README

Jiacheng Weng

How to use

You can also clone from github:

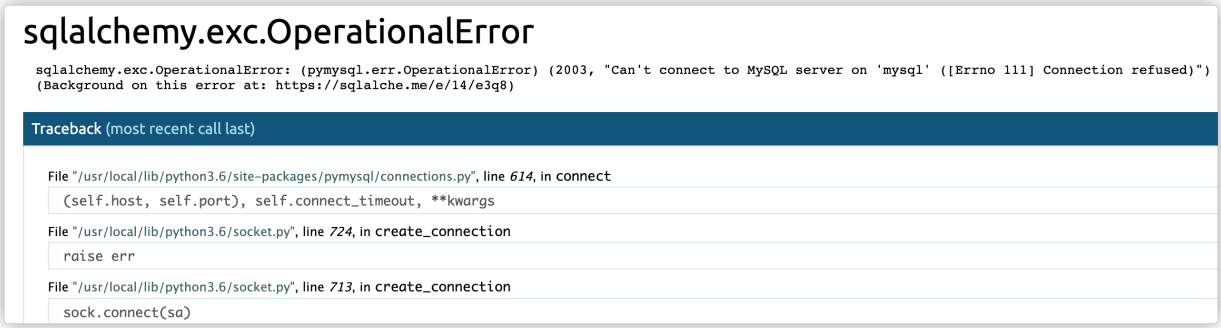
```
git clone https://github.com/JasonWong97/FlaskMarket
```

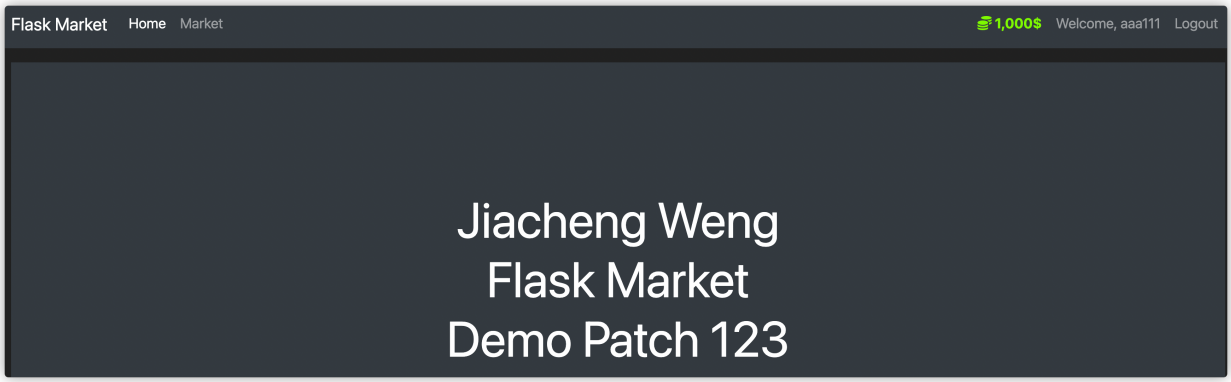
under FlaskMarket, in terminal

```
docker-compose up -d
```

```
~/Doc/FlaskMarket  git main  ✔  31
docker-compose up -d
[+] Running 2/2
 ⠿ Container flaskmarket_mysql_1  Start...
 ⠿ Container flaskmarket_app_1    Started
```

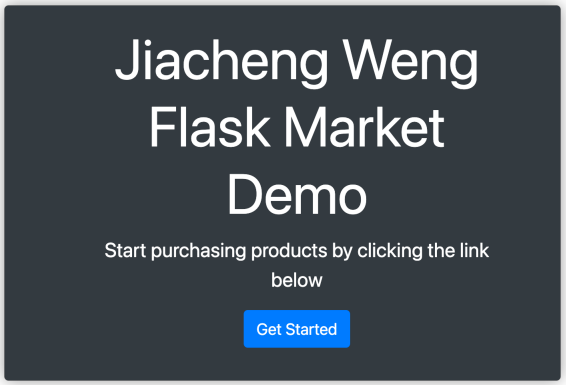
At first, `http://127.0.0.1:5001/` may looks like this. It takes one to two minutes to fully boot. Then you can see the home page.





1. git diff patch

> After patching, delete the flask container and image and rerun docker compose.



I can also use `"git diff"` to generate patch file.

```
~/Doc/FlaskMarket  git @40d03994 ?1
└ git format-patch 40d0399..f14456d
0001-add-patch.patch
0002-add-123.patch

~/Doc/FlaskMarket  git @40d03994 ?2
└ git am ./*.patch
Applying: add patch
Applying: add 123
```

> 💡 I can also do patching in the docker container. But in this way, we need to put credential in the container. This may bring potential security issues.

https://www.baeldung.com/ops/dockerfile-git-strategies

2. Database initialization

use model.py to generate database

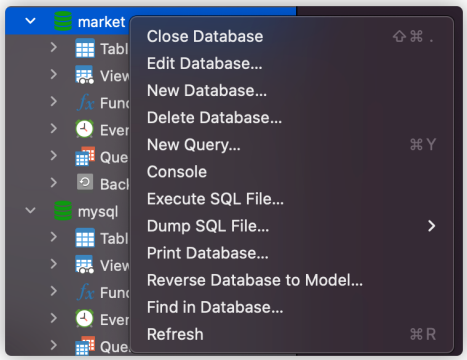
SQLAlchemy 1.4 Documentation

A table within a specific schema is referred towards explicitly using the syntax " . ". Constrast this to an architecture such as that of MySQL, where there are only "databases", however SQL statements can refer to multiple databases at once, using the same syntax except it is " .

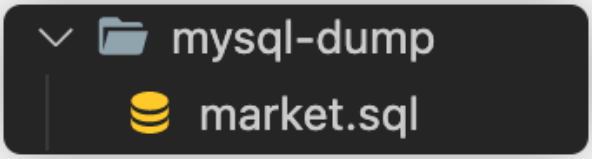
https://docs.sqlalchemy.org/en/14/core/metadata.html

```
# under FlaskMarket folder, in terminal
python
from market import db
db.create_all()
```

This will create database at Mysql. Then dump market.sql file



Then put `.sql` files in the mysql-dump directory.



Then import into docker image. This only execute once.

```
volumes:
  - ./mysql-dump:/docker-entrypoint-initdb.d
```

15 So I had the same issue for hours, and then decided to look into docker-entrypoint.sh. It turns out that the script checks for $DATADIR/mysql, typical /var/lib/mysql and skips the rest of the code if the datadir exists, incl. docker-entrypoint-initdb.d

MySQL scripts in docker-entrypoint-initdb are not executed

Thanks for contributing an answer to Stack Overflow! Please be sure to answer the question. Provide details and share your research! Asking for help, clarification, or responding to other answers. Making statements based on

https://stackoverflow.com/questions/38504257/mysql-scripts-in-docker-entrypoint-initdb-are-not-executed

Docker Compose mysql import .sql

I was having a similar issue with mysql where I would mount a local directory at /configs/mysql/data containing a mydatabasedump.sql file via docker-compose to the docker-entrypoint-initdb.d volume, the file would get loaded on to the container

https://stackoverflow.com/questions/36617682/docker-compose-mysql-import-sql

MySQL scripts in docker-entrypoint-initdb are not executed

You should clear data_volume before run the container and the sql files will be executed. This volume data_volume can be removed by using command: docker volume rm data_volume. The root cause of your problem can be found in docker-entrypoint.sh. When you run a mysql container, it checks mysql directory /var/lib/mysql exist or not.

https://newbedev.com/mysql-scripts-in-docker-entrypoint-initdb-are-not-executed

Hints

1. missing packages

1. Add the missing python packages to the docker image. The missing packages are:
    a. flask
    b. flask-mysqldb

market > requirements.txt

```
cryptography==36.0.1
email-validator==1.1.3
entrypoints==0.3
findspark==1.4.2
Flask==2.0.2
Flask-Bcrypt==0.7.1
Flask-Login==0.5.0
Flask-MySQLdb==0.2.0
Flask-SQLAlchemy==2.5.1
Flask-WTF==1.0.0
PyMySQL==1.0.2
```

```
COPY ./market/requirements.txt /market
RUN python -m pip install --upgrade pip
RUN pip install --trusted-host pypi.org --trusted-host pypi.python.org --trusted-host=files.pythonhosted.org --no-cache-dir -r requirements.txt
```

2. MySQL user and password

2. The mysql user name used in the app is ***user*** and the password is ***pass.***
   The root MySQL password is ***password***

In `docker-compose.yml`

```yaml
environment:
  MYSQL_ROOT_PASSWORD: password
  MYSQL_DATABASE: market
  MYSQL_USER: user
  MYSQL_PASSWORD: pass
restart: always
```

3. mysql data import

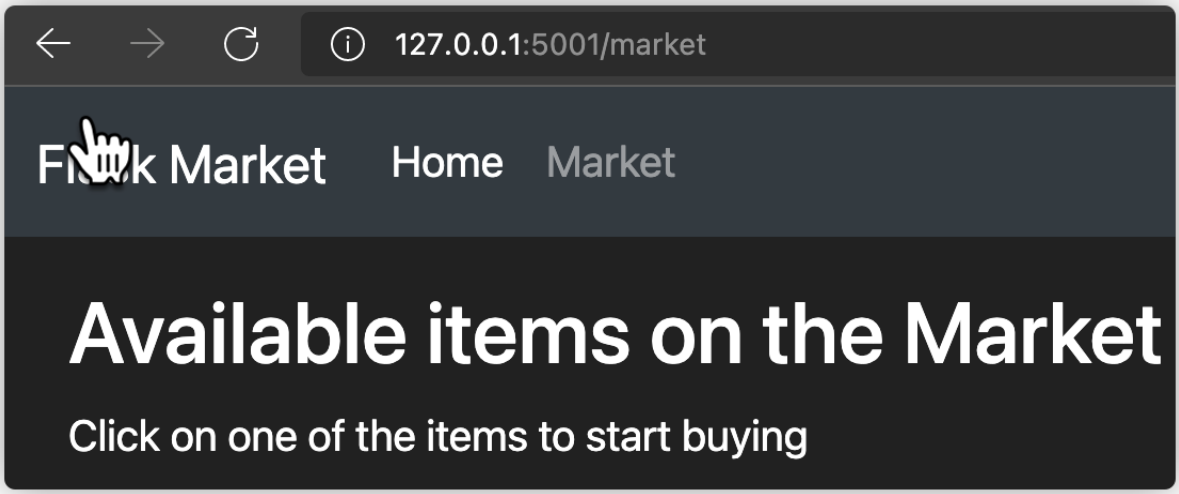
3. You can use additional service in docker compose that will run the mysql commands to create db, table and user on startup

see previous `2. Database initialization`

4. flask listening on 127.0.0.1

4. Keep in mind that by default flask is listening on 127.0.0.1

```python
if __name__ == '__main__':
    app.run(debug=True,host='0.0.0.0')
```



5. persistent mysql data and restarts

5. The mysql data is persistent and survives restarts

```yaml
mysql:
  image: mysql:8.0
  # build: ./mysql
  platform: linux/amd64
  volumes:
    - ./mysql-dump:/docker-entrypoint-initdb.d
  ports:
    - "33061:3306"
  environment:
    MYSQL_ROOT_PASSWORD: password
    MYSQL_DATABASE: market
    MYSQL_USER: user
    MYSQL_PASSWORD: pass
  restart: always
```

After I reboot my computer, the container will restart automatically.

Test

I also upload the files to github and test the program on digital ocean's Ubuntu machine. It works.

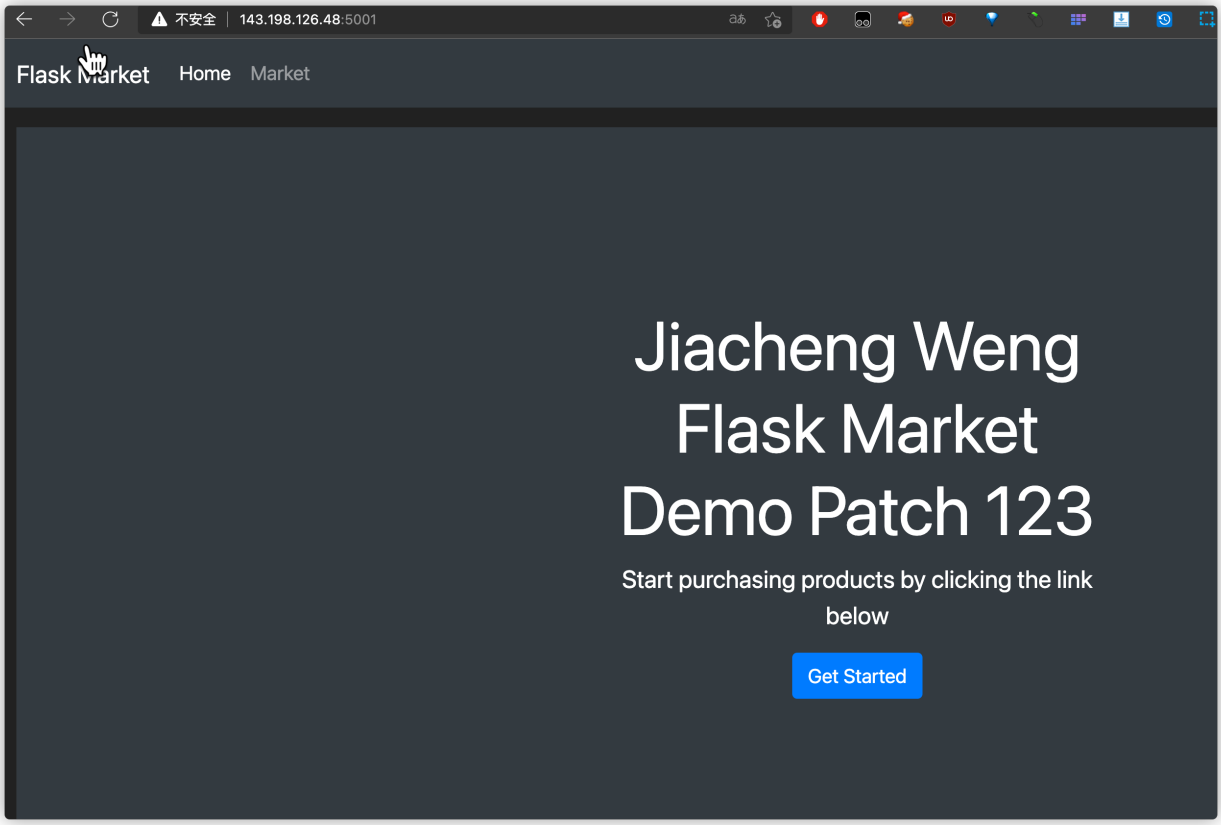
Jiacheng Weng Flask Demo

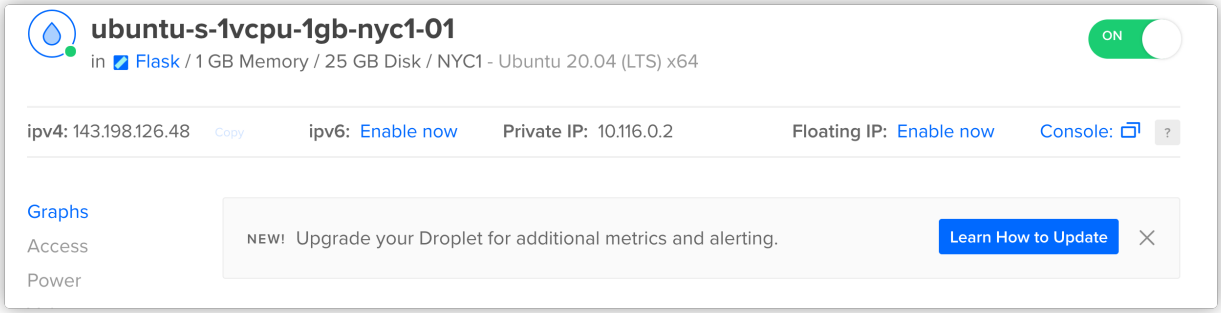
http://143.198.126.48:5001/

ubuntu-s-1vcpu-1gb-nyc1-01 - DigitalOcean Droplet Web Console

https://cloud.digitalocean.com/droplets/280986596/terminal/ui/?os_user=root

```
Collecting zipp>=0.5
  Downloading zipp-3.6.0-py3-none-any.whl (5.3 kB)
Building wheels for collected packages: Flask-Bcrypt, Flask-MySQLdb, mysqlclient
  Building wheel for Flask-Bcrypt (setup.py): started
  Building wheel for Flask-Bcrypt (setup.py): finished with status 'done'
  Created wheel for Flask-Bcrypt: filename=Flask_Bcrypt-0.7.1-py3-none-any.whl size=5030 sha256=ca698e3e4126d563d
d34e35fd5d3c9e7d5c54b2fc46f805cb8f377b15bdb1e59
  Stored in directory: /tmp/pip-ephem-wheel-cache-8ton5mtv/wheels/c9/5c/81/9da99305abd85cb148b6428bd1a8e37aff1090
8430b0e2474a
  Building wheel for Flask-MySQLdb (setup.py): started
  Building wheel for Flask-MySQLdb (setup.py): finished with status 'done'
  Created wheel for Flask-MySQLdb: filename=Flask_MySQLdb-0.2.0-py3-none-any.whl size=2666 sha256=fb059417fee15a2
8455b2eab404c2738110404f2feeebe12ecf5262030e68acf
  Stored in directory: /tmp/pip-ephem-wheel-cache-8ton5mtv/wheels/56/e8/13/f56acc8e609a558e5c6f9ae745ccd9843795ca
0e6a3fc019d8
  Building wheel for mysqlclient (setup.py): started
  Building wheel for mysqlclient (setup.py): finished with status 'done'
  Created wheel for mysqlclient: filename=mysqlclient-2.1.0-cp36-cp36m-linux_x86_64.whl size=109847 sha256=d32873
6aca3a7eb860e35e4d940186ac8be9099729f382925c289bb57751c275
  Stored in directory: /tmp/pip-ephem-wheel-cache-8ton5mtv/wheels/96/0d/a8/c8cd77741e717373250fffd50cb3b8cbe7ed0a
40aa3e11169c
Successfully built Flask-Bcrypt Flask-MySQLdb mysqlclient
Installing collected packages: zipp, typing-extensions, pycparser, MarkupSafe, importlib-metadata, dataclasses, W
erkzeug, six, Jinja2, itsdangerous, greenlet, click, cffi, WTForms, SQLAlchemy, mysqlclient, idna, Flask, dnspyth
on, bcrypt, PyMySQL, Flask-WTF, Flask-SQLAlchemy, Flask-MySQLdb, Flask-Login, Flask-Bcrypt, findspark, entrypoint
s, email-validator, cryptography
Successfully installed Flask-2.0.2 Flask-Bcrypt-0.7.1 Flask-Login-0.5.0 Flask-MySQLdb-0.2.0 Flask-SQLAlchemy-2.5.
1 Flask-WTF-1.0.0 Jinja2-3.0.3 MarkupSafe-2.0.1 PyMySQL-1.0.2 SQLAlchemy-1.4.29 WTForms-3.0.0 Werkzeug-2.0.2 bcry
pt-3.2.0 cffi-1.15.0 click-8.0.3 cryptography-36.0.1 dataclasses-0.8 dnspython-2.1.0 email-validator-1.1.3 entryp
oints-0.3 findspark-1.4.2 greenlet-1.1.2 idna-3.3 importlib-metadata-4.8.3 itsdangerous-2.0.1 mysqlclient-2.1.0 p
ycparser-2.21 six-1.16.0 typing-extensions-4.0.1 zipp-3.6.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the syste
m package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
Removing intermediate container b728df706ac9
 ---> 7d8334623ddd
Step 7/8 : COPY . /market
 ---> eba2a6938b80
Step 8/8 : CMD python run.py
 ---> Running in d91691482550
Removing intermediate container d91691482550
 ---> 6555243e9ce8
Successfully built 6555243e9ce8
Successfully tagged flaskmarket_app:latest
WARNING: Image for service app was built because it did not already exist. To rebuild this image you must use `do
cker-compose build` or `docker-compose up --build`.
Creating flaskmarket_mysql_1 ... done
Creating flaskmarket_app_1   ... done
root@ubuntu-s-1vcpu-1gb-nyc1-01:~/wjc/FlaskMarket#
```





Reference

[Flask Course - Python Web Application Development - YouTube]

Apple M1 Chip, No Matching Manifest For Linux/Arm64/V8 Docker MySql

If you are using Apple Silicon M1 chip then sometimes you have an issue with the apps, containers, etc. Because some images don't support the new Apple M1 Chip If you are getting this error `no matching manifest for Linux/arm64/v8 in the

https://onexlab-io.medium.com/apple-m1-chip-no-matching-manifest-for-linux-arm64-v8-docker-mysql-5142060a9309

Docker MYSQL [2003] Can't connect to MySQL server (111 Connection refused)

I'd like to connect python3 to mysql on Docker Container (I use ubuntu 18.04) Here is docker-compose.yml version: '3' services: # MySQL db: image: mysql:5.7 container_name: mysql_host environment: MYSQL_ROOT_PASSWORD: root MYSQL_DATABASE:

https://stackoverflow.com/questions/67241107/docker-mysql-2003-cant-connect-to-mysql-server-111-connection-refused

Deploy Flask-MySQL app with docker-compose - DevopsRoles.com