مستندات پروژه نهایی کنگرو

معرفی پروژه

در این پروژه قصد داریم تا یک سایت وردپرسی راه اندازی کنیم و می خواهیم این پروژه با قابلیت مقیاس پذیری همراه باشد.
این پروژه بر بستر Docker و با Docker Compose پیاده سازی شده است.
ما در این پروژه از دیتابیس MariaDB، سرویس ProxySQL، وردپرس، لود بالانسر Nginx استفاده میکنیم تا یک وب سایت وردپرسی سریع، دردسترس و مقیاس پذیر بسازیم.
در انتها قابلیت CI/CD خودکار برای اعمال تغییرات در تم وردپرس را پیاده سازی می کنیم.

نصب و راه اندازی

پیش نیازهای سخت افزاری:

- حداقل 2 هسته CPU.
- حداقل 2 گیگابایت RAM.

سیستم عامل پیشنهادی: Debian یا Ubuntu.

(برای اجرای صحیح پروژه، کل فایل های پروژه باید در دایرکتوری /opt/Amirali_Salari_CanGrow/ قرار بگیرند.)

1. نصب Docker Engine: برای نصب داکر روی سیستم عامل خودتان، می توانید از مستندات رسمی داکر استفاده کنید.
   در Debian، برای نصب Docker Engine ابتدا باید ریپازیتوری های Docker را روی سیستم خودتان اضافه کرده و سپس با دستور apt-get آن را نصب کنید.
2. نصب Docker Compose: برای نصب این پلاگین روی سیستم عامل خودتان، می توانید از مستندات رسمی داکر استفاده کنید.
   در Debian، برای نصب Docker Compose می توانید از دستور زیر استفاده کنید.

```
sudo apt-get install docker-compose-plugin
```

3. نصب Git: برای نصب سیستم کنترل نسخه Git در سیستم عامل Debian، از دستور زیر استفاده کنید:

```
sudo apt-get install git
```

4. دریافت پروژه: وارد پوشه opt/ شوید و برای دریافت پروژه دستور زیر را در ترمینال وارد کنید:

```
git clone https://github.com/Amirali-Salari/Amirali_Salari_CanGrow
```

5. وارد مسیر /opt/Amirali_Salari_CanGrow شوید و برای راه اندازی همه کانتینرها دستور زیر را وارد کنید:

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

6. در این مرحله با وارد کردن دستور docker ps -a از بالا بودن همه کانتینرها اطمینان حاصل کنید.

7. تبریک می گوییم! همه کانتینرها با موفقیت راه اندازی شدند. حالا برای راه اندازی Replication باید دستورات زیر را اجرا کنید تا اسکریپت های مربوطه در دو کانتینر mariadb_master و mariadb_replica اجرا شوند و تنظیمات لازم را اعمال کنند.

```
sudo docker exec amirali_salari_cangrow-mariadb_master-1 sh -c '/opt/masterdb/initial.sh'

sudo docker exec amirali_salari_cangrow-mariadb_replica-1 sh -c
'/opt/replicadb/initial.sh'
```

8. حالا برای ساخت Monitoring User مورد نیاز برای ProxySQL، از دستور زیر استفاده کنید:
(این کاربر، امکانات لازم برای بررسی دیتابیس ها را برای ProxySQL فراهم می کند)

```
sudo docker exec amirali_salari_cangrow-mariadb_master-1 sh -c
'/opt/proxysql/create-monitoring-user.sh'
```

9. حالا برای انجام تنظیمات و کانفیگ های ProxySQL باید کوئری های مختلفی در کانتینر ProxySQL اجرا کنیم، به دلیل اینکه تعداد این کوئری ها بسیار زیاد است و برای سهولت کار می توانید از یک اسکریپت استفاده کنید، برای اجرای این اسکریپت در کانتینر ProxySQL دستور زیر را در هاست وارد کنید:

```
sudo docker exec amirali_salari_cangrow-proxysql-1 sh -c
'/opt/proxysql/initial.sh'
```

10. هم اکنون در مرورگر خود آدرس http://localhost (اگر قصد اجرای وردپرس روی لوکال هاست دارید) یا آدرس IP عمومی سرور (اگر قصد اجرای وردپرس روی سرور دارید) را تایپ کنید و سپس به پورت 80 آن درخواست بفرستید (پورت http).
در این مرحله زبان سایت وردپرسی خود را انتخاب کرده و ادامه دهید.
در مرحله بعد نام سایت و اطلاعات کاربری خود را وارد کرده و روی دکمه "Install Wordpress" کلیک کنید.

11. همه چیز آماده است! وارد حساب کاربری وردپرس شوید و با بیشترین سرعت از سایت ساختن لذت ببرید.

نصب و راه اندازی خودکار

برای نصب و راه اندازی خودکار پروژه، می توانید فایل install.sh موجود در پوشه ریشه پروژه را با دسترسی root اجرا کنید.

مستندات فنی

فاز اول راه‌اندازی سایت وردپرس:

ابتدا فایل داکر کامپوز را ایجاد می کنیم. سپس یک سرویس mariadb از روی آخرین ایمیج راه اندازی کرده و با استفاده از متغیرهای محیطی مثل MYSQL_USER و MYSQL_DATABASE، یک دیتابیس با نام وردپرس ایجاد کرده و برای دسترسی به آن یک حساب کاربری (شامل نام کاربری و رمز عبور) ایجاد می کنیم، دیتای ذخیره شده در این دیتابیس نیز در یک volume با نام mariadb_data قرار می گیرد. (لازم به ذکر است برای ایجاد یک کانتینر mariadb مقداردهی به متغیر MYSQL_ROOT_PASSWORD الزامی است.)

در مرحله بعد دو نمونه وردپرس از روی آخرین نسخه ایمیج وردپرس ایجاد می کنیم. برای این سرویس ها نیز با استفاده از متغیرهای محیطی اتصال با دیتابیس برقرار می شود، برای مثال مقدار WORDPRESS_DB_HOST برابر با نام سرویس دیتابیس است.
بقیه متغیرها نیز باید دقیقا برابر با نام دیتابیس، نام کاربری و رمز عبور انتخاب شده در mariadb باشند. لازم به ذکر است برای اجرای کانتینر وردپرس، بالا بودن mariadb الزامی است از این رو در قسمت depends_on این مورد درج شده است.

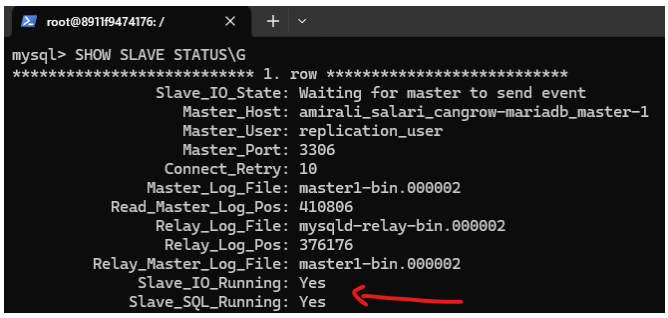
در مرحله آخر یک سرویس nginx ایجاد کرده و آن را به پورت 80 هاست مپ می کنیم تا از طریق هاست در دسترس باشد.
فایل nginx.conf که فایل پیکربندی nginx است از طریق یک volume از هاست به مکان مشخصی در کانتینر منتقل می شود.
در واقع ما در این پروژه از وب سرور nginx به عنوان یک متعادل کننده بار (Load Balancer) استفاده می کنیم تا مطمئن شویم سرویس ما همیشه در دسترس است.

توضیحات فایل nginx.conf: در ابتدای این فایل worker_proccess را روی خودکار تنظیم می کنیم تا nginx براساس تعداد core های پردازنده، تصمیم گیری های لازم را انجام دهد. در ادامه و در بخش upstream دو سرور وردپرس را مشخص می کنیم که nginx بر اساس الگوریتم ip_hash درخواست ها را به آنها ارسال می کند؛ در واقع این الگوریتم، آدرس IP سرور و مشتری را ترکیب می کند و از یک تابع ریاضی برای تبدیل آن به hash استفاده می کند. بر اساس hash، اتصال به یک سرور خاص اختصاص داده می شود.
این روش تضمین می کند که یک کاربر با IP ثابت، به همان سرور خاص متصل می شود و باعث می شود نشست های وردپرس از بین نروند.
در بخش location نحوه پروکسی شدن درخواست ها را تعیین می کنیم، این کار تضمین می کند همه درخواست ها به درستی از کاربر گرفته شده و به سرورهای وردپرس منتقل می شوند.

فاز دوم ایجاد دو دیتابیس:

در این فاز دو دیتابیس Mariadb ایجاد می کنیم و سپس عملیات Replication را روی آنها پیاده سازی می کنیم، در واقع ما دو دیتابیس master و replica ایجاد می کنیم و همه کوئری ها را برای master ارسال می کنیم و در این فرآیند همه دیتاها در زمان واقعی (Real Time) روی replica هم ذخیره می شوند.
برای تکمیل این فاز، ابتدا دو دیتابیس mariadb با نام های master و replica در فایل داکر کامپوز تعریف می کنیم و متغیرهای محیطی آنها را از فایل env. اضافه می کنیم. در دیتابیس master: ابتدا فایل my.cnf آن را تغییر می دهیم؛ در فایل کانفیگ جدید، یک شماره سرور تعریف می شود و قابلیت ایجاد لاگ تغییرات را فعالسازی می کنیم (این قابلیت برای فعالسازی Replication ضروری است)، پس از آن اسکریپت initial.sh را اجرا می کنیم، این اسکریپت ابتدا mysql-client را برای برقراری ارتباط با دیتابیس، نصب می کند.
سپس یک کاربر دیتابیس برای replication ایجاد می کند که بعدا توسط دیتابیس replica استفاده خواهد شد، جزئیات و شماره فایل لاگ master را در یک فایل ذخیره می کنیم. (زیرا این فایل بعدا در جریان راه اندازی دیتابیس replica استفاده خواهد شد) در این اسکریپت، یک متغیر جهانی read_only هم تنظیم می شود که مقدار آن در فاز بعد برای proxysql بسیار مهم است.
در دیتابیس replica: ابتدا فایل my.cnf آن را تغییر می دهیم، در فایل کانفیگ جدیدیک شماره سرور یکتا تعریف می کنیم. سپس اسکریپت initial.sh آن را اجرا می کنیم، در این اسکریپت ابتدا mysql-client را برای برقراری ارتباط با دیتابیس، نصب می کنیم.

در مرحله بعد کوئری های SQL لازم برای راه اندازی Replication را اجرا می کنیم.
ابتدا مشخصات کاربر root برای دسترسی به دیتابیس وارد می شود، در مرحله آدرس دیتابیس master، مشخصات کاربر ساخته شده، پورت دیتابیس، مشخصات فایل لاگ و تعداد دفعات تلاش برای اتصال را مشخص می کنیم و در انتها فرآیند Replication را آغاز می کنیم و وضعیت آن را نمایش می دهیم.



در این status اگر Slave_IO_Running و Slave_SQL_Running مقدار Yes را نمایش دهند، یعنی عملیات Replication با موفقیت راه اندازی شده است.
(لازم به ذکر است اسکریپت های initial.sh باید داخل کانتینر ها اجرا شوند اجرا شوند.)

فاز سوم: تقسیم بار بین دیتابیس ها

در این فاز قرار است سرویس ProxySQL را بین سرویس های وردپرس و دیتابیس های Mariadb قرار دهیم و همه کوئری ها را به سمت این سرویس ارسال کنیم.
در فایل داکر کامپوز، یک سرویس جدید به نام proxysql اضافه می کنیم که از روی ایمیج رسمی proxysql ساخته می شود و همیشه در صورت پایین آمدن، ریستارت می شود سپس volume لازم را اضافه می کنیم و پیش نیازهای آن را اجرا شدن دیتابیس های mariadb قرار می دهیم. همچنین این سرویس در شبکه پروژه یعنی cangrownet قرار می گیرد.
در پوشه proxysql اسکریپت های موردنیاز قرار گرفته اند. اسکریپت "create-monitoring-user.sh" باید در کانتینر mariadb_master اجرا شود زیرا وظیفه آن ایجاد کاربر مخصوص monitoring است. (این کاربر توسط سرویس proxysql برای نظارت به دیتابیس ها استفاده می شود.)
سپس باید اسکریپت initial.sh را در کانتینر proxysql اجرا کنیم؛ این اسکریپت با استفاده از mysql-client به proxysql وصل می شود و کوئری های لازم برای انجام تنظیمات و کانفیگ ها را ارسال می کند.
این کوئری ها کارهای زیر را انجام می دهند:

* سرورهای mysql را اضافه می کند.
* مشخصات کاربر monitor را اضافه می کند.
* متغیرهای لازم را تغییر می دهد.
* جدول hostgroups را می سازد و بر اساس متغیر read_only دیتابیس ها را در گروه "نویسنده" و "خواننده" قرار می دهد.
* جدول users را می سازد و دو کاربر root و database_user از دیتابیس ها را اضافه می کند.
* در نهایت در جدول query_rules قوانین توزیع بار تنظیم می شوند، بر اساس این قوانین به طور کلی درخواست های نوشتن به دیتابیس master ارسال می شوند و درخواست های خواندن به دیتابیس replica ارسال می شوند و بقیه کوئری های احتمالی که قانون ندارند به دیتابیس پیش فرض (یعنی master) هدایت می شوند. در نهایت همه تغییرات در لحظه اعمال می شوند و به دیسک منتقل می شوند.

فاز چهارم: راه اندازی CI/CD جهت انتشار خودکار

در این مرحله ابتدا باید یک جفت کلید SSH خصوصی و عمومی تولید کنید و کلید خصوصی را در گیت هاب به عنوان یک secret تعریف کنید. سپس اسکریپت initial.sh را اجرا کنید تا کلید عمومی ساخته شده به لیست کلیدهای مجاز اضافه شود و دسترسی های لازم اعطا شود، در آخر پیش نیاز این فرآیند یعنی rsync نصب می شود.

در مرحله بعد باید وارد ریپازیتوری گیت هاب خود شوید و فایل های قالب موردنظر را قرار دهید. سپس به بخش Actions بروید و یک workflow ایجاد کنید و فایل main.yml را ایجاد کنید.

به طور کلی این workflow با ایجاد تغییر در branch main ریپازیتوری، تغییرات را به سرور ارسال می کند.

این سرویس روی آخرین نسخه ubuntu اجرا می شود و از rsync برای ارسال تغییرات به سرور استفاده می کند.

متغیر path آدرس موردنظر در ریپازیتوری را مشخص می کند، remote_path آدرس مقصد در سرور را نمایش می دهد، remote_host آدرس آی پی سرور را نشان می دهد، remote_port پورتی که ssh روی آن فعال است را تعیین می کند (پیش فرض 22)، remote_user کاربر موردنظر روی سرور را نشان می دهد که در این سناریو root است، مقدار remote_key برابر با کلید SSH خصوصی است. (لازم به ذکر است برای مقداردهی باید یک متغیر secret با نام remote_key ایجاد کنید و کلید خصوصی خود را در آن قرار دهید)

عیب یابی و رفع مشکلات

همه لاگ های پروژه را می توانید با استفاده از دستور docker compose logs مشاهده کنید.

برای مشاهده لاگ های یک سرویس مشخص می توانید نام کانتینر موردنظر را پس از دستور بالا وارد کنید.

چند مشکل رایج در اجرای پروژه:

* در نظر نگرفتن پیش نیازهای سخت افزاری.
* مشکلات متعدد در دانلود پکیج ها، نصب داکر و دسترسی به ریجستری داکر. برای رفع این مشکل می توانید از راهکارهای متعددی استفاده کنید، یکی از این راهکارها استفاده از DNS شکن است.

نکته مهم: در اسکریپت های اجرایی خطاها مدیریت نمی شوند و خروجی استاندارد همه کدها به صورت کامل نمایش داده می شود، لذا باید به خروجی استاندارد اسکریپت ها دقت شود و در صورت مشاهده هرگونه خطا نسبت به برطرف کردن مشکل اقدام کنید.

منابع

- How to configure ProxySQL for the first time
- How to set up ProxySQL Read/Write Split
- HTTP Load Balancing
- Rsync Deployments Action

موفق باشید!