppingAppilcationSprintOne\k8s1>docker i
REPOSITORY                                TAG       IMAGE ID       CREATED        SIZE
online-shopping                           latest    cfc902d65bf3   3 hours ago    362MB
pranaysingireddy1/online-shopping         0.0.1     cfc902d65bf3   3 hours ago    362MB
pranaysingireddy1/online-shopping         latest    98d4261141f6   2 days ago     362MB
employee-service-postgres                 latest    b1fef619f0a6   3 days ago     366MB
pranaysingireddy1/employee-service-postgres  0.0.2  b1fef619f0a6   3 days ago     366MB
pranaysingireddy1/employee-service        0.0.2     cda44d03e482   3 days ago     366MB
employee-service                          latest    f0621f3fbb05   4 days ago     366MB
employee-service-h2                       latest    0a08eb75a4fa   5 days ago     365MB
pranaysingireddy1/employee-service-h2     latest    0a08eb75a4fa   5 days ago     365MB
ecommerce-rest-api                        latest    699fa6fcf410   10 days ago    903MB
<none>                                    <none>    aea9b698d7d1   2 weeks ago    113MB
postgres                                  latest    e94a3bb61224   2 weeks ago    374MB
gcr.io/k8s-minikube/kicbase               v0.0.28   e2a6c047bedd   2 months ago   1.08G
<none>                                    <none>    0f2d3075cca7   6 months ago   183MB
<none>                                    <none>    5ab125058b18   6 months ago   145MB
<none>                                    <none>    87c364bbeb7b   6 months ago   84.2M
```

Step-5 Creating yaml files using kubectl command:

- To create a file, the following command is used:

```
$ kubectl create –f <file-name>
```

- To view all the pods, deployments and services created, we use the following command:

```
$ kubectl get all
```

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppingAppilcationSprintOne\k8s1>kubectl get all
NAME                                          READY   STATUS             RESTARTS         AGE
pod/ecommerce-rest-api-v9fdg                  1/1     Running            5 (87m ago)      9d
pod/employee-service-h2-5fb6fc87f-6g98s       0/1     OOMKilled          97 (4m29s ago)   5d2h
pod/employee-service-h2-5fb6fc87f-8nt87       1/1     Running            94 (4m5s ago)    5d2h
pod/employee-service-postgres-89ddc95d7-2g52g 1/1     Running            53 (5m20s ago)   4d3h
pod/employee-service-postgres-89ddc95d7-2knp7 1/1     Running            61 (3m41s ago)   4d3h
pod/employee-service-postgres-89ddc95d7-44qxw 0/1     CrashLoopBackOff   38 (3m21s ago)   4d3h
pod/myapp-rc-6kbt2                            1/1     Running            5 (87m ago)      9d
pod/myapp-rc-9dpzw                            1/1     Running            5 (87m ago)      9d
pod/myapp-rc-bvhnm                            1/1     Running            5 (87m ago)      9d
pod/myapp-replicaset-c7f4b                    1/1     Running            5                9d
pod/myapp-replicaset-cnmz6                    1/1     Running            5                9d
pod/myapp-replicaset-qg87j                    1/1     Running            5 (87m ago)      9d
pod/nginx                                     1/1     Running            5 (87m ago)      10d
pod/online-shopping-7cc4bcfb8-bnr6s           1/1     Running            1 (2m23s ago)    5m23s
pod/online-shopping-7cc4bcfb8-bx67w           1/1     Running            1 (3m3s ago)     5m23s
pod/online-shopping-7cc4bcfb8-w4xv2           1/1     Running            0                5m23s
pod/postgres-6f4cd8968f-dwp5l                 1/1     Running            0                5m10s
pod/postgres-pod                              1/1     Running            4 (87m ago)      5d18h
pod/redis-pod                                 1/1     Running            4 (87m ago)      5d18h
pod/result-app-pod                            1/1     Running            4 (87m ago)      5d18h
pod/voting-app-pod                            1/1     Running            4 (87m ago)      5d18h
pod/worker-app-pod                            1/1     Running            38 (2m45s ago)   5d18h

NAME                              DESIRED   CURRENT   READY   AGE
replicationcontroller/myapp-rc    3         3         3       9d

NAME                                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
service/db                           ClusterIP   10.102.54.121    <none>        5432/TCP       5d18h
service/employee-service-h2          ClusterIP   10.107.98.37     <none>        8080/TCP       5d2h
service/employee-service-postgres    ClusterIP   10.97.234.250    <none>        8080/TCP       4d3h
service/kubernetes                   ClusterIP   10.96.0.1        <none>        443/TCP        10d
service/online-shopping              ClusterIP   10.107.154.197   <none>        8080/TCP       5m23s
service/postgres                     ClusterIP   None             <none>        5432/TCP       5m10s
service/redis                        ClusterIP   10.103.236.226   <none>        6379/TCP       5d18h
service/result-app-service           NodePort    10.96.4.23       <none>        80:30005/TCP   5d18h
service/voting-app-service           NodePort    10.111.14.59     <none>        80:30004/TCP   5d18h

NAME                                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/employee-service-h2        1/2     2            1           5d2h
deployment.apps/employee-service-postgres  2/3     3            2           4d3h
deployment.apps/online-shopping            3/3     3            3           5m23s
deployment.apps/postgres                   1/1     1            1           5m10s
```

Port Forwarding:

- To use clusterIP, we need to use port-forward command:

```
$ kubectl port-forward svc/image-name 9002:8080
```

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppingAppilcationSprintOne\k8s1>kubectl port-forward svc/online-shopping 9002:8080
Forwarding from 127.0.0.1:9002 -> 8080
Forwarding from [::1]:9002 -> 8080
Handling connection for 9002
Handling connection for 9002
```

http://localhost:8080/productController/addProduct

Save

POST http://localhost:8080/productController/addProduct Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
{
    "productId":"1218",
    "productName":"mobile",
    "price":10000,
    "color":"red",
    "dimension":"square",
    "specification":"i5",
    "manufacturer":"redmi",
    "quantity":1,
    "catId":"1218",
```

Body Cookies Headers (5) Test Results Status: 200 OK Time: 162 ms Size: 315 B Save Response

Pretty Raw Preview Visualize JSON

```
{
    "productId": "1218",
    "productName": "mobile",
    "price": 10000.0,
    "color": "red",
    "dimension": "square",
    "specification": "i5",
    "manufacturer": "redmi",
    "quantity": 1
}
```

http://localhost:8080/productController/viewAllProducts

Save

GET http://localhost:8080/productController/viewAllProducts Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results Status: 200 OK Time: 24 ms Size: 317 B Save Response

Pretty Raw Preview Visualize JSON

```
[
    {
        "productId": "1218",
        "productName": "mobile",
        "price": 10000.0,
        "color": "red",
        "dimension": "square",
        "specification": "i5",
        "manufacturer": "redmi",
        "quantity": 1
    }
]
```

Deploying in EKS Cluster:

Application Properties:

```
spring.datasource.driverClassName=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://${POSTGRES_HOST}:5432/postgres
spring.datasource.username=${POSTGRES_USER}
spring.datasource.password=${POSTGRES_PASSWORD}
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.hibernate.ddl-auto=update
```

Step-2 Creating manifest files:

- The manifest files also known as the "yaml" files are created just like the way these files are created in the kubernetes deployment.
- These files are as the following:
  - postgres-storage.yaml
  - postgres-secrets.yaml
  - postgres-deployment.yaml
  - postgres-deployment.yaml
  - springboot-deployment.yaml
  - springboot-service.yaml

Step-3 Installation of AWS CLI:

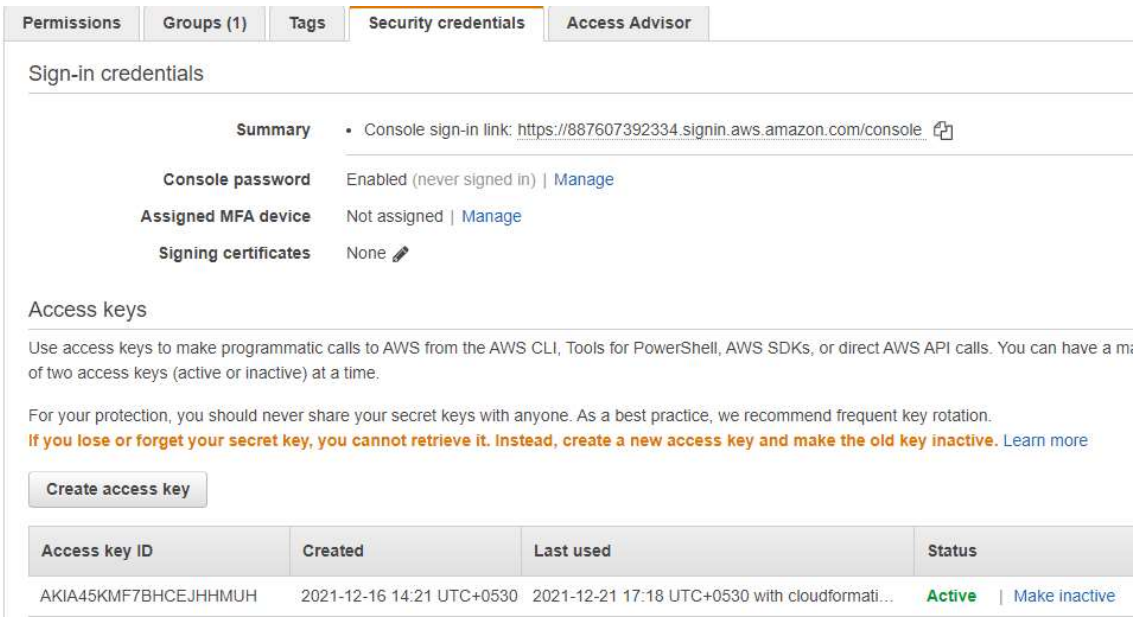
- Download and run the AWS CLI MSI installer for Windows (64-bit)

  https://awscli.amazonaws.com/AWSCLIV2.msi

- To confirm the installation, open the **Start** menu, search for `cmd` to open a command prompt window, and at the command prompt use the `aws --version` command.

```
C:\> aws --version
aws-cli/2.3.7 Python/3.8.8 Windows/10 exe/AMD64 prompt/off
```

- We need secret keys from AWS IAM account. Go to IAM in AWS and generate access key by going into the security credentials section in users.



- Download the access key generated.
- Now, in cmd configure the AWS by using the following command:

```
$ aws configure
```

Then enter the access key id, secret access

```
C:\Users\Lenovo>aws configure
AWS Access Key ID [****************HMUH]: AKIA45KMF7BHCEJHHMUH
AWS Secret Access Key [****************cC2e]: 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
Default region name [ap-south-1]: ap-south-1
Default output format [json]: json

C:\Users\Lenovo>
```

Step-4 Installation of eksctl:

- For installing the eksctl, chocolatey has to be installed first.
- In order to install Chocolatey, first, ensure that you are using an **administrative shell.**
- Copy the text specific to your command shell **- cmd.exe**.
- Paste the copied text into your shell and press Enter.
  @"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "[System.Net.ServicePointManager]::SecurityProtocol = 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"

- Wait a few seconds for the command to complete.
- After installing eksctl, run the commands as shown in the attached screenshot.

To install or upgrade `eksctl` on Windows using Chocolatey

1. If you do not already have Chocolatey installed on your Windows system, see Installing Chocolatey.
2. Install or upgrade `eksctl` .
   - Install the binaries with the following command:

     ```
     choco install -y eksctl
     ```

   - If they are already installed, run the following command to upgrade:

     ```
     choco upgrade -y eksctl
     ```

3. Test that your installation was successful with the following command.

   ```
   eksctl version
   ```

```
C:\Users\Lenovo>eksctl version
0.76.0
```

Step-5 Create a cluster:

- In order to create a cluster, the following command is used:

```
$ eksctl create cluster --name <cluster-name>  --version 1.21 --region <region-name>
--nodegroup-name <node-group-name> --node-type t2.micro –nodes 2
```

- To create or update kubeconfig for our cluster:

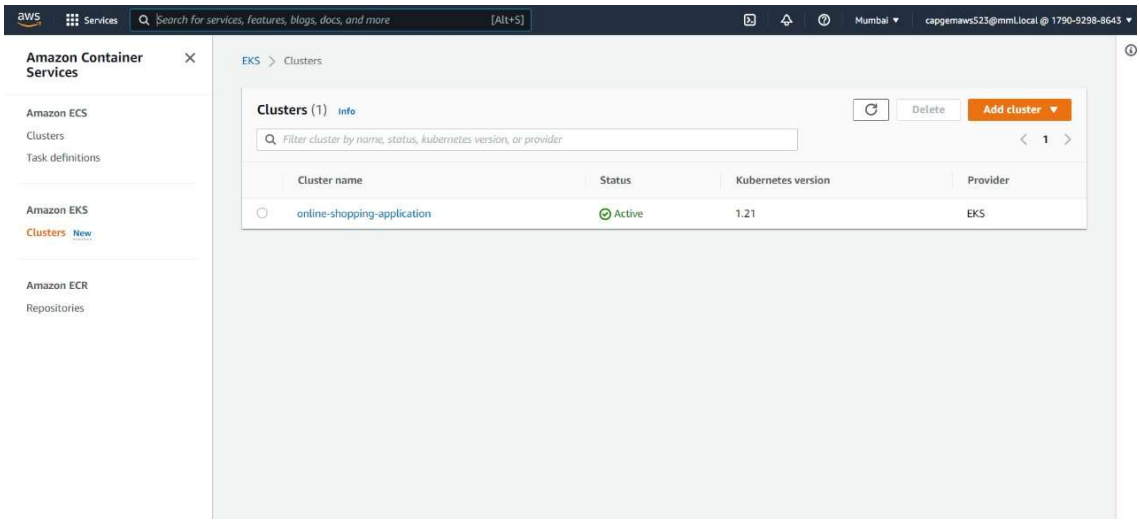
```
$ aws eks --region <region-code> update-kubeconfig --name <cluster-name>
```

- Now, create files using the kubectl command:

```
$ kubectl apply –f <file-name>
```

```
2021-12-20 21:37:03 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:38:03 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:39:04 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:40:05 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:41:05 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:42:06 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:43:07 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:44:07 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:45:08 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:46:10 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-cluster"
2021-12-20 21:48:14 [ℹ]  building managed nodegroup stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:48:15 [ℹ]  deploying stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:48:15 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:48:31 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:48:48 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:49:09 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:49:26 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:49:46 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:50:06 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:50:25 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:50:42 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:51:01 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:51:18 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:51:35 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:51:53 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:52:10 [ℹ]  waiting for CloudFormation stack "eksctl-online-shopping-application-nodegroup-online-shopping-application-node-group"
2021-12-20 21:52:10 [ℹ]  waiting for the control plane availability...
2021-12-20 21:52:11 [ℹ]  saved kubeconfig as "C:\\Users\\hp\\.kube\\config"
2021-12-20 21:52:11 [ℹ]  no tasks
2021-12-20 21:52:11 [ℹ]  all EKS cluster resources for "online-shopping-application" have been created
2021-12-20 21:52:12 [ℹ]  nodegroup "online-shopping-application-node-group" has 2 node(s)
2021-12-20 21:52:12 [ℹ]  node "ip-192-168-4-148.ap-south-1.compute.internal" is ready
2021-12-20 21:52:12 [ℹ]  node "ip-192-168-51-237.ap-south-1.compute.internal" is ready
2021-12-20 21:52:12 [ℹ]  waiting for at least 2 node(s) to become ready in "online-shopping-application-node-group"
2021-12-20 21:52:12 [ℹ]  nodegroup "online-shopping-application-node-group" has 2 node(s)
2021-12-20 21:52:12 [ℹ]  node "ip-192-168-4-148.ap-south-1.compute.internal" is ready
2021-12-20 21:52:12 [ℹ]  node "ip-192-168-51-237.ap-south-1.compute.internal" is ready
2021-12-20 21:52:19 [ℹ]  kubectl command should work with "C:\\Users\\hp\\.kube\\config", try 'kubectl get nodes'
2021-12-20 21:52:19 [ℹ]  EKS cluster "online-shopping-application" in "ap-south-1" region is ready
```

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppingAppilcationSprintOne\test>aws eks --region ap-south-1 update-kubeconfig --name online-shopping-application
Added new context arn:aws:eks:ap-south-1:179092988643:cluster/online-shopping-application to C:\Users\hp\.kube\config
```



- To view all the pods, deployments and services use the following kubectl command:

```
$ kubectl get all
```

```
C:\Users\hp\git\OnlineShoppingApplication\OnilineShoppingAppilcationSprintOne\test>kubectl get all
NAME                              READY   STATUS    RESTARTS   AGE
pod/postgres-5bdb4fc5f9-vrvhh     1/1     Running   0          2m18s

NAME                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
service/kubernetes   ClusterIP   10.100.0.1       <none>        443/TCP          24m
service/postgres     NodePort    10.100.189.189   <none>        5432:30656/TCP   75s

NAME                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/postgres   1/1     1            1           2m19s

NAME                                  DESIRED   CURRENT   READY   AGE
replicaset.apps/postgres-5bdb4fc5f9   1         1         1       2m19s
```

- Now, check in the browser by pasting the IP address in the browser:

ab066272dd14e4ac3b23116e0571cfee-2028212639.ap-south-1.elb.amazonaws.com:8080/productController/viewAllProducts