SYSTEM MANUAL

BACKEND

Set Up Without Docker-Compose

This project is implemented by using Python 3.8.0 and Flask 1.1.2. To achive more information about the versions of the libraries that are used, it can be useful to examine "requirements.txt" file in the backend folder of the repository. To comply with compatibility issues, please use the following command to install related libraries.

1. virtualenv <Virtual Environment Name>
2. source <Virtual Environment Name>/bin/activate
3. cd <Path of Repository>/Platon/backend
4. pip/pip3 install -r requirements.txt

To make the database migrations:

5. Assign FLASK_APP environment variable as platon_backend.py
6. Go to <Path of Repository>/platon/backend/config.py file
7. Select the version that you will use.

DEBUG = True and DEVELOPMENT = True => Connects backend to the the MySQL Server in the Live Production Server

DEBUG = False and DEVELOPMENT = True => Connects backend to the the Test MySQL Server

DEBUG = False and DEVELOPMEN = False => This is the Docker-Compose configuration. (Use it only with Docker-Compose Setup)

8. flask db migrate
9. flask db upgrade

To run the backend server:

10. python/python3 platon_backend.py

Set Up with Docker-Compose

It was much easier to set up our backend server by using docker compose. In addition, there is not any external MySQL server required for this setting up process. This commands only runs backend and database servers. To set up with Docker-Compose use the following commands:

1. Install Docker and Docker-Compose
2. cd <Path of Repository>/platon
3. sudo docker-compose up --build (-d flag can be used to run container silently)

To stop all of the containers:

1. cd <Path of Repository>/platon

2. sudo docker-compose down

Set Up All System (Frontend-Backend_Database) With Docker-Compose

In this approach it is possible to set up backend, frontend and database server together. To set up with Docker-Compose use the following commands:

1. Install Docker and Docker-Compose

2. Adjust the Frontend Host Name:

IMPORTANT: This step is only required if you want to deploy frontend into a remote machine. Currently the delivered code is arraged to run in localhost.

Change the FRONTEND_HOSTNAME variable in the <Path of Repository>/platon/config.py file.

Example: FRONTEND_HOSTNAME = "http://ec2-3-120-98-39.eu-central-1.compute.amazonaws.com/"

3. Restore Database Dump(If you want an empty database, skip this step)

3.1 cd <Path of Repository>/platon

3.2 sudo bash restore_database.sh

4. Run All System

4.1 cd <Path of Repository>/platon

4.2 sudo bash run.sh

Database username and password can be found in the path: <Path of Repository>/platon/docker-compose.yml

MYSQL_DATABASE=platondb

MYSQL_ROOT_PASSWORD=rootpassword

MYSQL_USER=root

FRONTEND

Requirement:

You need to install Node.js before you run the following scripts. Your Node.js version must be above 5.0.0.

Recommendation:

In order to have full experience of the app, it is recommended that you should use Google Chrome as the browser.

In the project directory ("platon/frontend" directory), first run:

> npm install

This installs all the packages in package.json.

To open the app, you need to run:

> npm start

This runs the app in the development mode. You can open http://localhost:3000 to view the app in the browser. The page will reload if you make any changes in the code.

To run tests, you need to run:

> npm test

This runs the provided package's "test" script.

To have production build, you need to run:

> npm run build

This builds the app for production to the build folder.

Congrats. Your app is ready to be deployed!

This manual is created according to the information on the official create-react-app website. For more detailed information, visit the website. (https://create-react-app.dev/docs/getting-started/)

ANDROID

Requirements

- Android Studio
- Android SDK Tools (minSdkVersion:25 targetSdkVersion:29)
- JDK
- Android 6.0 and above (API Level 23)

Setup

Clone the project from Github Repository.

- git clone https://github.com/bounswe/bounswe2020group7.git

Open the source code in Android Studio.

Install a virtual device on Android Studio or connect an Android phone.

Run the project

Select a virtual device or connected Android phone in Android Studio.

Build and run the project on the selected device.