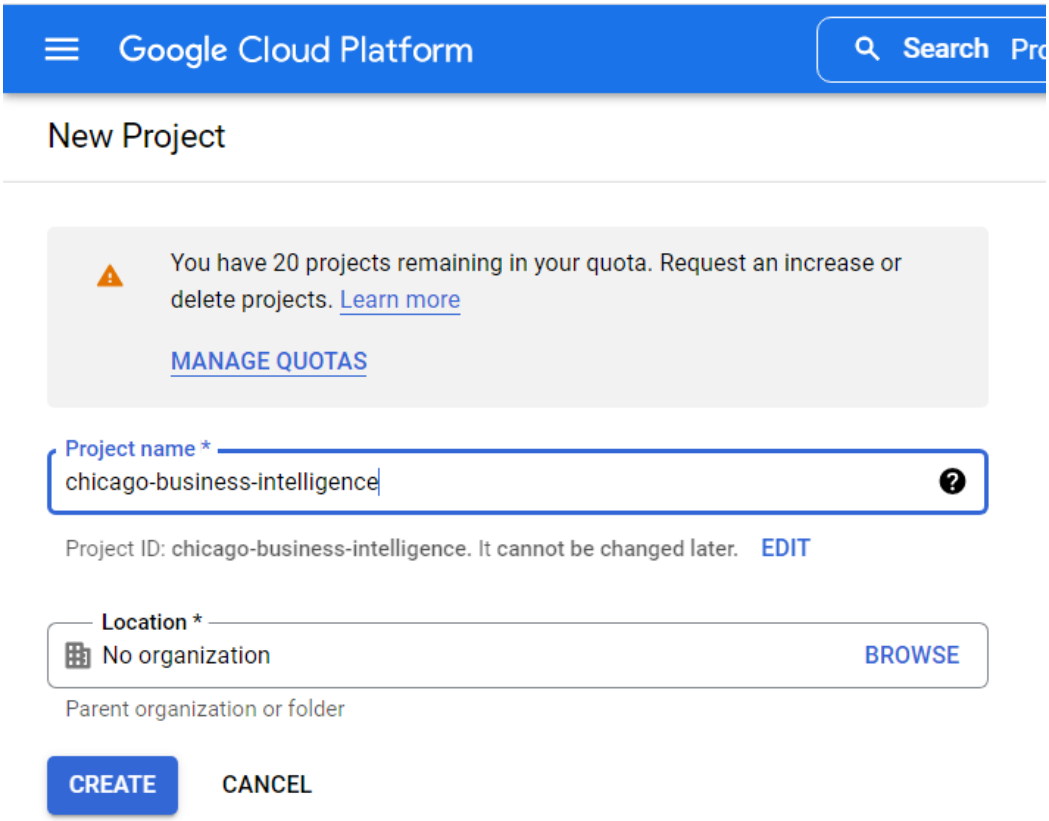
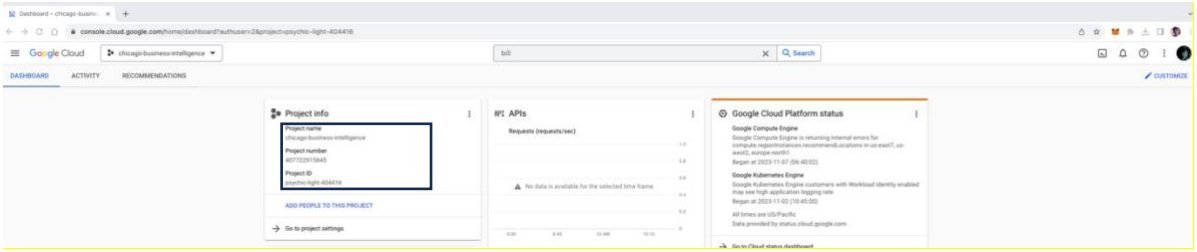
Steps to Deploy Go Microservice for Chicago Business Intelligence on GCP

Step1: Initial Setup for Google Cloud Platform

- Install the google cloud CLI on your local machine.
- Create a new project on your google cloud console. Make a note of the project id and project Name.





- After creating a project on Google Cloud Console execute "**gcloud init**" command on your local machine and select the project created above when prompted.

```
Your current project has been set to: [chicago-business-intelligence].
```

Step 2: Postgres database Setup

- Create database instance of postgres using the following command.
  "**gcloud sql instances create mypostgres --database-version=POSTGRES_14 --cpu=2 --memory=7680MB --region=us-central**"

```
❯ gcloud sql instances create mypostgres --database-version=POSTGRES_14 --cpu=2 --memory=7680MB --region=us-central
API [sqladmin.googleapis.com] not enabled on project [psychic-light-404416]. Would you like to enable and retry (this will take a few minutes)? (y/N)?  y

Enabling service [sqladmin.googleapis.com] on project [psychic-light-404416]...
Operation "operations/acat.p2-407722915845-aec12022-7284-47cf-9d6f-6729f60c583a" finished successfully.
Creating Cloud SQL instance for POSTGRES_14...done.
Created [https://sqladmin.googleapis.com/sql/v1beta4/projects/psychic-light-404416/instances/mypostgres].
NAME: mypostgres
DATABASE_VERSION: POSTGRES_14
LOCATION: us-central1-b
TIER: db-custom-2-7680
PRIMARY_ADDRESS: 34.170.237.130
PRIVATE_ADDRESS: -
STATUS: RUNNABLE
~/Documents/CS588 TA/3. GCP-Docker-PG-Go_Source_code
```

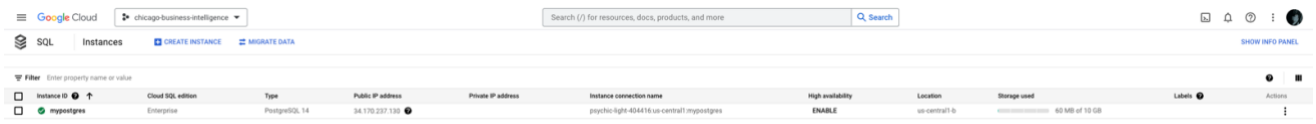
- Create sql users on the database instance using the following command.
  “gcloud sql users set-password postgres --instance=mypostgres --password=root”

```
❯ gcloud sql users set-password postgres --instance=mypostgres --password=root
Updating Cloud SQL user...done.
~/Documents/CS588 TA/3. GCP-Docker-PG-Go_Source_code
```

- Create a database for our microservice using the following command.
  “gcloud sql databases create chicago_business_intelligence --instance=mypostgres”

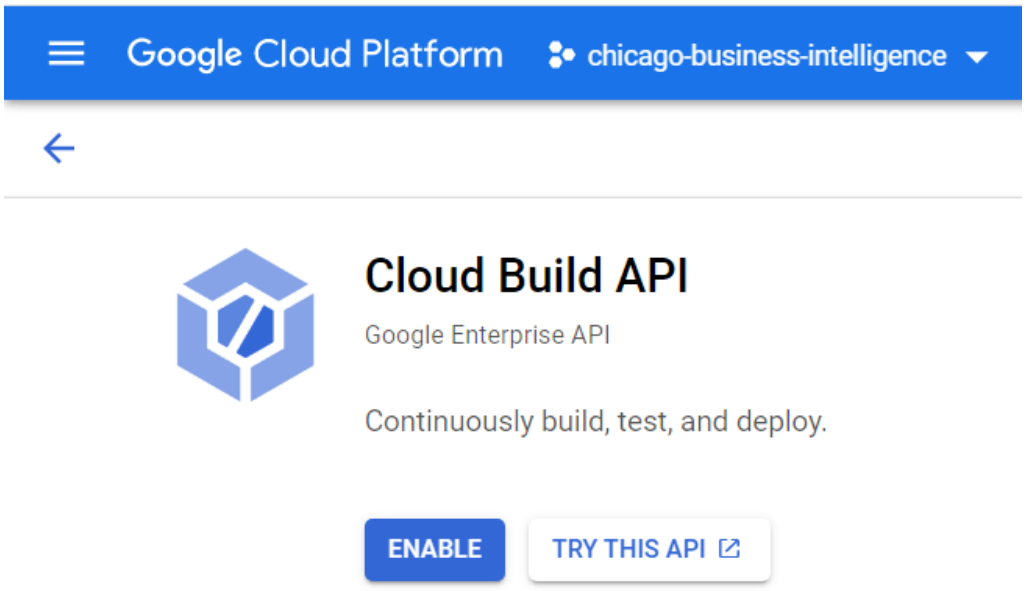
```
❯ gcloud sql databases create chicago_business_intelligence --instance=mypostgres
Creating Cloud SQL database...done.
Created database [chicago_business_intelligence].
instance: mypostgres
name: chicago_business_intelligence
project: psychic-light-404416
~/Documents/CS588 TA/3. GCP-Docker-PG-Go_Source_code
```

- Open Google Cloud console, search for SQL and confirm that database instance is up and running

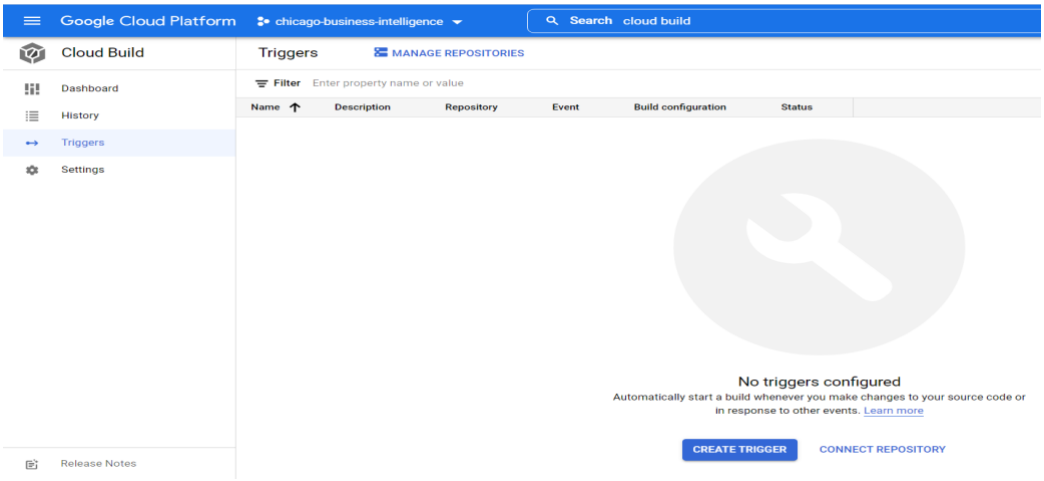


Step 3: Setting up continuous deployment using cloud build.

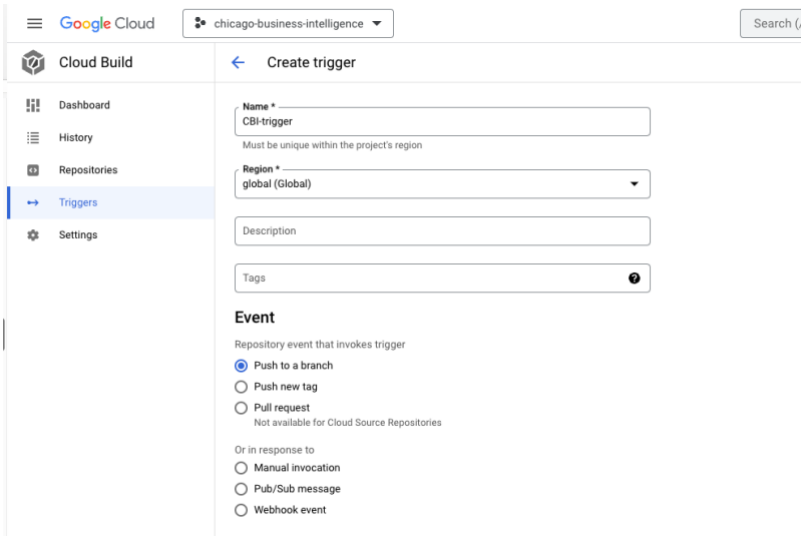
- Create a repository on GitHub to store the source code for our project.
- Open Google Cloud Console, Search for Cloud build API, and Enable it for your project



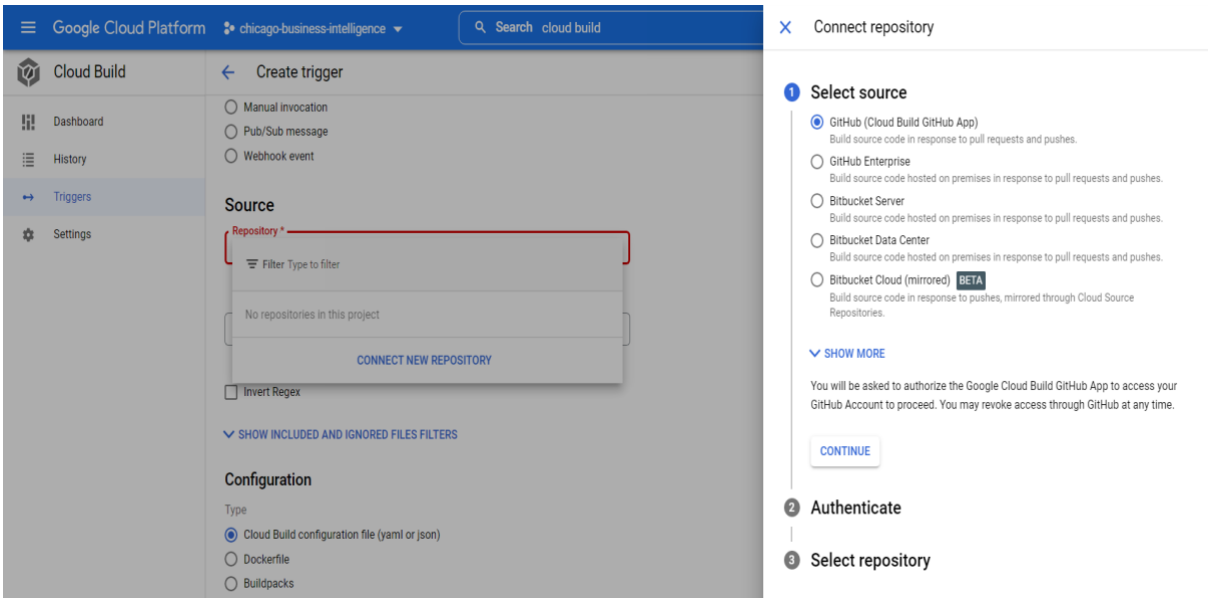
- After the API is enabled, click on the create trigger button.



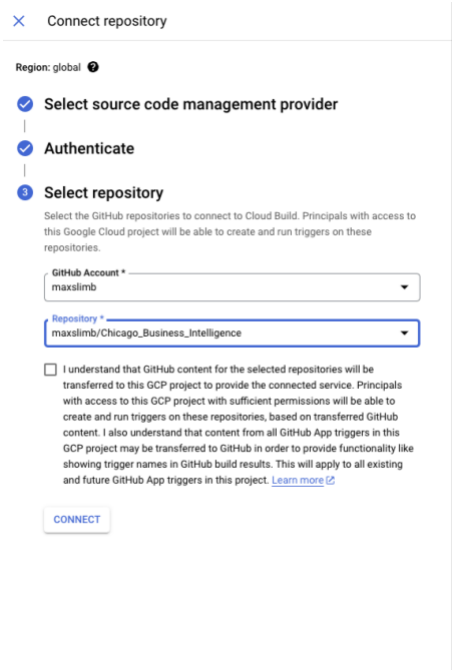
- Fill the details for the trigger as shown in the below images.



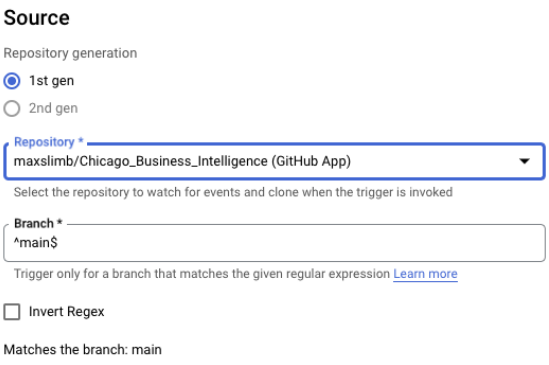
- Click on connect repository, select github and authenticate.



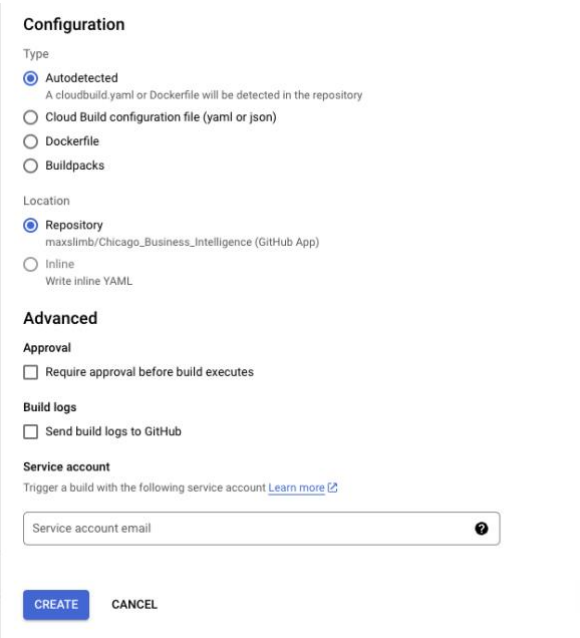
- After authentication select the repository created for Chicago business intelligence.



- Select the repository after connecting the project.

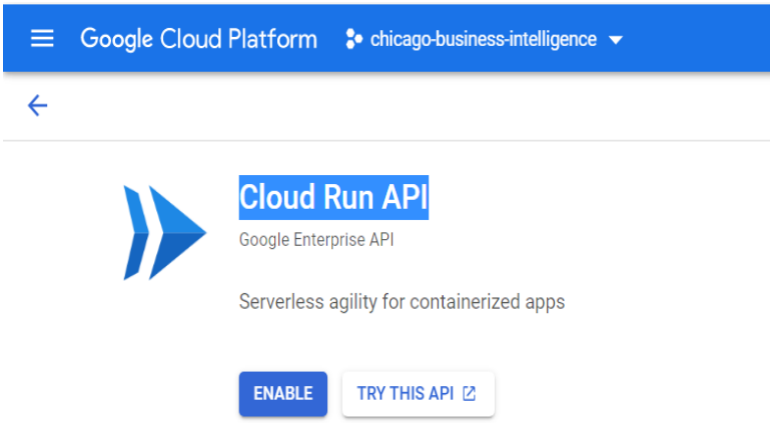


- Click on Create to create the trigger.

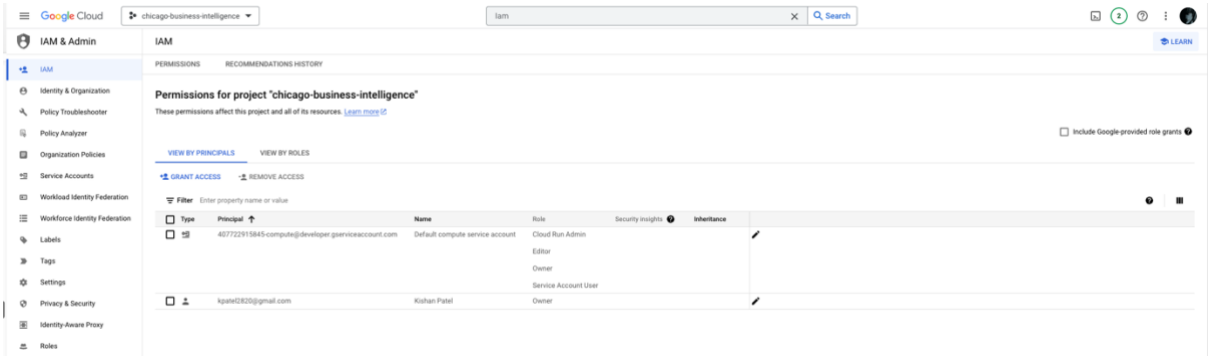


Step 4: Setting up the containers for Go-microservice and Pgadmin

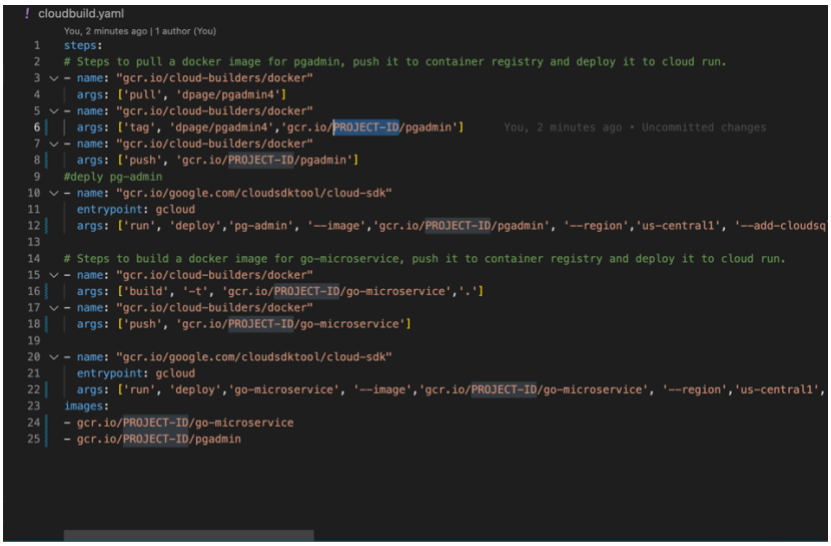
- Enable Cloud Run API for your project.



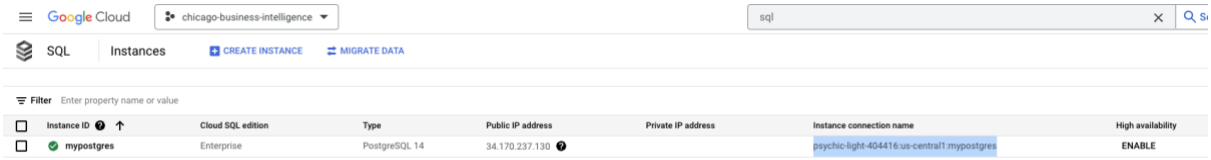
- Go to IAM page and make sure all the required roles are enabled for the project.



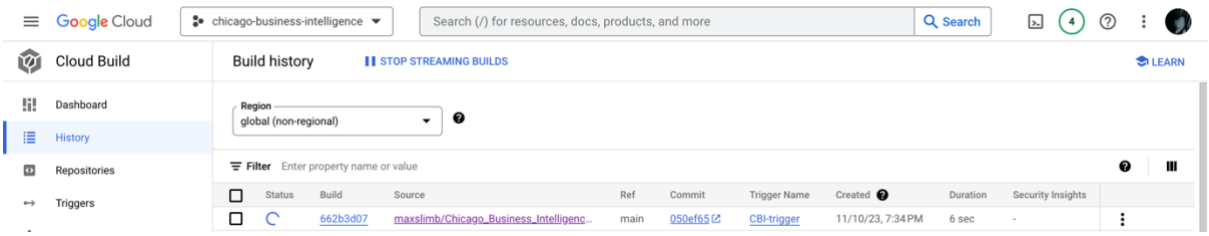
- The images for the go microservice and pgadmin are created with the help of cloudbuild.yaml file, you have to **update your Project ID** in various places in the cloudbuild.yaml file



- Go to the postgres instance created in the previous steps and copy the instance connection name.

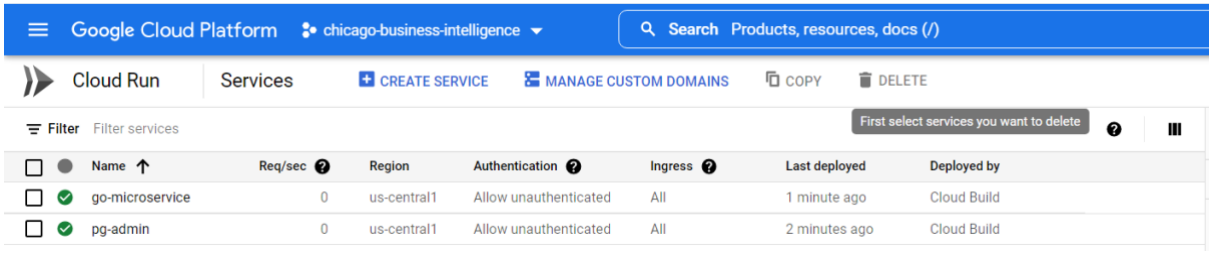


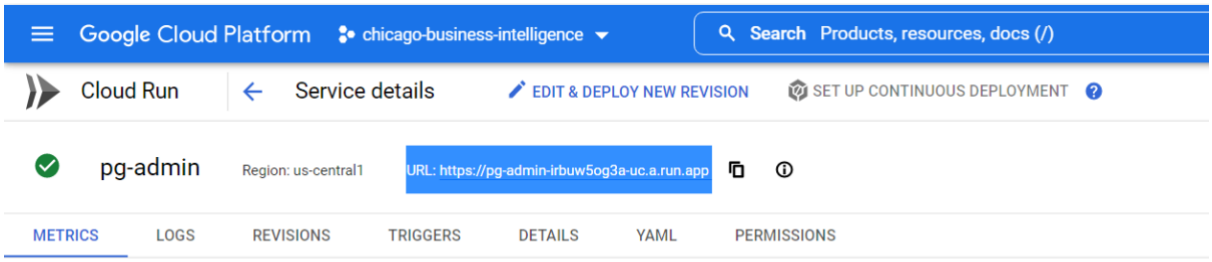
- Update line 198 of your main.go source code file and update the connection string with your Instance connecton name as shown below.
- connectionName := "**psychic-light-404416:us-central1:mypostgres"**
- Update the line 285 with the Geocoder API-KEY
- Push the source code along with the cloudbuild.yaml file to the repository created in the above steps
- A build is triggered in the cloud build immediately after pushing the code to GitHub.



- Wait for the build to be complete. Build logs can be viewed by clicking on the build id.

- Go to Cloud Run, click on pgadmin, copy the highlighted URL





Open the URL in a Browser and Login to pgadmin to validate that tables are created.