Full Stack Engineer

Module 6.2: Deployment

Prerequisites

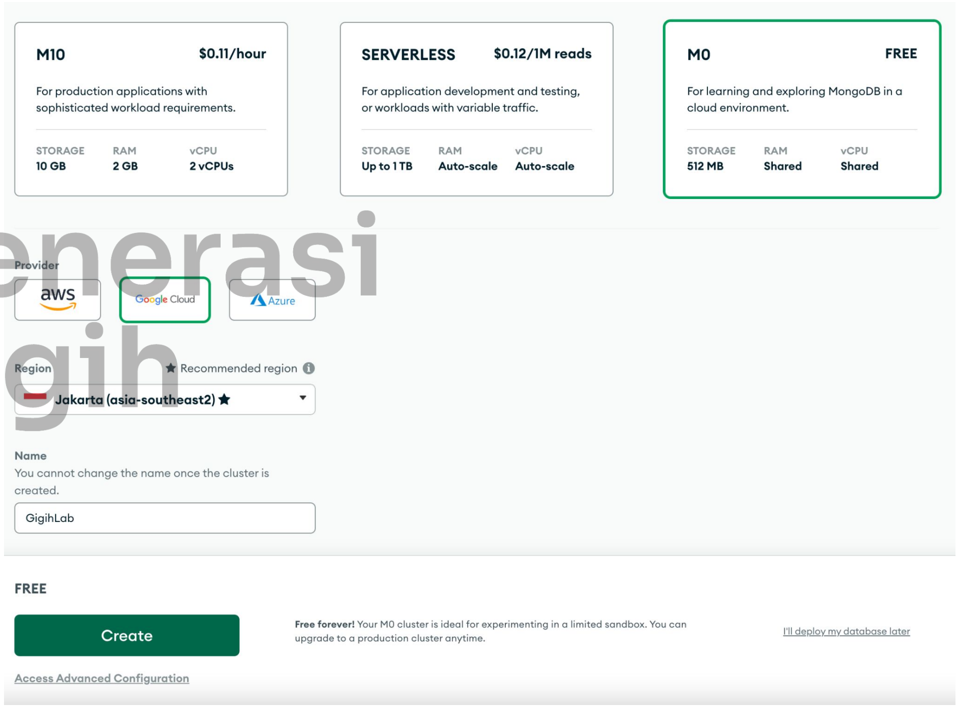
The following prerequisites must be prepared before the class started!

Generasi Gigih

1: Setup MongoDB Atlas (1)

Register a free account at MongoDB Atlas and create a new cluster.

For the purpose of this module, you can just create an M0 cluster (because it's free!) and select Google Cloud as the provider and Jakarta as the region. You can name the cluster with any name you want.



M10 $0.11/hour
For production applications with sophisticated workload requirements.

STORAGE	RAM	vCPU
10 GB	2 GB	2 vCPUs

SERVERLESS $0.12/1M reads
For application development and testing, or workloads with variable traffic.

STORAGE	RAM	vCPU
Up to 1 TB	Auto-scale	Auto-scale

M0 FREE
For learning and exploring MongoDB in a cloud environment.

STORAGE	RAM	vCPU
512 MB	Shared	Shared

Provider: aws [✓ Google Cloud] Azure

Region ★ Recommended region
Jakarta (asia-southeast2) ★

Name
You cannot change the name once the cluster is created.
GigihLab

FREE

Create

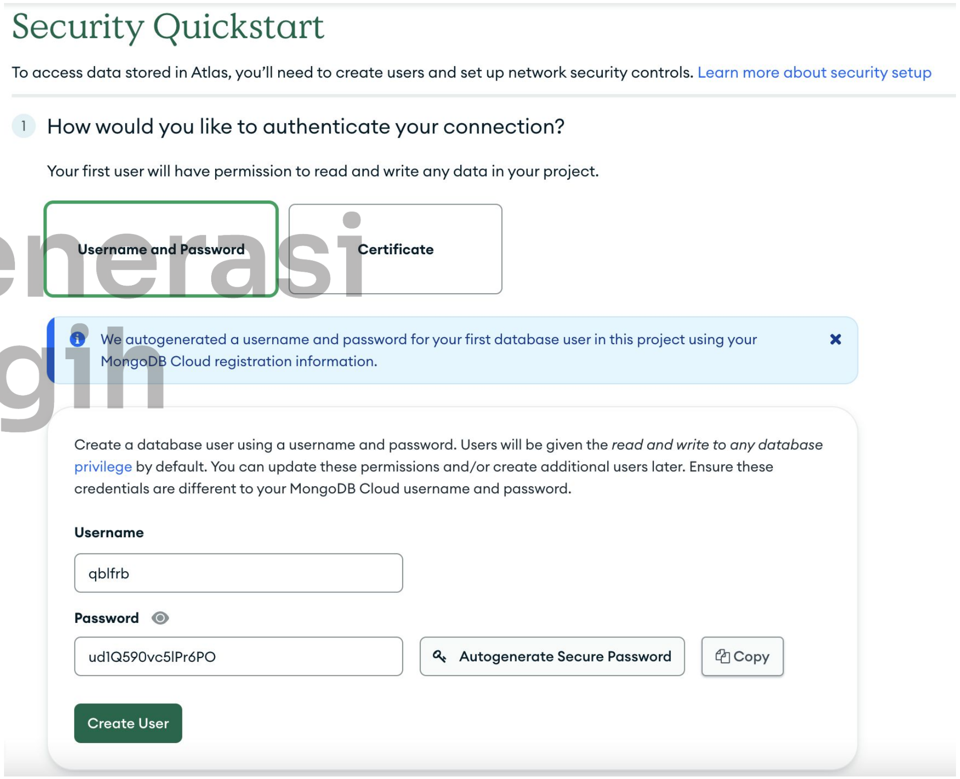
Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

I'll deploy my database later

Access Advanced Configuration

1: Setup MongoDB Atlas (2)

When prompted with Security Quickstart, for this module, you can just choose username and password as your authentication method. Don't forget to store the generated password somewhere save.



Security Quickstart

To access data stored in Atlas, you'll need to create users and set up network security controls. Learn more about security setup

1 How would you like to authenticate your connection?

Your first user will have permission to read and write any data in your project.

We autogenerated a username and password for your first database user in this project using your MongoDB Cloud registration information.

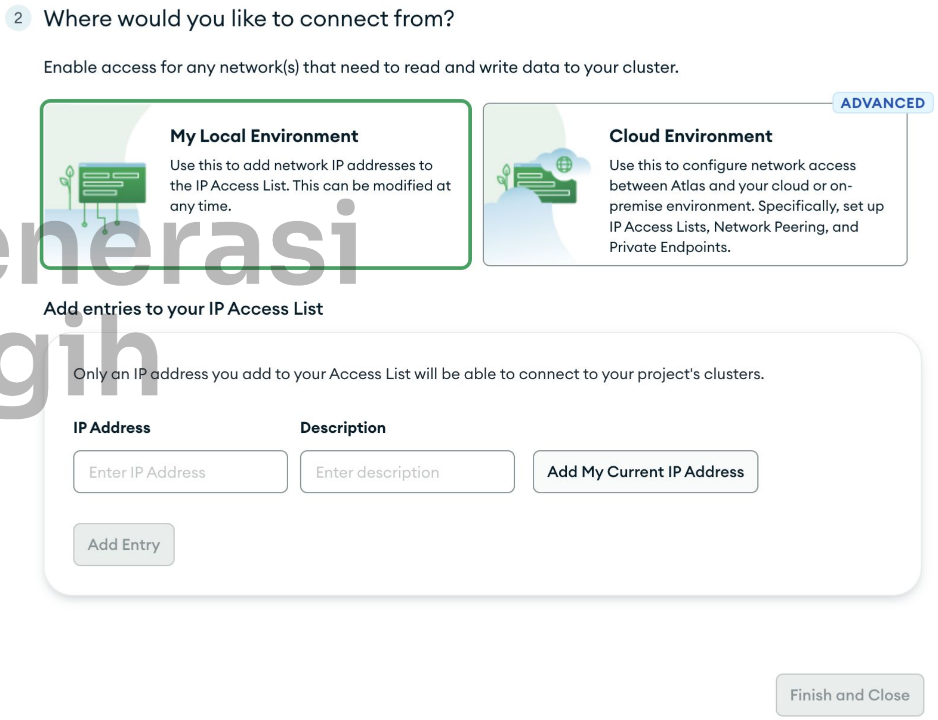
Create a database user using a username and password. Users will be given the *read and write to any database privilege* by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.

Username: qblfrb

Password: ud1Q590vc5lPr6PO

1: Setup MongoDB Atlas (3)

Choose “My Local Environment” and click “Add My Current IP Address” in the connection setting section below. Click “Finish and Close” once you’re done.


2 Where would you like to connect from?

Enable access for any network(s) that need to read and write data to your cluster.

My Local Environment
Use this to add network IP addresses to the IP Access List. This can be modified at any time.

ADVANCED
Cloud Environment
Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.

Add entries to your IP Access List

Only an IP address you add to your Access List will be able to connect to your project's clusters.

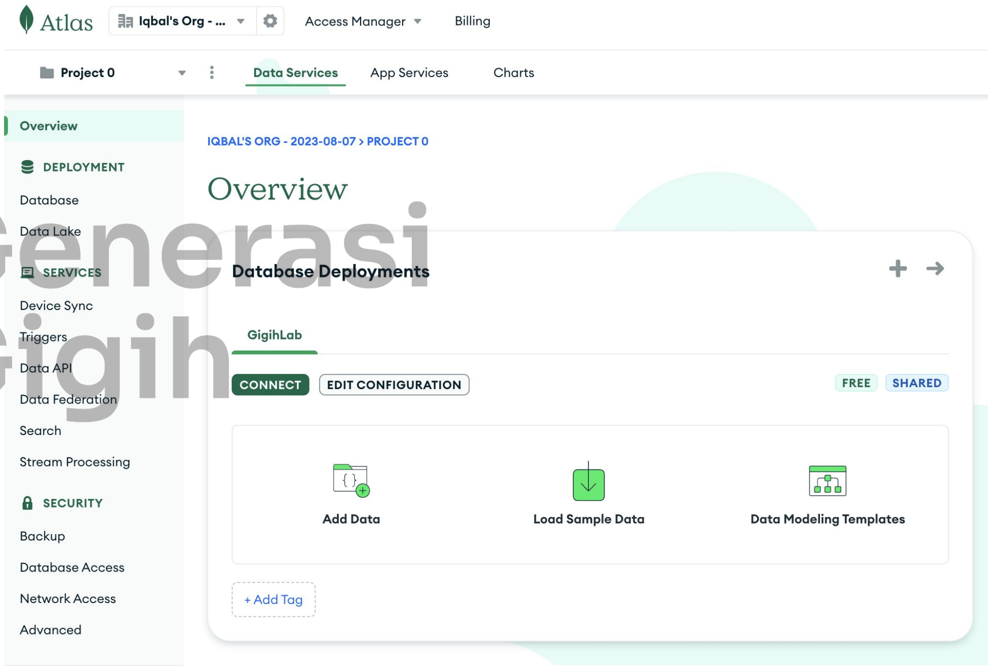
IP Address	Description	
Enter IP Address	Enter description	Add My Current IP Address

Add Entry

Finish and Close

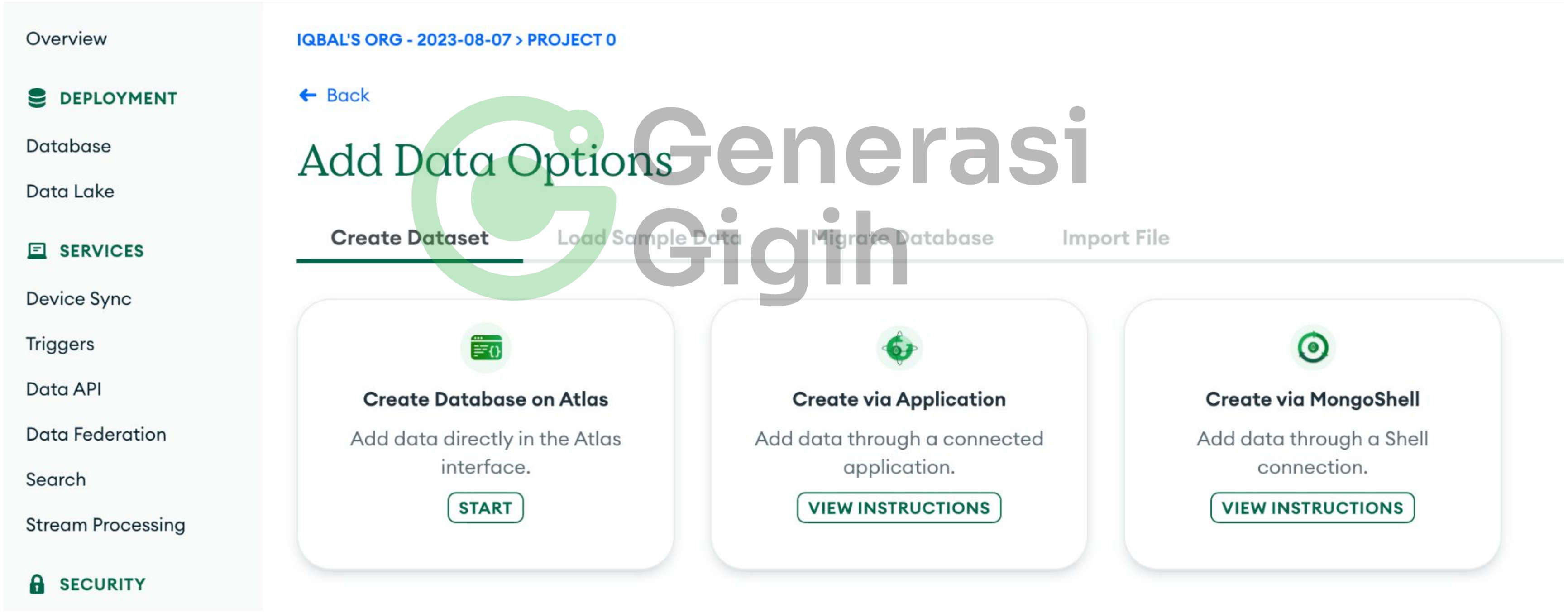
1: Setup MongoDB Atlas (4)

Next, we'll try to add sample data. From this page, click "Add Data".



1: Setup MongoDB Atlas (5)

Then, “Create Database on Atlas”.



Overview

DEPLOYMENT

Database

Data Lake

SERVICES

Device Sync

Triggers

Data API

Data Federation

Search

Stream Processing

SECURITY

IQBAL'S ORG - 2023-08-07 > PROJECT 0

← Back

Add Data Options

Create Dataset | Load Sample Data | Migrate Database | Import File

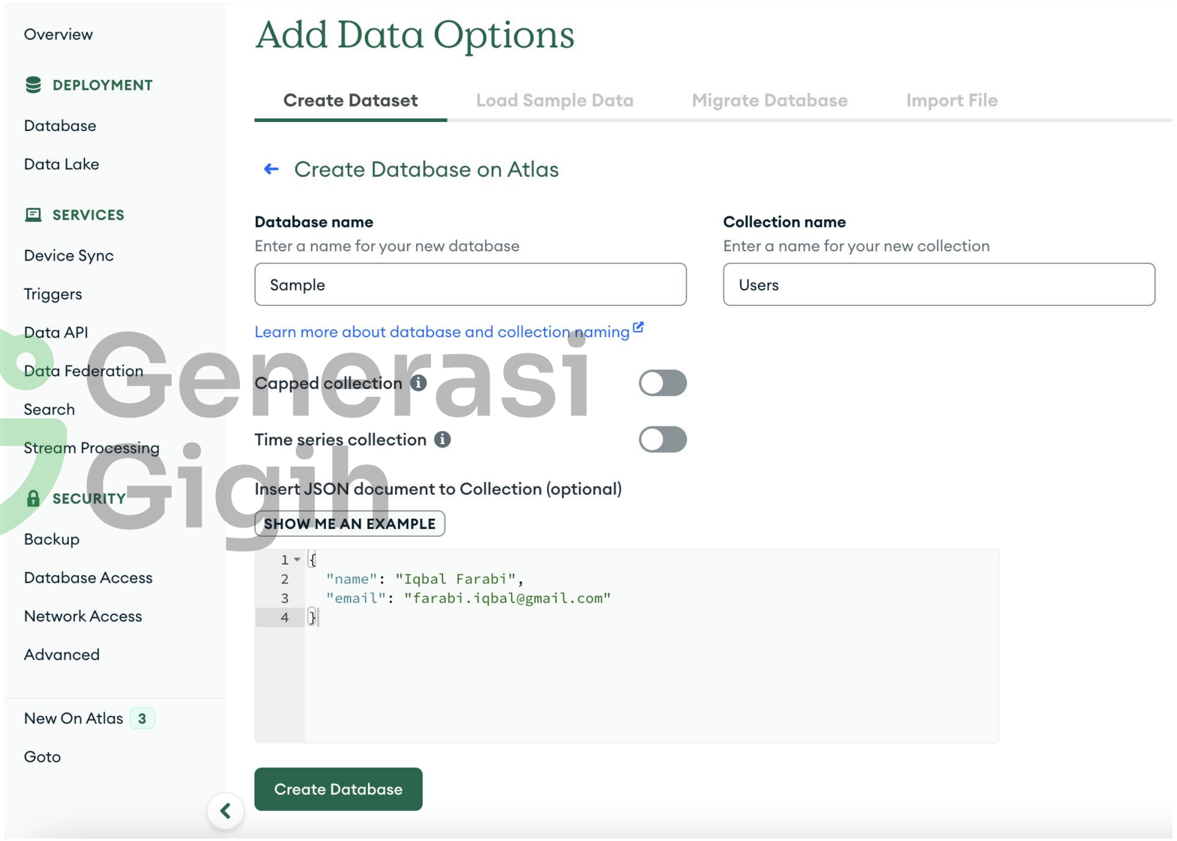
Create Database on Atlas
Add data directly in the Atlas interface.
START

Create via Application
Add data through a connected application.
VIEW INSTRUCTIONS

Create via MongoShell
Add data through a Shell connection.
VIEW INSTRUCTIONS

1: Setup MongoDB Atlas (6)

Fill in the database name, collection name, and document as you like. In this example, we'll create a database called "Sample", a collection called "Users", and we insert one document.



Database name
Enter a name for your new database
Sample

Collection name
Enter a name for your new collection
Users

Capped collection: off

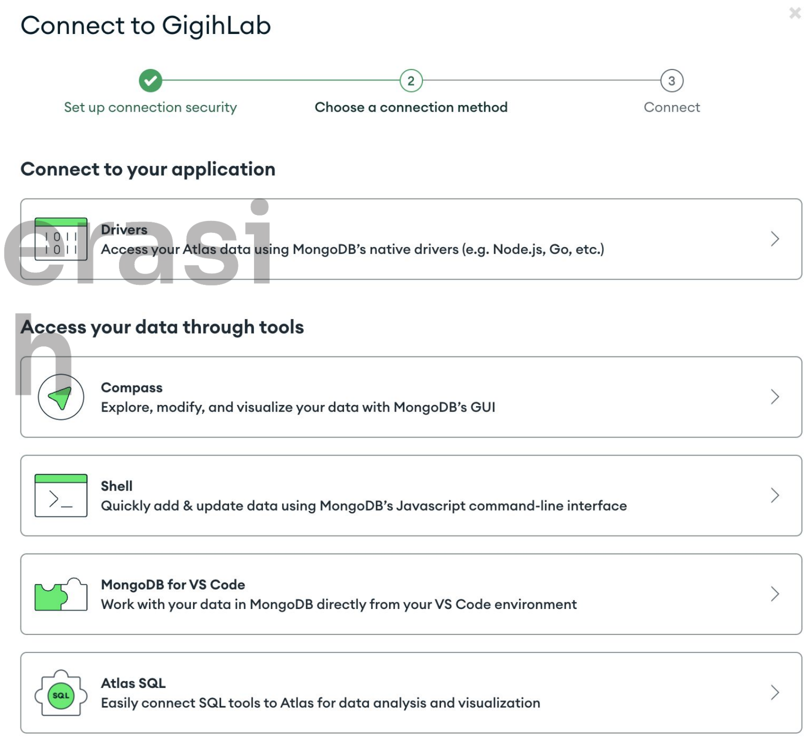
Time series collection: off

Insert JSON document to Collection (optional)

```
{
  "name": "Iqbal Farabi",
  "email": "farabi.iqbal@gmail.com"
}
```

1: Setup MongoDB Atlas (7)

Next, we'll set up a connection to our MongoDB Atlas. In this preparation step, we'll connect with Compass. After selecting Compass, you'll be asked to download and install MongoDB Compass.


Connect to GigihLab

Connect to your application

Drivers
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

Access your data through tools

Compass
Explore, modify, and visualize your data with MongoDB's GUI

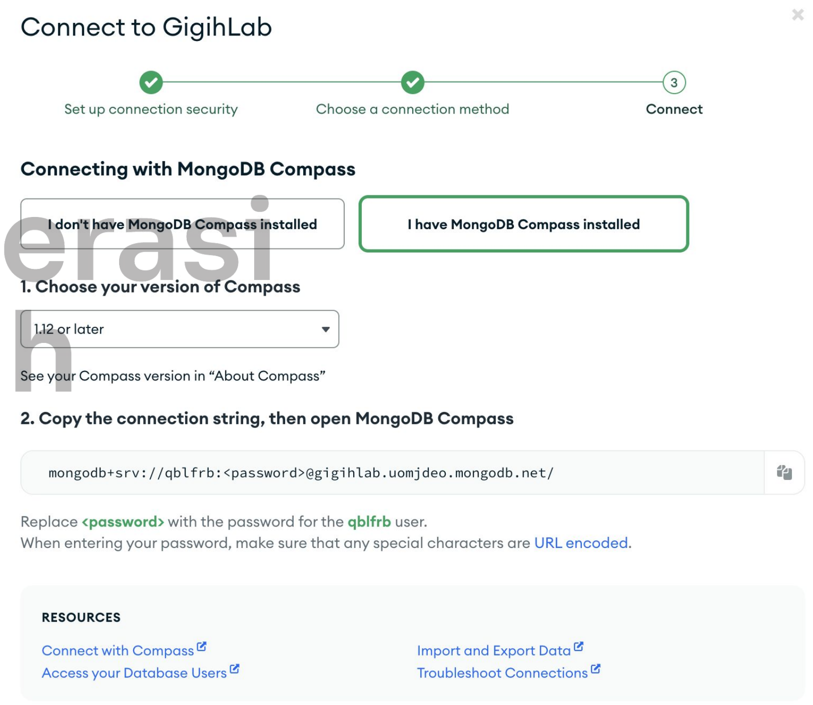
Shell
Quickly add & update data using MongoDB's Javascript command-line interface

MongoDB for VS Code
Work with your data in MongoDB directly from your VS Code environment

Atlas SQL
Easily connect SQL tools to Atlas for data analysis and visualization

1: Setup MongoDB Atlas (8)

Once you choose connect with MongoDB Compass, you'll be given the connection string.


Connect to GigihLab
Set up connection security
Choose a connection method
3
Connect
I don't have MongoDB Compass installed
I have MongoDB Compass installed
RESOURCES
Connect with Compass
Import and Export Data
Access your Database Users
Troubleshoot Connections

Connecting with MongoDB Compass

1. Choose your version of Compass

1.12 or later

See your Compass version in "About Compass"

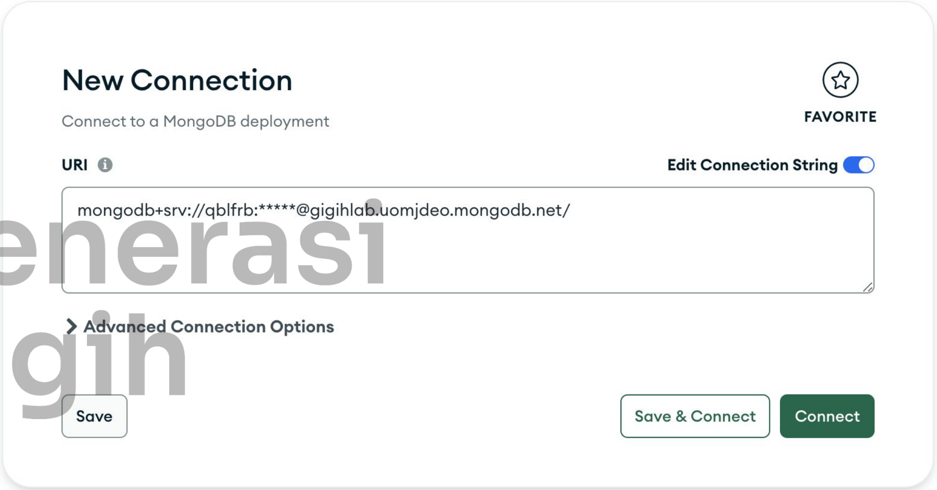
2. Copy the connection string, then open MongoDB Compass

```
mongodb+srv://qblfrb:<password>@gigihlab.uomjdeo.mongodb.net/
```

Replace **<password>** with the password for the **qblfrb** user.
When entering your password, make sure that any special characters are URL encoded.

1: Setup MongoDB Atlas (9)

Once your MongoDB Compass is installed, you can connect with your connection string.


New Connection

Connect to a MongoDB deployment

URI

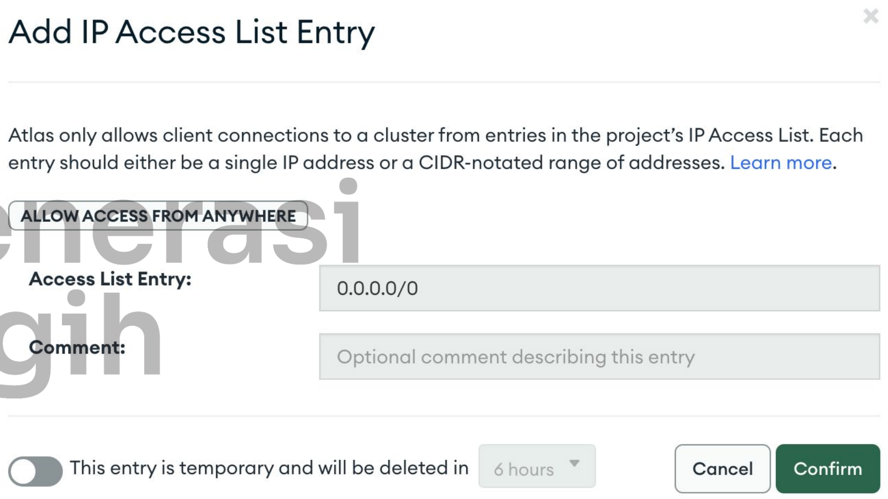
Edit Connection String

```
mongodb+srv://qblfrb:*****@gigihlab.uomjdeo.mongodb.net/
```

1: Setup MongoDB Atlas (10)

If you encounter problem to connect via MongoDB Compass, change your network access setting in MongoDB Atlas. From “Network Access” menu, click “+ ADD IP ADDRESS”. Click “ALLOW ACCESS FROM ANYWHERE” in the following pop up form.

IMPORTANT: In a real production system, you don’t want to do this as you want to only allow access from the IP addresses of your backend endpoints.


Add IP Access List Entry

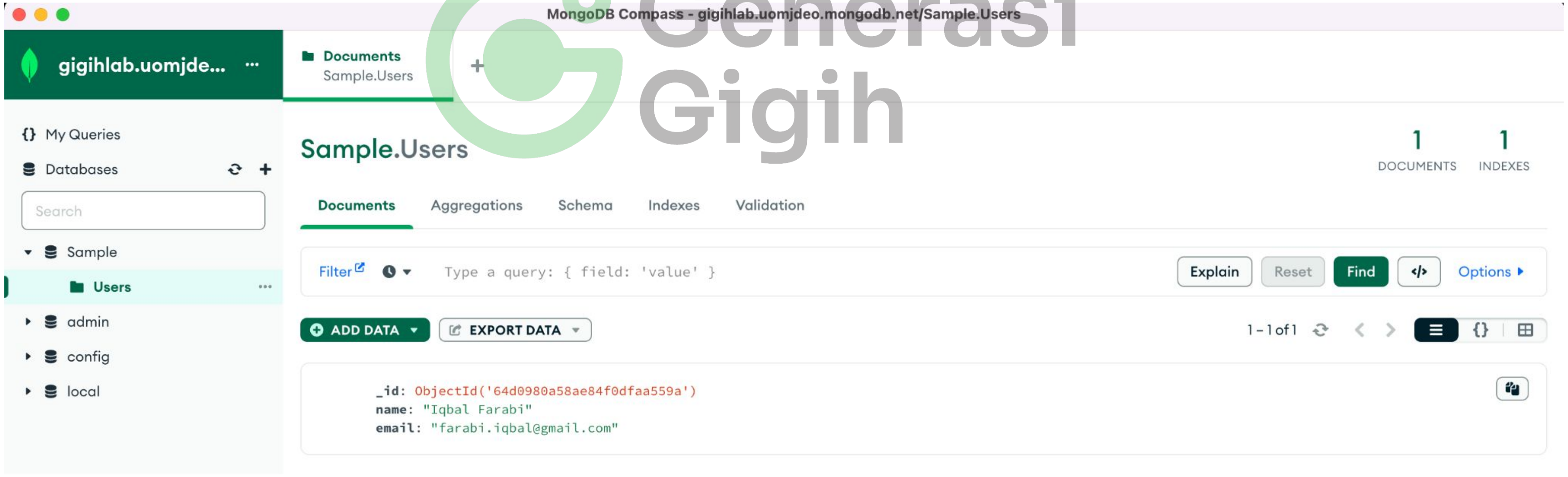
Atlas only allows client connections to a cluster from entries in the project’s IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. Learn more.

Access List Entry:	0.0.0.0/0
Comment:	Optional comment describing this entry

This entry is temporary and will be deleted in 6 hours

1: Setup MongoDB Atlas (11)

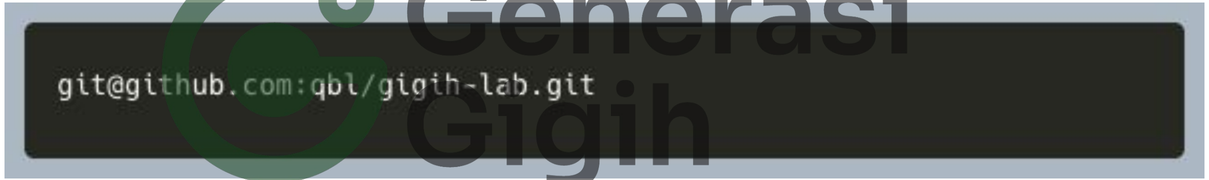
If you're able to connect and see the sample database, collection, and document that you set up before, then congratulations! Your MongoDB Atlas setup is complete.



```
_id: ObjectId('64d0980a58ae84f0dfaa559a')
name: "Iqbal Farabi"
email: "farabi.iqbal@gmail.com"
```

2: Setup Sample Project (1)

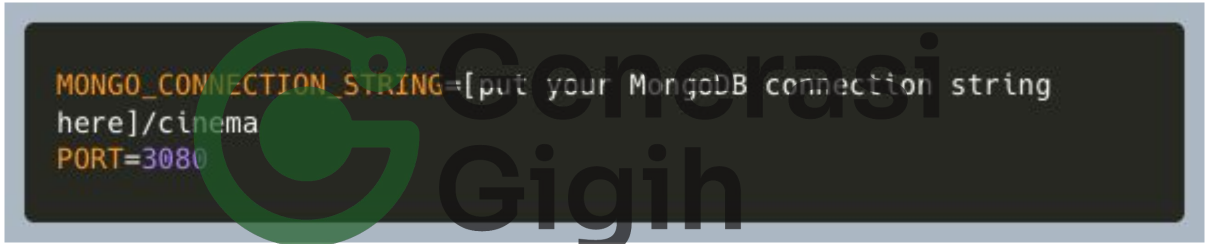
Clone the sample project using the following command:



```
git@github.com:qbl/gigih-lab.git
```

2: Setup Sample Project (2)

In your .env file, add the following environment variables:



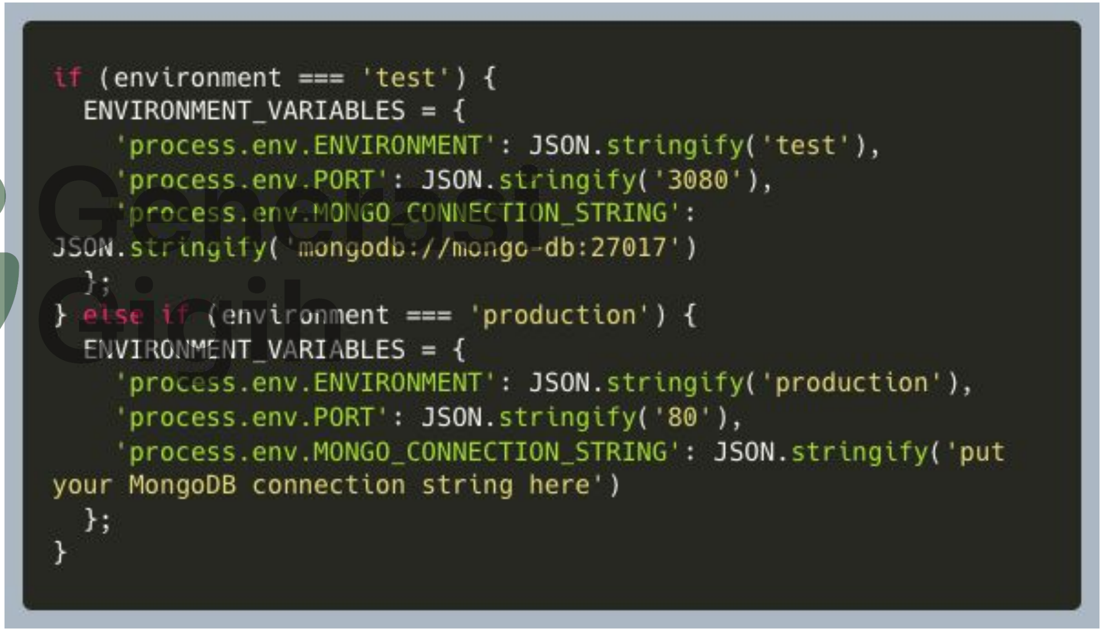
```
MONGO_CONNECTION_STRING=[put your MongoDB connection string
here]/cinema
PORT=3080
```

In the example above, we're going to name our database "cinema".

2: Setup Sample Project (3)

For deployment in production with Docker, put your MongoDB Atlas connection string in webpack.config.js file.

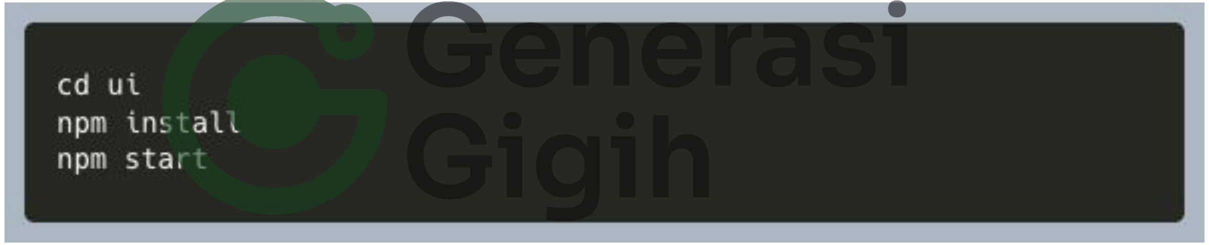
IMPORTANT: In actual production environment, this is not the correct way to store credentials and you should never check in your credentials into your git repositories.



```
if (environment === 'test') {
  ENVIRONMENT_VARIABLES = {
    'process.env.ENVIRONMENT': JSON.stringify('test'),
    'process.env.PORT': JSON.stringify('3080'),
    'process.env.MONGO_CONNECTION_STRING':
JSON.stringify('mongodb://mongo-db:27017')
  };
} else if (environment === 'production') {
  ENVIRONMENT_VARIABLES = {
    'process.env.ENVIRONMENT': JSON.stringify('production'),
    'process.env.PORT': JSON.stringify('80'),
    'process.env.MONGO_CONNECTION_STRING': JSON.stringify('put
your MongoDB connection string here')
  };
}
```

2: Setup Sample Project (3)

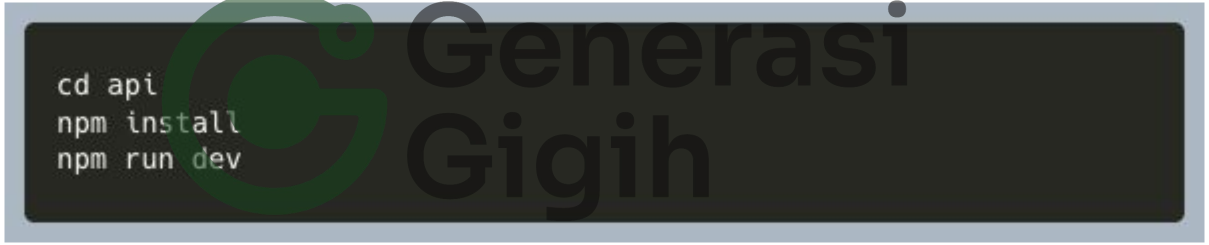
From the project's root directory, run the following commands to run the frontend:



```
cd ui
npm install
npm start
```

2: Setup Sample Project (4)

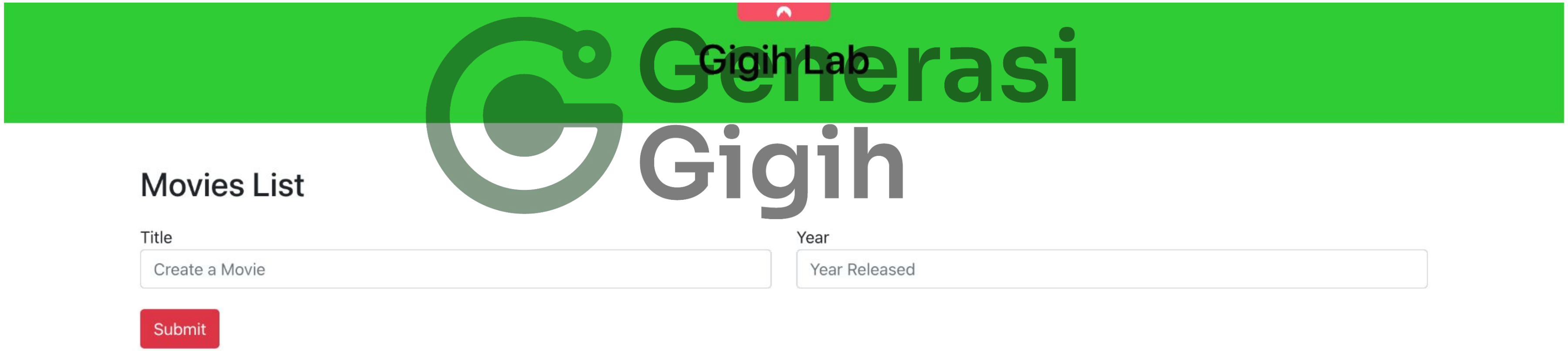
From the project's root directory, run the following commands to run the backend:



```
cd api
npm install
npm run dev
```

2: Setup Sample Project (5)

Open http://localhost:3000/ in your browser, you should see something like:


Gigih Lab

Movies List

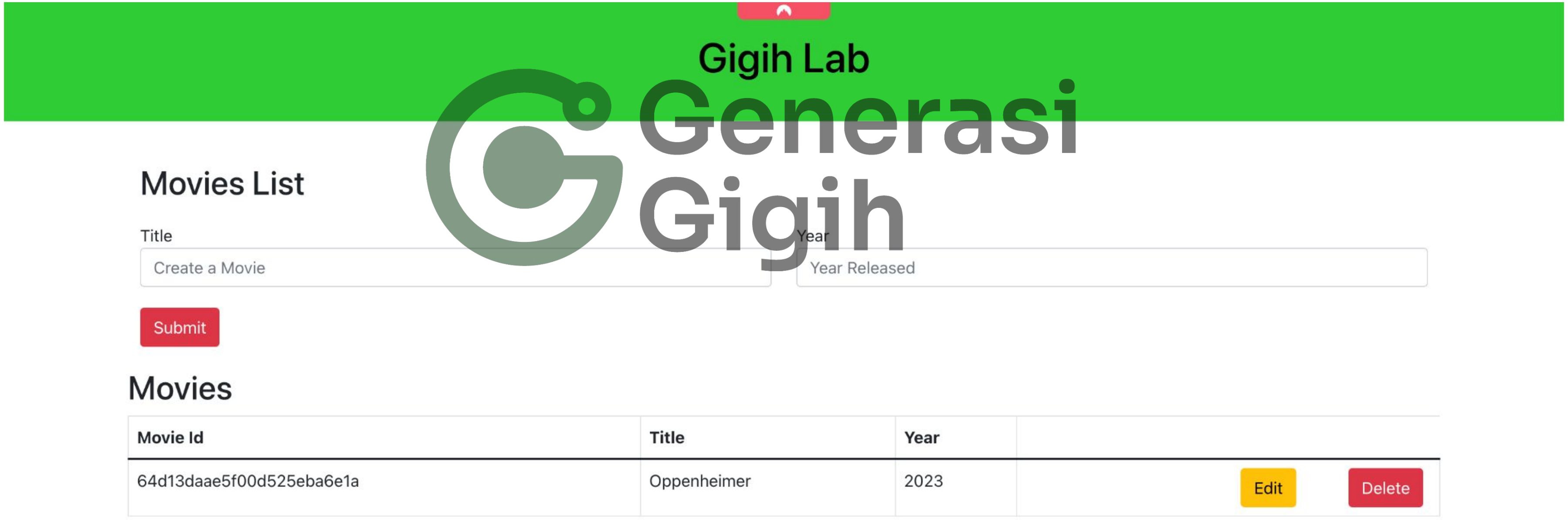
Title: Create a Movie

Year: Year Released

Submit

2: Setup Sample Project (6)

Try add some movies data, your app should be able to store save the data:



Gigih Lab

Movies List

Title

Create a Movie

Year

Year Released

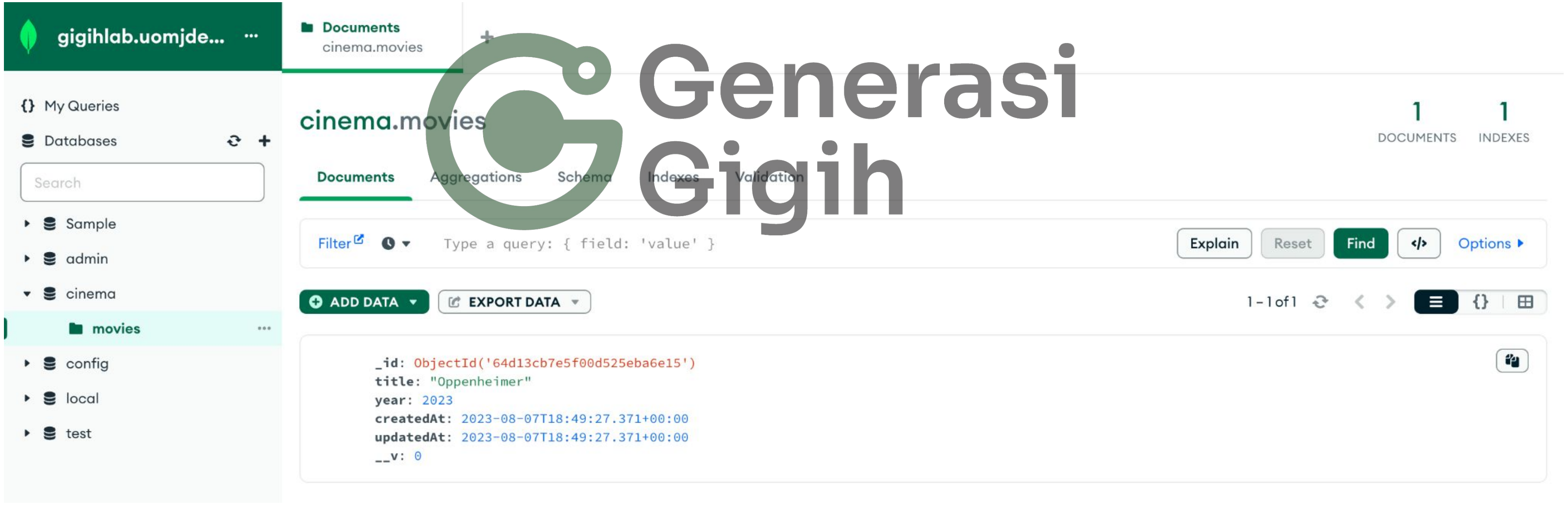
Submit

Movies

Movie Id	Title	Year	
64d13daae5f00d525eba6e1a	Oppenheimer	2023	Edit Delete

2: Setup Sample Project (7)

From your MongoDB Compass, you should also be able to see the same data:



```
_id: ObjectId('64d13cb7e5f00d525eba6e15')
title: "Oppenheimer"
year: 2023
createdAt: 2023-08-07T18:49:27.371+00:00
updatedAt: 2023-08-07T18:49:27.371+00:00
__v: 0
```

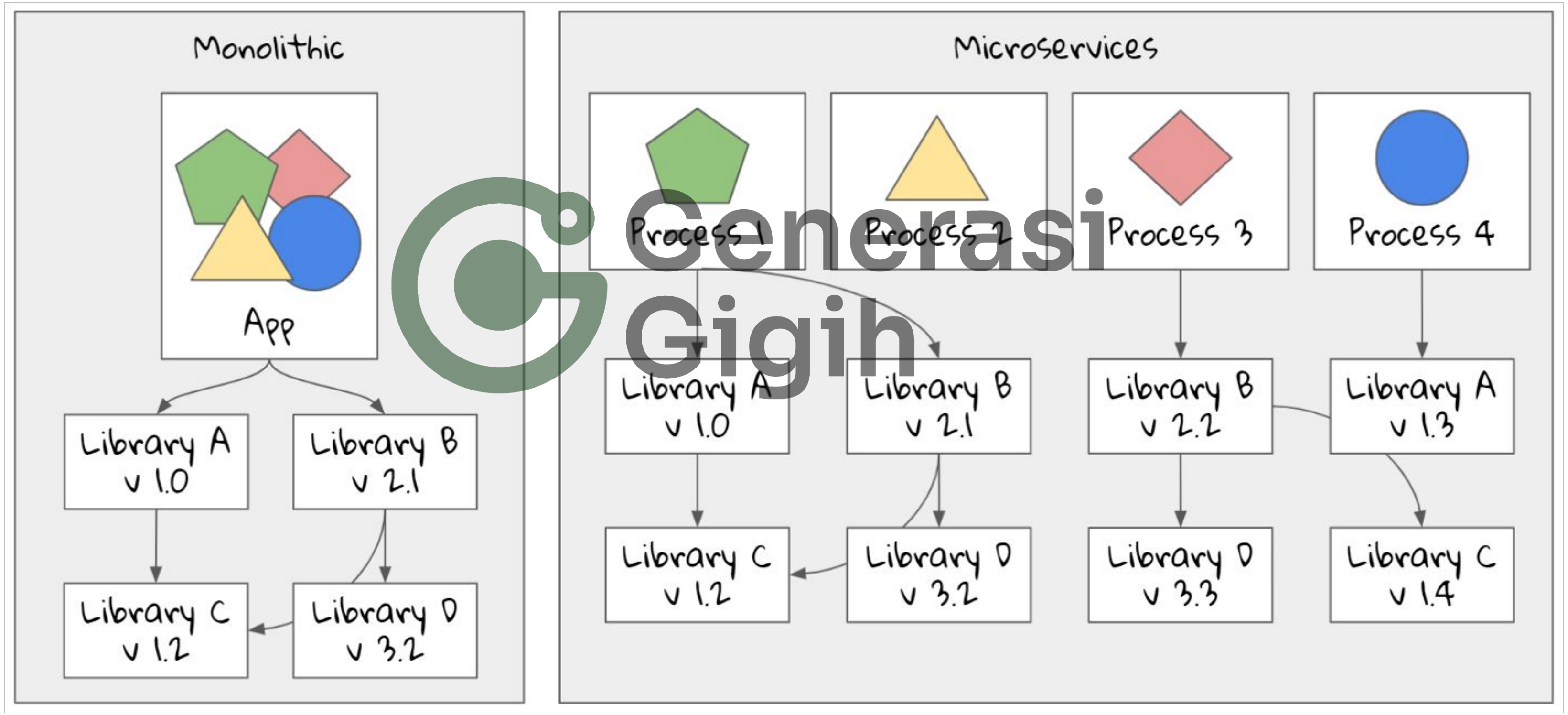
3: Setup Docker

- Follow the instructions to install Docker on your local machine in this page.
- Register yourself to Docker Hub.

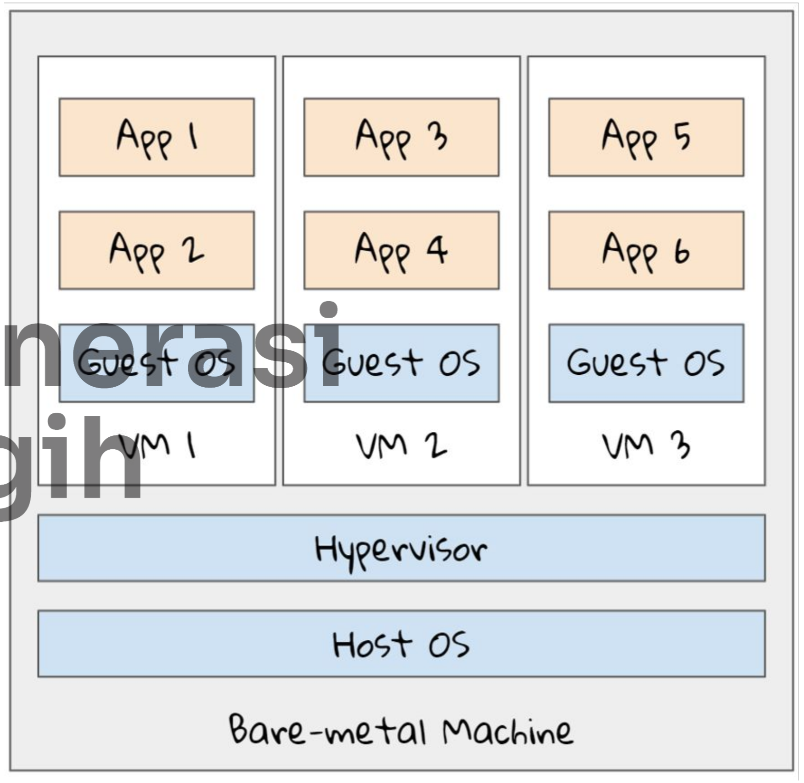
Generasi Gigih

Intro to Containers

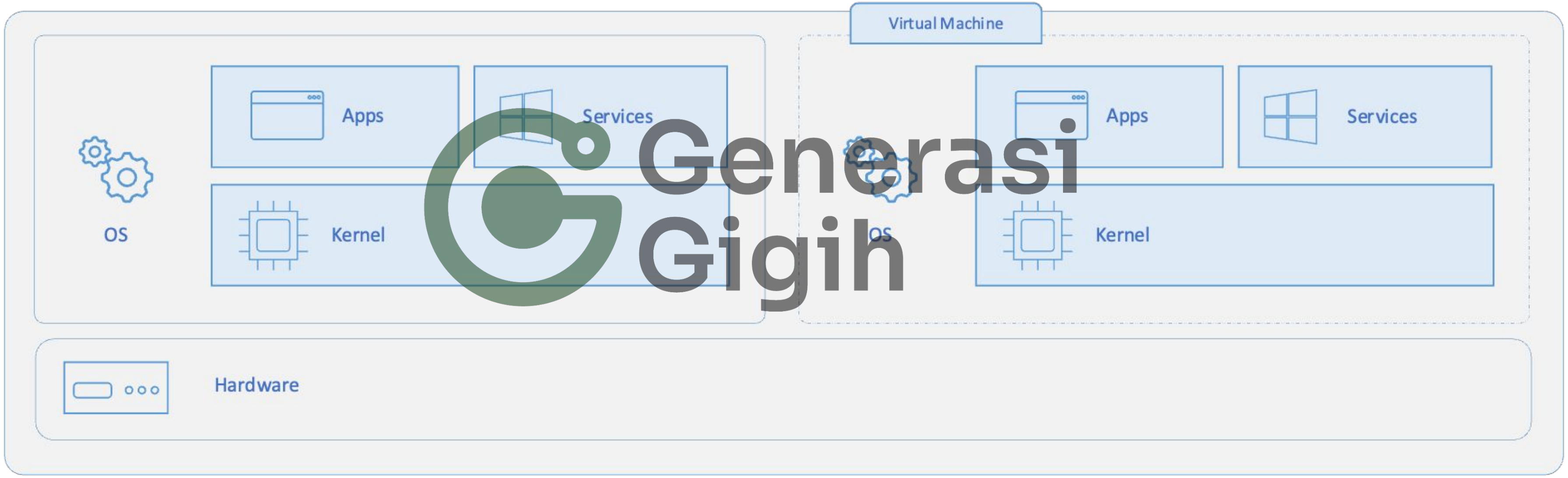
Deployment, Once Upon A Time



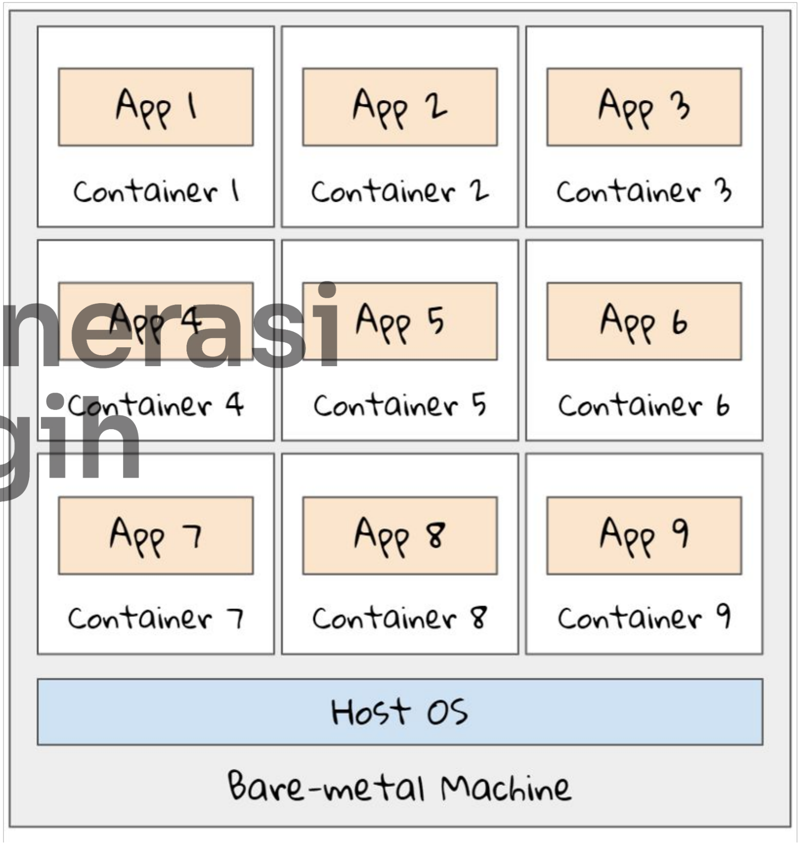
Along Came Virtual Machines



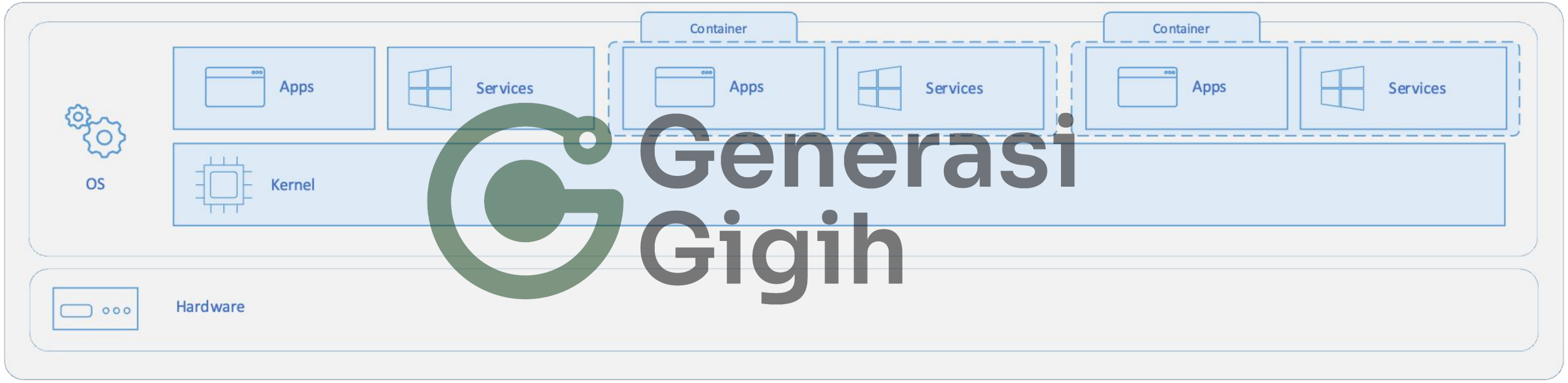
A Typical Virtual Machine Architecture



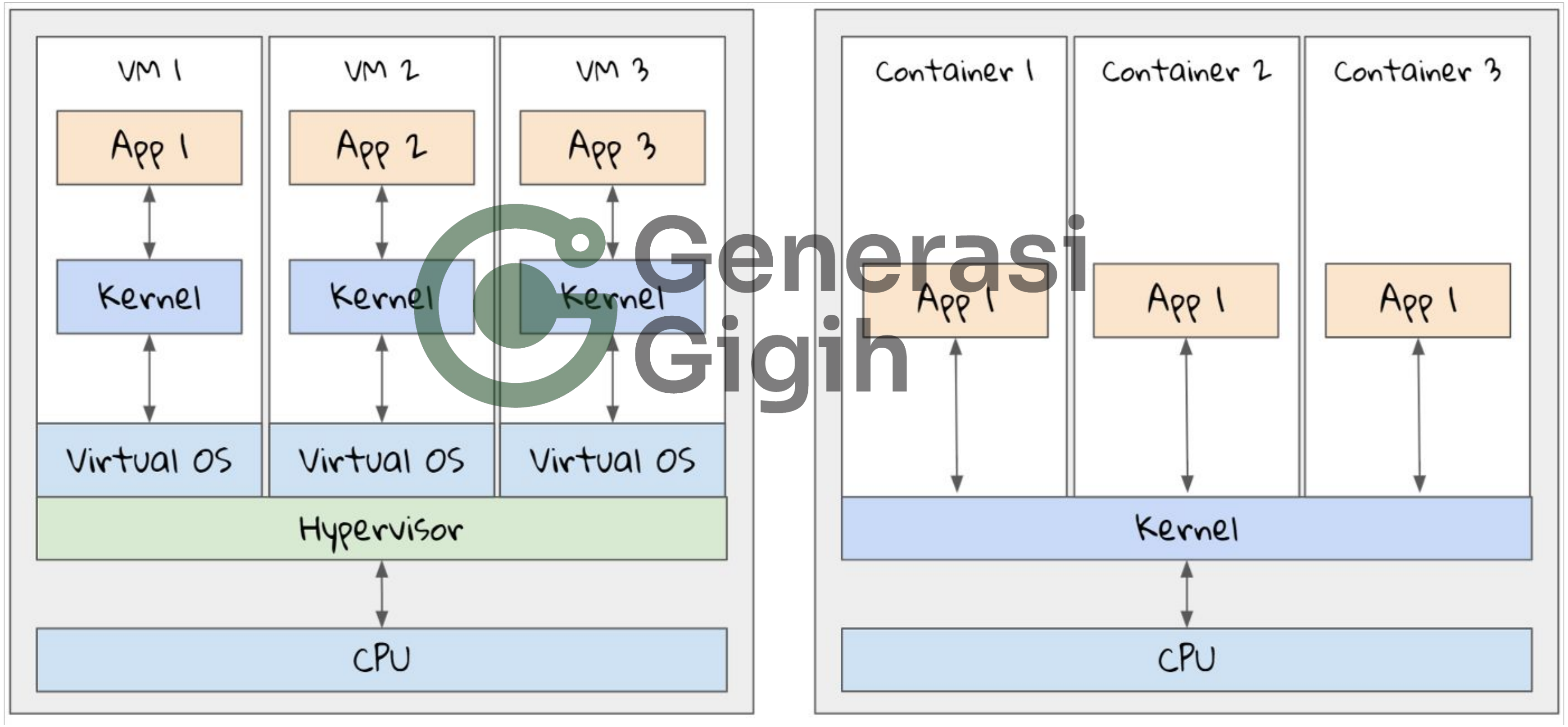
Then, Containers



A Typical Container Architecture



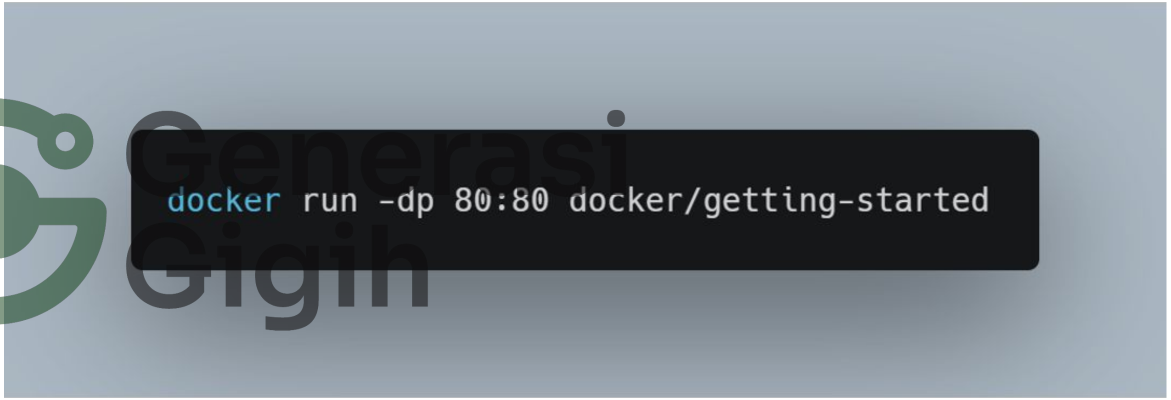
Virtual Machine vs Containers



Generasi Gigih

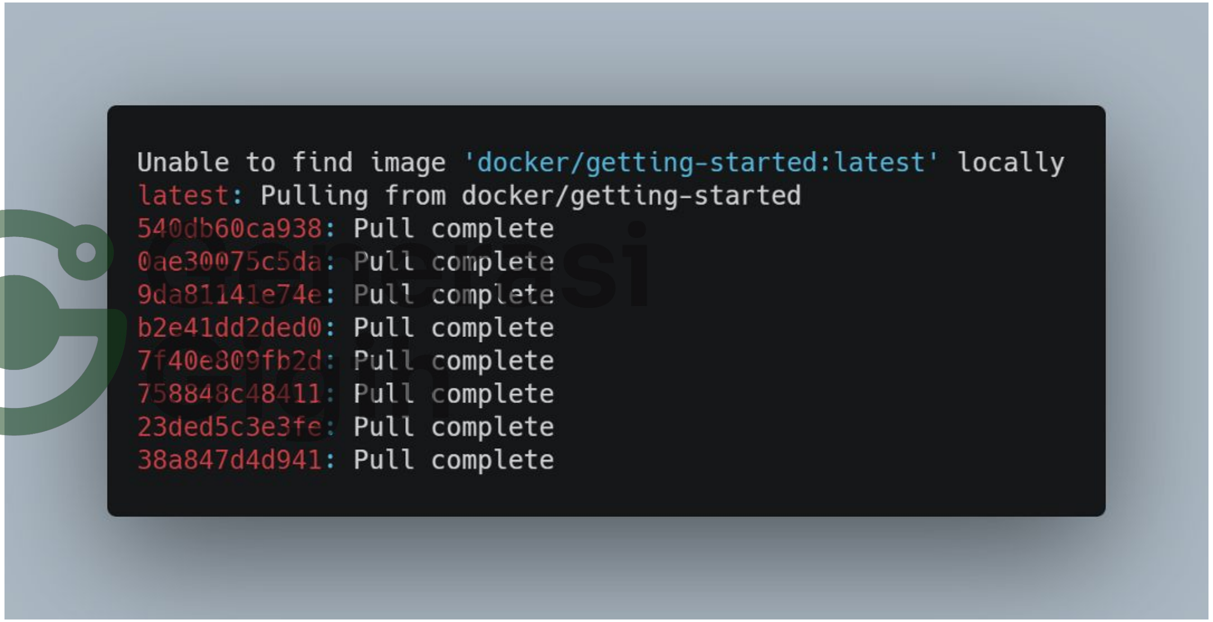
Intro to Docker

Hello, Docker!



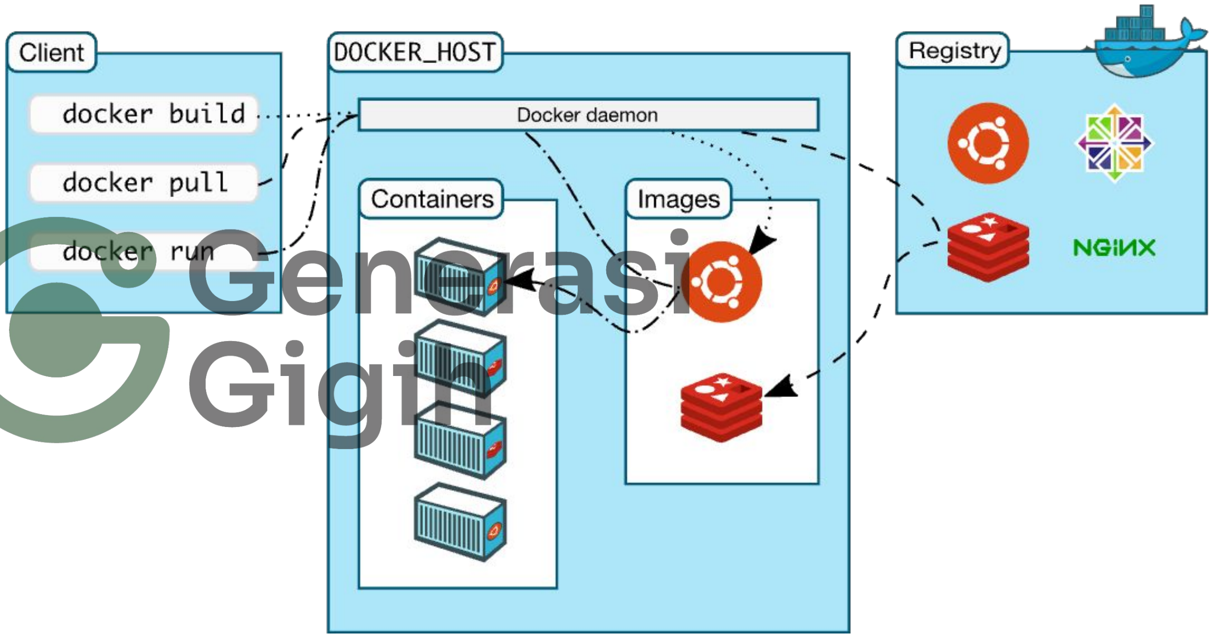
```
docker run -dp 80:80 docker/getting-started
```

Hello, Docker!

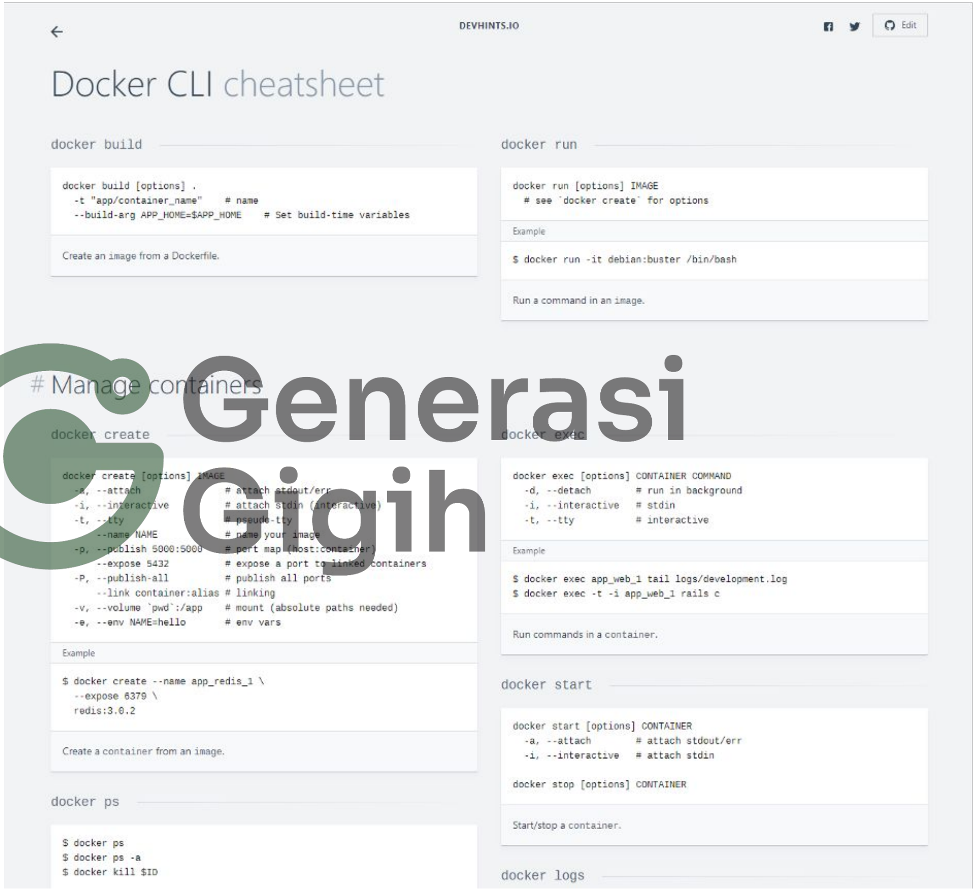


```
Unable to find image 'docker/getting-started:latest' locally
latest: Pulling from docker/getting-started
540db60ca938: Pull complete
0ae30075c5da: Pull complete
9da81141e74e: Pull complete
b2e41dd2ded0: Pull complete
7f40e809fb2d: Pull complete
758848c48411: Pull complete
23ded5c3e3fe: Pull complete
38a847d4d941: Pull complete
```

Docker Architecture



Cheat Sheet



DEVHINTS.IO

Edit

Docker CLI cheatsheet

docker build

```
docker build [options] .
  -t "app/container_name"    # name
  --build-arg APP_HOME=$APP_HOME    # Set build-time variables
```

Create an image from a Dockerfile.

docker run

```
docker run [options] IMAGE
  # see `docker create` for options
```

Example

```
$ docker run -it debian:buster /bin/bash
```

Run a command in an image.

Manage containers

docker create

```
docker create [options] IMAGE
  -a, --attach               # attach stdout/err
  -i, --interactive          # attach stdin (interactive)
  -t, --tty                  # pseudo-tty
      --name NAME            # name your image
  -p, --publish 5000:5000    # port map (host:container)
      --expose 5432          # expose a port to linked containers
  -P, --publish-all          # publish all ports
      --link container:alias # linking
  -v, --volume `pwd`:/app    # mount (absolute paths needed)
  -e, --env NAME=hello       # env vars
```

Example

```
$ docker create --name app_redis_1 \
  --expose 6379 \
  redis:3.0.2
```

Create a container from an image.

docker exec

```
docker exec [options] CONTAINER COMMAND
  -d, --detach        # run in background
  -i, --interactive   # stdin
  -t, --tty           # interactive
```

Example

```
$ docker exec app_web_1 tail logs/development.log
$ docker exec -t -i app_web_1 rails c
```

Run commands in a container.

docker start

```
docker start [options] CONTAINER
  -a, --attach        # attach stdout/err
  -i, --interactive   # attach stdin

docker stop [options] CONTAINER
```

Start/stop a container.

docker ps

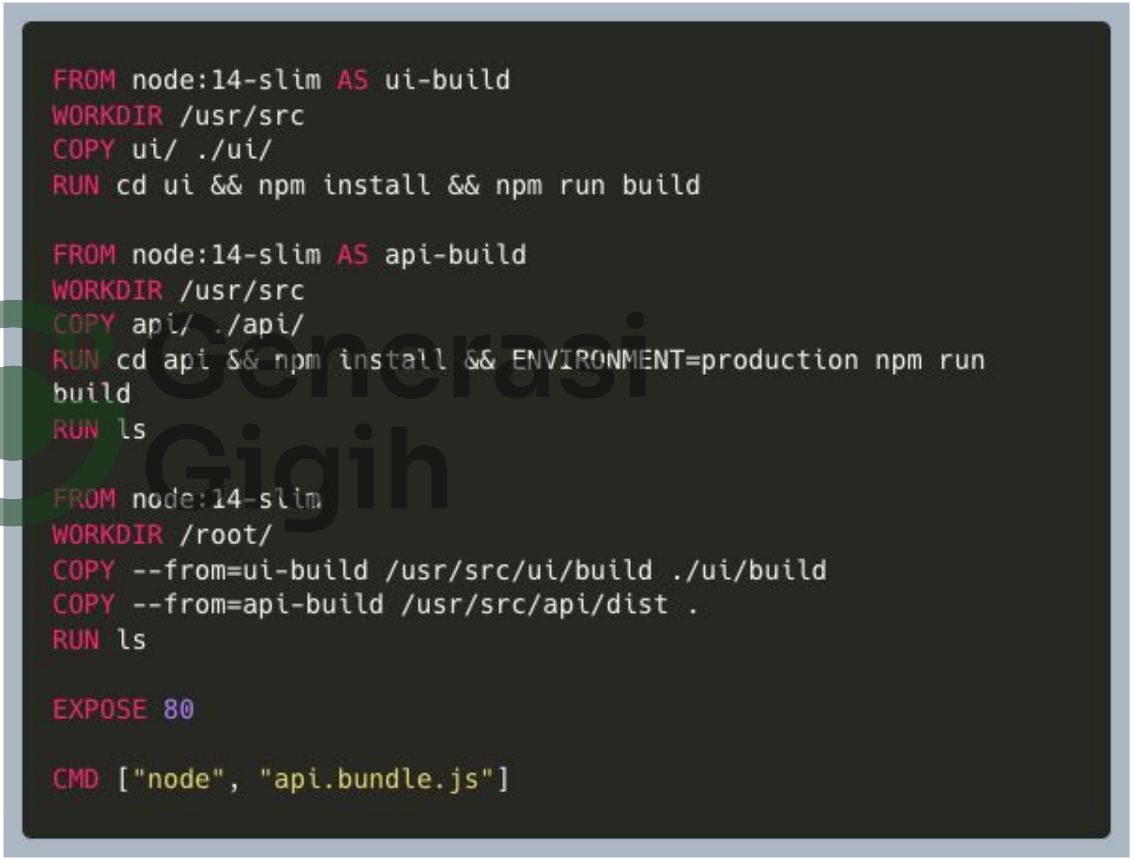
```
$ docker ps
$ docker ps -a
$ docker kill $ID
```

docker logs

Generasi Gigih

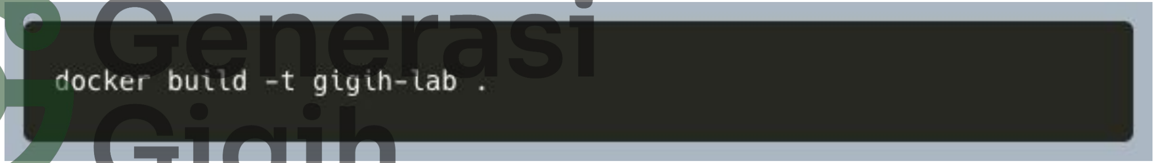
Deploying Our App to Container

Dockerfile



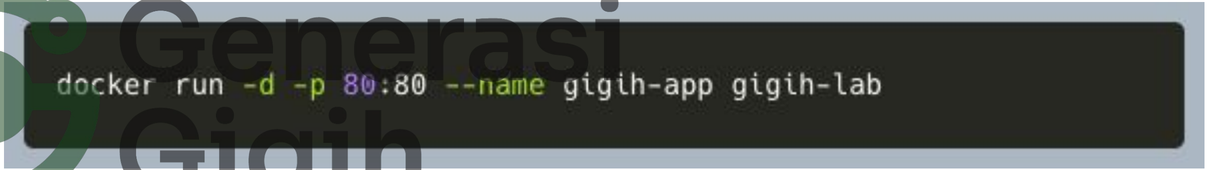
```
FROM node:14-slim AS ui-build
WORKDIR /usr/src
COPY ui/ ./ui/
RUN cd ui && npm install && npm run build

FROM node:14-slim AS api-build
WORKDIR /usr/src
COPY api/ ./api/
RUN cd api && npm install && ENVIRONMENT=production npm run
build
RUN ls

FROM node:14-slim
WORKDIR /root/
COPY --from=ui-build /usr/src/ui/build ./ui/build
COPY --from=api-build /usr/src/api/dist .
RUN ls

EXPOSE 80

CMD ["node", "api.bundle.js"]
```

Build The Image



```
docker build -t gigih-lab .
```

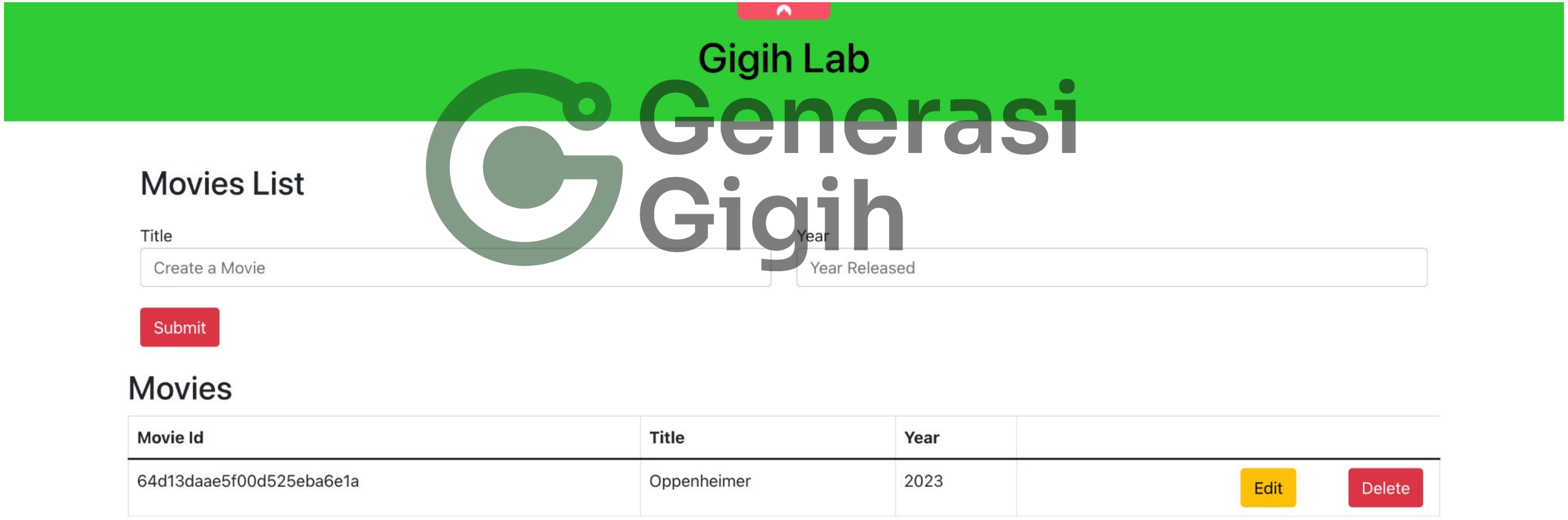
Run
The Container



```
docker run -d -p 80:80 --name gigih-app gigih-lab
```

Confirm It Works

Open http://localhost/ and check if you see the following:



Gigih Lab

Movies List

Title: Create a Movie

Year: Year Released

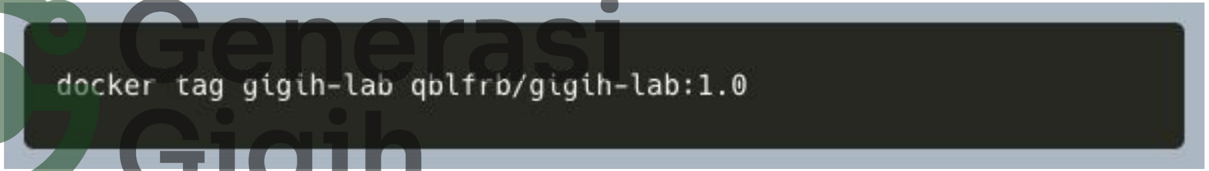
Movies

Movie Id	Title	Year	
64d13daae5f00d525eba6e1a	Oppenheimer	2023	Edit Delete

Generasi Gigih

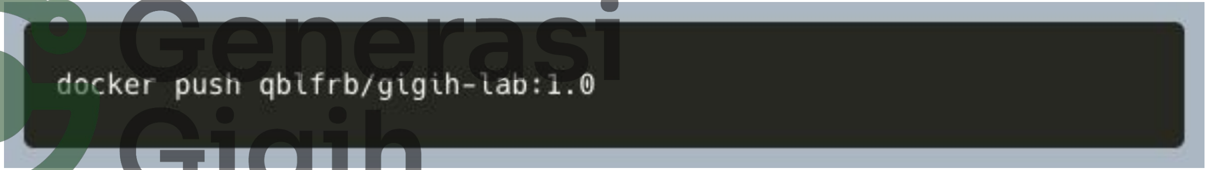
Store Our Image to Docker Hub

Tag The Image



```
docker tag gigih-lab qblfrb/gigih-lab:1.0
```

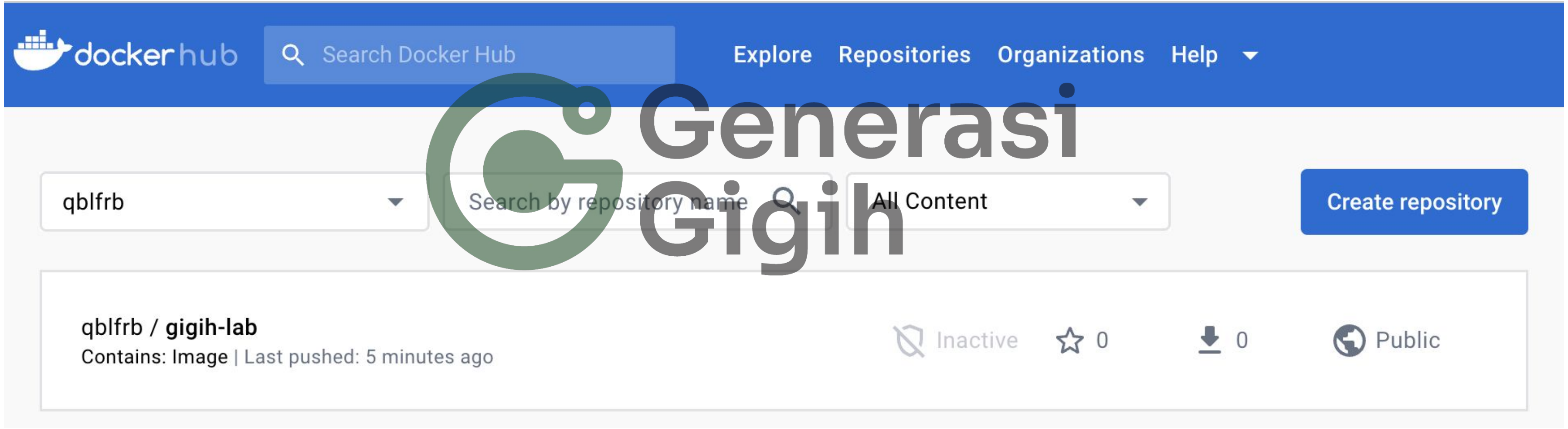
Push The Image



```
docker push qblfrb/gigih-lab:1.0
```

Verify It's Stored in Docker Hub

Go to https://hub.docker.com/ and check if your image is there:



qblfrb / gigih-lab
Contains: Image | Last pushed: 5 minutes ago

Inactive ☆ 0 ⤓ 0 Public

Generasi Gigih

Showcase Time!

Generasi Gigih

Q&A!

Generasi Gigih

Finally, Let’s Wrap Up!

See you in the next session!