Title: Connecting to MySQL from Flask Application using docker-compose

Post Body:

I have an application using Flask and MySQL. The application does not connect to MySQL container from the Flask Application but it can be accessed using Sequel Pro with the same credentials.

Docker Compose File

```
version: '2'  services:   web:     build: flask-app     ports:      - '5000:5000'     volumes:      - .:/code   mysql:     bui
```

Docker file for MySQL

The docker file for MySQL will add schema from `test.dump` file.

```
FROM mysql/mysql-server ADD test.sql  /docker-entrypoint-initdb.d
```

Docker file for Flask

```
FROM python:latest COPY . /app WORKDIR /app RUN pip install -r requirements.txt ENTRYPOINT ['python'] CMD ['app.py']
```

Starting point `app.py`

```
from flask import Flask, request, jsonify, Response import json import mysql.connector from flask_cors import CORS, cross_orig
```

When I do a GET request on `http://localhost:5000/` using REST Client I get a valid response.

A GET request on `http://localhost:5000/api/getMonths` gives error message:

```
mysql.connector.errors.InterfaceError: 2003: Can't connect to MySQL server on '0.0.0.0:3306' (111 Connection refused)
```

When the same credentials were used on Sequel Pro, I was able to access the database.

Please advice me on how to connect the MySQL container from the Flask Application. This is my first time suing Docker and do forgive me if this is a silly mistake from my part.

Accepted Answer:

Change this

```
return mysql.connector.connect(user='testing', host='0.0.0.0', port='3306', password='testing', database='test')
```

to

```
return mysql.connector.connect(user='testing', host='mysql', port='3306', password='testing', database='test')
```

Your code is running inside the container and not on your host. So you need to provide it a address where it can reach within container network. For `docker-compose` each service is reachable using its name. So in your it is `mysql` as that is name you have used for the service

Highest Rated Answer:

For others who encounter similar issue, if you are mapping different ports from host to container for the MySQL service, make sure that container that needs to connect to the MySQL service is using the port for the container not for the host.

Here is an example of a docker compose file. Here you can see that my application (which is running in a container) will be using port 3306 to connect to the MySQL service (which is also running in a container on port 3306). Anyone connecting to this MySQL service from the outside of the 'backend' network which is basically anything that does not run in a container with the same network will need to use port 3308 to connect to this MySQL service.

```
version: '3' services:    redis:     image: redis:alpine     command: redis-server --requirepass imroot     ports:       - '63
```