E5 Toolkit Tutorial

Musicminion/e5-toolkit (github.com)

版权声明

- 使用者可以将全部或部分本内容免费用于非商业用途
- 使用者在使用全部或部分本内容时请注明来源
- 对于不遵守此声明或者其他违法使用本内容者，将依法保留追究权
- 本内容的发布采用 Creative Commons Attribution 4.0 License
  - 完整文本：https://creativecommons.org/licenses/by/4.0/legalcode

PREPRATION

What do you need?

- To build a server with e5-renew(qyi), you need:
  - A Server (At least 2 core and 2G memory, Ubuntu 20 LTS)
  - A Domain(Also need matched SSL cert, Nginx type)
  - Configure Domain Name System(DNS) resolution to your server‘s IP address(check by ` ping your.domain.com`)
- Deploying to your localhost is **Not Recommended** at all
  - E5-renew(qyi) run with 5 containers in total.
  - A waste of your computer resource.
  - Modify env file to add `:port` to related domain URI.

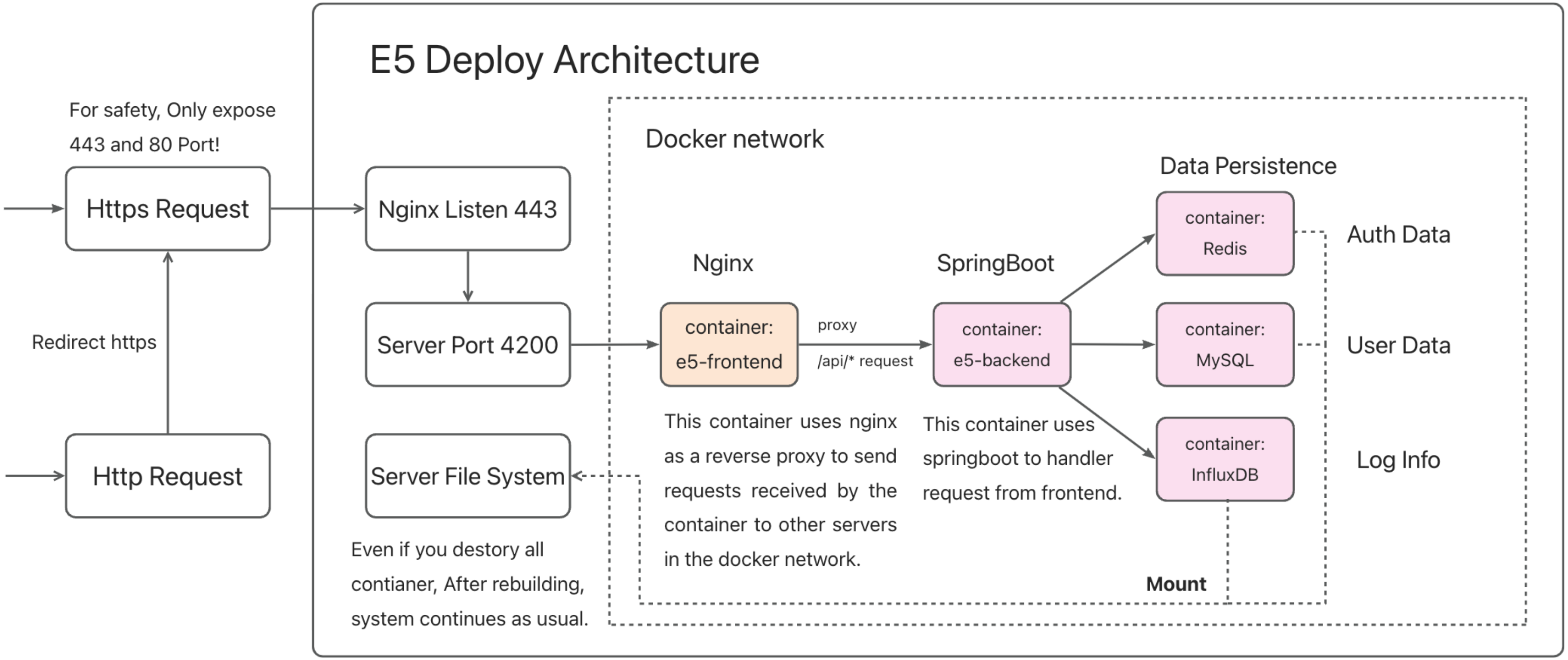
INTRODUCTIONS

Basic Concept

- This is a tool for quick build qyi-e5 web. The original Author is qyi. if you want to support or check his original repository, refer to: luoye663/e5: e5-backend and luoye663/e5-html (GitHub)
- E5 is a developer project for Microsoft office 365, only if you keep active in developing !
- Official Website: **Microsoft E5 Developer**

Project Architecture

Before you start, you'd better have a glimpse of the architecture of this project. Take a look of this!



Project Architecture(Nginx)

- **Nginx(on your host):**
  - Redirect http:80 to https:443
  - Enable SSL to make your site more safe
- **Nginx(inside container):**
  - Server frontend context
  - Proxy `/api/*` request to backend container

NGINX®
Part of F5

Project Architecture(Data)

- **All data related to your database will mount to your e5-toolkit directory, which named `data`**
- **After you launch your containers, you will find three folders under data folder**
  - MySQL
  - Influx DB
  - Redis

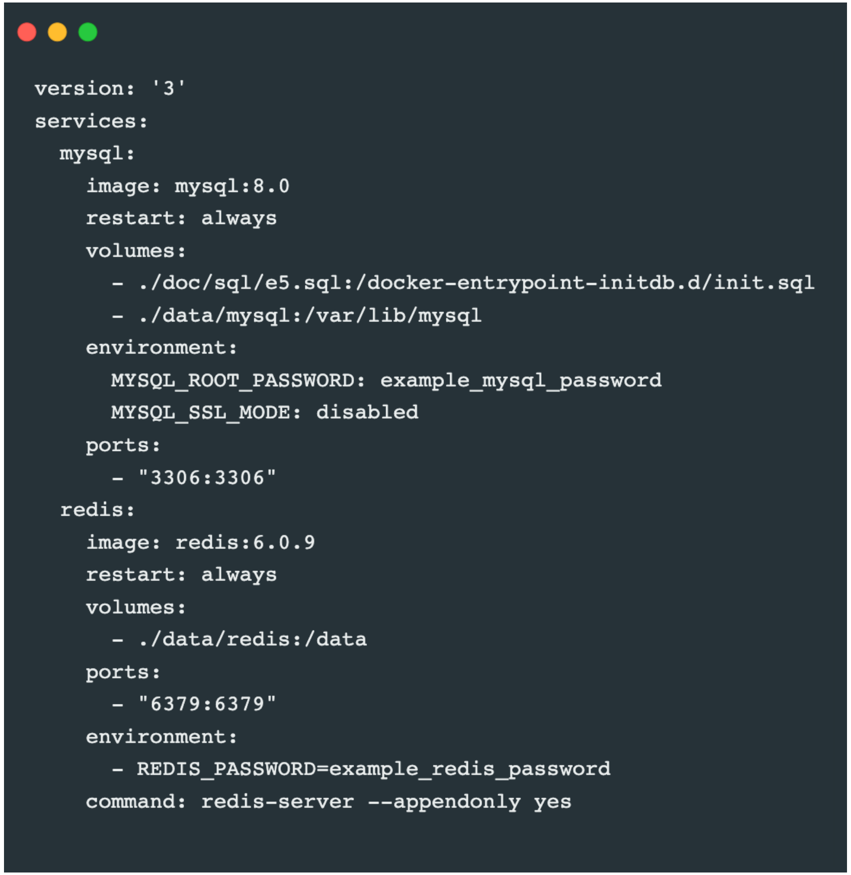
Project Architecture(Data)

- **Want to move to another server?**
- **Just backup `data` folder**
- **Want to clear all data?**
- **Just remove `data` folder**

```
dev@node-1:~/e5$ tree -L 2
├── config.env
├── data
│   ├── influxdb
│   ├── mysql
│   └── redis
├── doc
│   └── sql(for init db)
├── docker-compose.yml
├── Dockerfile
├── LICENSE
├── logs
│   └── log
```

Safety is important !

- However, In docker-compose file, I also opened database related ports for debug (Like SQL 3389)
- So, If you want to make your SQL or Influx data safer, **you need to restrict Safety Groups Or Open Only 443 Port For your data safety.** Or you can modify **related Password and token,** and env file.
- The Only needed Ports are :
  - TCP/443 (https)
  - TCP/80 (http)

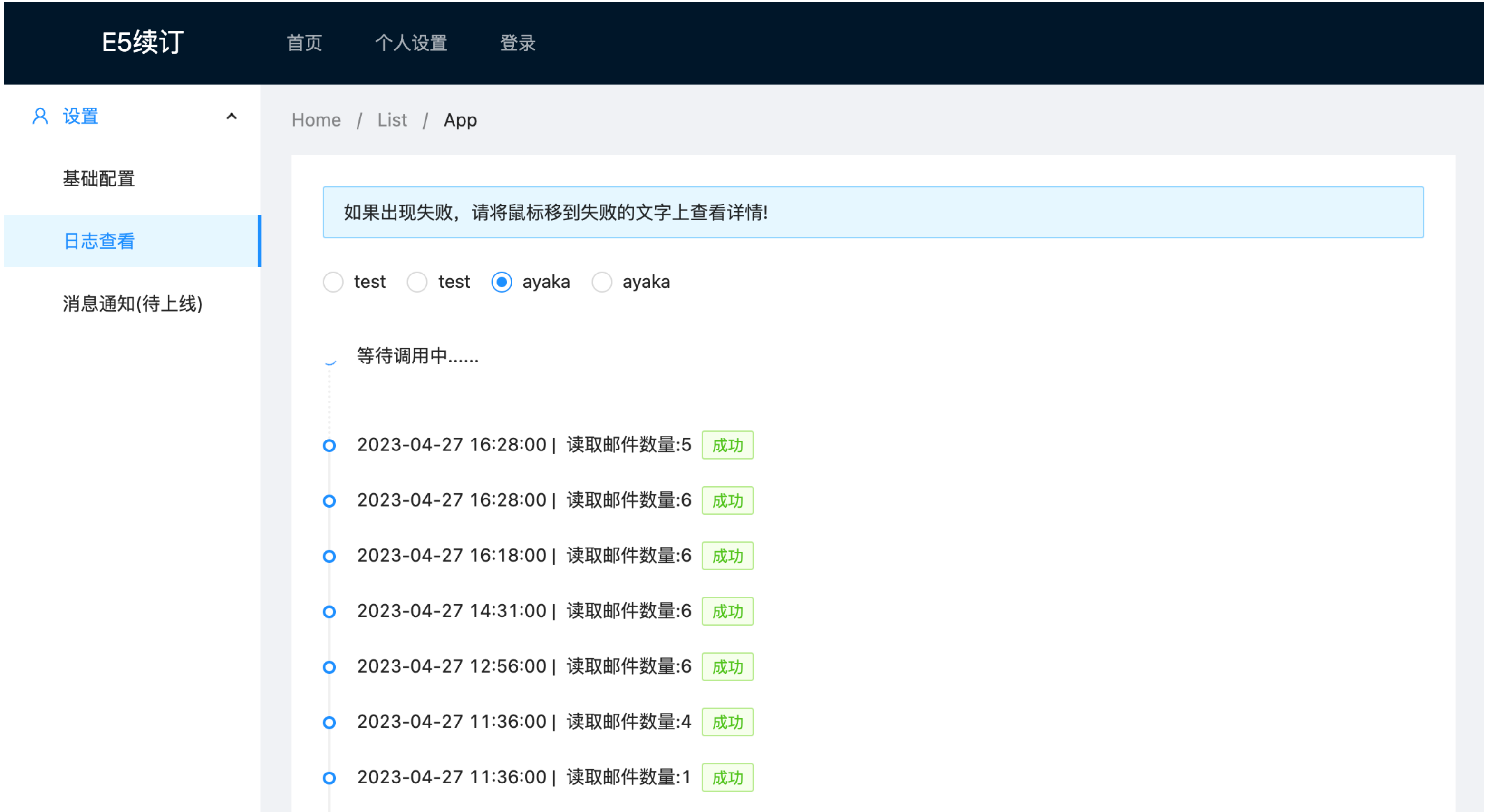


```
version: '3'
services:
  mysql:
    image: mysql:8.0
    restart: always
    volumes:
      - ./doc/sql/e5.sql:/docker-entrypoint-initdb.d/init.sql
      - ./data/mysql:/var/lib/mysql
    environment:
      MYSQL_ROOT_PASSWORD: example_mysql_password
      MYSQL_SSL_MODE: disabled
    ports:
      - "3306:3306"
  redis:
    image: redis:6.0.9
    restart: always
    volumes:
      - ./data/redis:/data
    ports:
      - "6379:6379"
    environment:
      - REDIS_PASSWORD=example_redis_password
    command: redis-server --appendonly yes
```

Change Default Password

- **All password related starts with example_**
  - Like: `example_redis_password`
  - Like: `example_influxdb_token`
- **How to modify password of token?**
  - Use searching feature of vscode
  - modify both config.env and docker-compose.yaml

Effect display



如果出现失败，请将鼠标移到失败的文字上查看详情!

○ test ○ test ● ayaka ○ ayaka

等待调用中......

2023-04-27 16:28:00 | 读取邮件数量:5 成功

2023-04-27 16:28:00 | 读取邮件数量:6 成功

2023-04-27 16:18:00 | 读取邮件数量:6 成功

2023-04-27 14:31:00 | 读取邮件数量:6 成功

2023-04-27 12:56:00 | 读取邮件数量:6 成功

2023-04-27 11:36:00 | 读取邮件数量:4 成功

2023-04-27 11:36:00 | 读取邮件数量:1 成功

DEPLOYMENT

Setup Docker And Docker-Compose

DEPLOYMENT

Setup Docker

- **Refer to docker official website**
- following the instruction from docker official Install Docker Engine on Ubuntu | Docker Documentation
- **https://docs.docker.com/engine/install/ubuntu/**
- **Commands are attached in following ppts**
- **Run them one by one.**

Setup Docker

- sudo apt-get remove docker docker-engine docker.io containerd runc
- sudo apt-get install ca-certificates curl gnupg
- sudo install -m 0755 -d /etc/apt/keyrings curl -fsSL https://download.docker.com/linux/ubuntu/gpg |
- sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
- sudo chmod a+r /etc/apt/keyrings/docker.gpg

Setup Docker

- echo \ "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \ "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
- sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
- sudo apt-get update
- sudo docker run hello-world

Add user to DockerGroup

- sudo groupadd docker
- sudo usermod -aG docker $USER
- newgrp docker
- **Reboot Your System**
- docker run hello-world

Setup Git And Repository

DEPLOYMENT

Setup Git and Pull Repo

- sudo apt-get update
- sudo apt-get install git
- git clone https://github.com/Musicminion/e5-toolkit.git ~/e5-toolkit
- cd ~/e5-toolkit

Setup Nginx

DEPLOYMENT

NGINX
Part of F5

Setup nginx

- sudo apt-get update
- sudo apt-get install nginx-full
- sudo service nginx status # check if running
- sudo service nginx start # if not running

Modify Nginx Conf

- Reposiroty Example config file: e5-toolkit/nginx.conf at main · Musicminion/e5-toolkit (github.com)
- Nginx default config file is under /etc/nginx/
- You need to put ssl cert under: /etc/nginx/cert
- Then Replace All `your.domain.com` to your actual domain name(in example nginx config file)
- Finally, copy example nginx.cont to /etc/nginx/nginx.cont
- Restart Nginx by running `sudo service nginx restart`

Setup Env Var

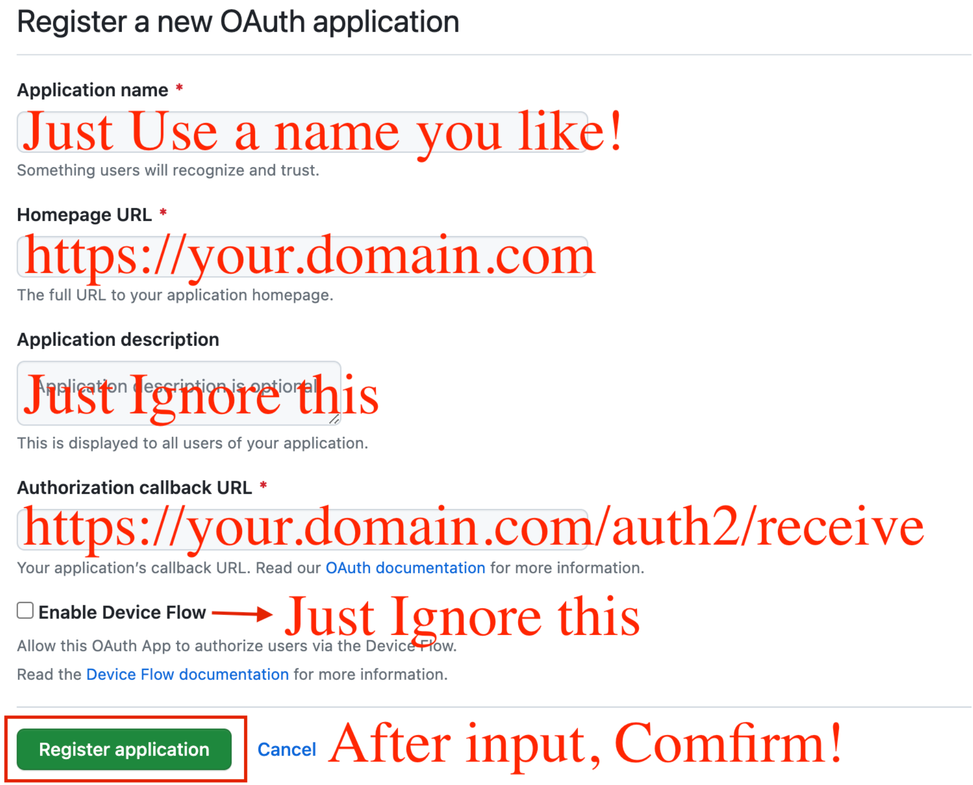
DEPLOYMENT

Env var is important!

- **First, modify config.env file you clone from my repo just now**
- **Running cmd: nano ~/e5-toolkit/config.env**
- **Search for [[[!!!changeIt!!!]]], you will find four in total**
- Go to New OAuth Application (github.com), following next slide and change your.domain.com to your own domain name.

Config Github Oauth

- **Follow this:**



Register a new OAuth application

Application name *

Just Use a name you like!

Something users will recognize and trust.

Homepage URL *

https://your.domain.com

The full URL to your application homepage.

Application description

Application description is optional

Just Ignore this

This is displayed to all users of your application.

Authorization callback URL *

https://your.domain.com/auth2/receive

Your application's callback URL. Read our OAuth documentation for more information.

☐ **Enable Device Flow** → Just Ignore this

Allow this OAuth App to authorize users via the Device Flow.

Read the Device Flow documentation for more information.

After input, Comfirm!

Env Description

- After create your OAuth App, you can get its ClientID, and then create your Client secrets
- Copy your Client secrets Immediately, since you will never see it any more in latter time
- GITHUB_CLIENT_ID: apply from New OAuth Application (github.com)
- GITHUB_CLIENT_SECRET: create your OAuth APP and create your secrets
- GITHUB_REPLYURL: the same as your input in upper table(Authorization callback url), like https://your.domain.com/auth2/receive
- OUTLOOK_REPLYURL: https://your.domain.com/outlook/auth2/%s/receive
- Launch and enjoy yourself.

Launch APP

- cd ~/e5-toolkit
- sudo docker-compose up –d
- Wait for a moment, and check your website.