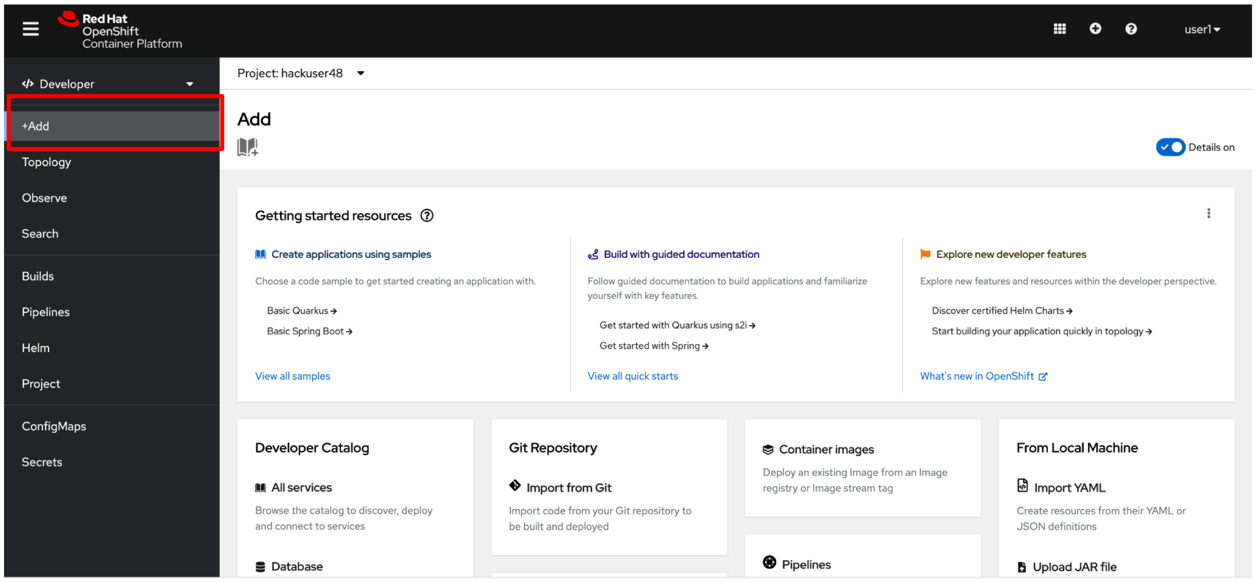
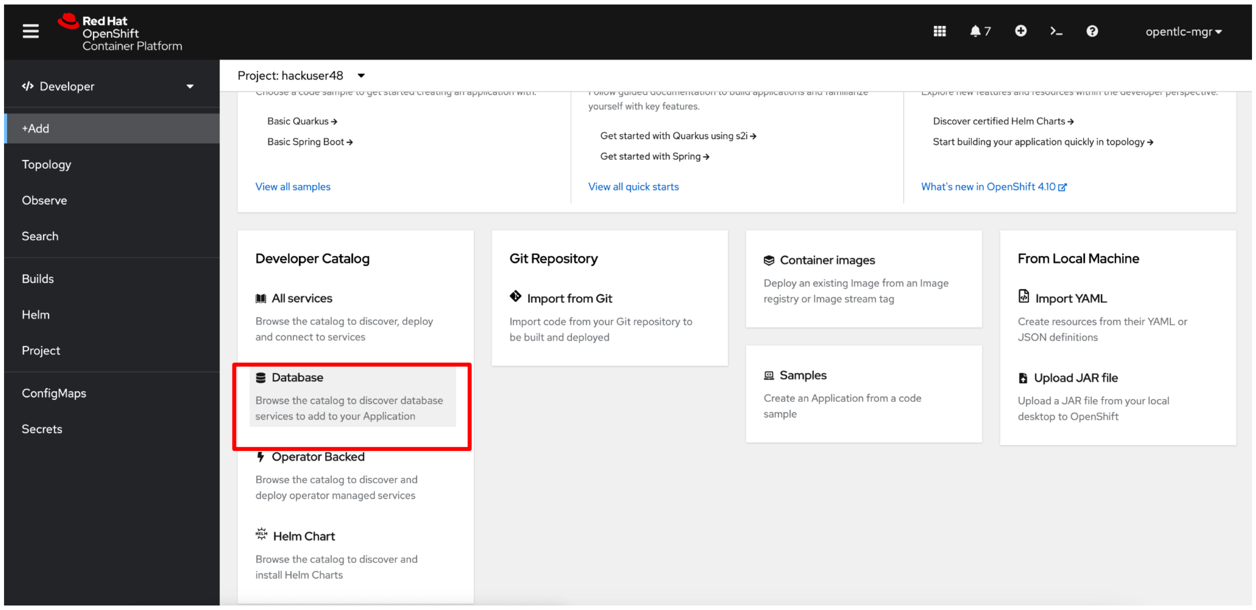
Deploying Database on OpenShift

Now before you start working on code development, first deploy the database which will be required for your backend application to connect to it. Either you can deploy a MySQL or PostgreSQL or MongoDB database using the below screenshots.

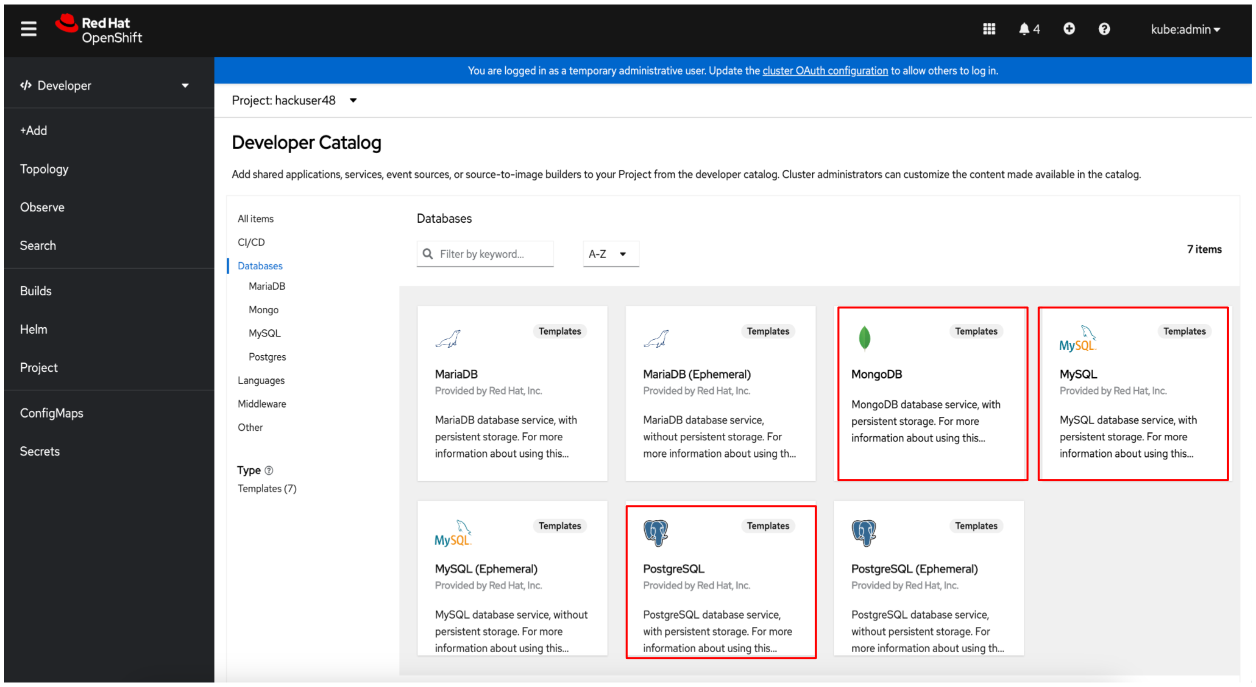
Click on the "**+ Add"** button in Openshift Web console at Developer perspective


Red Hat OpenShift Container Platform
user1
Developer
+Add
Topology
Observe
Search
Builds
Pipelines
Helm
Project
ConfigMaps
Secrets
Project: hackuser48
Add
Details on
Getting started resources
Create applications using samples
Choose a code sample to get started creating an application with.
Basic Quarkus
Basic Spring Boot
View all samples
Build with guided documentation
Follow guided documentation to build applications and familiarize yourself with key features.
Get started with Quarkus using s2i
Get started with Spring
View all quick starts
Explore new developer features
Explore new features and resources within the developer perspective.
Discover certified Helm Charts
Start building your application quickly in topology
What's new in OpenShift
Developer Catalog
All services
Browse the catalog to discover, deploy and connect to services
Database
Git Repository
Import from Git
Import code from your Git repository to be built and deployed
Container images
Deploy an existing Image from an Image registry or Image stream tag
Pipelines
From Local Machine
Import YAML
Create resources from their YAML or JSON definitions
Upload JAR file

Then on the form that opens, scroll the screen down & select "Database" under the "Developer Catalog" section.

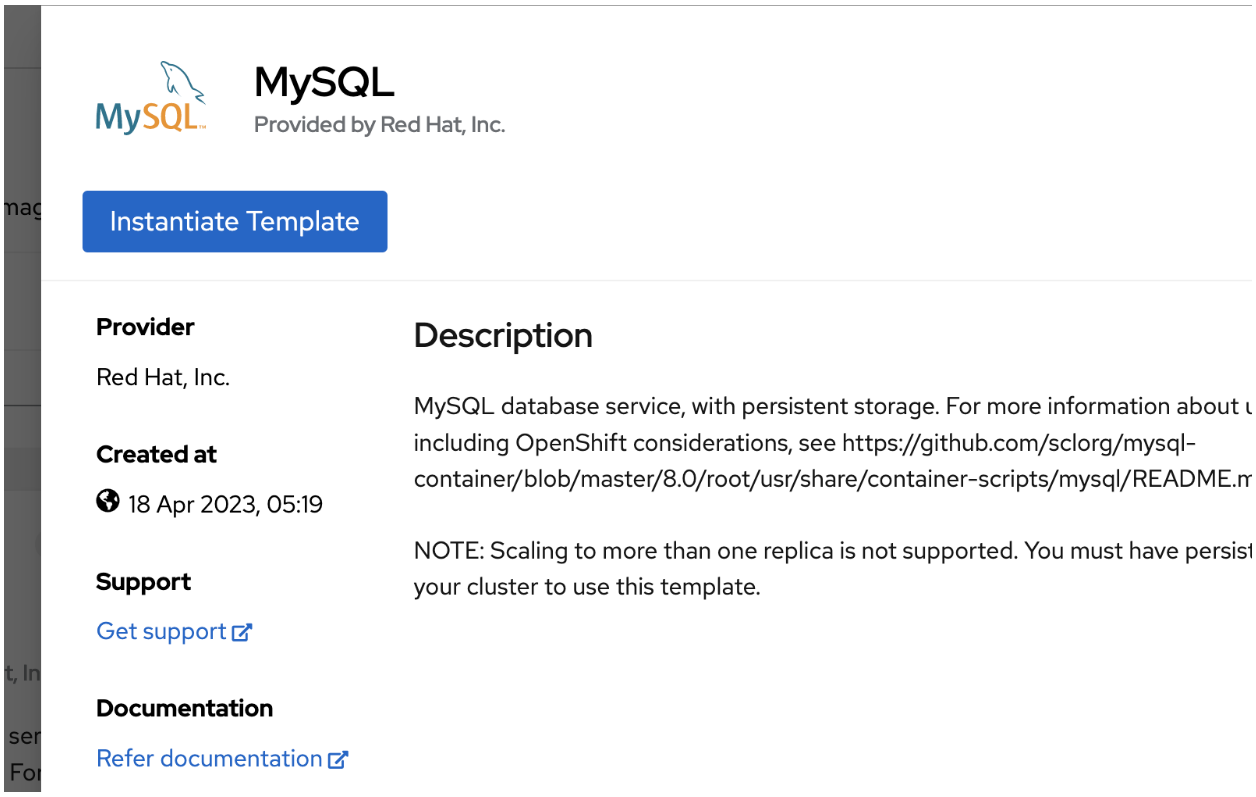

Red Hat OpenShift Container Platform
7
opentlc-mgr
Developer
+Add
Topology
Observe
Search
Builds
Helm
Project
ConfigMaps
Secrets
Project: hackuser48
Basic Quarkus
Basic Spring Boot
View all samples
yourself with key features.
Get started with Quarkus using s2i
Get started with Spring
View all quick starts
Discover certified Helm Charts
Start building your application quickly in topology
What's new in OpenShift 4.10
Developer Catalog
All services
Browse the catalog to discover, deploy and connect to services
Database
Browse the catalog to discover database services to add to your Application
Operator Backed
Browse the catalog to discover and deploy operator managed services
Helm Chart
Browse the catalog to discover and install Helm Charts
Git Repository
Import from Git
Import code from your Git repository to be built and deployed
Container images
Deploy an existing Image from an Image registry or Image stream tag
Samples
Create an Application from a code sample
From Local Machine
Import YAML
Create resources from their YAML or JSON definitions
Upload JAR file
Upload a JAR file from your local desktop to OpenShift

Select either “MySQL” or PostgreSQL or MongoDB database as per your requirements & it will display the next screen for Instantiate the template . Do not select the Ephemeral template as this will not provide data persistence.


Red Hat OpenShift
4
kube:admin
You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.
Developer
+Add
Topology
Observe
Search
Builds
Helm
Project
ConfigMaps
Secrets
Project: hackuser48
Developer Catalog
Add shared applications, services, event sources, or source-to-image builders to your Project from the developer catalog. Cluster administrators can customize the content made available in the catalog.
All items
CI/CD
Databases
MariaDB
Mongo
MySQL
Postgres
Languages
Middleware
Other
Type
Templates (7)
Databases
Filter by keyword...
A-Z
7 items
Templates
MariaDB
Provided by Red Hat, Inc.
MariaDB database service, with persistent storage. For more information about using this...
MariaDB (Ephemeral)
Provided by Red Hat, Inc.
MariaDB database service, without persistent storage. For more information about using th...
MongoDB
MongoDB database service, with persistent storage. For more information about using this...
MySQL
Provided by Red Hat, Inc.
MySQL database service, with persistent storage. For more information about using this...
MySQL (Ephemeral)
Provided by Red Hat, Inc.
MySQL database service, without persistent storage. For more information about using this...
PostgreSQL
Provided by Red Hat, Inc.
PostgreSQL database service, with persistent storage. For more information about using this...
PostgreSQL (Ephemeral)
Provided by Red Hat, Inc.
PostgreSQL database service, without persistent storage. For more information about using th...

Depending on the selection click on Instantiate template for MySQL or PostgreSQL or MongoDB database.

For MySQL (Other databases – Go to Next section)


MySQL

Provided by Red Hat, Inc.

Instantiate Template

Provider

Red Hat, Inc.

Created at

18 Apr 2023, 05:19

Support

Get support

Documentation

Refer documentation

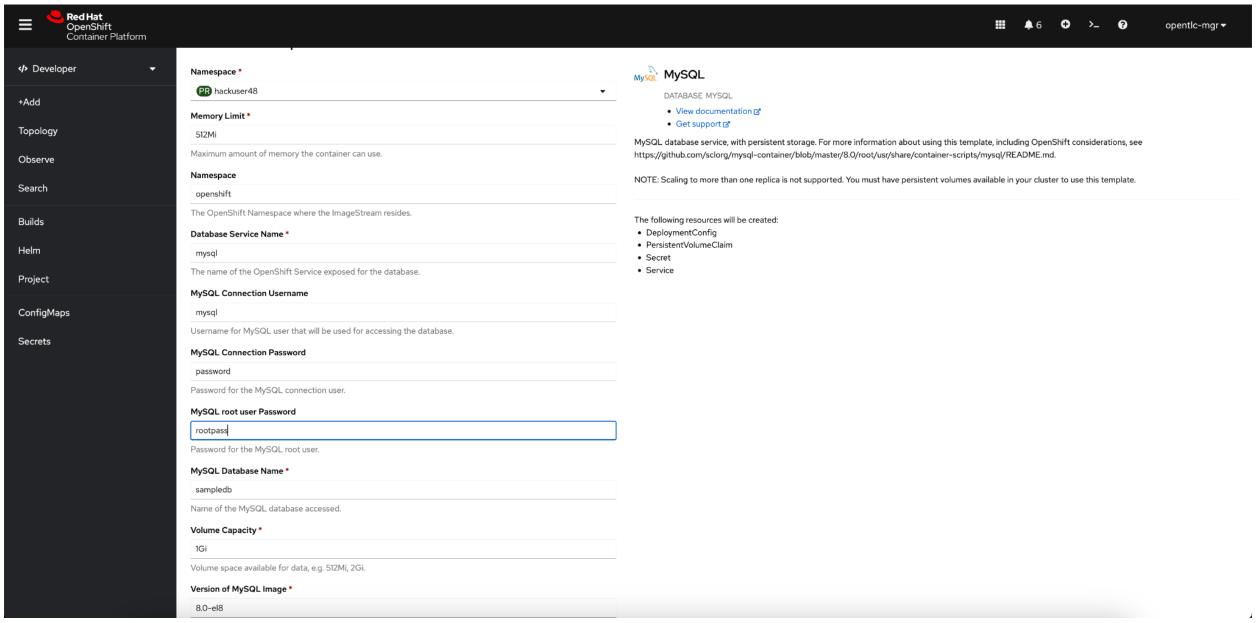
Description

MySQL database service, with persistent storage. For more information about u
including OpenShift considerations, see https://github.com/sclorg/mysql-
container/blob/master/8.0/root/usr/share/container-scripts/mysql/README.m

NOTE: Scaling to more than one replica is not supported. You must have persist
your cluster to use this template.

On the form that opens, enter the following configurations. Use the below provided values & leave other configurations as default. The below example is for MySQL database

- **Database Service Name : mysql** (This will be the DB hostname which will be used later for your backend code to connect to the database. Port will be 3306 for MySQL. You can hard code in your backend code)
- **MySQL connection Username: mysql** (DB connection username for your backend code configuration. You can hard code in your backend code)
- **MySQL Connection Password: password** (DB connection password for your backend code configuration. You can hard code in your backend code)
- **MySQL root user password : rootpass**
- **MySQL Database name: sampledb** (This database will be created for you which you can configure in your backend code configuration. You can hard code in your backend code)



Namespace *
hackuser48

Memory Limit *
512Mi
Maximum amount of memory the container can use.

Namespace
openshift
The OpenShift Namespace where the ImageStream resides.

Database Service Name *
mysql
The name of the OpenShift Service exposed for the database.

MySQL Connection Username
mysql
Username for MySQL user that will be used for accessing the database.

MySQL Connection Password
password
Password for the MySQL connection user.

MySQL root user Password
rootpass
Password for the MySQL root user.

MySQL Database Name *
sampledb
Name of the MySQL database accessed.

Volume Capacity *
1Gi
Volume space available for data, e.g. 512Mi, 2Gi.

Version of MySQL Image *
8.0-el8

MySQL

DATABASE MYSQL

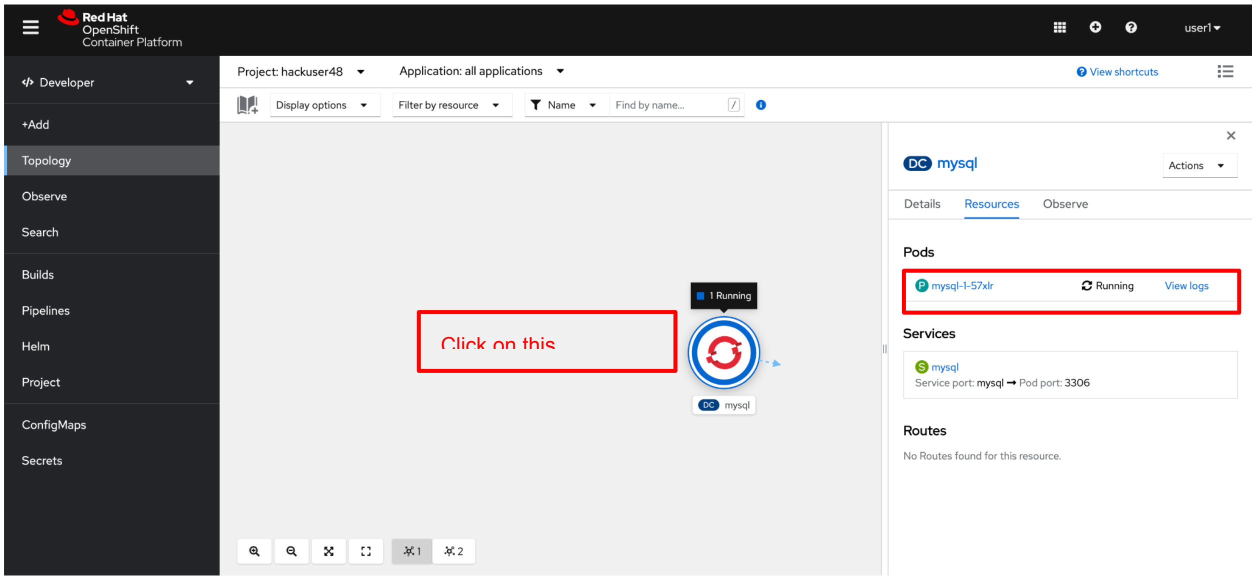
- View documentation
- Get support

MySQL database service, with persistent storage. For more information about using this template, including OpenShift considerations, see https://github.com/sclorg/mysql-container/blob/master/8.0/root/usr/share/container-scripts/mysql/README.md.

NOTE: Scaling to more than one replica is not supported. You must have persistent volumes available in your cluster to use this template.

The following resources will be created:
- DeploymentConfig
- PersistentVolumeClaim
- Secret
- Service

Scroll down the form and click on the "Create" button below. This will display the deployment screen & click on deployment as shown below. MySQL pod will be in running status in a while.



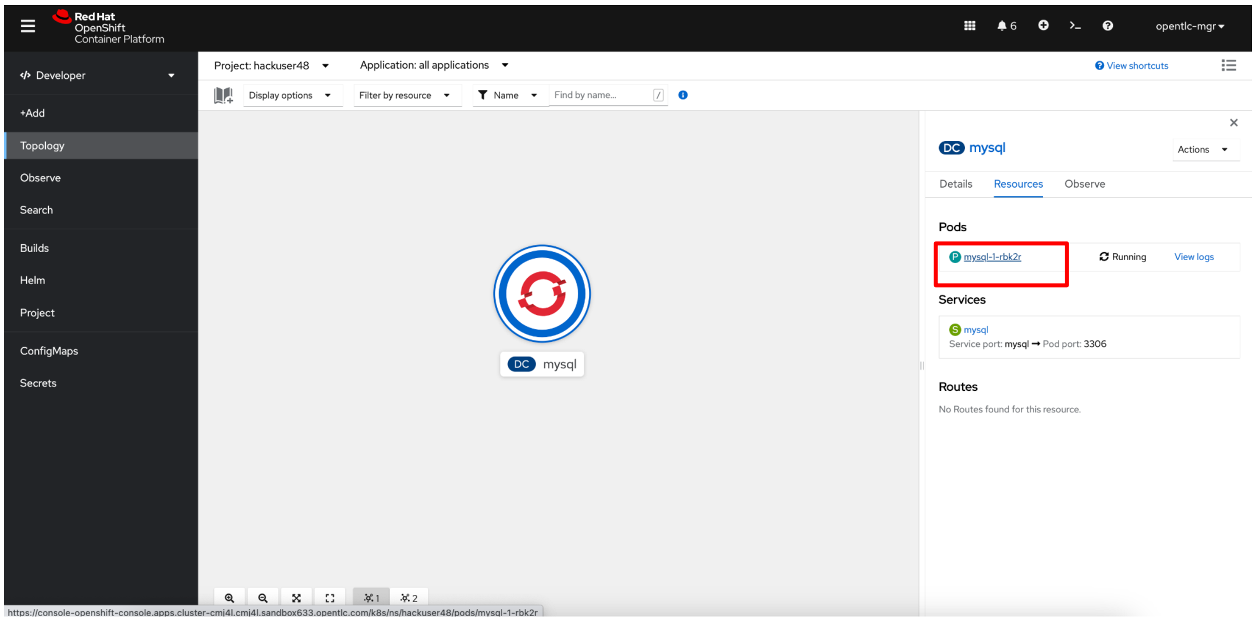
DC mysql

Pods
P mysql-1-57xlr Running View logs

Services
S mysql
Service port: mysql → Pod port: 3306

Routes
No Routes found for this resource.

If you need to execute SQL commands, you need to login to the terminal of the MySQL pod. Click on the pod as shown below.

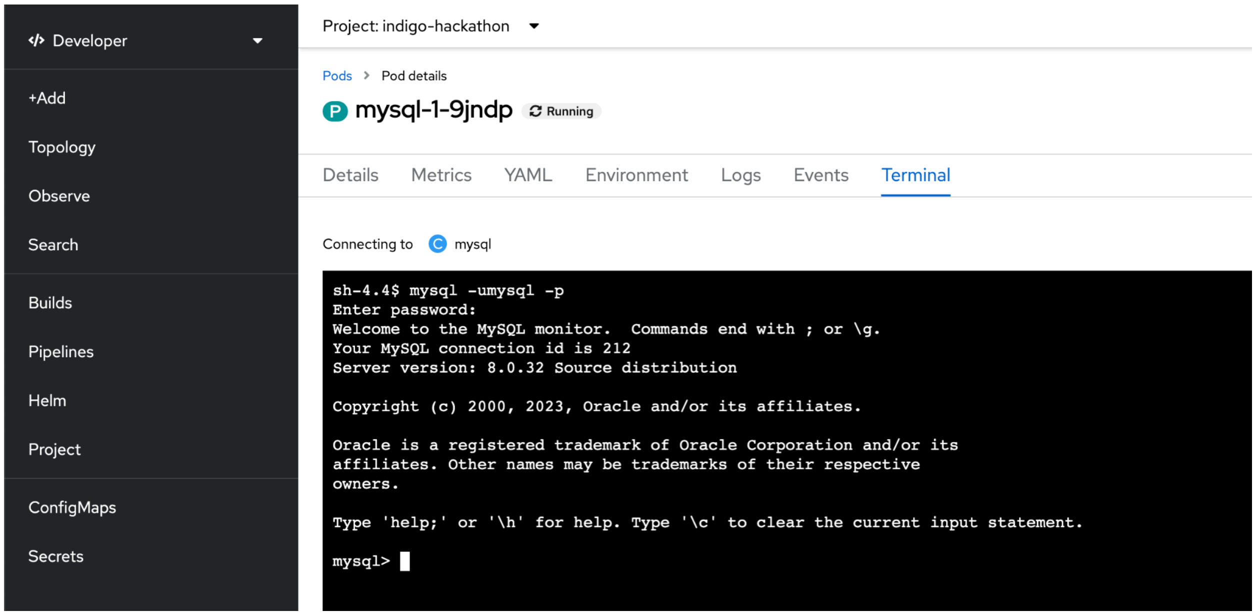


Navigate to the Terminal tab and then enter the following commands in the terminal that opens.

`mysql -umysql -p`

And enter the password when prompted.

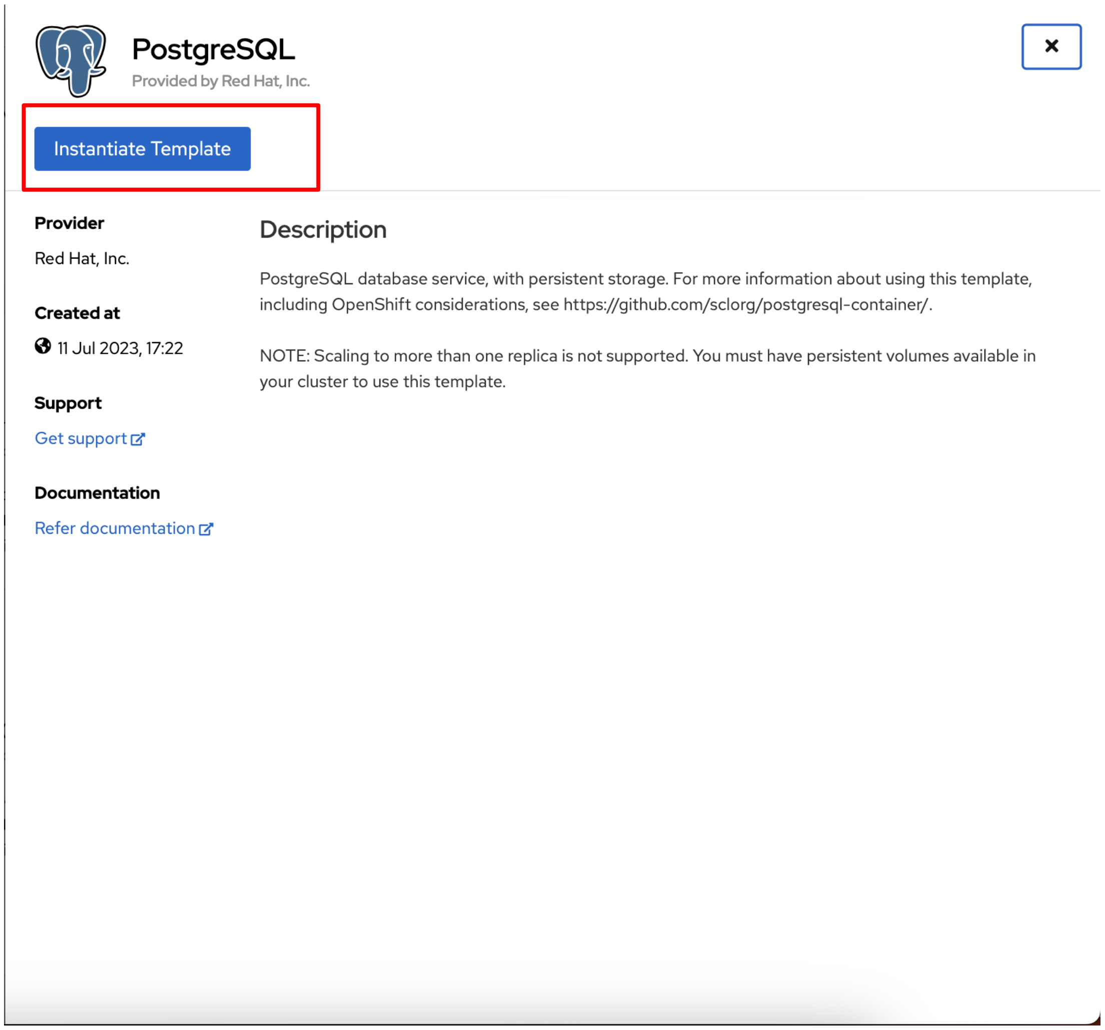
Now you can enter any SQL command that you need to manage the database.



```
sh-4.4$ mysql -umysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 212
Server version: 8.0.32 Source distribution

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```

Note: When you will be working on your backend application code using Red Hat Openshift Dev Spaces below, how to connect to the database which is deployed above with screenshots are provided for you to run DDL/DML statements for Database CRUD operations. Please refer to those screenshots for executing operations on the database.

For PostgreSQL



PostgreSQL
Provided by Red Hat, Inc.

Provider
Red Hat, Inc.

Created at
11 Jul 2023, 17:22

Support
Get support

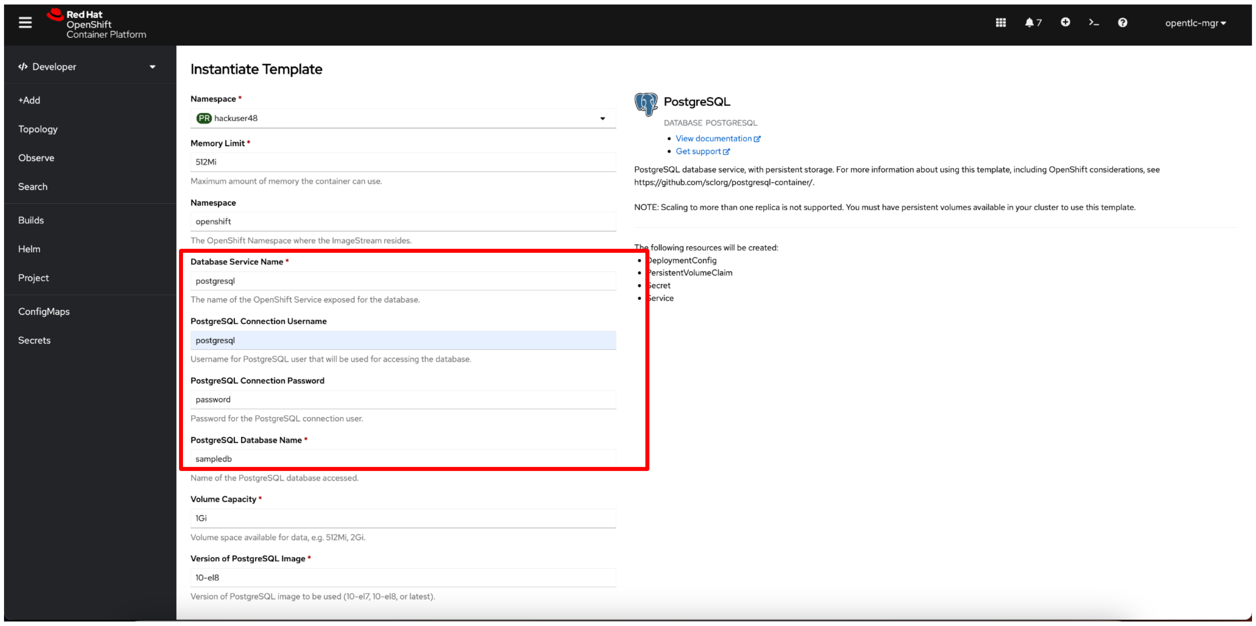
Documentation
Refer documentation

Description

PostgreSQL database service, with persistent storage. For more information about using this template, including OpenShift considerations, see https://github.com/sclorg/postgresql-container/.

NOTE: Scaling to more than one replica is not supported. You must have persistent volumes available in your cluster to use this template.

Use the following values for configuring backend code to the database. Use the below provided values & leave other configurations as default



Instantiate Template

Namespace *
hackuser48

Memory Limit *
512Mi
Maximum amount of memory the container can use.

Namespace
openshift
The OpenShift Namespace where the ImageStream resides.

Database Service Name *
postgresql
The name of the OpenShift Service exposed for the database.

PostgreSQL Connection Username
postgresql
Username for PostgreSQL user that will be used for accessing the database.

PostgreSQL Connection Password
password
Password for the PostgreSQL connection user.

PostgreSQL Database Name *
sampledb
Name of the PostgreSQL database accessed.

Volume Capacity *
1Gi
Volume space available for data, e.g. 512Mi, 2Gi.

Version of PostgreSQL Image *
10-el8
Version of PostgreSQL image to be used (10-el7, 10-el8, or latest).

PostgreSQL
DATABASE POSTGRESQL
- View documentation
- Get support

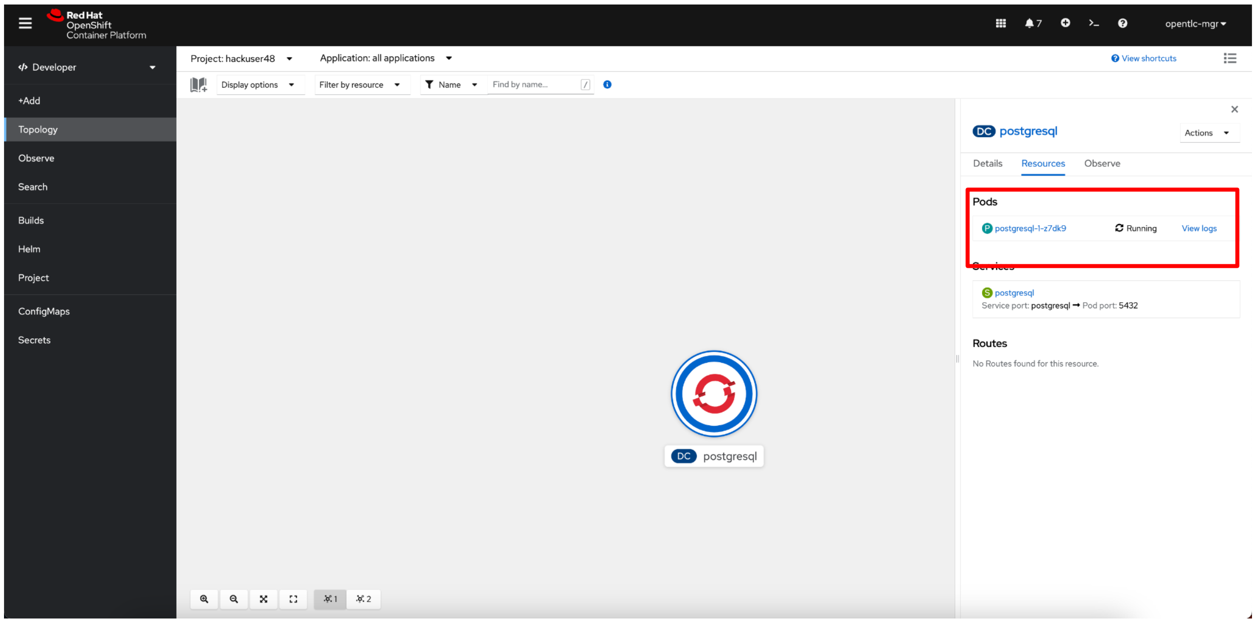
PostgreSQL database service, with persistent storage. For more information about using this template, including OpenShift considerations, see https://github.com/sclorg/postgresql-container/.

NOTE: Scaling to more than one replica is not supported. You must have persistent volumes available in your cluster to use this template.

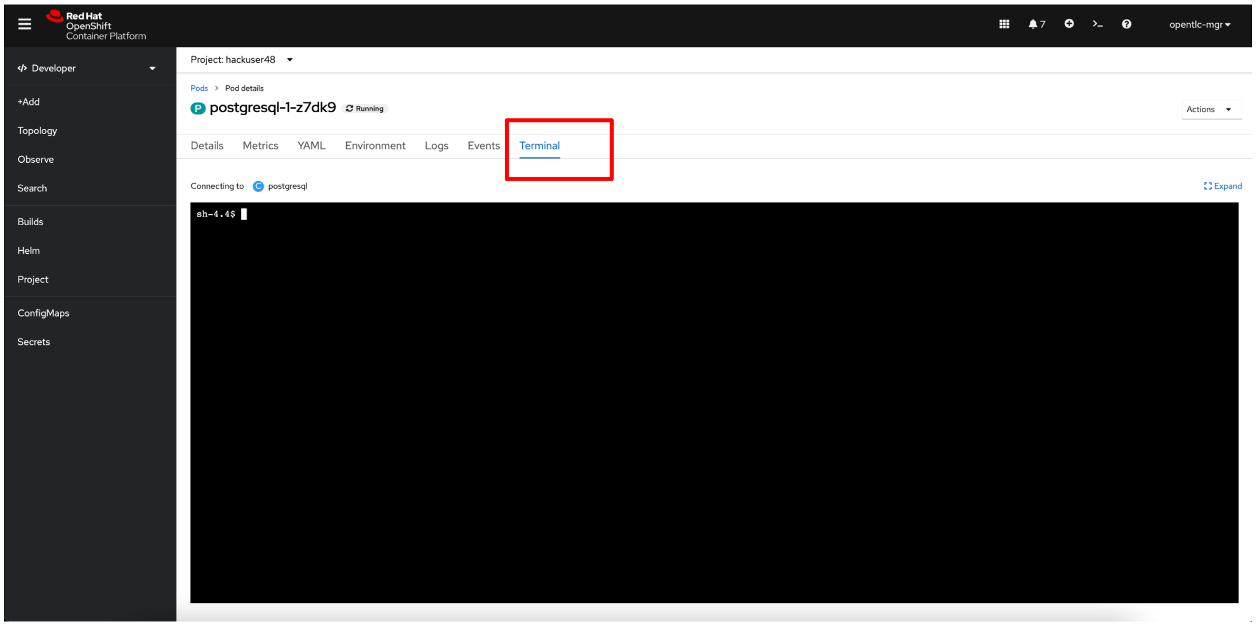
The following resources will be created:
- DeploymentConfig
- PersistentVolumeClaim
- Secret
- Service

- **Database Service Name : postgresql (**This will be the DB hostname which will be used later for your backend code to connect to the database. Port will be default which is 5432 for PostgreSQL. You can hard code in your backend code)
- **PostgreSQL connection Username: postgresql** (DB connection username for your backend code configuration. You can hard code in your backend code)
- **PostgreSQL Connection Password: password** (DB connection password for your backend code configuration. You can hard code in your backend code)
- **PostgreSQL Database name: sampledb** (This database will be created for you which you can configure in your backend code configuration. You can hard code in your backend code)

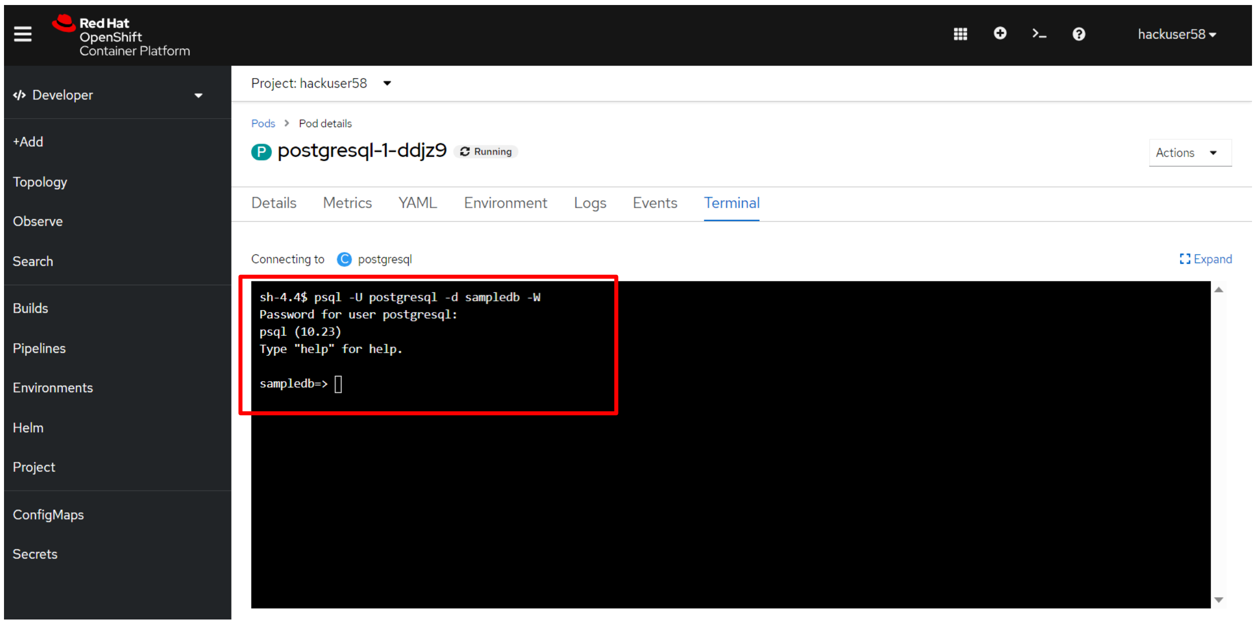
Scroll down the form and click on the "Create" button below. This will display the deployment screen & click on deployment as shown below. PostgreSQL pod will be in running status in a while. Click on the pod to access the terminal of PostgreSQL database to execute the database commands if required.



Navigate to the terminal tab & connect & execute the database commands if required.



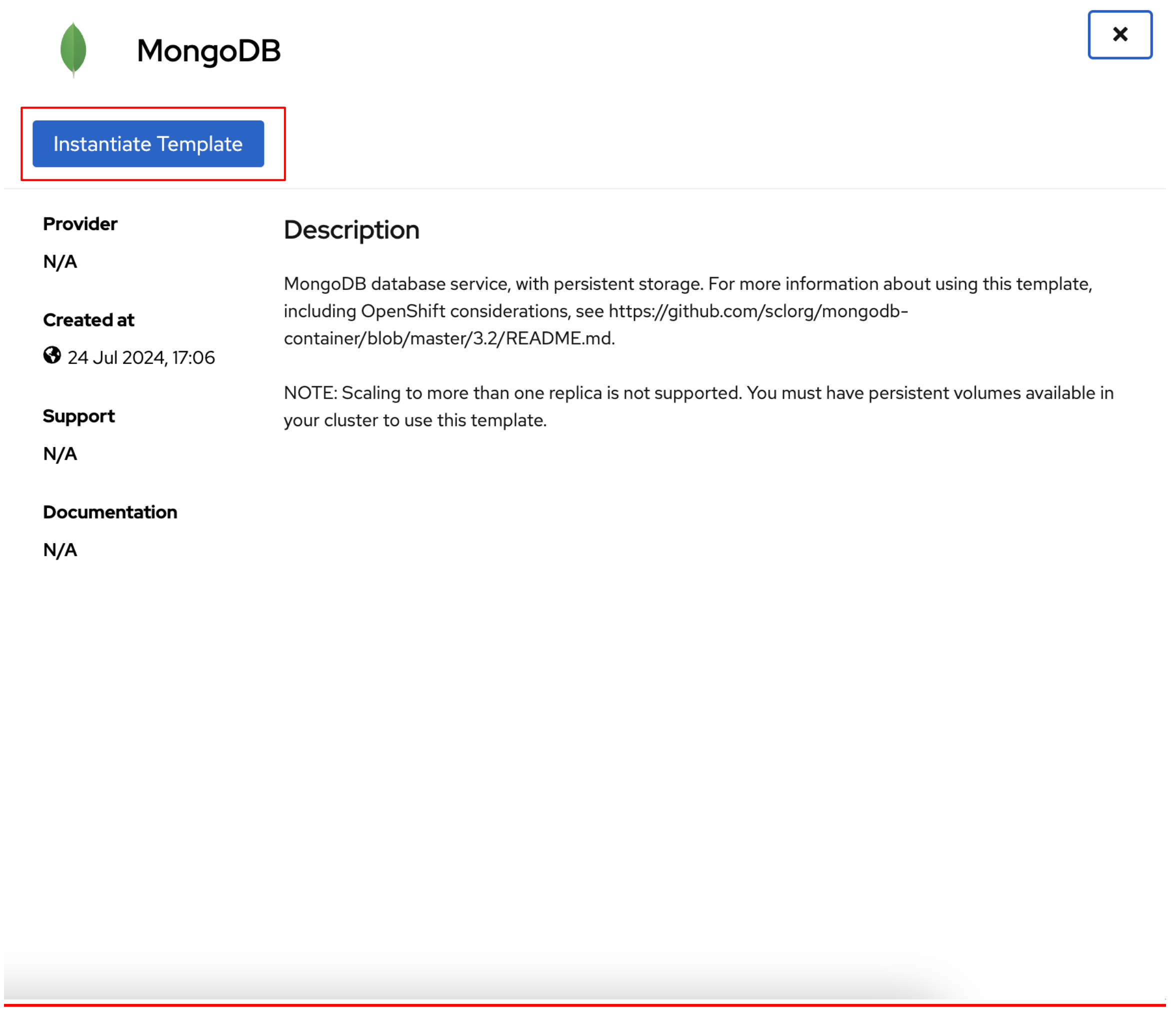
```
sh-4.4$
```



```
sh-4.4$ psql -U postgresql -d sampledb -W
Password for user postgresql:
psql (10.23)
Type "help" for help.

sampledb=>
```

Note: When you will be working on your backend application code using Red Hat Openshift Dev Spaces below, how to connect to the database which is deployed above with screenshots are provided for you to run DDL/DML statements for Database CRUD operations. Please refer to those screenshots for executing operations on the database.

For MongoDB



Provider

N/A

Created at

24 Jul 2024, 17:06

Support

N/A

Documentation

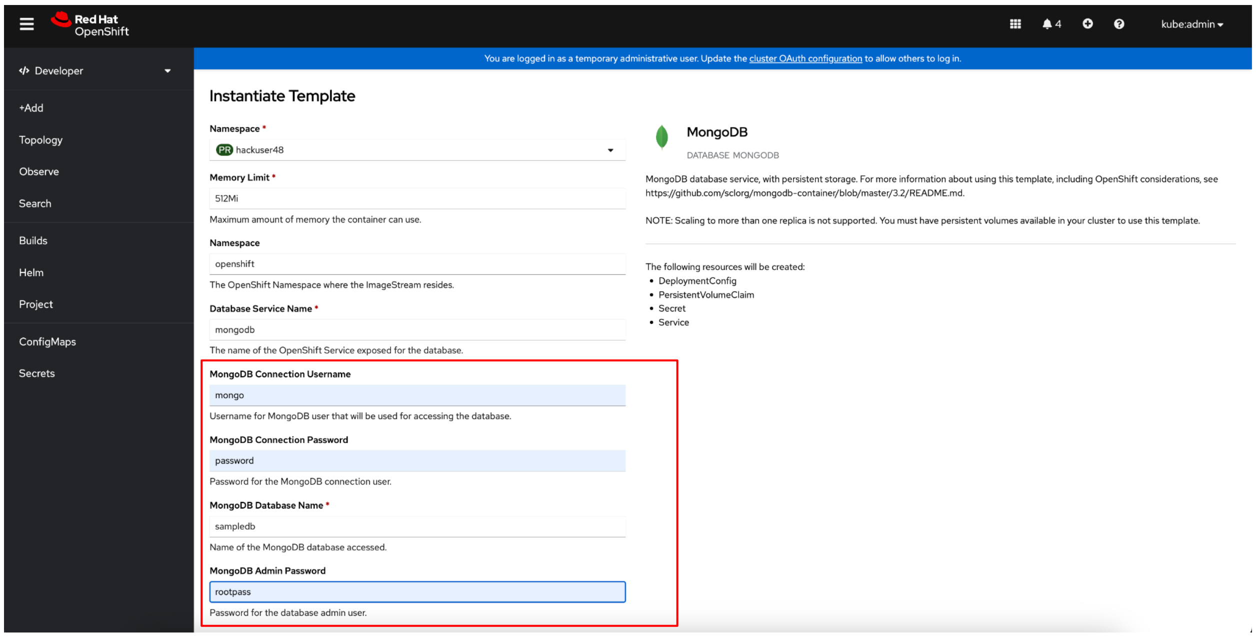
N/A

Description

MongoDB database service, with persistent storage. For more information about using this template, including OpenShift considerations, see https://github.com/sclorg/mongodb-container/blob/master/3.2/README.md.

NOTE: Scaling to more than one replica is not supported. You must have persistent volumes available in your cluster to use this template.

Use the following values for configuring backend code to the database. Use the below provided values & leave other configurations as default



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Instantiate Template

Namespace *	PR hackuser48
Memory Limit *	512Mi
	Maximum amount of memory the container can use.
Namespace	openshift
	The OpenShift Namespace where the ImageStream resides.
Database Service Name *	mongodb
	The name of the OpenShift Service exposed for the database.
MongoDB Connection Username	mongo
	Username for MongoDB user that will be used for accessing the database.
MongoDB Connection Password	password
	Password for the MongoDB connection user.
MongoDB Database Name *	sampledb
	Name of the MongoDB database accessed.
MongoDB Admin Password	rootpass
	Password for the database admin user.

MongoDB

DATABASE MONGODB

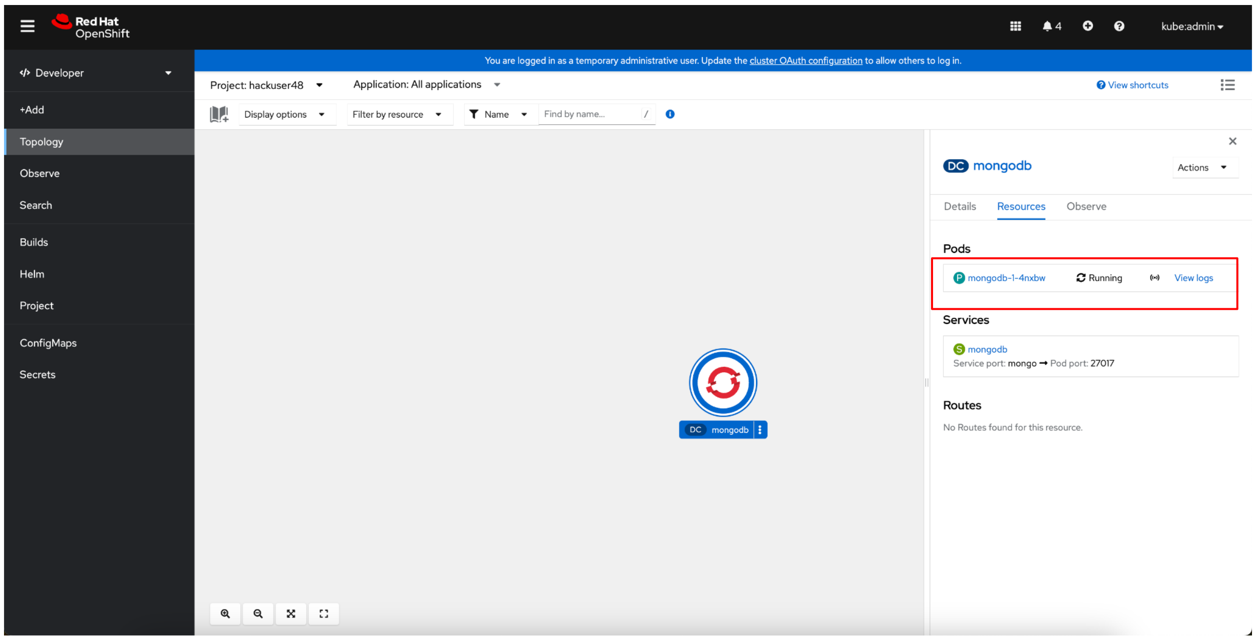
MongoDB database service, with persistent storage. For more information about using this template, including OpenShift considerations, see https://github.com/sclorg/mongodb-container/blob/master/3.2/README.md.

NOTE: Scaling to more than one replica is not supported. You must have persistent volumes available in your cluster to use this template.

The following resources will be created:
- DeploymentConfig
- PersistentVolumeClaim
- Secret
- Service

- **Database Service Name : mongodb (**This will be the DB hostname which will be used later for your backend code to connect to the database. Port will be default which is 27017 for MongoDB. You can hard code in your backend code)
- **MongoDB connection Username: mongo** (DB connection username for your backend code configuration. You can hard code in your backend code)
- **MongoDB Connection Password: password** (DB connection password for your backend code configuration. You can hard code in your backend code)
- **MongoDB Database name: sampledb** (This database will be created for you which you can configure in your backend code configuration. You can hard code in your backend code)
- **MongoDB Admin Password: rootpass** (Admin password)

Scroll down the form and click on the "Create" button below. This will display the deployment screen & click on deployment as shown below. MongoDB pod will be in running status in a while. Click on the pod to access the terminal of MongoDB database to execute the database commands if required.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: hackuser48

Application: All applications

DC mongodb

Pods

mongodb-1-4nxbw Running View logs

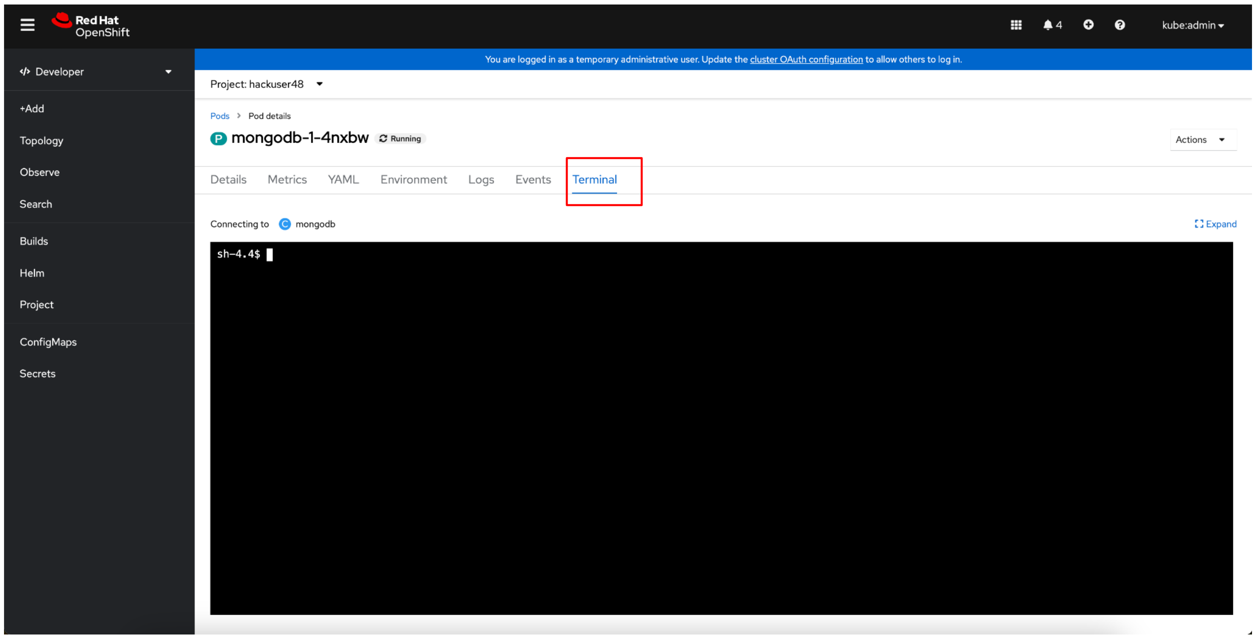
Services

mongodb
Service port: mongo → Pod port: 27017

Routes

No Routes found for this resource.

Navigate to the terminal tab & connect & execute the database commands if required.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

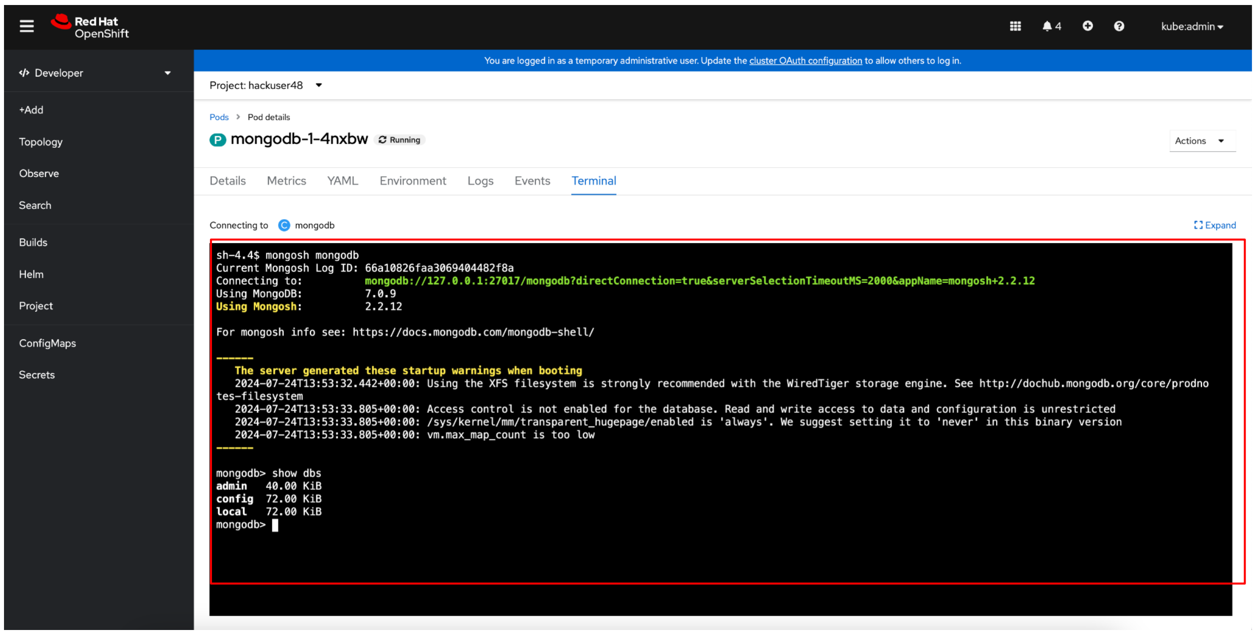
Project: hackuser48

Pods > Pod details

P mongodb-1-4nxbw Running

Connecting to C mongodb

```
sh-4.4$
```



```
sh-4.4$ mongosh mongodb
Current Mongosh Log ID: 66a10826faa3069404482f8a
Connecting to:          mongodb://127.0.0.1:27017/mongodb?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.12
Using MongoDB:          7.0.9
Using Mongosh:          2.2.12

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2024-07-24T13:53:32.442+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodno
tes-filesystem
   2024-07-24T13:53:33.805+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
   2024-07-24T13:53:33.805+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We suggest setting it to 'never' in this binary version
   2024-07-24T13:53:33.805+00:00: vm.max_map_count is too low
------

mongodb> show dbs
admin   40.00 KiB
config  72.00 KiB
local   72.00 KiB
mongodb> 
```

Note: When you will be working on your backend application code using Red Hat Openshift Dev Spaces below, how to connect to the database which is deployed above with screenshots are provided for you to run DDL/DML statements for Database CRUD operations. Please refer to those screenshots for executing operations on the database.

Next step - Move to Guide 3-IDEWorkspace

After deploying the database on Openshift platform, you would need an IDE workspace environment to clone your code from GitHub repositories & perform build & testing of your code.

So move to Guide 3 - IDE workspace to perform code development.