Build and Deploy Documentation

SightScan | AMOS Project Team 2

SERVICES

Crawler

The crawler is responsible of crawling images for a given city and inserting them into the data warehouse.

Environment Variables

PGHOST → ****URL of the data warehouse

PGDATABASE → ****database name you want to access

PGPORT → **port** of the data warehouse

PGUSER → **user** of the data warehouse

PGPASSWORD → **password** of the data warehouse

apikey → **open maps api key**

maps_key → **open maps key**

Docker

- `cd ./amos/crawler`
- `docker build -t crawler:v1 .`
- `docker run -e PGHOST=<PGHOST> -e PGDATABASE=<PGDATABASE> -e PGPORT=<PGPORT> -e PGUSER=<PGUSER> -e PGPASSWORD=<PGPASSWORD> -e apikey=<apikey> -e maps_key=<maps_key> crawler:v1 "city_name"`

Locally

1. `cd ./amos/crawler`
2. `pip install -r requirements.txt`
3. `python main.py --<argument>=<argument_value>`

- **Command Line Arguments**
  --skip=true Skips keyword if downloaded directory already exists. This is needed when re-downloading.

--threads=4 Number of threads to download.
--google=true Download from google.com (boolean)
--full=false Download full resolution image instead of thumbnails (slow)
--face=false Face search mode
--no_gui=auto No GUI mode. (headless mode) Acceleration for full_resolution mode, but unstable on thumbnail mode. Default: "auto" - false if full=false, true if full=true (can be used for docker linux system)
--limit=0 Maximum count of images to download per site. (0: infinite)
--no_driver=false Whether a driver should be used
--location='Berlin' The location keywords need to be found for.
--sights_limit The limit of sights to be found by the collector api

IMPORTANT

You need an .env-file in the **respective directory** to run it **locally**. Otherwise the env-variables won't be readable for the program.

Data Mart Refresher

The Data Mart Refresher is responsible for three things. First checking if enough labeled images are available in the DWH if there are it starts a model training process, if not it starts the image labeling process and updating our data marts.

Data Marts can be seen in the technical documentation

Environment Variables

DATA_MART_MTS_ENDPOINT_URL → URL of the data warehouse.

DATA_MART_ILS_ENDPOINT_URL → Database name you want to access.

DATA_MART_REFRESH_DATA_MARTS_EVERY_SECONDS → Interval of seconds to refresh the data marts

DATA_MART_ENABLE_MODEL_TRAINING_EVERY_SECONDS → Interval of seconds to check and trigger Model Training Service

DATA_MART_ENABLE_LABELLING_REQUESTS_EVERY_SECONDS → Interval of seconds to check and trigger Image Labeling Service

PGPORT → **Port** of the data warehouse

PGUSER → **User** of the data warehouse

PGPASSWORD → **Password** of the data warehouse

Docker

1. `cd ./amos/data_mart_refresher`
2. `docker build -t data_mart_refresher:v1 .`
3. `docker run -e DATA_MART_MTS_ENDPOINT_URL=<DATA_MART_MTS_ENDPOINT_URL> -e DATA_MART_ILS_ENDPOINT_URL=<DATA_MART_ILS_ENDPOINT_URL> -e DATA_MART_REFRESH_DATA_MARTS_EVERY_SECONDS=5 -e DATA_MART_ENABLE_MODEL_TRAINING_EVERY_SECONDS=10 -e DATA_MART_ENABLE_LABELLING_REQUESTS_EVERY_SECONDS=10 -e PGHOST=<PGHOST> -e PGDATABASE=<PGDATABASE> -e PGPORT=<PGPORT> -e PGUSER=<PGUSER> -e PGPASSWORD=<PGPASSWORD> data_mart_refresher:v1`

IMPORTANT

If you have environment variables in an .env-file you can remove the variables from the docker command.

Locally

1. `cd ./amos/data_mart_refresher`
2. `pip install -r requirements.txt`
3. `python ./cron_jobs/main.py`

IMPORTANT

You need an .env-file in the **respective directory** to run it **locally**. Otherwise the env-variables won't be readable for the program.

Postgres

Is an open source object-relational database, which runs our data warehouse.

Docker

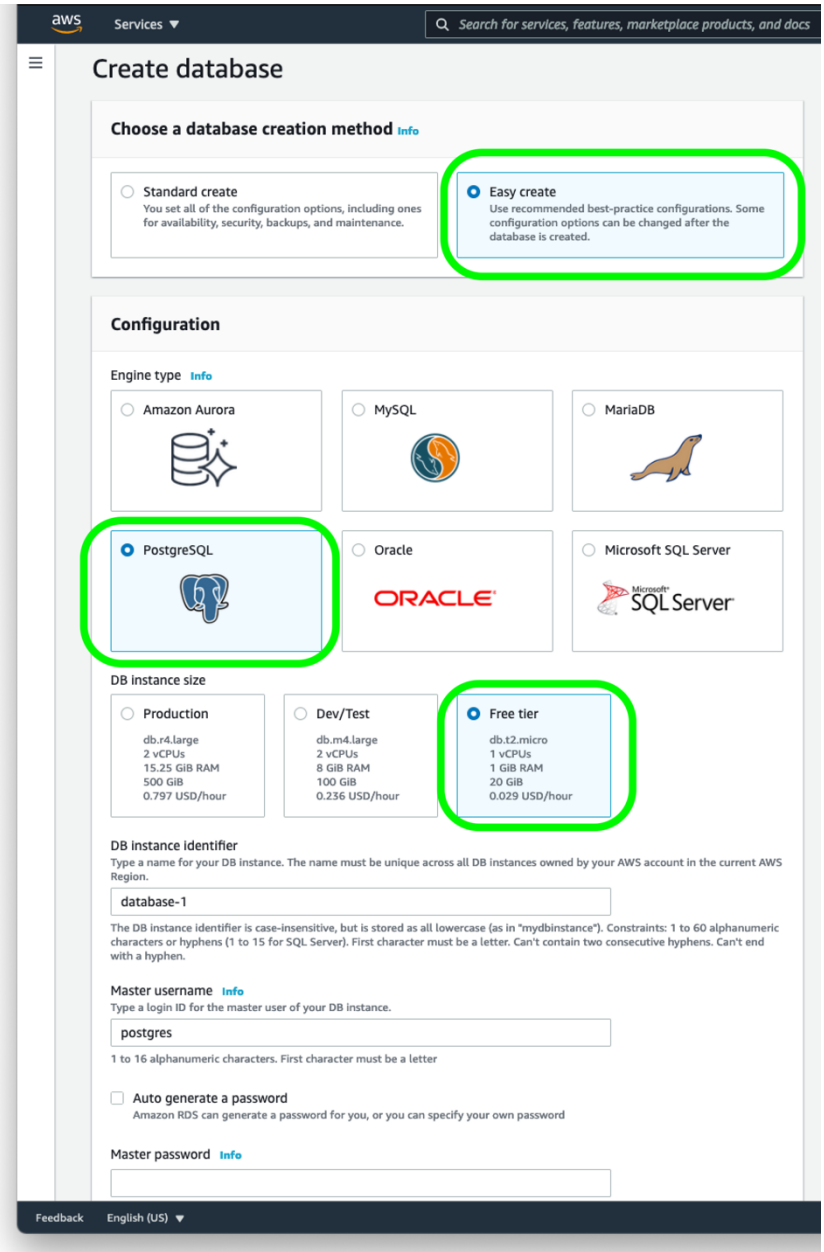
1. `cd ./postgres`
2. `docker build -t postgres:v1 .`
3. `docker run postgres:v1 -p 5432:5432`

Important

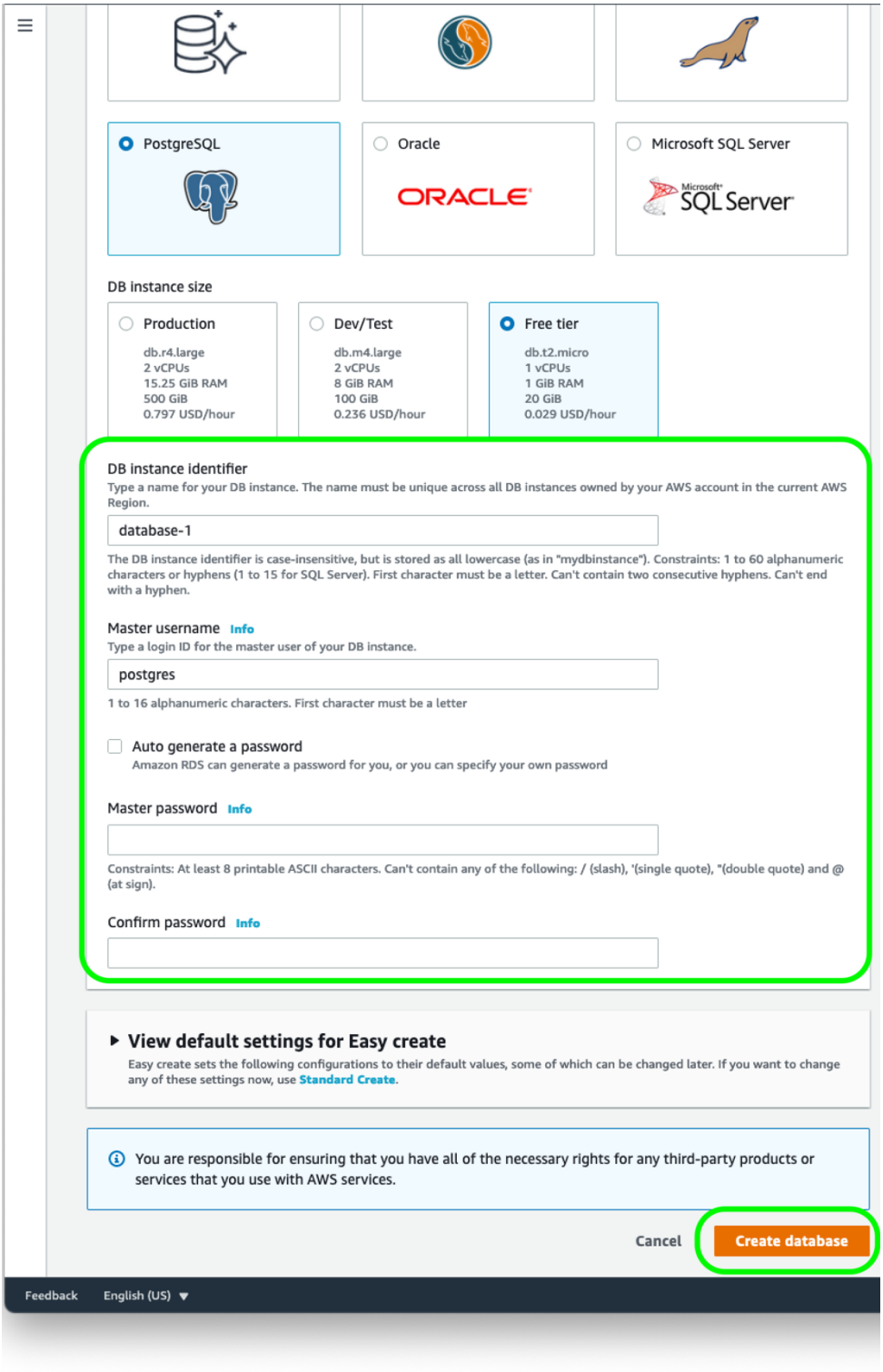
If you run a PostgreSQL without our Dockerfile, run the `database_init.sql` file to initialise the data warehouse.

Deployment

- The data warehouse is deployed using Amazon RDS and creating a PostgreSQL database instance using the free tier version from AWS. It is as se



1. Simply choose PostgreSQL as a Database engine and the Free tier version



2. Add your dataset password and username and create the database.
3. Once the database is created connect to the database via the credentials and run the database_init.sql file.

Desktop App

The desktop app, is the user interface for the SightScan project it allows to interact with images and download trained models to use them for identifying images.

- Locally
    1. `cd ./amos/gui`
    2. `pip install -r requirements.txt`
    3. `python main.py`

Sight Detector

The Sight Detector is our Model Training Service which uses a Yolov5 algorithm to detect sights in a given image.

- Locally
    1. `cd ./amos/sight_detector`
    2. `pip install -r requirements.txt`
    3. `python -m pip install -r requirements.txt`
    4. `python detect.py --weights weights/best.pt --save-txt`

DEPLOYMENT

Right now deployment is happening locally, since we don't have the correct configured access to AWS to deploy our docker images.