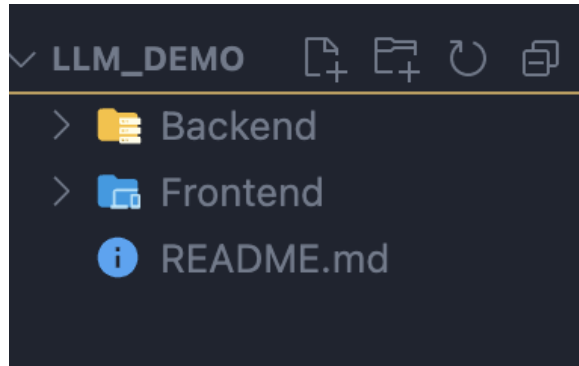


Project Deployment of GCP cloud run

LLM-BI

Steps for Setting up Locally :

⇒ File Structure



For backend

1. Go to directory:

```
cd Backend
```

2. Create a python virtual environment using venv names (env):

```
python3 -m venv env
```

3. Activating the virtual environment

```
# Linux / Unix  
source /env/bin/activate
```

```
# Windows  
env/Scripts/activate
```

4. Installing necessary dependencies :

```
pip install -r requirements.txt
```

5. Running the server

```
uvicorn main:app --reload
```

FastAPI 0.1.0 OAS 3.1
/openapi.json

No operations defined in spec!

Now the Backend is fully Up.

For Frontend :

1. Go to frontend directory

```
cd ../Frontend
```

2. Install Node

```
Brew install node  
(make sure brew is installed)
```

3. Install node modules

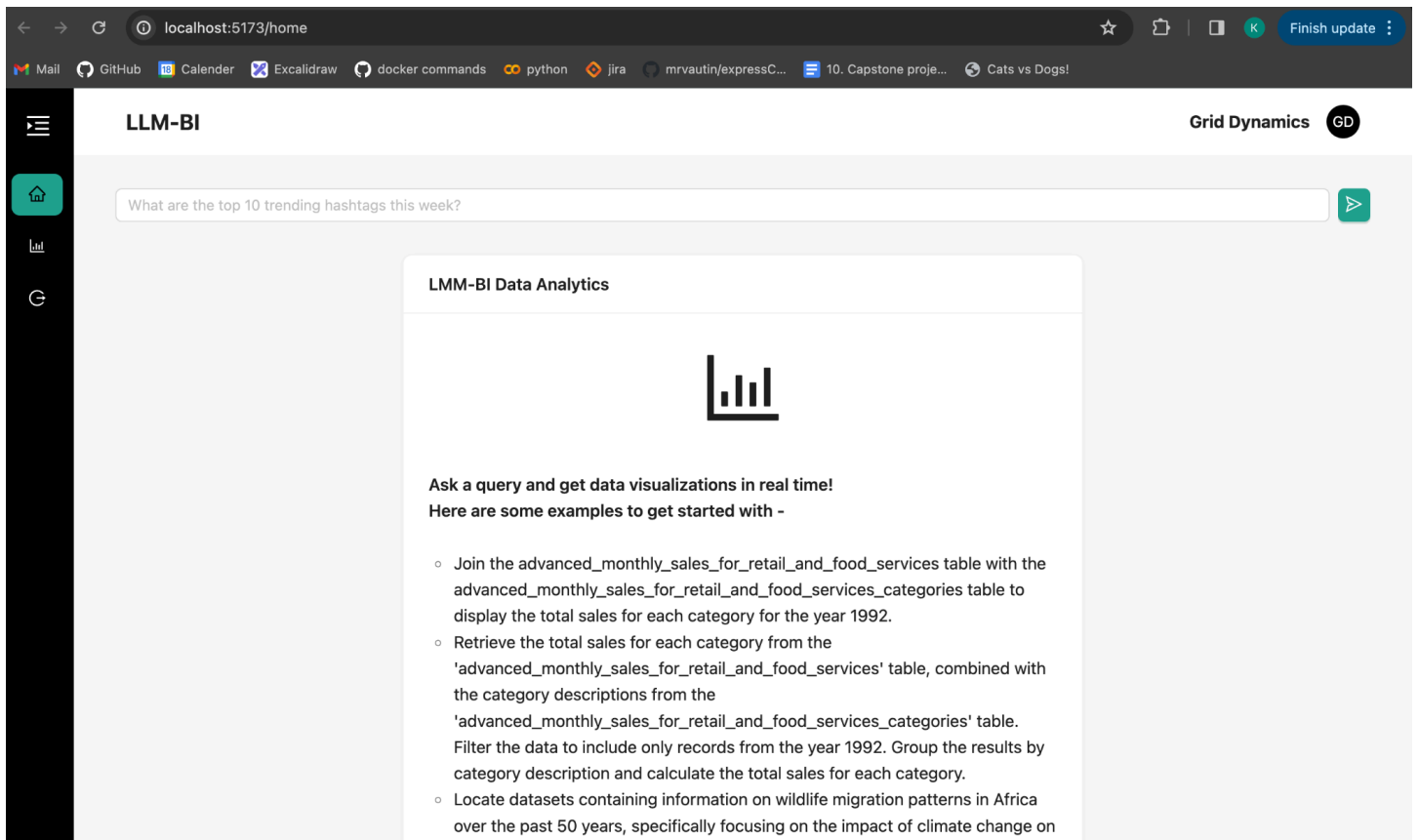
```
npm i
```

4. Run the server

```
npm run dev
```

5. For building

```
npm run build
```



Now the project is fully deployed locally.

For deploying it in Google Cloud (Cloud Run) First we have to make the docker image for both frontend and backend separately.

STEPS :

1. Install docker on your machine by following the instructions given in this doc <https://docs.docker.com/engine/install/>
2. Make Dockerfile for both frontend and backend by going in their respective directories.

For Backend

```
1 FROM python
2
3 WORKDIR ./app
4
5 COPY . .
6
7 RUN pip install -r requirements.txt
8
9 CMD uvicorn main:app --port=8000 --host=0.0.0.0
```

For frontend

```

1  FROM node as build
2
3  WORKDIR /app
4
5  COPY . .
6
7  RUN npm i --force
8
9  CMD ["npm", "run", "build"]
10
11 FROM nginx
12
13 COPY --from=build /app/dist /usr/share/nginx/html
14

```

3. Now we have make docker images -
Run following commands by going to their respective directories.

```
docker build . -t backend:v1
```

```
docker build . -t frontend:v1
```

```
docker buildx build --platform linux/amd64 -t frontend .
docker buildx build --platform linux/amd64 -t backend .
```

You will something this type output in terminal

```

[+] Building 14.9s (14/14) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 196B
=> [internal] load metadata for docker.io/library/nginx:latest
=> [internal] load metadata for docker.io/library/node:latest
=> [auth] library/nginx:pull token for registry-1.docker.io
=> [auth] library/node:pull token for registry-1.docker.io
=> [build 1/4] FROM docker.io/library/node@sha256:b9ccc4aca32eebf124e0ca0fd573dacffba2b9236987a1d4d2625ce3c162ecc8
=> [internal] load build context
=> => transferring context: 2.96MB
=> [stage-1 1/2] FROM docker.io/library/nginx@sha256:6db391d1c0cfb30588ba0bf72ea999404f2764feb0f1f196acd5867ac7efa7e
=> CACHED [build 2/4] WORKDIR /app
=> [build 3/4] COPY . .
=> [build 4/4] RUN npm i --force
=> CACHED [stage-1 2/2] COPY --from=build /app/dist /usr/share/nginx/html
=> exporting to image
=> => exporting layers
=> => writing image sha256:c8be3362c68239843abea8c59ac35d6df737f08026b41d9c1d0eb8d24ab5ba0c
=> => naming to docker.io/library/frontend:v3

```

View build details: [docker-desktop://dashboard/build/desktop-linux/desktop-linux/4fakzqm2g2wtup1ku9jdfcma2](#)

What's Next?

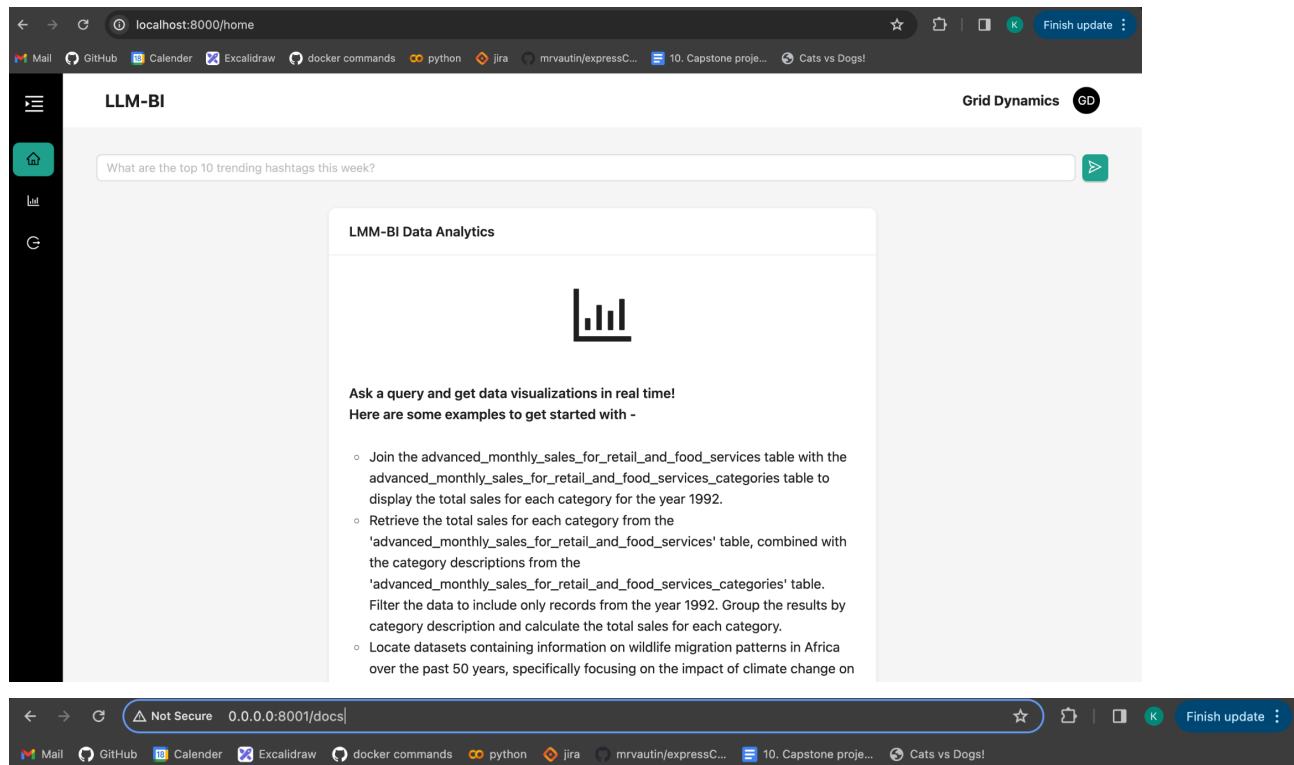
View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

Now we can run these images by making their container and port binding with the host.

```
# for frontend
docker run -p 8000:80 frontend:v1
```

```
# for backend
docker run -p 8001:8000 backend:v1
```

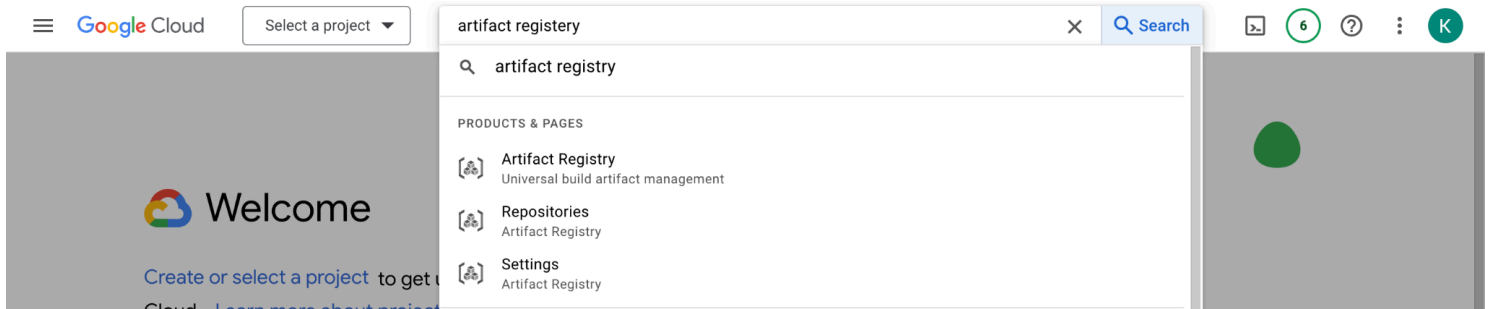
Now we can see these containers deployed on Docker



Now here come GCP part ⇒

For deploying in it google cloud platform first we have to push the container of both frontend and backend to the artifact registry.

Search on GCP platform “artifact registry click on that”



By going to the artifact registry.

⇒ **Click on “Create repository”**
Interface will come like this

← Create repository

Name *

llm-demo

Format

☒ Docker

☐ Maven

☐ npm

☐ Python

☐ Apt

☐ Yum

☐ Kubeflow Pipelines

☐ Go

Mode

☒ Standard

☐ Remote

☐ Virtual

Location type

☒ Region

☐ Multi-region

Region *

us-central1 (Iowa)

Description

Labels

Provide docker repository name and region and leave rest of the settings.

Once the repo is created we have to push the docker images which we have built earlier in our machine for this we need to setup **gcloud** locally.

1. Follow the instructions written in this doc
<https://cloud.google.com/sdk/docs/install>

2. For authorize yourself run :
`gcloud auth login`

Page will open u need to authenticate yourself

3. For authorize yourself run :
`gcloud auth login`

4. Configure docker

Run the following command to configure `gcloud` as the credential helper for the Artifact Registry domain associated with this repository's location:

```
gcloud auth configure-docker \  
us-central1-docker.pkg.dev
```

5. Necessary step tag the images with given command

```
docker tag {image-name}  
us-central1-docker.pkg.dev/gd-gcp-internship-cd/llm-demo/frontend
```

```
docker tag {image-name}  
us-central1-docker.pkg.dev/gd-gcp-internship-cd/llm-demo/backend
```

6. Push the docker images

```
docker push us-central1-docker.pkg.dev/gd-gcp-internship-cd/llm-demo/frontend  
docker push us-central1-docker.pkg.dev/gd-gcp-internship-cd/llm-demo/backend
```

After pushing interface will look like this

[←](#) Images for llm-demo [DELETE](#) [EDIT REPOSITORY](#) [SETUP INSTRUCTIONS](#)



us-central1-docker.pkg.dev > gd-gcp-internship-cd > llm-demo

Repository Details

Format	Docker
Type	Standard


[SHOW MORE](#)

[Filter](#) Enter property name or value

<input type="checkbox"/>	Name ↑	Created	Updated
<input type="checkbox"/>	 backend	11 hours ago	11 hours ago
<input type="checkbox"/>	 frontend	11 hours ago	11 hours ago

Click on **Deploy on Cloud Run**

[Filter](#) Enter property name or value

<input type="checkbox"/>	Name	Description	Tags	Created	Updated ↓	
<input type="checkbox"/>	 8d4d0bdbc78f		latest	11 hours ago	11 hours ago	<div><div>Show pull command</div><div>Edit tags</div><div>Deploy to Cloud Run</div><div>Deploy to GKE</div><div>Deploy to GCE</div></div>

Just change the container port to 80
And click on create.



A service exposes a unique endpoint and automatically scales the underlying infrastructure to handle incoming requests. Service name and region cannot be changed later.



Artifact Registry



Docker Hub



GitHub



Deploy one revision from an existing container image



Continuously deploy from a repository

Container image URL

us-central1-docker.pkg.dev/gd-gcp-internship-cd/llm-demo/frontend:lates [SELECT](#)

TEST WITH A SAMPLE CONTAINER

Should listen for HTTP requests on \$PORT and not rely on local state. [How to build a container?](#)

Configure

Service name *

frontend

Region *

us-central1 (Iowa)

[How to pick a region?](#)

Authentication *

You do not have the permission to change IAM policy on Cloud Run services. You need the `run.services.setIamPolicy` permission, which is included in the [Cloud Run Admin IAM](#)

[CREATE](#)[CANCEL](#)

After this it takes some time to deploy. . . .

Final Deployed Screenshot :

Cloud Run

Service details

EDIT & DEPLOY NEW REVISION

SET UP CONTINUOUS DEPLOYMENT

LEARN

frontend

Region: us-central1

URL: <https://frontend-mcwhdaqkga-uc.a.run.app>

Min instances: 0

METRICS

SLOS

LOGS

REVISIONS

NETWORKING

SECURITY

TRIGGERS

INTEGRATIONS

PREVIEW

YAML

Revisions

MANAGE TRAFFIC

Filter

Filter revisions

	Name	Traffic
<input checked="" type="radio"/>	frontend-00002-2sn	100% (to latest)
<input type="radio"/>	frontend-00001-rhv	0%

frontend-00002-2sn

Deployed by krishnasahk8356@gmail.com using Cloud Console

CONTAINERS

VOLUMES

NETWORKING

SECURITY

YAML

General

CPU allocation

CPU is only allocated during request processing

Startup CPU boost

Enabled

Concurrency

80

Request timeout

300 seconds

Execution environment

Default

Autoscaling

Max instances

100

frontend-mcwhdaqkga-uc.a.run.app/home


Finish update

LLM-BI

Grid Dynamics GD

What are the top 10 trending hashtags this week?

LMM-BI Data Analytics



Ask a query and get data visualizations in real time!

Here are some examples to get started with -

- Join the advanced_monthly_sales_for_retail_and_food_services table with the advanced_monthly_sales_for_retail_and_food_services_categories table to display the total sales for each category for the year 1992.
- Retrieve the total sales for each category from the 'advanced_monthly_sales_for_retail_and_food_services' table, combined with the category descriptions from the 'advanced_monthly_sales_for_retail_and_food_services_categories' table. Filter the data to include only records from the year 1992. Group the results by category description and calculate the total sales for each category.
- Locate datasets containing information on wildlife migration patterns in Africa over the past 50 years, specifically focusing on the impact of climate change on migration routes of endangered species such as elephants and rhinoceroses.