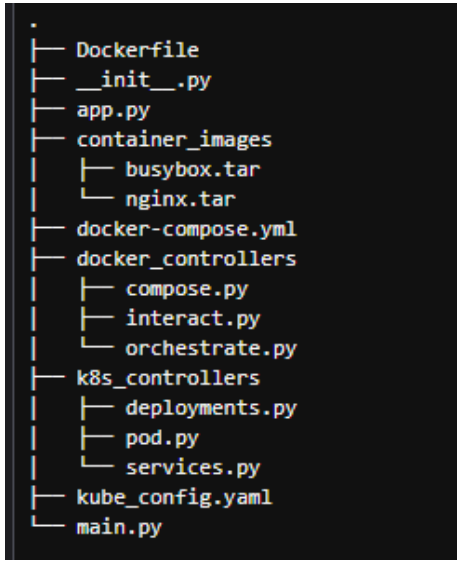
Container Service Manager (CSM) APP User Manual

Introduction

Description

The CSM command-line app facilitates the management of containerized services/apps. Upon successful app Launch, users can interact with either docker or Kubernetes through the menu system.

Below is the snapshot of the Apps file structure:



Dependencies

- python3: v3.6 and above
- Kubernetes python client: v28.1.0

- Docker python client: v7.0.0
- Docker deamon: v24.0.7
- Pandas: v2.1.4
- Python_on_whales: v0.68.0

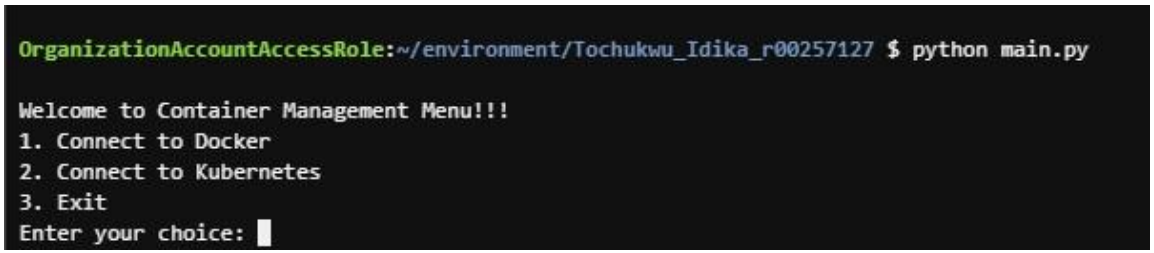
Note: Development of this was done in **AWS cloud9 utilizing ubuntu 22.04 OS**.

ASM Application Usage

How to Launch the APP

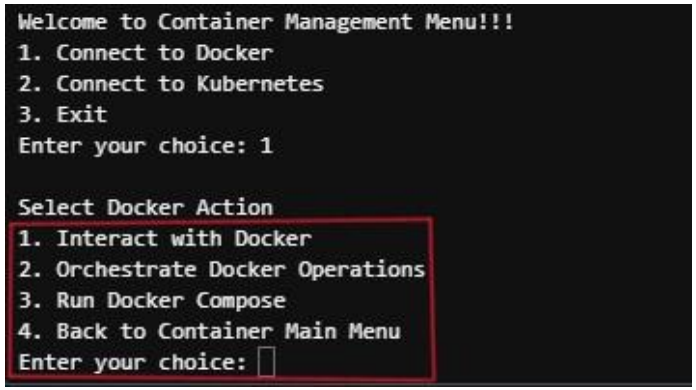
To Launch the APP, follow these steps:

1. Access the terminal and navigate to “Tochukwu_Idika_r00257127” directory. Execute **“python main.py”** to launch the application.
2. Upon successful launch of the app, you are provided access to the Container Service Menu which allows interaction and management of containers as shown in the below image.



Docker

Upon selecting to connect to **docker**, you are provided with the below **sub menus**.



Interact with Docker Menu

This section outlines the methods accessible within the **Interact with docker** menu, coupled with the anticipated result that demonstrate expected behavior upon successful execution of each method.

- **List all Containers:** It lists all the running containers in docker.

```
Select an 'Interact with Docker' Action
1. List All Container
2. List all Stopped Containers
3. Run Containers
4. View Port Mappings
5. Stop and Remove all Containers
6. Save Image to File
7. Back to 'Docker Actions' Menu

Enter your choice: 1
All Containers:
Container ID: cd0644131651a1e09d3e4cdbb0269a48b2d826c2878bebbafe1120a190599734, Name: admiring_lamport
```

- **List stopped containers:** It lists all the containers that have exited or stopped.

```
Select an 'Interact with Docker' Action
1. List All Container
2. List all Stopped Containers
3. Run Containers
4. View Port Mappings
5. Stop and Remove all Containers
6. Save Image to File
7. Back to 'Docker Actions' Menu

Enter your choice: 2
Stopped/Exited Containers:
Container ID: d1c5cf0093385847428e77760db2adf2509a4604afef56871300880e3d104dbc, Name: hungry_leakey
Container ID: 53c999549b822e7cc00a38a106e74443a5ee5305b51c8dc5fbfef6633d12f766, Name: admiring_ptolemy
Container ID: fec2d33d9602826178c22dddd08139e8c276c07db631b0bbbf61ef15ad7f3f95, Name: goofy_goodall
Container ID: 23cbbb39b73bbb895d7c924685759a188acbb07bb3cbcc2c64311946a176071d, Name: tochi_busybox
```

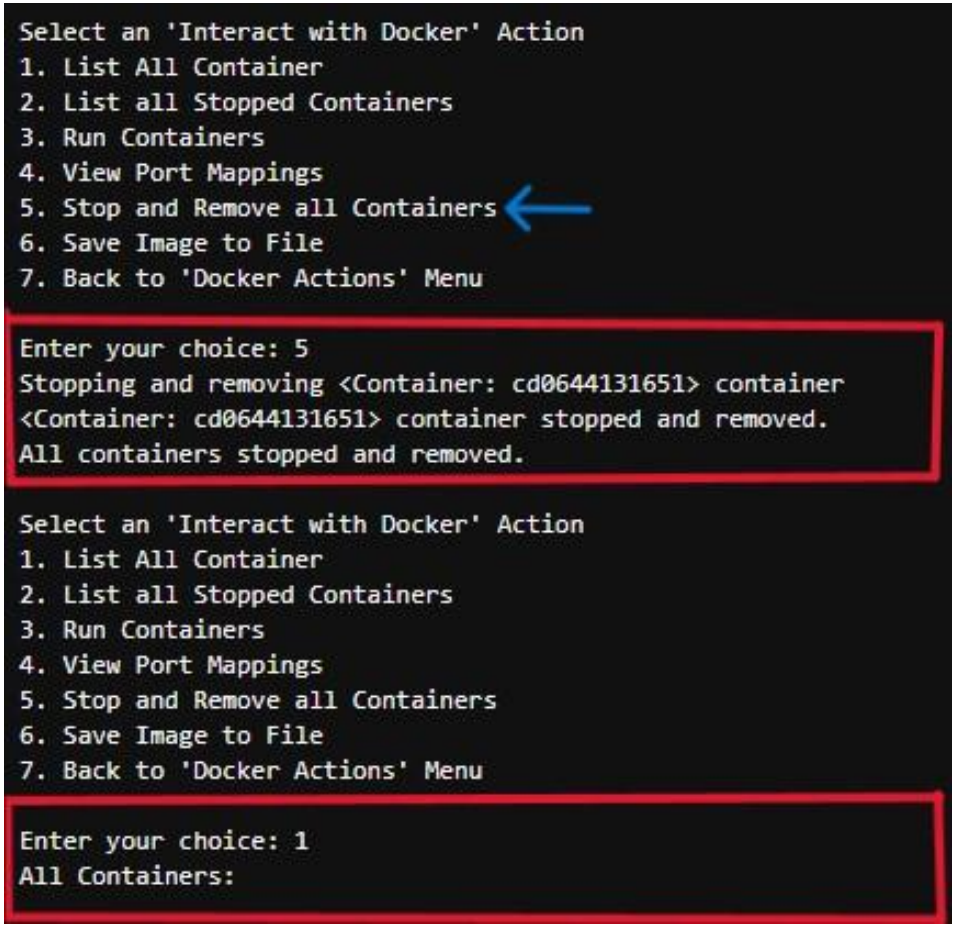
- **Run a Container:** It runs a container based on the image specified by the user.

```
Select an 'Interact with Docker' Action
1. List All Container
2. List all Stopped Containers
3. Run Containers
4. View Port Mappings
5. Stop and Remove all Containers
6. Save Image to File
7. Back to 'Docker Actions' Menu

Enter your choice: 3

Enter the image or type 'exit' to leave: nginx
Enter a name for the container (press Enter for a random name):
Container admiring_lamport is running.
```

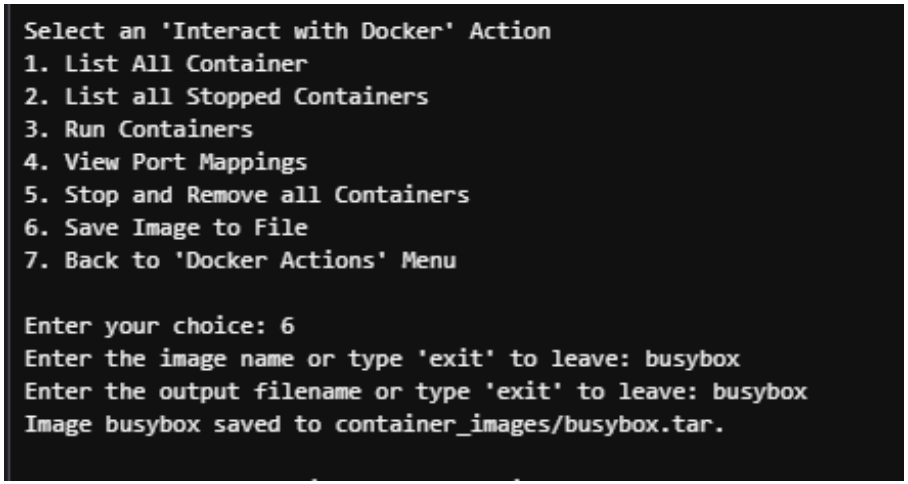
- **View Port Mappings:** It shows the port mappings of a running container.

```
Select an 'Interact with Docker' Action
1. List All Container
2. List all Stopped Containers
3. Run Containers
4. View Port Mappings
5. Stop and Remove all Containers
6. Save Image to File
7. Back to 'Docker Actions' Menu

Enter your choice: 4

Select a container ID:
1. 9e1a5badc80a1b5073c43cb656f14b1152050e8125426e8cfe2b29b9bc6c75e7

Enter your number choice or type 'exit' to leave: 1

Port mappings for Container 9e1a5badc80a1b5073c43cb656f14b1152050e8125426e8cfe2b29b9bc6c75e7:
8000/tcp -> [{'HostIp': '0.0.0.0', 'HostPort': '8005'}, {'HostIp': '::', 'HostPort': '8005'}]
```

- **Stop and remove all containers:** It stops and removes both running and stopped containers.



- **Save Image to file:** It saves an image specified by the user to **'container_images'** folder.



Orchestrate Docker Operation Menu

This menu demonstrates **how to build and run a container image**. It expects an **app.py file** at the root of the folder that will be used to build the image.

```
Select an 'Orchestration' Action
1. Build and Run a Container Image
2. Back to 'Docker Actions' Menu

Enter your choice: 1
Type 'exit' to leave
Enter the Python version (from 3.6 and above): 3.11
Enter the path to the Python program: app.py
Enter the desired image name: MountPointe

Creating dockerfile ...
Dockerfile created successfully.
Building the mountpointe image
Image 'mountpointe' created successfully.
here is the created image (<Image: 'mountpointe:latest'>, <itertools._tee object at 0x7f9817d69f80>)
Container 'dreamy_bassi' is running.
Container paused.
Container unpaused.
```

Below is the deployed web app running

Hi and welcome to MountPointe

Your go-to place for everything machine learning operations.

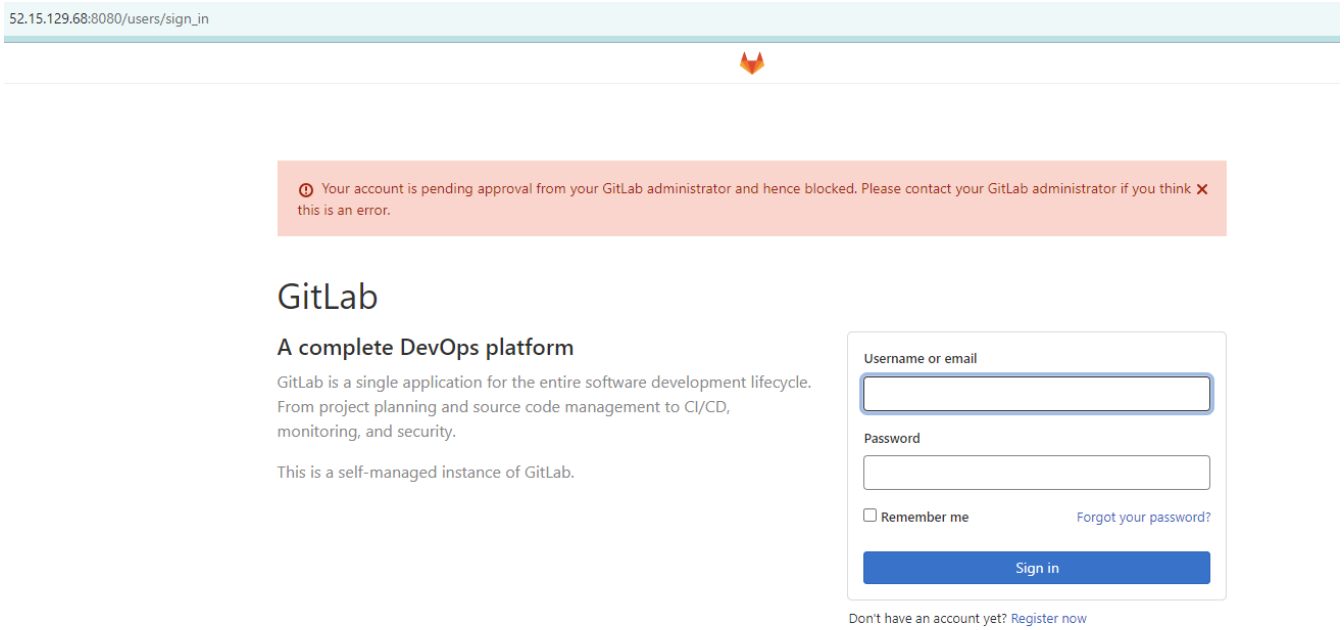
Run Docker Compose Menu

This section outlines the methods accessible within the 'Run docker compose menu', coupled with the anticipated result that demonstrate expected behavior upon successful execution of each method.

- **Start Docker Compose:** It runs the the docker-compose file. The docker compose file **Must** be located at the **root folder**. The docker file used for demonstration was retrieved from this site.

```
Select a 'Docker Compose' Action
1. Start Docker Compose Containers
2. Stop Docker Compose Containers
3. Back to 'Docker Actions' Menu

Enter your choice: 1
Enter docker compose filename: docker-compose.yml
starting up docker compose containers ...
[+] Running 3/0
 ✓ Container tochukwu_idika_r00257127-postgresql-1  Running
 ✓ Container tochukwu_idika_r00257127-redis-1       Running
 ✓ Container tochukwu_idika_r00257127-gitlab-1      Running
Docker Compose started successfully.
```

Note: The installed application is **Gitlab**. I was able **to register successfully as a new user**. However, when I attempted to log in, I encountered an error as shown in the image below. However, I am convinced that the front end connected successfully with the backend otherwise, I would not have been able to register successfully as a new user.

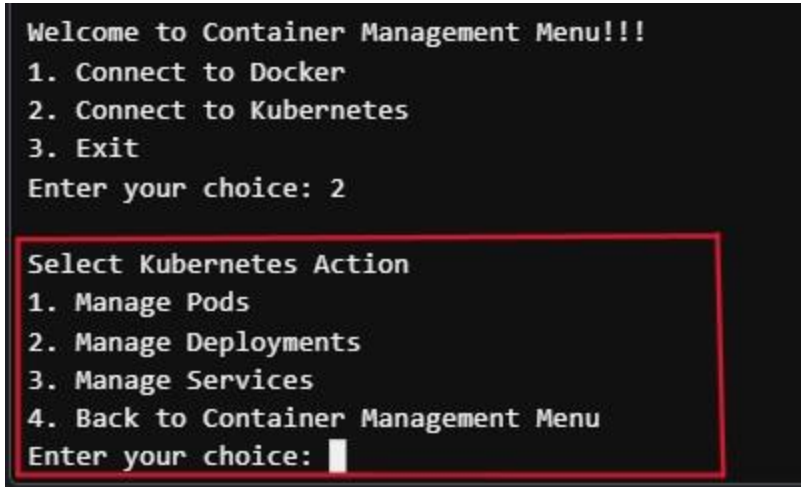


- **Stop Docker Compose: It tears down the application.**

```
Select a 'Docker Compose' Action
1. Start Docker Compose Containers
2. Stop Docker Compose Containers
3. Back to 'Docker Actions' Menu

Enter your choice: 2
Enter docker compose filename: docker-compose.yml
stopping docker compose containers ...
[+] Running 4/4
 ✓ Container tochukwu_idika_r00257127-gitlab-1      Removed
 ✓ Container tochukwu_idika_r00257127-postgresql-1  Removed
 ✓ Container tochukwu_idika_r00257127-redis-1       Removed
 ✓ Network tochukwu_idika_r00257127_default         Removed
```

Kubernetes

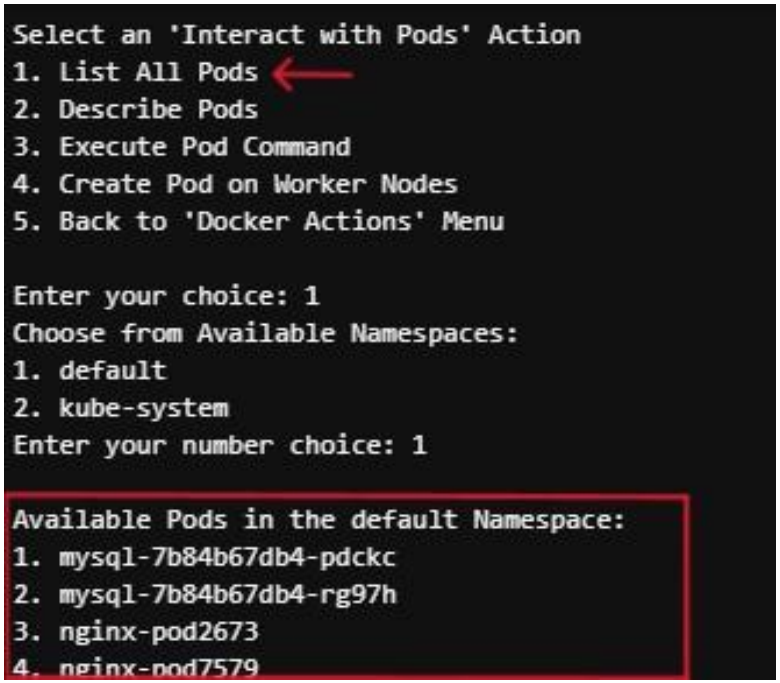
Upon selecting to connect to **kubernetes**, you are provided with the below **sub menus**.



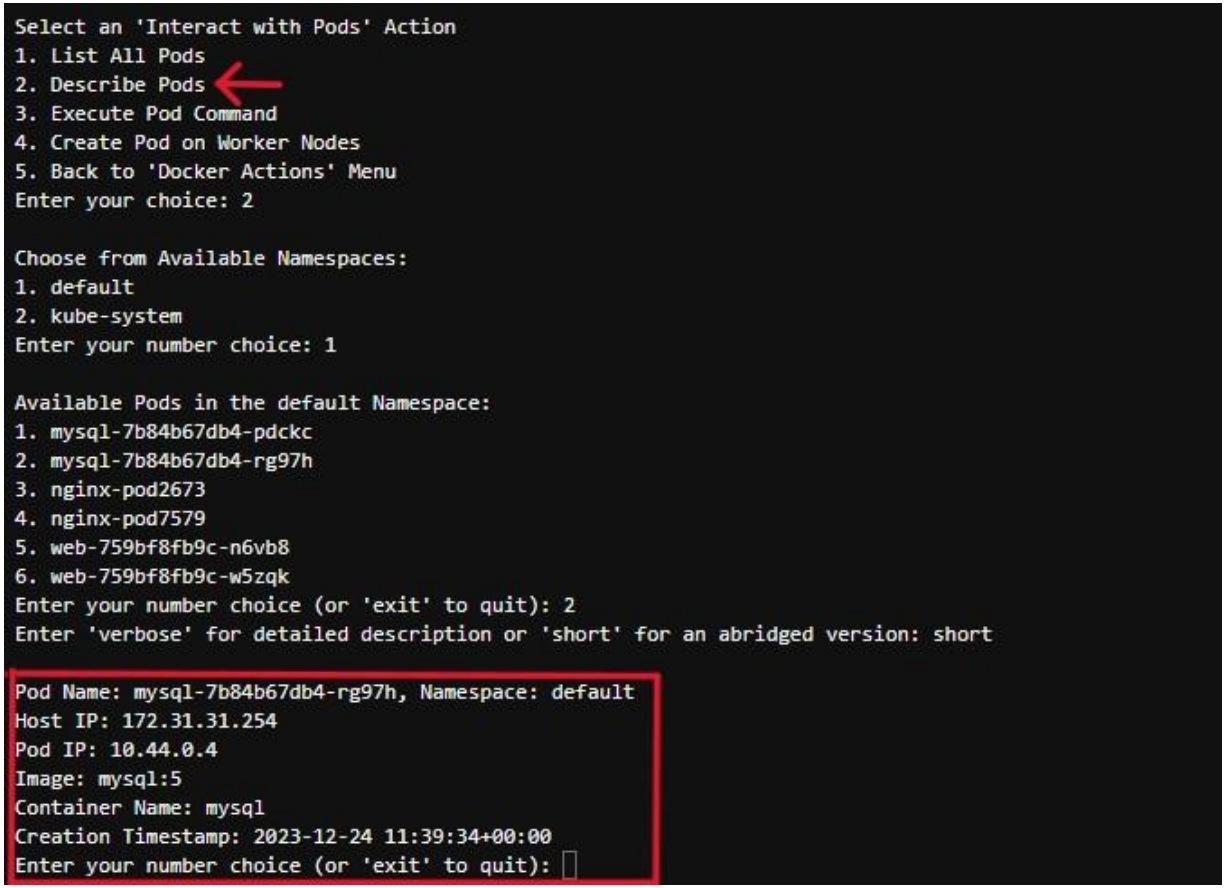
Manage Pods Menu

This section outlines the methods accessible within the pod menu, coupled with the anticipated result that demonstrate expected behavior upon successful execution of each method.

- **List Pods Command:** Lists all the Pods within the selected namespace.



- **Describe Pods:** It provides information regarding the selected Pod. You can choose **a verbose or short description**.



- **Execute Pod Command:** It executes a shell command on a running pod.

```
Available Pods in the default Namespace:
1. mysql-7b84b67db4-pdckc
2. mysql-7b84b67db4-rg97h
3. nginx-pod2673
4. nginx-pod7579
5. web-759bf8fb9c-n6vb8
6. web-759bf8fb9c-w5zqk

Enter your number choice (or 'exit' to quit): 3

Enter the command to run (or 'exit' to quit): echo Tochukwu Idika
WHILEEEE ::: True
WHILEEEE ::: True
Executing command sucessfully.


Enter the command to run (or 'exit' to quit):
```

- **Create Pod on Worker Nodes:** It creates a pod with the same image on all the worker nodes.

```
Select Kubernetes Action
1. Manage Pods
2. Manage Deployments
3. Manage Services
4. Back to Container Management Menu
Enter your choice: 1

Select an 'Interact with Pods' Action
1. List All Pods
2. Describe Pods
3. Execute Pod Command
4. Create Pod on Worker Nodes
5. Back to 'Docker Actions' Menu
Enter your choice: 4
Enter the container image: nginx
Enter the pod name (default is nginx-pod):


Choose from Available Namespaces:
1. default
2. kube-system

Enter your number choice: 1
Pod 'nginx-pod6937' created successfully on node 'ip-172-31-24-166' in 'default' namespace
Pod 'nginx-pod6264' created successfully on node 'ip-172-31-31-254' in 'default' namespace
```

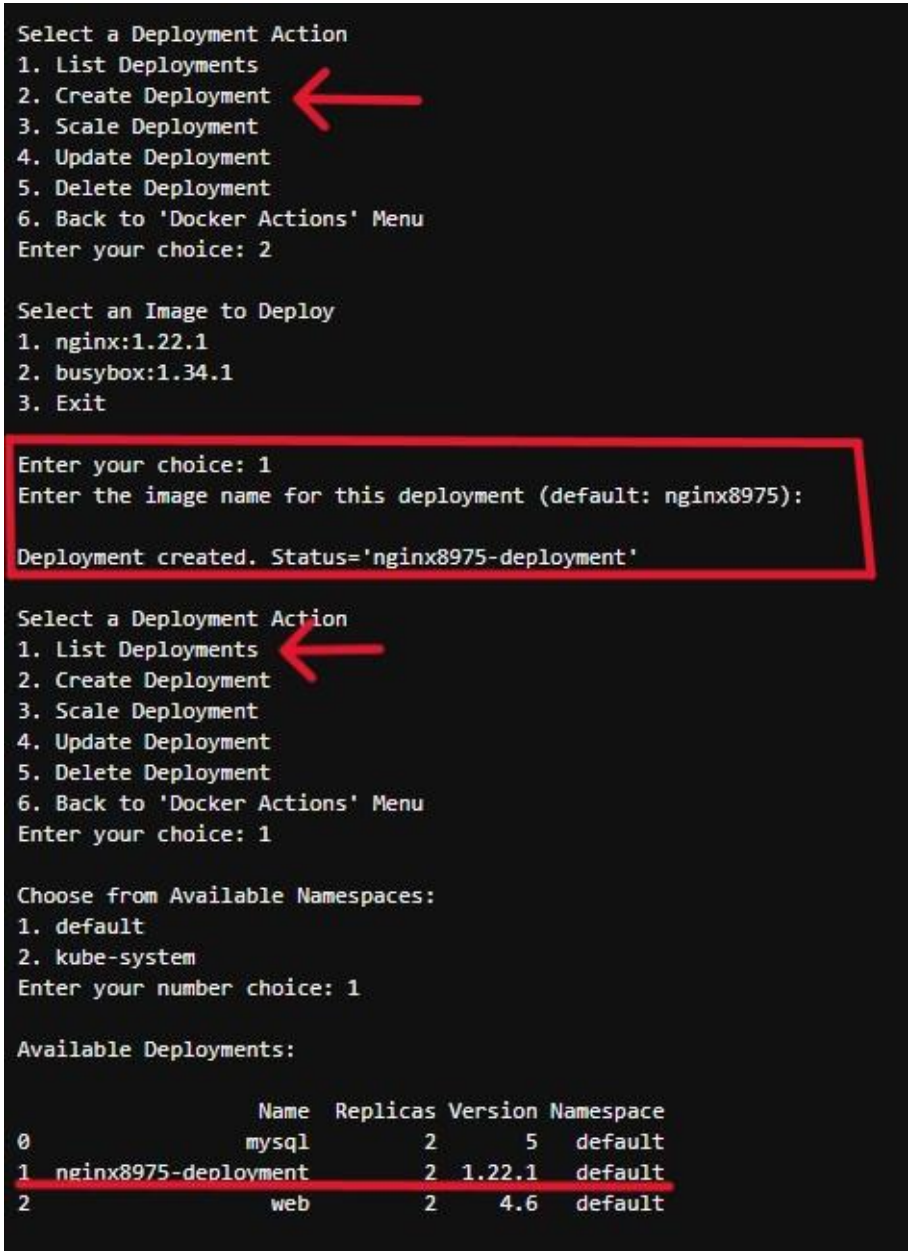
Manage Deployments Menu

This section outlines the methods accessible within the Deployment menu, coupled with the anticipated result that demonstrate expected behavior upon successful execution of each method.

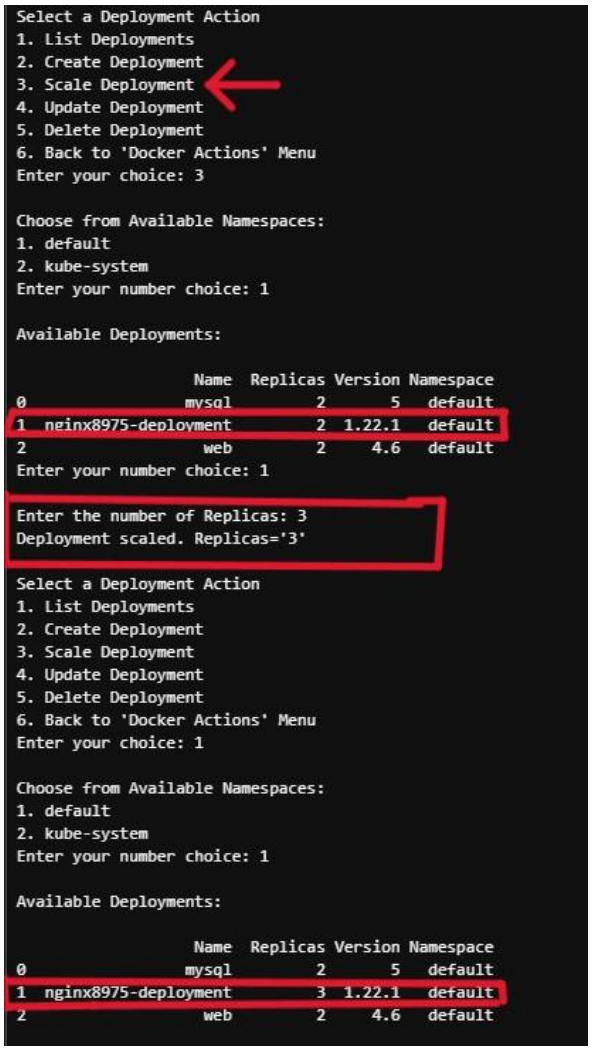
- **List Deployments:** It lists all the deployments within the namespace selected by the user.

```
Select Kubernetes Action
1. Manage Pods
2. Manage Deployments
3. Manage Services
4. Back to Container Management Menu
Enter your choice: 2

Select a Deployment Action
1. List Deployments
2. Create Deployment
3. Scale Deployment
4. Update Deployment
5. Delete Deployment
6. Back to 'Docker Actions' Menu
Enter your choice: 1

Choose from Available Namespaces:
1. default
2. kube-system
Enter your number choice: 1

Available Deployments:

    Name  Replicas Version Namespace
0  mysql         2       5   default
1    web         2     4.6   default
```

- **Create deployment:** It creates a Kubernetes deployment based on the image selected by the user.

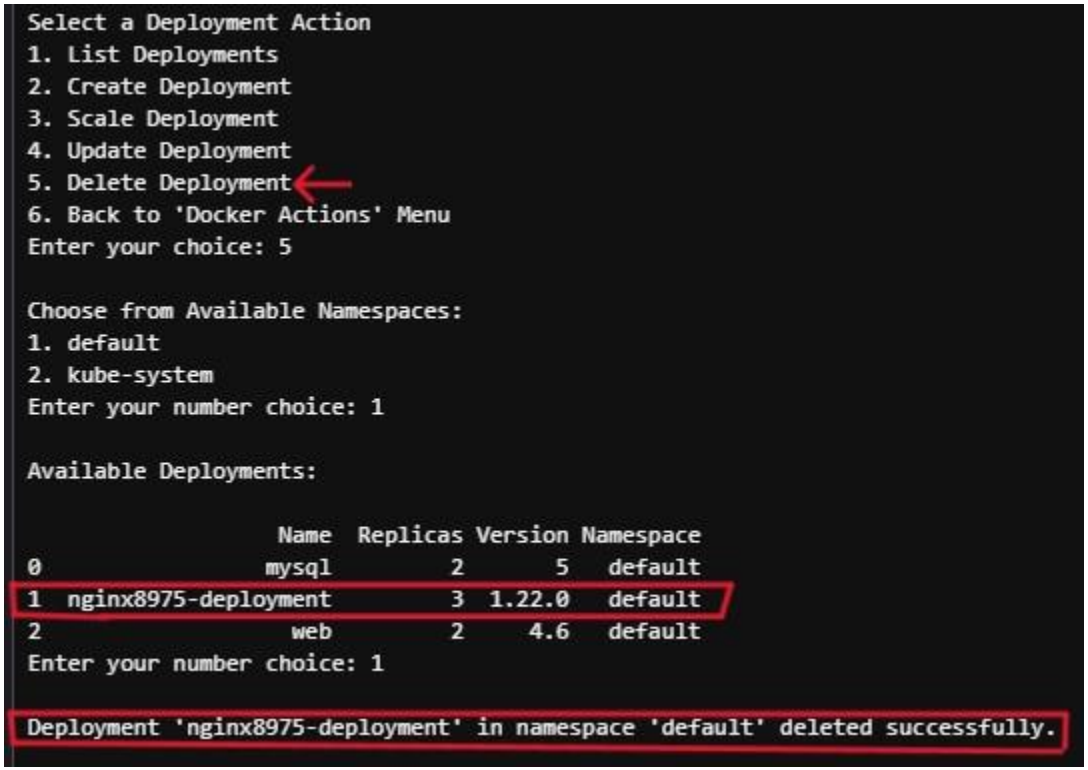

Select a Deployment Action
1. List Deployments
2. Create Deployment
3. Scale Deployment
4. Update Deployment
5. Delete Deployment
6. Back to 'Docker Actions' Menu
Enter your choice: 2
Select an Image to Deploy
1. nginx:1.22.1
2. busybox:1.34.1
3. Exit
Enter your choice: 1
Enter the image name for this deployment (default: nginx8975):
Deployment created. Status='nginx8975-deployment'
Select a Deployment Action
1. List Deployments
2. Create Deployment
3. Scale Deployment
4. Update Deployment
5. Delete Deployment
6. Back to 'Docker Actions' Menu
Enter your choice: 1
Choose from Available Namespaces:
1. default
2. kube-system
Enter your number choice: 1
Available Deployments:
Name Replicas Version Namespace
0 mysql 2 5 default
1 nginx8975-deployment 2 1.22.1 default
2 web 2 4.6 default

- **Scale Deployment:** It scales up or down an existing deployment on the cluster.



- **Update Deployment:** It updates an existing deployment within the cluster.

- **Deletes Deployment:** It deletes a deployment.



Manage Services Menu

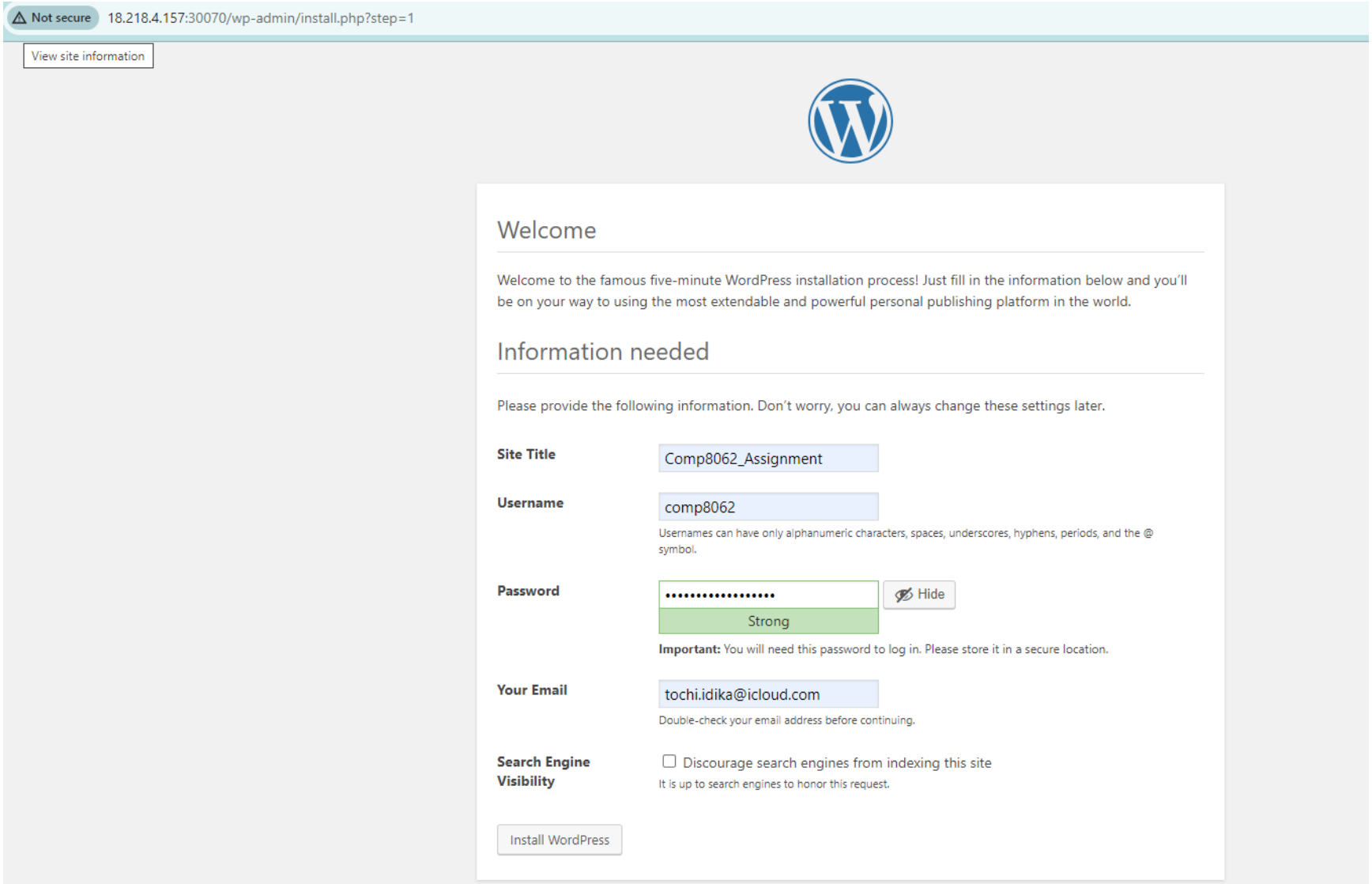
This menu focuses on deploying a WordPress Application as a Kubernetes service. The WordPress application has two microservices: the web app (acting as the front end) and the Mysql (acting as the backend).

I got the inspiration on how to deploy the WordPress app as a Kubernetes microservice by going through this website.

- To confirm the service deployment is successful, run the command **"kubectl get services"**

```
OrganizationAccountAccessRole:~/environment/Tochukwu_Idika_r00257127 $ kubectl get deployments
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
mysql   2/2     2            2           2m22s
web     2/2     2            2           2m22s
OrganizationAccountAccessRole:~/environment/Tochukwu_Idika_r00257127 $ kubectl get services
NAME         TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP      10.96.0.1        <none>        443/TCP        3d1h
mysql        ClusterIP      10.96.43.173     <none>        3306/TCP       2m26s
web          LoadBalancer   10.109.250.236   <pending>     80:30070/TCP   2m26s
OrganizationAccountAccessRole:~/environment/Tochukwu_Idika_r00257127 $
```

- To confirm the application is accessible through the **loadbalancer service port,** get on the browser using <**WorkerNodeIPaddress**>:30070 as shown in the **below image**.



- **However, it did not connect properly with the database as shown in the image below.**

18.218.4.157:30070/wp-admin/install.php?step=2

Can't select database

We were able to connect to the database server (which means your username and password is okay) but not able to select the `wordpress` database.

- Are you sure it exists?
- Does the user `root` have permission to use the `wordpress` database?
- On some systems the name of your database is prefixed with your username, so it would be like `username_wordpress`. Could that be the problem?

If you don't know how to set up a database you should **contact your host**. If all else fails you may find help at the WordPress Support Forums.