Project Setup

Objective :- In this document we are going to learn how to start new a project in django from the django boiler plate template

Requirements :

1. You will be provided with an github repository to start with.
2. You will need AWS IAM Keys

Important:- If you do not have any of the above please contact the team lead or project manager.

Note : Here Project Name is Assumed to be sulzer
Steps:

1. Clone the repository on your system and follow step 1 to 4 from the readme.
    a. Clone the repository

    ```
    git clone <repository_url>
    cd <repo-name>
    git checkout -b dev
    ```

    b. Install AWS CLI and configurations as per your os

    - Install AWSCLI
    - Install & Configure AWSCLI

    c. Create a new virtualenv and install requirements

    i. Windows

    ```
    virtualenv venv
    .\venv\Scripts\activate
    pip install -r requirements.txt
    pip install "drf-yasg[validation]"
    pip install git+https://github.com/atomic-loops/atomicloops-django-logger
    ```

    ii. Linux/MacOs

    ```
    virtualenv venv
    source venv/bin/activate
    pip install -r requirements.txt
    pip install "drf-yasg[validation]"
    pip install git+https://github.com/atomic-loops/atomicloops-django-logger
    ```

2. Download vault.py file
    Linux/MacOs

    ```
    wget -O src/vault.py https://atomicloops-dev.s3.ap-south-1.amazonaws.com/vault.py
    ```

    Windows

    ```
    curl -o src/vault.py https://atomicloops-dev.s3.ap-south-1.amazonaws.com/vault.py
    ```

3. Update the Project Name in src/settings/base.py

```
# Before
PROJECT_NAME = "******"

# After
PROJECT_NAME = "sulzer"
```

4. Update the SIGNING_KEY in SIMPLE_JWT config in src/settings/base.py
   Note to generate a random key you can enter the following command in terminal

   ```
   # Linux/MacOs
   python3 -c 'from django.core.management.utils import get_random_secret_key; print(get_random_secret_key())'

   # Windows
   python -c "from django.core.management.utils import get_random_secret_key; print(get_random_secret_key())"
   ```

   Code

   ```
   # Before
   "SIGNING_KEY": "******"

   # After
   "SIGNING_KEY":"$mu(_-ax_h6u9#jf_3l8@k18%6fwn(i7i=%i%r)kyyify=o$%u"
   ```

   Same for `SECRET_KEY` in base.py

5. Update the backend service container_name to <project_name>-backend in docker-compose-dev.yml
   Please find the PROJECT NAME IN THE README

   ```
   # Before
   container_name: "******"

   # After
   container_name: sulzer-backend
   ```

   Note :- Same for db, pgadmin, flower, celery, rabbit-mq, redis services → <project_name>-<service_name>

6. Update the db service <volume_name> to <project_name>-db in docker-compose-dev.yml Please find the PROJECT NAME IN THE README

   ```
   # Before
   <volume_name>:/var/lib/postgresql/data

   # After
   sulzer-db:/var/lib/postgresql/data
   ```

7. Please Update the db service <volume_name> to <project_name>-admin in docker-compose-dev.yml Please find the PROJECT NAME IN THE README

   ```
   <volume_name>:/var/lib/pgadmin # Before

   sulzer-admin:/var/lib/pgadmin # After
   ```

Note:- Also uncomment the last volume block in the docker-compose-dev.yml

```
###########BEFORE#############
volumes:
  prometheus_data: {}
  grafana_data: {}
#   <volume_name>:
#   <volume_name>:

##########AFTER#############
volumes:
  prometheus_data: {}
  grafana_data: {}
  sulzer-db:
  sulzer-admin:
```

Note :- Same changes 5, 6, 7 needs to be done in docker-compose.yml file
The docker-compose.yml does not contain db section since the db is/will be hosted on cloud.

8. Update the following variables in src/vault.py file
   Contanct ADMIN for AWS Details
   a. S3_BUCKET
   b. AWS_KEY_ID
   c. AWS_SECRET_KEY
   d. REGION
   e. AWS_URL
   f. EMAIL
   g. PASSWORD
   h. DB_HOST
      Note:- DB_HOST value is same as the container name for db service in the docker-compose-dev.yml file

9. Check for project setup

```
python manage.py check_setup
```

Output : It will give the following output is the setup are done are per the document

```
Project Setup Done Successfully.
```