포팅 메뉴얼

Project

```
Spring Boot 3.2.5 / Java 17 / Gradle Project
```

도커 파일

- Gateway

```
FROM openjdk:17-jdk-slim
ARG JAR_FILE=build/libs/FluffitGateway-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar", "app.jar"]
```

- Eureka

```
FROM openjdk:17-jdk-slim
ARG JAR_FILE=build/libs/FluffitEureka-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar", "app.jar"]
```

- Config

```
FROM openjdk:17-jdk-slim
ARG JAR_FILE=build/libs/FluffitConfig-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar", "app.jar"]
```

- Member

```
FROM openjdk:17-jdk-slim
ARG JAR_FILE=build/libs/FluffitMember-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar", "app.jar"]
```

- Flupet

```
FROM openjdk:17-jdk-slim
ARG JAR_FILE=build/libs/FluffitFlupet-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar", "app.jar"]
```

- Battle

```
FROM openjdk:17-jdk-slim
ARG JAR_FILE=build/libs/fluffitbattle-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar", "app.jar"]
```

Docker-compose.yml

```
version: '3.8'

services:
  member-mariadb:
    image: mariadb:latest
    container_name: member-mariadb
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: ssafyd204
      MYSQL_USER: d204
      MYSQL_PASSWORD: gumid204
      MYSQL_DATABASE: fluffit_member
      TZ: Asia/Seoul
    volumes:
      - /var/lib/docker/volumes/mariadb_data_member:/var/lib/mysql
    restart: unless-stopped
    command:
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
      - --max_connections=300
    networks:
      - dev-net

  battle-mariadb:
    image: mariadb:latest
    container_name: battle-mariadb
    ports:
      - "3307:3306"
    environment:
      MYSQL_ROOT_PASSWORD: ssafyd204
      MYSQL_USER: d204
      MYSQL_PASSWORD: gumid204
      MYSQL_DATABASE: fluffit_battle
      TZ: Asia/Seoul
    volumes:
      - /var/lib/docker/volumes/mariadb_data_battle:/var/lib/mysql
```

```
    restart: unless-stopped
    command:
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
      - --max_connections=300
    networks:
      - dev-net

  flupet-mariadb:  # 새로 추가할 서비스
    image: mariadb:latest
    container_name: flupet-mariadb
    ports:
      - "3308:3306"
    environment:
      MYSQL_ROOT_PASSWORD: d204
      MYSQL_USER: d204
      MYSQL_PASSWORD: gumid204
      MYSQL_DATABASE: fluffit_flupet
      TZ: Asia/Seoul
    volumes:
      - /var/lib/docker/volumes/mariadb_data_flupet:/var/lib/mysql
    restart: unless-stopped
    command:
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
      - --max_connections=300
    networks:
      - dev-net

  redis:
    image: redis:latest
    container_name: redis
    ports:
      - "6379:6379"
    volumes:
      - /var/lib/docker/volumes/redis_data:/data
    command:
      - redis-server
      - --requirepass gumid204
      - --maxmemory-policy allkeys-lru
    networks:
      - dev-net
    restart: unless-stopped

  redis-refresh:
    image: redis:latest
    container_name: redis-refresh
    ports:
      - "6380:6379"
    volumes:
      - /var/lib/docker/volumes/redis_refresh_data:/data
    command:
      - redis-server
      - --requirepass gumid204
      - --maxmemory-policy allkeys-lru
    networks:
      - dev-net
    restart: unless-stopped

  jenkins:
    image: jenkins/jenkins:latest
    container_name: jenkins
    user: root
    ports:
      - "18080:8080"
    volumes:
      - /var/lib/docker/volumes/jenkins_data:/var/jenkins_home
      - /var/run/docker.sock:/var/run/docker.sock
      - /usr/bin/docker:/usr/bin/docker
    environment:
      TZ: Asia/Seoul
    restart: unless-stopped
    networks:
      - dev-net

  rabbitmq:
    image: rabbitmq:management
    container_name: rabbitmq
    ports:
      - "5672:5672"
      - "15672:15672"
    restart: unless-stopped
    volumes:
      - /var/lib/docker/volumes/rabbitmq_data:/var/lib/rabbitmq
    environment:
      TZ: Asia/Seoul
      RABBITMQ_DEFAULT_USER: "d204"
      RABBITMQ_DEFAULT_PASS: "gumid204"
    networks:
      - dev-net

  zookeeper:
    image: bitnami/zookeeper:latest
    container_name: zookeeper
    ports:
      - "2181:2181"
```

```
    environment:
      ALLOW_ANONYMOUS_LOGIN: "yes"
    networks:
      dev-net:
        ipv4_address: 172.18.0.20

  kafka:
    image: bitnami/kafka:latest
    container_name: kafka
    depends_on:
      - zookeeper
    ports:
      - "9092:9092"
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:9092,PLAINTEXT_HOST://43.201.50.244:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT_HOST:PLAINTEXT
      KAFKA_LISTENERS: PLAINTEXT://kafka:9092,PLAINTEXT_HOST://0.0.0.0:29092
      KAFKA_CREATE_TOPICS: "test:1:1"
      TZ: Asia/Seoul
    networks:
      dev-net:
        ipv4_address: 172.18.0.101
networks:
  dev-net:
    external: true
```

- kafka 관련 docker-compose.yml

```
version: '3.8'

services:
  zookeeper:
    image: bitnami/zookeeper:latest
    container_name: zookeeper
    ports:
      - "2181:2181"
    environment:
      ALLOW_ANONYMOUS_LOGIN: "yes"
    networks:
      dev-net:
        ipv4_address: 172.18.0.100

  kafka:
    image: bitnami/kafka:latest
    container_name: kafka
    depends_on:
      - zookeeper
    ports:
      - "9092:9092"
      - "29092:29092"
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
      KAFKA_LISTENERS: PLAINTEXT://kafka:9092,PLAINTEXT_HOST://0.0.0.0:29092
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:9092,PLAINTEXT_HOST://43.201.50.244:29092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT_HOST:PLAINTEXT
      KAFKA_CREATE_TOPICS: "test:1:1"
      TZ: Asia/Seoul
    networks:
      dev-net:
        ipv4_address: 172.18.0.101

networks:
  dev-net:
    external: true
```

- dev-net 네트워크 생성

```
docker network create --subnet=172.18.0.0/16 dev-net
```

yml 파일

- 프로젝트 yml 파일
- gateway

```
#application.yml
server:
  port: 8000
```

```
#bootstrap.yml
spring:
  cloud:
    config:
      uri: https://config.fluffit.shop #http://127.0.0.1:8888
      name: gateway-service
  #Profiles을 사용한 Configuration 적용
  #.yml 파일명은 user-service-dev로 설정
```

```
#  profiles:
#    active: dev
```

- eureka

```
#application.yml
server:
  port: 8761

spring:
  application:
    name: eureka-service

eureka:
  client:
    register-with-eureka: false
    fetch-registry: false
  server:
    enabled-self-preservation: false
```

- config

```
#application.yml
server:
  port: 8888

spring:
  application:
    name: config-service
  cloud:
    config:
      server:
        git: #default
          uri: ENC(bdmJoKBdlVr73s3mqmoUM+x0tHQkFVp182YiDbQIYcvZKTgy5kxV6MRQO+zQFhO2OEtFsjkAaGo=)
          username: ENC(d2IM6gDqSIXJAaWijxYzKA==)
          password: ENC(jTO/0HgYmORW8MA2DL1phlAilH8uSWQ2xYhZ+gLcltgndgasZ5L+krePRwVzviLDwtX3btSwH70=)
          default-label: master #기본이 main 브랜치이므로 master 브랜치에서 가져오기 위해 설정
        bootstrap: true
  rabbitmq:
    host: rabbitmq
    port: 5672
    username: ENC(7CDqSrW1frk32qx8yIyHqA==)
    password: ENC(h8qqcxuF8aPm5dhKVFfLmMbYg2paxNem)

management:
  endpoints:
    web:
      exposure:
        include: health, busrefresh, refresh, metrics

jasypt:
  key: ${JASYPT_KEY}
```

```
encrypt:
  key-store:
    # location: file://${user.home}/Desktop/git/toy-msa/config-service/apiEncryptionKey.jks
    # Mac 또는 Linux용
    #location: file:/apiEncryptionKey.jks
    # Window용
    #location: file:apiEncryptionKey.jks
    location: file:/config/apiEncryptionKey.jks
    password: ${JKS_PWD} #ENC(xPv3Jlo1m5zpY4127BdHEA==)
    alias: ${JKS_ALIAS} #ENC(aOf2VmtI4OpHwENDzUnOx7zyx1DpvLpHUu4jhC86OOI=)

#jasypt:
#  key: ${JASYPT_KEY}
```

- member

```
#application.yml
server:
  port: 0 #랜덤포트
```

```
#bootstrap.yml
spring:
  cloud:
    config:
#      uri: https://config.fluffit.shop #local용
      uri : http://config:8888
      name: member-service
  #Profiles을 사용한 Configuration 적용
  #.yml 파일명은 user-service-dev로 설정
#  profiles:
#    active: dev
```

- flupet

```
#application.yml
server:
  port: 0
```

```
#bootstrap.yml
spring:
  cloud:
    config:
      uri: https://config.fluffit.shop #http://127.0.0.1:8888, https://config.fluffit.shop
      name: flupet-service
  #Profiles을 사용한 Configuration 적용
  #.yml 파일명은 user-service-dev로 설정
#  profiles:
#    active: dev
```

```
#data.sql
# INSERT INTO flupet (`name`, `img_url`, `stage`) VALUES
#                                  ('알', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/egg.gif', 1),
#                                  ('흰토끼', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/rabbit_white1.png,https://my-fluffit-app-ser
#                                  ('네로', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/cat_gray.png,https://my-fluffit-app-service-b
#                                  ('코기', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/dog_corgi.png,https://my-fluffit-app-service-
```

```
#schema.sql
drop table if exists `member_flupet`;
drop table if exists `flupet`;
CREATE TABLE `flupet` (
                          `id`  int unsigned auto_increment
                              primary key,
                          `name`    varchar(10) NOT NULL,
                          `img_url` MEDIUMTEXT  NOT NULL, -- 터지면 MEDIUMTEXT로 설정
                          `stage`   int NOT NULL
);

# drop table if exists `member_flupet`;
CREATE TABLE `member_flupet` (
                    `id` int unsigned auto_increment
                        primary key,
                    `flupet_id`  int unsigned    NOT NULL,
                    `member_id`  varchar(255)    NOT NULL,
                    `name`   varchar(10)    NOT NULL,
                    `exp`    int NOT NULL    DEFAULT 0,
                    `steps`  int unsigned    NOT NULL    DEFAULT 0,
                    `is_dead`    boolean NOT NULL    DEFAULT false,
                    `create_time`    datetime(6) NOT NULL,
                    `end_time`   datetime(6) NULL,
                    `fullness`   int NOT NULL    DEFAULT 100,
                    `health` int NOT NULL    DEFAULT 100,
                    `pat_cnt`    int NOT NULL    DEFAULT 5,
                    `acha_time`  datetime(6) NULL,
                    `fullness_update_time`   datetime(6) NULL,
                    `health_update_time` datetime(6) NULL,
                    CONSTRAINT `fk_member_flupet_flupet_id` FOREIGN KEY (`flupet_id`) REFERENCES `flupet` (`id`),
                    INDEX `idx_member_id` (`member_id`),
                    INDEX `idx_member_id_is_dead` (`member_id`, `is_dead`) -- 복합 인덱스 추가(이게 성능상 더 좋은지 판단 필요)
);

drop table if exists `food_type`;
CREATE TABLE `food_type` (
                    `id` int unsigned auto_increment
                        primary key,
                    `name`   varchar(10) NOT NULL,
                    `img_url`    varchar(255)    NOT NULL,
                    `fullness_effect`    int NOT NULL,
                    `health_effect`  int NOT NULL,
                    `price`  int NOT NULL,
                    `stock`  int NOT NULL    DEFAULT 30,
                    `description`    varchar(255)    NOT NULL
);

INSERT INTO flupet (`name`, `img_url`, `stage`) VALUES
                ('알', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/ezgif.com-animated-gif-maker.gif', 1),
                ('흰토끼', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/baby_rabbit_1.png,https://my-fluffit-app-service-bucket.s3.ap-nort
                ('네로', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/baby_nero_1.png,https://my-fluffit-app-service-bucket.s3.ap-northea
                ('코기', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/corgi-baby-1.png,https://my-fluffit-app-service-bucket.s3.ap-northe
                ('흰토끼', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/rabbit_white1.png,https://my-fluffit-app-service-bucket.s3.ap-nort
                ('네로', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/cat_gray.png,https://my-fluffit-app-service-bucket.s3.ap-northeast-
                ('코기', 'https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/dog_corgi.png,https://my-fluffit-app-service-bucket.s3.ap-northeast

INSERT INTO food_type (name, img_url, fullness_effect, health_effect, price, stock, description) VALUES
                                                                                                    ('기본 사료', 'https://my-fluffit-app-service-bucket.s3.ap-northeast
                                                                                                    ('인스턴트 사료', 'https://my-fluffit-app-service-bucket.s3.ap-northe
                                                                                                    ('고급 사료', 'https://my-fluffit-app-service-bucket.s3.ap-northeast

-- ALTER TABLE `members` ADD CONSTRAINT `PK_MEMBERS` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `memberFlupets` ADD CONSTRAINT `PK_MEMBERFLUPETS` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `steps` ADD CONSTRAINT `PK_STEPS` PRIMARY KEY (
--  `id`
-- );
```

```
--
-- ALTER TABLE `foodTypes` ADD CONSTRAINT `PK_FOODTYPES` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `runnings` ADD CONSTRAINT `PK_RUNNINGS` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `flupets` ADD CONSTRAINT `PK_FLUPETS` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `battleRankings` ADD CONSTRAINT `PK_BATTLERANKINGS` PRIMARY KEY (
--  `Key`
-- );
--
-- ALTER TABLE `battle` ADD CONSTRAINT `PK_BATTLE` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `AppVersion` ADD CONSTRAINT `PK_APPVERSION` PRIMARY KEY (
--  `id`
-- );
--
-- ALTER TABLE `PetRangkings` ADD CONSTRAINT `PK_PETRANGKINGS` PRIMARY KEY (
--  `Key`
-- );
--
-- ALTER TABLE `battleTypes` ADD CONSTRAINT `PK_BATTLETYPES` PRIMARY KEY (
--  `id`
-- );
```

- battle

```
server:
  port: 8082
```

```
spring:
  cloud:
    config:
      uri: https://config.fluffit.shop
      name: battle-service #깃허브 레포지토리에 추가한 .yml파일 이름
```

Private Repository에 저장된 yml 파일 (config에서 해당 yml을 가져와 사용)

- gateway

```
eureka:
  client:
    register-with-eureka: true
    fetch-registry: true
    service-url:
      defaultZone: https://eureka.fluffit.shop/eureka

spring:
  application:
    name: gateway-service
  rabbitmq:
    host: rabbitmq
    port: 5672
  username: "{cipher}AQAXruH8dEU0vcoeLyYHLZ30F2Mk76dobsDcp/XAJtwD9rVJFaygQBqrS30wSpG1RQ/ScIlF2u0BXAqofBBcRcwM8yELQ4dFrABHvBQ/NilbY3XPtxrzf8USFdIRkgVS4tIuPnaaX
  password: "{cipher}AQBmToBLwwVrP/uE4k+dJZiC7NmGrqkWLOZGp2xaKVdmcOjej+PhDuyW1U1h7LM/vyWg17v+Zn4MPi8inbG/GKR6lFdkgWbfL7qRmHi34jFnlnTF7hZb9CZpChUp86Qg3Src+71WT
cloud:
  gateway:
    default-filters:
      - name: GlobalFilter
        args:
          baseMessage: Spring Cloud Gateway Global Filter
          preLogger: true
          postLogger: true

    routes:
      #member-service auth
      - id: member-service
        uri: lb://MEMBER-SERVICE
        predicates:
          - Path=/member-service/auth/**
          - Method=GET,POST,DELETE,PUT
        filters:
          - RemoveRequestHeader=Cookie
          - RewritePath=/member-service/(?<segment>.*), /$\{segment}
      #member-service app-status
      - id: member-service
        uri: lb://MEMBER-SERVICE
        predicates:
          - Path=/member-service/app-status/**
          - Method=GET,POST,DELETE,PUT
        filters:
          - RemoveRequestHeader=Cookie
          - RewritePath=/member-service/(?<segment>.*), /$\{segment}
```

```
        #member-service actuator
        - id: member-service
          uri: lb://MEMBER-SERVICE
          predicates:
            - Path=/member-service/actuator/**
            - Method=GET,POST
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/member-service/(?<segment>.*), /$\{segment}
        #member-service member
        - id: member-service
          uri: lb://MEMBER-SERVICE
          predicates:
            - Path=/member-service/member/**
            - Method=GET,POST,DELETE,PUT
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/member-service/(?<segment>.*), /$\{segment}
            - CustomFilter
        #member-service exercise
        - id: member-service
          uri: lb://MEMBER-SERVICE
          predicates:
            - Path=/member-service/exercise/**
            - Method=GET,POST,DELETE,PUT
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/member-service/(?<segment>.*), /$\{segment}
            - CustomFilter
        #member-service swagger
        - id: member-service
          uri: lb://MEMBER-SERVICE
          predicates:
            - Path=/member-service/swagger-ui/**
            - Method=GET,POST,DELETE,PUT
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/member-service/(?<segment>.*), /$\{segment}
        #flupet-service
        - id: flupet-service
          uri: lb://FLUPET-SERVICE
          predicates:
            - Path=/flupet-service/webjars/**
            - Method=GET,POST
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/flupet-service/(?<segment>.*), /$\{segment}
        #flupet-service
        - id: flupet-service
          uri: lb://FLUPET-SERVICE
          predicates:
            - Path=/flupet-service/**
            - Method=GET,POST,DELETE,PUT
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/flupet-service/(?<segment>.*), /$\{segment}
            - CustomFilter
        #battle-service
        - id: battle-service
          uri: lb://BATTLE-SERVICE
          predicates:
            - Path=/battle-service/**
            - Method=GET,POST,DELETE,PUT
          filters:
            - RemoveRequestHeader=Cookie
            - RewritePath=/battle-service/(?<segment>.*), /$\{segment}
            - CustomFilter
      data:
        redis:
          host: redis-refresh
          port: 6379
          password: "{cipher}AQB24TI/0km0wKPxPZzzdpb0H4q6LWDaHC+OZJqic+bbWfoz/XmhNeo2YAe8pshZ4EWUTKVu/tFsC6Ho5TGGgXUFRYgHt6U+pQopTS1fsYu1xNGX76j5I/RegPBnfVI1tiqlLjx

    management:
      endpoint:
        web:
          exposure:
            include: refresh, health, beans, busrefresh, info, metrics, prometheus
    # jwt:
    #   secret: '{cipher}AQBe16BkxG+YwGC+FbGnGNoEE9KcSj5eUcXiUKsHWEnDlkeK7ugbKpGQ3weuZAnGGSq2F9HyPjTB0ifLkXWpsX7ybJjUqK8RRydV9t5cIVDQTRZbhgUVKpYoggdD6pyDEdMwr3Lgaja
```

- member

```
eureka:
 instance:
   instance-id: ${spring.application.name}:${spring.application.instance_id:${random.value}}
 client:
   service-url:
     defaultZone: https://eureka.fluffit.shop/eureka
   fetch-registry: true
   register-with-eureka: true

management:
  endpoints:
    web:
```

```
      exposure:
        include: refresh, health, beans, busrefresh, info, metrics, prometheus

spring:
  cloud:
  application:
    name: member-service
  rabbitmq:
    host: rabbitmq
    port: 5672
    username: '{cipher}AQB/VuK4mCRtOoBLTl+WwhWa3LzvfJGqaI1MkFeKYqe8WIA7x/utg84dMUHEzTa8h4KCs2mqfW7QqXRB3y/zfipIdMeiQQ4sUuJoHKWhtwtN+ESgkD2BRCwmXZoNtHAxtUxN4t00i
    password: '{cipher}AQA1TImpHxPML7Q4nPdV4AIiXmcO48bvyUndEel6gCnMol9B+RHjHh3HnHPrWJu0RqwCJYdTvKgi2jlEkft39Y9koGSynUfZ+158Ibd+X9EHcLCfJmF2vmpzfjnLKFKr0y6pRGli9

  jpa:
    hibernate:
      naming:
        implicit-strategy: org.springframework.boot.orm.jpa.hibernate.SpringImplicitNamingStrategy
      ddl-auto: create-drop
    properties:
      hibernate:
        show_sql: false
        format_sql: false
        highlight_sql: false
    defer-datasource-initialization: true
  sql:
    init:
        mode: never
  datasource:
      driver-class-name: org.mariadb.jdbc.Driver
      username: '{cipher}AQATe2qDY71R2EqiOFprZKCtjfdVeyifu6u3ZaPD3bKrn9RhTEb+2nIXCrNILdGD8MPiOh+3zQIrF9sxMoMVImPG8UbEEBsRE9Sylz9RZ0Ml2luw8LC7l/zo4KGAy7eHf/nO/Mr
      password: '{cipher}AQCWw0jxvMW0eID68PqmUFkTHRFyJX8xWTuDmSz/LpmytenL+5QjPKjfdNPglLsHWadORzpXbXV4HUSTocN9n2gpdhlVl3KxThgQyhXU0TelJN8CmXTPWhaBp6V90uHgLWokmc2
      url: '{cipher}AQA7HLDxFpOClW8OQyQXJHOypCwNs6P93Ozk8MR4xHOhOwetT2Zba9WY2QWmTC4IXepv+4nA/a+o9evFB/Q2oNSlQp34GooFEfx5Xsm3nadZs3EcohnjZMtsd94Jv3VEXWy9s+xNq0pm
      #url: '{cipher}AQAjQTTLjPjgo9eMk49H5ZCydJMgwkVp65EEYUFn8bCWO2+qrjCLDyPi68MKkJ3MQ7gD/82+guYK4/V8zZkRZZFgNP58gKWFT3C3YoVK00cwqWgJLuwET+eabuXND8s7yDxLyK9E95r

  data:
    redis:
      host: redis-refresh
      port: 6379
      password: '{cipher}AQBDrFVMwfOEh9LhndW4Zid1GaNavWf1/dfyhSrIVpeXSkoVQ2CC6jyM4NCLx5LpcUkfWH8hkcbIL2ckqrdmQ09PoPfmx0RCLvoabPyTFJHE+jMauc6YJK4y83WpdJVftrNuyVt
security:
  hmackey: '{cipher}AQB8EdnL61lkhho5BDdiO2UAeT5TpZJSYyrbG2wFK/+3tpiFwKu6hrqyHDPw5r4ojziffjfWfifK/w+jkYWWpDClp0LVAwY3rVLGqoWWkuGozdl4CRn1KXFmRwJa8YL51VSR+cdzomK+
jwt:
  secret: '{cipher}AQAnglOaC+9UralXzmXb0ZEi7TT6gLvQQc3AgYCVtAmd5u9lEVUzkNswvUcL0MniF15e6hjYwXAPR+/McvNjzgeiIRtLPQtmn7SYL2uioiG88Ee5ZCdRXwlUdn5MQHkK4WMtFR/9LuaBY
```

- flupet

```
spring:
  cloud:
  #    config:
  #      enabled: false
  application:
    name: flupet-service
  sql:
    init:
      mode: never
  rabbitmq:
    host: rabbitmq #rabbitmq
    port: 5672
    username: "{cipher}AQCAb+VtSRcBrOqH4nU/oL3K26hZuo6ogRFkEr8BTBrpe9vLeOvNg6xnXoGpcw7ONF49Chd/oygTdRCpPRM1zeZU9GUP2fu7fRNO6T6DQ5SgYO703hGBK73ahBzlV8hNaYjbI1qmd
    password: "{cipher}AQBjsI8rNvF7/h76nNQzd79291ZsLgPuGF+eM5m93lS1N5DKzYSOUP0UfTw63DHf6hAFcHqq1eyHprjLaunMutSq0di9rlSdP022kxy4mmt+X3hg/KqielO2WAR1lRCF7LpNVo0ot/
  r2dbc:
    url: "{cipher}AQA/csaCqTcRM1UneZXgWCD8ebmVWJx+uSQdl1ld7STrVoOxta0iSgEYKqy5RdTEI3UURjoptYrocoi/cILfdm2KE1iVceAChEswRchoZQpCv4eY1VQ/YEiiA3X2ybXwtXqf3Sme0vay1P
  #    username: [db아이디]
  #    password: [비번]
  data:
    redis:
      host: redis #redis
      port: 6379
      password: "{cipher}AQBjsI8rNvF7/h76nNQzd79291ZsLgPuGF+eM5m93lS1N5DKzYSOUP0UfTw63DHf6hAFcHqq1eyHprjLaunMutSq0di9rlSdP022kxy4mmt+X3hg/KqielO2WAR1lRCF7LpNVo0
      lettuce:
        pool:
          max-active: 10
          max-idle: 5
          min-idle: 0

eureka:
  instance:
    instance-id: ${spring.application.name}:${spring.application.instance_id:${random.value}}
  client:
    service-url:
      defaultZone: https://eureka.fluffit.shop/eureka
    fetch-registry: true
    register-with-eureka: true

management:
  endpoints:
    web:
      exposure:
        include: refresh, health, beans, busrefresh, info, metrics, prometheus

logging:
  level:
    com.ssafy.fluffitflupet.client: DEBUG
    org.springframework.r2dbc.core: debug
```

```
  pattern:
    level: "%5p [${spring.application.name:},%X{traceId:-},%X{spanId:-}]"

schedule:
  cron: 0 1 0 * * ? #매 00시 1분마다
  use: true

total:
  stage: 3

tombstone:
  img: https://my-fluffit-app-service-bucket.s3.ap-northeast-2.amazonaws.com/tombstone.png

kafka:
  addr: kafka:9092
#eureka등록 잠시 비활성화, 사용할때는 주석(Application실행 파일에 주석도 처리)
# eureka:
#   client:
#     enabled: false
```

- battle

```
spring:
  cloud:
  #    config:
  #      enabled: false
  application:
    name: battle-service
  sql:
    init:
      mode: always
  rabbitmq:
    host: fluffit.shop
    port: 5672
    username: '{cipher}AQAo2izQv1JUK9BwpUDwyMYoWX3nZssgRNcvzM6+LkHYfmEVg7moH8xmZoiQqG2s7MHRlOzWTwra21iaOw5sf5lUdLtqPVLutkh0FOKrIh76Yr4hEEpUwRz1neDGiZStvCL/jWuV4
    password: '{cipher}AQCYNnrPyH0e75wyeI6jVYRbQzaMHBBcY+rhLk9IB7mjVrSkmA+Mg2uax3Hw/MLY/r+HPI00PPnloXa8bxk7Rt8BSHQntjYO+A3lz2A33b9J5EU/woHaBHiOVvjEDEOfDMRaRRf1d
  datasource:
    url: '{cipher}AQAPzBtxTl2t3yOx3h4SFdQB3qXzLIUXxtjLGhHNMO34ub4DnYXxGVA0zDZw7ICu2k910aVzMrkHcCEwpaXgk2LRBmNgk1YMW0v8+gGNSDMFMr3JHVttsLG6fsqKHlSvckk7DIkK3hBXsPl
    username: d204
    password: gumid204
    driver-class-name: org.mariadb.jdbc.Driver
  r2dbc:
    url: '{cipher}AQAPzBtxTl2t3yOx3h4SFdQB3qXzLIUXxtjLGhHNMO34ub4DnYXxGVA0zDZw7ICu2k910aVzMrkHcCEwpaXgk2LRBmNgk1YMW0v8+gGNSDMFMr3JHVttsLG6fsqKHlSvckk7DIkK3hBXsPl
    username: root
    password: ssafy
    driver-class-name: io.r2dbc.mariadb.MariadbConnectionFactory
  jpa:
    hibernate:
      ddl-auto: update
    show-sql: true
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MariaDBDialect
#    username: [db아이디]
#    password: [비번]
  jmx:
    enabled: true
  data:
    redis:
      host: fluffit.shop
      port: 6379
      password: '{cipher}AQBjsI8rNvF7/h76nNQzd79291ZsLgPuGF+eM5m93lS1N5DKzYSOUP0UfTw63DHf6hAFcHqq1eyHprjLaunMutSq0di9rlSdP022kxy4mmt+X3hg/KqielO2WAR1lRCF7LpNVo0
      lettuce:
        pool:
          max-active: 10
          max-idle: 5
          min-idle: 1

eureka:
 # instance: // 포트번호 0번으로 할 때만 필요
 #   instance-id: ${spring.application.name}:${spring.application.instance_id:${random.value}}
 client:
   service-url:
     defaultZone: https://eureka.fluffit.shop/eureka
   fetch-registry: true
   register-with-eureka: true

management:
  endpoints:
    web:
      exposure:
        include: refresh, health, beans, busrefresh, info, metrics, prometheus
    jmx:
      exposure:
        include: '*'

springdoc:
  swagger-ui:
    path: /swagger-ui.html
  api-docs:
    path: /api-docs


logging:
  level:
```

```
    root: info
  pattern:
    level: '%5p [${spring.application.name:},%X{traceId:-},%X{spanId:-}]'

# #eureka등록 잠시 비활성화 사용할때는 주석(Application실행 파일에 주석도 처리)
# eureka:
#   client:
#     enabled: false
```

NginX 설정 파일

```
server {
    listen 80;
    server_name fluffit.shop;

    if ($host = fluffit.shop){
        return 301 https://$host$request_uri;
    }

    return 404;
}

server {
        listen 443 ssl http2;
        server_name fluffit.shop;
        ssl_certificate /etc/letsencrypt/live/fluffit.shop/fullchain.pem; # managed by Certbot
        ssl_certificate_key /etc/letsencrypt/live/fluffit.shop/privkey.pem; # managed by Certbot
        include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
        ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

         location / {
                proxy_pass http://localhost:8000;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }
         location /encrypt {
                proxy_pass http://localhost:8888/encrypt;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }
         location /decrypt {
                proxy_pass http://localhost:8888/decrypt;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }
}

server {
        listen 443 ssl http2;
        server_name jenkins.fluffit.shop;
    ssl_certificate /etc/letsencrypt/live/jenkins.fluffit.shop/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/jenkins.fluffit.shop/privkey.pem; # managed by Certbot
        include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
        ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

        location / {
                proxy_pass http://localhost:18080;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }
}

server {
        listen 443 ssl http2;
        server_name config.fluffit.shop;
    ssl_certificate /etc/letsencrypt/live/config.fluffit.shop/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/config.fluffit.shop/privkey.pem; # managed by Certbot
        include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
        ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

        location / {
                proxy_pass http://localhost:8888;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }
}

server {
        listen 443 ssl http2;
        server_name eureka.fluffit.shop;
    ssl_certificate /etc/letsencrypt/live/eureka.fluffit.shop/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/eureka.fluffit.shop/privkey.pem; # managed by Certbot
        include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
        ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
```

```
        location /eureka {
                proxy_pass http://localhost:8761/eureka;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }
         location / {
                proxy_pass http://localhost:8761;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
       }
}
```

- 외부 접속 경로

```
#jenkins
https://jenkins.fluffit.shop/
#eureka
https://eureka.fluffit.shop/
#config
https://config.fluffit.shop/
```

fluffit.shop

- **레코드 개수 :** 5개
- **최근 업데이트 :** 2024-05-07 18:57:08

타입	호스트	값/위치
A	@	43.201.50.244
A	www	43.201.50.244
A	jenkins	43.201.50.244
A	config	43.201.50.244
A	eureka	43.201.50.244

+ 레코드 추가

DNS 설정 목록