

## **TEAM. 7**

**Payal Pawale**

**Monika S A**

**Amol Mundhe**

**Siddharth katule**

### **Docker:**

Docker is an open source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.

#### **Installation steps for docker:**

1. Download Docker:

<https://docs.docker.com/desktop/windows/install/>

2. Double –click Install Docker.
3. Follow the install wizard: accept the license, authorize the installer, and proceed with the install
4. Click finish to launch Docker.
5. Docker starts automatically.
6. Docker loads a “Welcome” windows giving you tips and access to the Docker documentation.

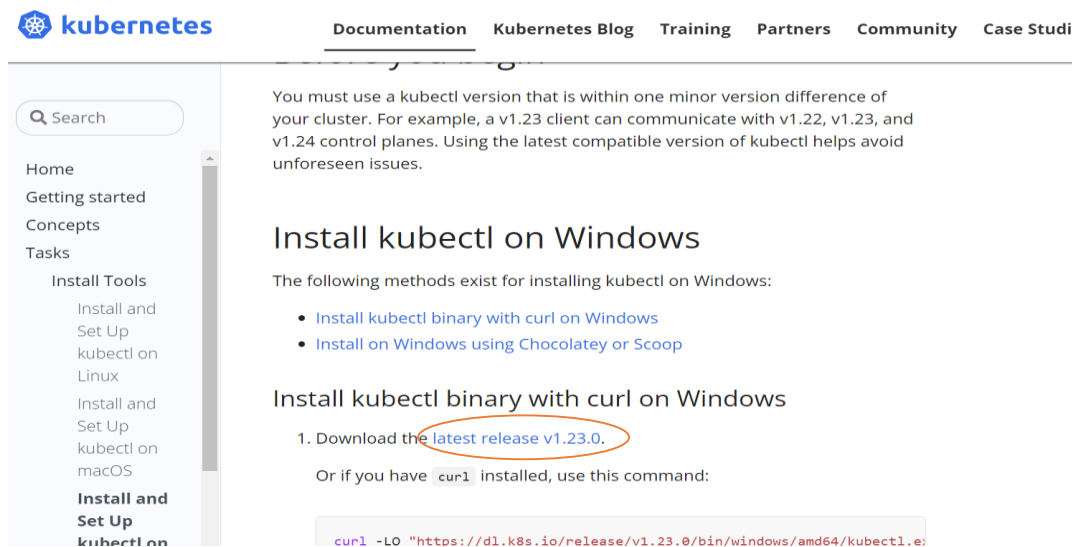
### **Kubernetes:**

Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation

1. To Install Kubernetes on windows go to this website:

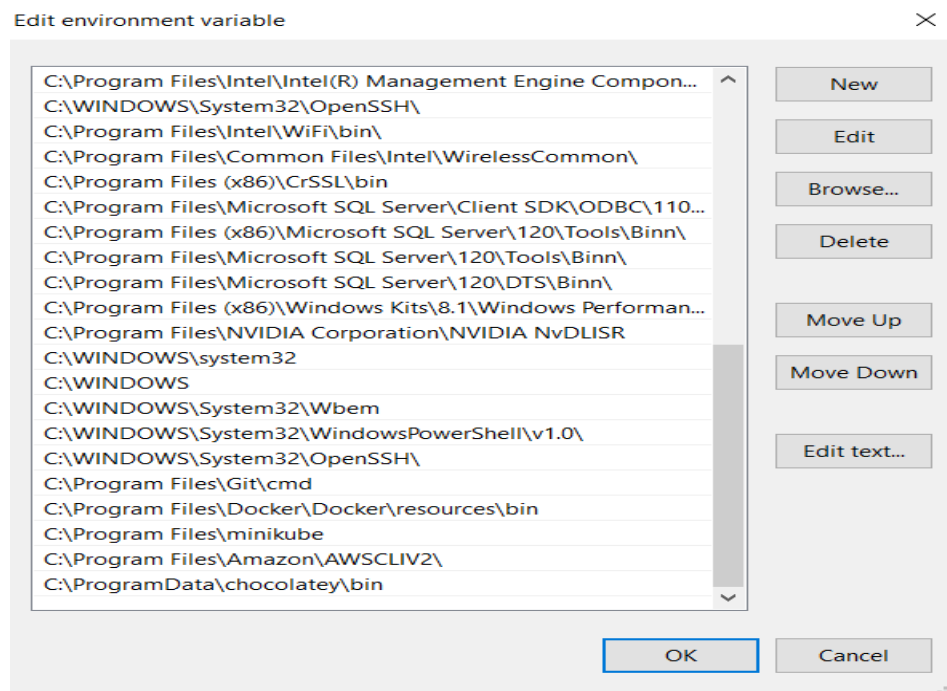
<https://kubernetes.io/docs/tasks/tools/install-kubectl-windows/>

2. Then click on latest version v1.23.0.to download the Kubernetes.



The screenshot shows the Kubernetes documentation website. The left sidebar contains navigation links: Home, Getting started, Concepts, Tasks, and Install Tools. Under Install Tools, there are links for Linux, macOS, and Windows. The main content area is titled "Install kubectI on Windows" and includes a warning about version compatibility. It lists two methods for installation: using curl or Chocolatey/Scoop. The first method, "Install kubectI binary with curl on Windows", is selected and shows a command to download the latest release v1.23.0. The command is: `curl -LO "https://dl.k8s.io/release/v1.23.0/bin/windows/amd64/kubectI.exe"`

3. Add the path in Environment variables.



4. Test to ensure the version of kubectI is the same as downloaded.

\$kubectI version --client

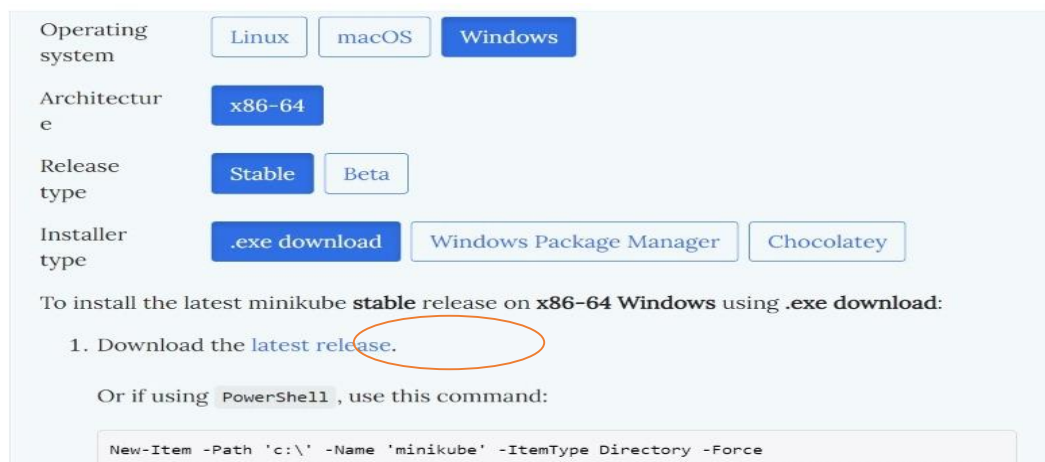
## MINIKUBE

Like kind, minikube is a tool that lets you run Kubernetes locally. Minikube runs a single-node Kubernetes cluster on your personal computer (including Windows, macOS and Linux PCs) so that you can try out Kubernetes, or for daily development work.

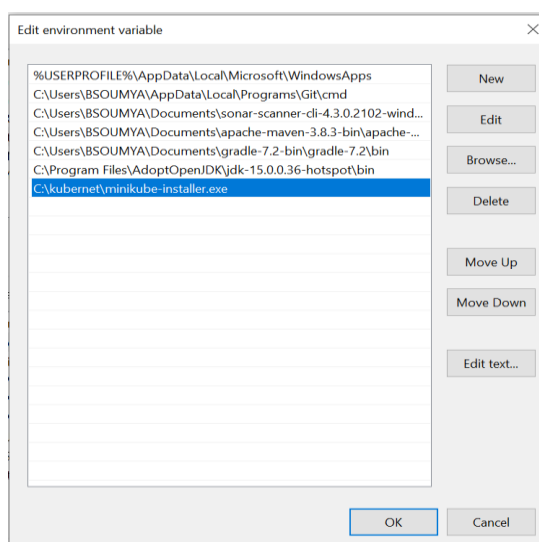
1. To Install minikube on windows go to this website:. To install minikube on windows go to this website:

<https://minikube.sigs.k8s.io/docs/start/>

2. Then click on latest release and download.



3. Add the path in Environment variables.



4. Start your cluster by using below command.

\$ Minikube start

5. Then it will be shown below.

```
Exiting due to PROVIDER_DOCKER_NOT_RUNNING: deadline exceeded running "docker version --format -": exit status 1
Suggestion: Restart the Docker service
Documentation: https://minikube.sigs.k8s.io/docs/drivers/docker/

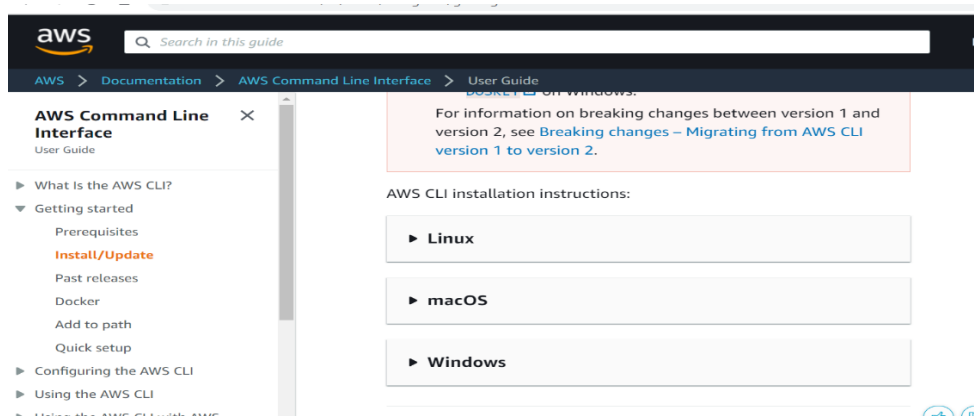
:\Users\ADINATH N MUNDHE>minikube start
minikube v1.24.0 on Microsoft Windows 10 Home Single Language 10.0.19043 Build 19043
Using the docker driver based on existing profile
Starting control plane node minikube in cluster minikube
Pulling base image ...
Restarting existing docker container for "minikube" ...
Preparing Kubernetes v1.22.3 on Docker 20.10.8 ...
Verifying Kubernetes components...
Executing "docker container inspect minikube --format={{.State.Status}}" took an unusually long time: 15.8778241s
Restarting the docker service may improve performance.
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: default-storageclass
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

## AWSCLI:

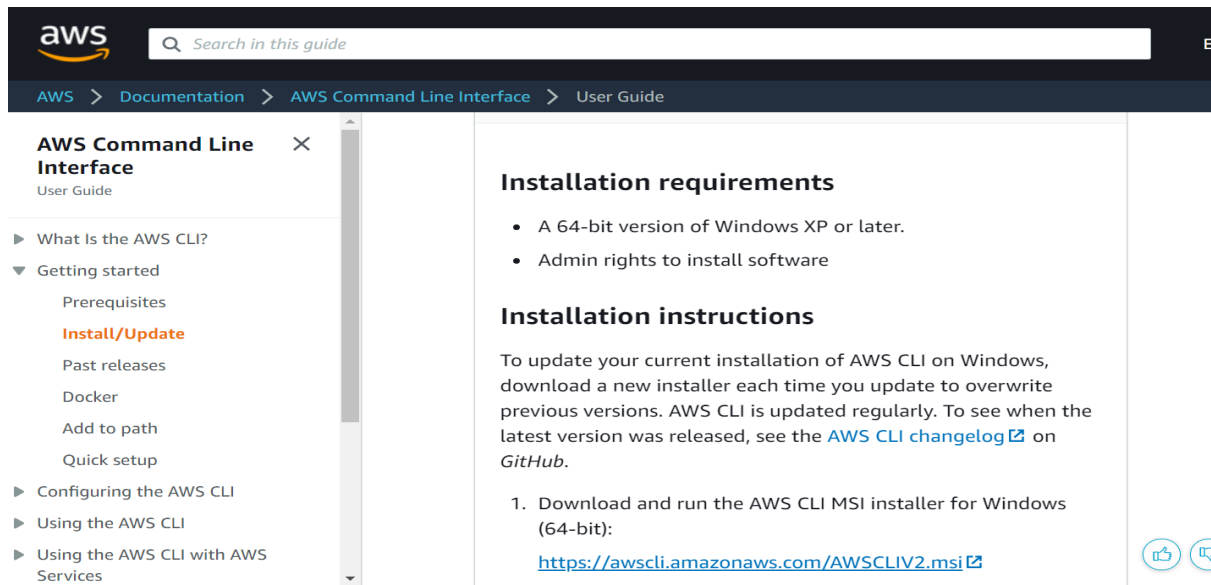
1. To Install AWSCLI on windows go to this website:

<https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting->

2. Then go to install and click on windows.



### 3. Click on link to download the awscli



The screenshot shows the AWS Command Line Interface User Guide page. The left sidebar contains a navigation menu with the following items: What is the AWS CLI?, Getting started (expanded), Prerequisites, Install/Update (highlighted in orange), Past releases, Docker, Add to path, Quick setup, Configuring the AWS CLI, Using the AWS CLI, and Using the AWS CLI with AWS Services. The main content area is titled 'Installation requirements' and 'Installation instructions'. The 'Installation requirements' section lists two bullet points: 'A 64-bit version of Windows XP or later.' and 'Admin rights to install software'. The 'Installation instructions' section contains a paragraph about updating the AWS CLI and a numbered list starting with '1. Download and run the AWS CLI MSI installer for Windows (64-bit):' followed by a link to <https://awscli.amazonaws.com/AWSCLIV2.msi>.

4. Follow the install wizard: accept the license, authorize the installer, and proceed with the install
5. Click finish to launch AWSCLI.
6. 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.

## CHOCOLATEY:

Chocolatey is a software management solution that gives you the freedom to create a simple software package and then deploy it anywhere you have Windows using any of your familiar configuration or system management tools

1. Goto chocolatey website:  
<https://chocolatey.org/install>
2. Click on take the installation course.



The screenshot shows the 'Basic Chocolatey Install' page. It includes a heading 'Basic Chocolatey Install' and a subheading 'Chocolatey installs in seconds. You are just a few steps from running choco right now!'. There are two numbered steps: '1. First, ensure that you are using an **administrative shell** - you can also install as a non-admin, check out [Non-Administrative Installation](#).' and '2. Copy the text specific to your command shell below.' Below these steps is a 'NOTE' box with a warning icon, stating: 'NOTE: Please inspect <https://community.chocolatey.org/install.ps1> prior to running any of these scripts to ensure safety. We already know it's safe, but you should verify the security and contents of **any** script from the internet you are not familiar with. All of these scripts download a remote PowerShell script and execute it on your machine. We take security very seriously. [Learn more about our security protocols](#).' Below the note are two buttons: 'Install with cmd.exe' and 'Install with powershell.exe'. Below these buttons is the text 'Install with cmd.exe' and 'Run the following command:'. At the bottom, there is a code block with the command: `> @"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -Inp`.

3. Paste the copied text into your shell and press Enter.
4. Wait a few seconds for the command to complete.
5. Then upgrade by using the below command:

```
$ choco upgrade chocolatey
```

### **EKSCTL:**

Eksctl is a tool jointly developed by AWS and Weave works that automates much of the experience of creating EKS clusters. In this module, we will use eksctl to launch and configure our EKS cluster and nodes

1. To install eksctl goto these AWS website and open Amazon EMR on EKS Development Guide.
2. If you do not already have Chocolatey installed on your Windows system, see [Installing Chocolatey](#)
3. Install or upgrade eksctl

```
$ choco install -y eksctl
```

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

```
$ choco upgrade -y eksctl
```

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

```
$ eksctl version
```

# Document – Online-Food-Delivery

## Procedure:

### 1. Steps To create Docker file:

- a. Install Docker on machine.
- b. Create project.
- c. Create a file called Docker File.
- d. Build your Docker File using properties.
- e. Save the file.

### 2. Steps to create docker image:

- a. Create a Base Container
  - b. Inspect Images
  - c. Inspect Containers
  - d. Start the Container
  - e. Modify the Running Container
  - f. Create an Image from a Container by using below command.
- **docker build -t food-delivery krumoni/ food-delivery .**
    - g. Tag the Image
  - **docker tag food-delivery krumoni/ food-delivery**
    - h. push that image into docker hub using below command.
  - **docker push krumoni/ food-delivery**

The screenshot shows the Docker Hub interface for the repository **krumoni/food-delivery**. The page includes a navigation bar with the Docker Hub logo, a search bar, and links to Explore, Repositories, Organizations, and Help. The repository page itself has tabs for General, Tags, Builds, Collaborators, Webhooks, and Settings. A banner for "Advanced Image Management" is visible. The repository details section shows the name **krumoni/food-delivery**, a note that it has no description, and the last push time of 5 hours ago. A "Docker commands" section provides the command `docker push krumoni/food-delivery:tagname`. Below this, the "Tags and Scans" section shows a table with one tag, **latest**, which was pulled 25 minutes ago and pushed 5 hours ago. A "VULNERABILITY SCANNING - DISABLED" warning is present. The "Automated Builds" section explains how to connect to GitHub or Bitbucket for automatic builds and includes an "Upgrade to Pro" button.

TAG	OS	PULLED	PUSHED
latest	linux	25 minutes ago	5 hours ago

### 3. Steps to create docker-compose file:

- create the docker-compose.yml
- Define services in a Compose file
- Run the application with compose using the below command

- **docker-compose up**

```
C:\Windows\System32\cmd.exe - docker-compose up
0270706166b39: Mounted from krumoni/spring-food-delivery-postgres
1119ff3fda9: Mounted from krumoni/spring-food-delivery-postgres
latest: digest: sha256:14502e53025ea8bcb7d51197b43d38386af8269d22c425f0331deb62dda12c size: 1163

C:\Users\Monika S A\Desktop>SPRINT-FoodDelivery>docker-compose up
Starting sprint-fooddelivery_postgresqldb_1 ... done
Recreating spring-food-delivery ... done
Attaching to sprint-fooddelivery_postgresqldb_1, food-delivery
postgresqldb_1 | PostgreSQL Database directory appears to contain a database; Skipping initialization
food-delivery_1 | 2021-12-21 10:21:45.584 UTC [1] LOG: starting PostgreSQL 14.1 (Debian 14.1-1.pgdg11+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 10.2.1-6) 10.2.1 20210110, 64-bit
food-delivery_1 | 2021-12-21 10:21:45.643 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
food-delivery_1 | 2021-12-21 10:21:45.643 UTC [1] LOG: listening on IPv6 address ":::", port 5432
food-delivery_1 | 2021-12-21 10:21:46.120 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
food-delivery_1 | 2021-12-21 10:21:46.451 UTC [27] LOG: database system was interrupted; last known up at 2021-12-20 04:58:11 UTC
food-delivery_1 | 2021-12-21 10:21:18.945 UTC [27] LOG: database system was not properly shut down; automatic recovery in progress
food-delivery_1 | 2021-12-21 10:21:59.131 UTC [27] LOG: redo starts at 0/173D428
food-delivery_1 | 2021-12-21 10:21:59.131 UTC [27] LOG: invalid record length at 0/173D510: wanted 24, got 0
food-delivery_1 | 2021-12-21 10:21:59.132 UTC [27] LOG: redo done at 0/173D408 system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s
food-delivery_1 | 2021-12-21 10:22:00.101 UTC [1] LOG: database system is ready to accept connections
food-delivery |
food-delivery | _____
food-delivery | \   /__  /  ___/  _/_/_/  /___/
food-delivery |  _/  /  /_/_/  /_/_/  /_/_/  /_/_/
food-delivery | /___/  /_/_/  /_/_/  /_/_/  /_/_/
food-delivery |
food-delivery | :: Spring Boot ::                (v2.6.1)
food-delivery |
food-delivery | 2021-12-21 10:22:01.618 INFO 1 --- [main] c.c.g.ofda.SpringFoodDeliveryApplication : Starting SpringFoodDeliveryApplication v0.0.1-SNAPSHOT on h33aa694b193 with PID 1 (/app.jar started by root in /)
food-delivery | 2021-12-21 10:22:01.642 INFO 1 --- [main] c.c.g.ofda.SpringFoodDeliveryApplication : No active profile set, falling back to default profile
es: default
food-delivery | 2021-12-21 10:22:04.669 INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
food-delivery | 2021-12-21 10:22:05.058 INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 342 ms. Found 6 JPA repository interfaces.
food-delivery | 2021-12-21 10:22:09.783 INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
food-delivery | 2021-12-21 10:22:09.788 INFO 1 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
food-delivery | 2021-12-21 10:22:09.849 INFO 1 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.55]
food-delivery | 2021-12-21 10:22:10.146 INFO 1 --- [main] o.a.c.c.c.[Tomcat].[/localhost]/[/] : Initializing Spring embedded WebApplicationContext
food-delivery | 2021-12-21 10:22:10.147 INFO 1 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: Initialization completed
```

```
C:\Windows\System32\cmd.exe - docker-compose up
```

```
food-delivery | [Spring Boot] :: Spring Boot :: (v2.6.1)
```

```
food-delivery | 2021-12-21 18:22:01.618 INFO 1 --- [main] c.g.ofda.SprintFoodDeliveryApplication : Starting SprintFoodDeliveryApplication v0.0.1-SNAPSHOT
```

```
food-delivery | Using Java 16-ea on b334a624b193 with PID 1 (/app.jar started by root in /)
```

```
food-delivery | 2021-12-21 18:22:01.642 INFO 1 --- [main] c.g.ofda.SprintFoodDeliveryApplication : No active profile set, falling back to default profile
```

```
food-delivery | 2021-12-21 18:22:04.669 INFO 1 --- [main] s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
```

```
food-delivery | 2021-12-21 18:22:05.058 INFO 1 --- [main] s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 342 ms. Found 6 JPA repository interfaces.
```

```
food-delivery | 2021-12-21 18:22:09.783 INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
```

```
food-delivery | 2021-12-21 18:22:09.848 INFO 1 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
```

```
food-delivery | 2021-12-21 18:22:09.940 INFO 1 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.55]
```

```
food-delivery | 2021-12-21 18:22:10.146 INFO 1 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
```

```
food-delivery | 2021-12-21 18:22:10.147 INFO 1 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 8183 ms
```

```
food-delivery | 2021-12-21 18:22:10.977 INFO 1 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
```

```
food-delivery | 2021-12-21 18:22:11.214 INFO 1 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 5.6.1.Final
```

```
food-delivery | 2021-12-21 18:22:11.938 INFO 1 --- [main] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
```

```
food-delivery | 2021-12-21 18:22:12.402 INFO 1 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
```

```
food-delivery | 2021-12-21 18:22:13.928 INFO 1 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
```

```
food-delivery | 2021-12-21 18:22:14.025 INFO 1 --- [main] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.PostgreSQLDialect
```

```
food-delivery | 2021-12-21 18:22:19.679 INFO 1 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
```

```
food-delivery | 2021-12-21 18:22:19.789 INFO 1 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
```

```
food-delivery | 2021-12-21 18:22:22.847 WARN 1 --- [main] JpaBaseConfigurations$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
```

```
food-delivery | 2021-12-21 18:22:24.442 INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path = ''
```

```
food-delivery | 2021-12-21 18:22:24.497 INFO 1 --- [main] c.g.ofda.SprintFoodDeliveryApplication : Started SprintFoodDeliveryApplication in 25.741 seconds ( JVM running for 33.924 )
```

```
food-delivery | Spring is Started
```

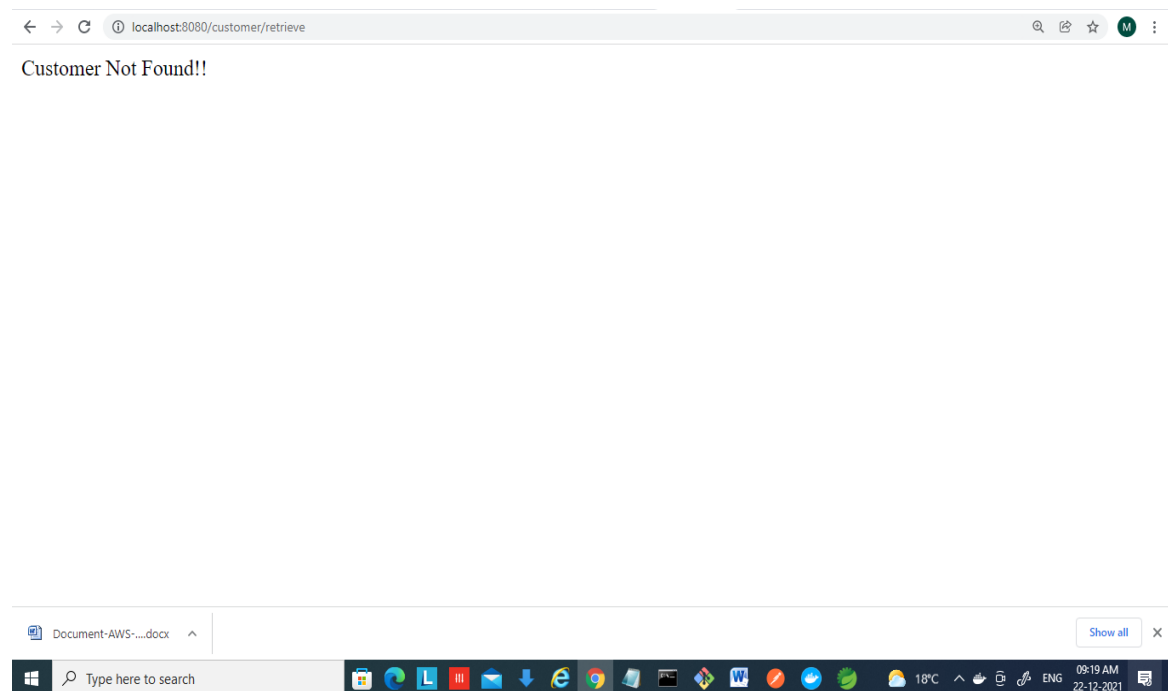
```
food-delivery | 2021-12-21 18:22:50.419 INFO 1 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
```

```
food-delivery | 2021-12-21 18:22:50.420 INFO 1 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
```

```
food-delivery | 2021-12-21 18:22:50.429 INFO 1 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 8 ms
```



Check whether the application is working with the port in chrome.



#### 4. Create the manifest files(yaml):

- a. change the application properties
- b. write yaml files
  - deployment.yaml
  - postgres-credentials.yaml
  - postgres-configmap.yaml
  - postgres-deployment.yaml
- c. Again build the jar
- d. In cmd start the minikube

#### 5. Deploy the application on Kubernetes environment:

Create an image food-delivery-system and push the image.

- **docker build -t food-delivery-system .**
- **docker tag food-delivery-system krumoni/ food-delivery-system**
- **docker push krumoni/food-delivery-system**

Change the directory using the command **cd k8s** ( In folder k8s yaml files are created)

a. write the below deployment commands in cmd:

- **kubectrl create -f deployment.yaml**
- **kubectrl create -f postgres-credentials.yaml**
- **kubectrl create -f postgres-configmap.yaml**
- **kubectrl create -f postgres-deployment.yaml**

The screenshot shows the Docker Hub interface for the repository **krumoni / food-delivery-system**. The page includes a search bar, navigation tabs (General, Tags, Builds, Collaborators, Webhooks, Settings), and a sidebar with options like Advanced Image Management, Docker commands, Tags and Scans, and Automated Builds. The main content area displays the repository name, a description (none), last pushed time (5 hours ago), and a table of tags. The 'latest' tag is highlighted, showing it was pulled 37 minutes ago and pushed 5 hours ago. A 'See all' link is provided for the tags. The 'Automated Builds' section is also visible, indicating that the repository is not currently configured for automated builds.

**krumoni / food-delivery-system**

This repository does not have a description

Last pushed: 5 hours ago

**Tags and Scans**

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
latest	linux	37 minutes ago	5 hours ago

[See all](#)

**Automated Builds**

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions.

[Upgrade to Pro](#) [Learn more](#)

b. After successful deployment ,forward the port

- **kubectl port-forward svc/ food-delivery-system 9094:8080**

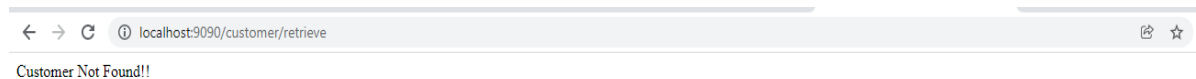
```
^C
E:\d drive\All Workspace\AWS-Workspace\SPRINT-FoodDelivery3\K8S>kubectl get all
NAME                                READY    STATUS    RESTARTS   AGE
pod/food-delivery-5b94cf476-c9kps   1/1     Running   1 (2m26s ago)    4m39s
pod/food-delivery-5b94cf476-lh4nb   1/1     Running   1 (2m27s ago)    4m39s
pod/food-delivery-5b94cf476-nsmbt   1/1     Running   1 (2m27s ago)    4m39s
pod/postgres-6f4cd8968f-7vgwg       1/1     Running   0            2m49s

NAME                                TYPE      CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/food-delivery               ClusterIP  10.104.156.70 <none>         8080/TCP    4m40s
service/kubernetes                   ClusterIP  10.96.0.1     <none>         443/TCP     11m
service/postgres                     ClusterIP  None          <none>         5432/TCP    2m50s

NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/food-delivery        3/3      3              3             4m39s
deployment.apps/postgres              1/1      1              1             2m49s

NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/food-delivery-5b94cf476  3          3          3        4m39s
replicaset.apps/postgres-6f4cd8968f      1          1          1        2m49s

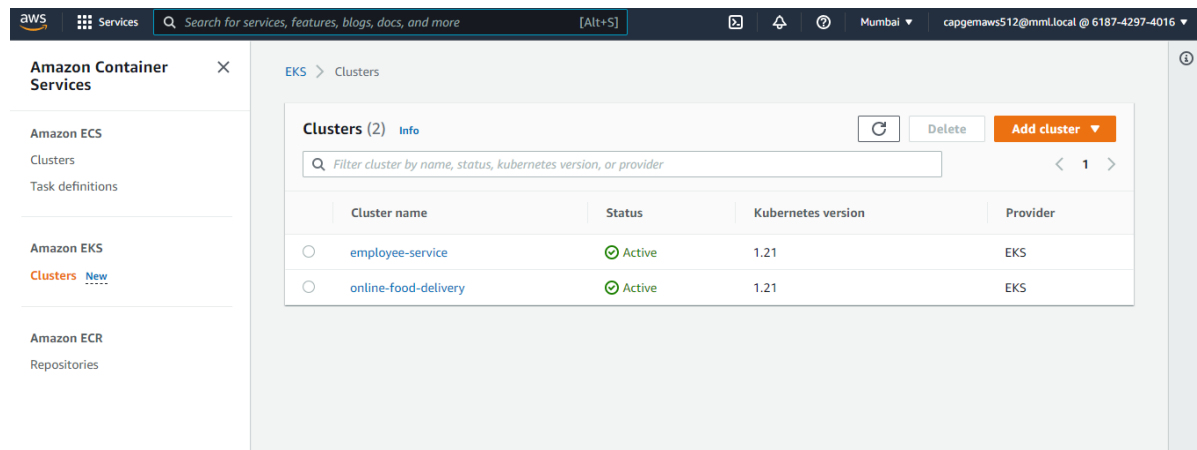
E:\d drive\All Workspace\AWS-Workspace\SPRINT-FoodDelivery3\K8S>kubectl port-forward svc/food-delivery 9090:8080
Forwarding from 127.0.0.1:9090 -> 8080
Forwarding from [::1]:9090 -> 8080
Handling connection for 9090
Handling connection for 9090
Handling connection for 9090
```



## 6. Deploy the application on EKS cluster:

a. Create a cluster in EKS with eksctl command

- **eksctl create cluster --name online-food-delivery --version 1.21 --region ap-south-1 --nodegroup-name online-food-delivery-node-group --node-type t2.micro --nodes 2**
- **aws eks --region ap-south-1 update-kubeconfig --name online-food-delivery**



```
C:\Windows\System32\cmd.exe
C:\Users\Monika S A\Desktop\SPRINT-FoodDelivery>eksctl create cluster --name online-food-delivery --version 1.21 --region ap-south-1 --nodegroup-name online-food-delivery-node-group --node-type t2.micro --nodes 2
[0] eksctl version 0.76.0
[0] using region ap-south-1
[0] setting availability zones to [ap-south-1c ap-south-1a ap-south-1b]
[0] subnets for ap-south-1c - public:192.168.0.0/19 private:192.168.0.0/19
[0] subnets for ap-south-1a - public:192.168.32.0/19 private:192.168.128.0/19
[0] subnets for ap-south-1b - public:192.168.64.0/19 private:192.168.160.0/19
[0] nodegroup "online-food-delivery-node-group" will use "" [AmazonLinux2/1.21]
[0] using Kubernetes version 1.21
[0] creating EKS cluster "online-food-delivery" in "ap-south-1" region with managed nodes
[0] will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
[0] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=ap-south-1 --cluster=online-food-delivery'
[0] CloudWatch logging will not be enabled for cluster "online-food-delivery" in "ap-south-1"
[0] you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=ap-south-1 --cluster=online-food-delivery'
[0] Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "online-food-delivery" in "ap-south-1"
[0]
2 sequential tasks: { create cluster control plane "online-food-delivery",
  2 sequential sub-tasks: {
    wait for control plane to become ready,
    create managed nodegroup "online-food-delivery-node-group",
  }
}
[0] building cluster stack "eksctl-online-food-delivery-cluster"
[0] deploying stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-cluster"
[0] building managed nodegroup stack "eksctl-online-food-delivery-nodegroup-online-food-delivery-node-group"
[0] deploying stack "eksctl-online-food-delivery-nodegroup-online-food-delivery-node-group"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-nodegroup-online-food-delivery-node-group"
[0] waiting for CloudFormation stack "eksctl-online-food-delivery-nodegroup-online-food-delivery-node-group"
```

b. Create the docker image and push on docker hub

c. Deploying the application on eks cluster using the following commands

Change the directory using the command **cd test** ( In folder test yaml files are created)

- **kubectl apply -f postgres-storage.yml**
- **kubectl apply -f postgres-secrets.yml**
- **kubectl apply -f postgres-deployment.yml**
- **kubectl apply -f postgres-service.yml**

```
C:\Windows\System32\cmd.exe
C:\Users\Monika S A\Desktop\SPRINT-FoodDelivery\test>kubectl create configmap hostname-config --from-literal=postgres_host=10.100.211.188
configmap/hostname-config created

C:\Users\Monika S A\Desktop\SPRINT-FoodDelivery\test>kubectl apply -f springboot-deployment.yml
deployment.apps/online-food-delivery created

C:\Users\Monika S A\Desktop\SPRINT-FoodDelivery\test>kubectl apply -f springboot-service.yml
service/online-food-delivery created

C:\Users\Monika S A\Desktop\SPRINT-FoodDelivery\test>kubectl get all
NAME                                READY   STATUS    RESTARTS   AGE
pod/online-food-delivery-7d669864b-n2kjs  1/1     Running   0           36s
pod/postgres-5bdb4fc5f9-69q9f          1/1     Running   0           2m32s

NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/kubernetes                 ClusterIP           10.100.0.1      <none>            443/TCP           25m
service/online-food-delivery       LoadBalancer        10.100.42.156   ac6102de69c5448ec9ec6654a61d7b23-980927595.ap-south-1.elb.amazonaws.com  8080:31529/TCP   15s
service/postgres                   NodePort            10.100.211.188 <none>            5432:30863/TCP   2m20s

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/online-food-delivery  1/1     1             1           37s
deployment.apps/postgres              1/1     1             1           2m33s

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/online-food-delivery-7d669864b  1         1         1       37s
replicaset.apps/postgres-5bdb4fc5f9            1         1         1       2m33s

C:\Users\Monika S A\Desktop\SPRINT-FoodDelivery\test>kubectl get all
NAME                                READY   STATUS    RESTARTS   AGE
pod/online-food-delivery-7d669864b-n2kjs  1/1     Running   0           2m32s
pod/postgres-5bdb4fc5f9-69q9f          1/1     Running   0           4m28s

NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/kubernetes                 ClusterIP           10.100.0.1      <none>            443/TCP           26m
service/online-food-delivery       LoadBalancer        10.100.42.156   ac6102de69c5448ec9ec6654a61d7b23-980927595.ap-south-1.elb.amazonaws.com  8080:31529/TCP   2m10s
service/postgres                   NodePort            10.100.211.188 <none>            5432:30863/TCP   4m15s

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/online-food-delivery  1/1     1             1           2m33s
deployment.apps/postgres              1/1     1             1           4m29s

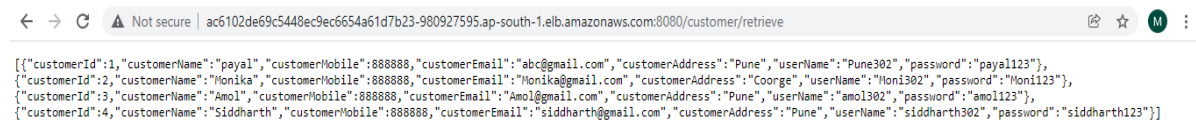
NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/online-food-delivery-7d669864b  1         1         1       2m33s
replicaset.apps/postgres-5bdb4fc5f9            1         1         1       4m29s
```

- **kubectl get all**
  - d. Set the config map
  - e. Get the Postgres Host IP Address:
- **kubectl get svc postgres -o jsonpath='{.spec.clusterIP}'**
  - f. get the IP Address and put in the below command

- **kubectl create configmap hostname-config --from-literal=postgres\_host=10.100.211.188**
- **kubectl apply -f springboot-deployment.yml**
- **kubectl apply -f springboot-service.yml**

## EKS Cluster Link:-

<http://ac6102de69c5448ec9ec6654a61d7b23-980927595.ap-south-1.elb.amazonaws.com:8080/customer/retrieve>

















← → ↻ ⚠ Not secure | ac6102de69c5448ec9ec6654a61d7b23-980927595.ap-south-1.elb.amazonaws.com:8080/customer/retrieve

```
[{"customerId":1,"customerName":"payal","customerMobile":888888,"customerEmail":"abc@gmail.com","customerAddress":"Pune","userName":"Pune302","password":"payal123"}, {"customerId":2,"customerName":"Monika","customerMobile":888888,"customerEmail":"Monika@gmail.com","customerAddress":"Coorge","userName":"Moni302","password":"Moni123"}, {"customerId":3,"customerName":"Amol","customerMobile":888888,"customerEmail":"Amol@gmail.com","customerAddress":"Pune","userName":"amol302","password":"amol123"}, {"customerId":4,"customerName":"Siddharth","customerMobile":888888,"customerEmail":"siddharth@gmail.com","customerAddress":"Pune","userName":"siddharth302","password":"siddharth123"}]
```







## Git hub link:

1. <https://github.com/aMOL156/Food-Delivery.git>
2. [https://github.com/MonikaProductOwner/Food-Delivery\\_Project.git](https://github.com/MonikaProductOwner/Food-Delivery_Project.git)
3. <https://github.com/siddharth7575/Online-food-delivery.git>
4. <https://github.com/pawalepayal/Online-Food-Delivery.git>

 <b>aMOL156</b> Add files via upload	989d5bb 15 minutes ago	 2 commits
 .classpath	Add files via upload	15 minutes ago
 .gitignore	Add files via upload	15 minutes ago
 .project	Add files via upload	15 minutes ago
 Dockerfile	Add files via upload	15 minutes ago
 HELP.md	Add files via upload	15 minutes ago
 OrderReadMe	Add files via upload	15 minutes ago
 README.md	Create README.md	20 minutes ago
 ReadMeCart	Add files via upload	15 minutes ago
 ReadMeCustomer	Add files via upload	15 minutes ago
 ReadMeFood	Add files via upload	15 minutes ago
 amol	Add files via upload	15 minutes ago
 amol.pub	Add files via upload	15 minutes ago

No description, website, or topics provided

-  Readme
-  0 stars
-  1 watching
-  0 forks

### Releases

No releases published  
[Create a new release](#)

### Packages

No packages published  
[Publish your first package](#)