[AuthUser] Guia para Configuração Ambientes (dev e prod) e Deploy do Microservice no Heroku

1. Configuração Ambiente **DEV** para AuthUser Microservice

1. [AuthUser Microservice] Criar branch local dev: **git checkout -b dev**
2. [AuthUser Microservice] Renomear o arquivo application.yaml para application-dev.yaml para propriedades exclusivas do ambiente de dev que são diferentes em outros ambientes;
3. [AuthUser Microservice] Criar outro arquivo application.yaml apenas com configurações de spring.application.name, recortar essa propriedade de application-dev.yaml;
4. [AuthUser Microservice] No arquivo application.yaml criado no passo 1.3 acima, incluir spring.profiles.active=dev;
5. [AuthUser Microservice] Fazer o commit das alterações;
6. [ConfigServer Repositório Git] Renomear o arquivo ead-authuser-service.yaml para ead-authuser-service-dev.yaml
7. [ConfigServer Repositório Git] Fazer commit das alterações e subir para o Github.

2. Configuração Ambiente **PROD** para AuthUser Microservice

1. [AuthUser Microservice] Criar branch local prod: **git checkout -b prod**
2. [AuthUser Microservice] No arquivo application.yaml alterar spring.profiles.active=dev para spring.profiles.active=prod ;
3. [AuthUser Microservice] Criar arquivo application-prod.yaml dentro do microservice.
4. [AuthUser Microservice] Em application-prod.yaml, inserir configuração e variável de ambiente para produção:

```
spring:
  config:
    import: 'configserver:${CONFIG_SERVER_URL}'
```

5. [AuthUser Microservice] Na raiz do diretório da aplicação, inserir novo arquivo system.properties contendo a versão do java utilizado (java.runtime.version=11).
6. [AuthUser Microservice] Fazer o commit das alterações realizadas para ambiente de prod
7. [ConfigServer Repositório Git] Duplicar o arquivo ead-authuser-service-dev.yaml e renomear essa cópia para ead-authuser-service-prod.yaml
8. [ConfigServer Repositório Git] Neste novo arquivo ead-authuser-service-prod.yaml alterar as configurações para procfile prod e incluir variáveis de ambiente necessárias (lembrar de remover a configuração do ansi):

```
server:
  port: ${PORT}
```

```
  servlet:
    context-path: '/ead-authuser/'

spring:
  application:
    name: ead-authuser-service
  datasource:
    url:  ${JDBC_DATABASE_URL}
    username: ${JDBC_DATABASE_USERNAME}
    password: ${JDBC_DATABASE_PASSWORD}
  jpa:
    hibernate:
      ddl-auto: update
      dialect: org.hibernate.dialect.PostgreSQLDialect
      jdbc:
        lob.non-contextual-creation: true
    properties:
      hibernate:
        show_sql: true
  rabbitmq:
    addresses: ${CLOUDAMQP_URL}

ead:
  api:
    url:
      course: 'http://ead-course-service/ead-course'
  broker:
    exchange:
      userEvent: ead.userevent

authuser:
  refreshscope:
    name: Michelli Brito UPDATE!!!!!!!

management:
  endpoints:
    web:
      exposure:
        include:
          - refresh
          - health

resilience4j:
  circuitbreaker:
    instances:
      circuitbreakerInstance:
        slidingWindowSize: 30
        permittedNumberOfCallsInHalfOpenState: 2
        slidingWindowType: TIME_BASED
        minimumNumberOfCalls: 2
        waitDurationInOpenState: 30s
        failureRateThreshold: 80
  retry:
    instances:
      retryInstance:
        maxRetryAttempts: 3
        waitDuration: 5s
```

9. Fazer o commit das alterações e subir para o Github.

3. **Deploy** de AuthUser Microservice no Heroku Platform

1. Realizar login no heroku via terminal: **heroku login**
2. Criar app heroku utilizando o seguinte command line via terminal:
    a. **heroku create -a <app-name> --remote heroku-prod**
    b. Verificar a criação do app no dashboard do Heroku
    c. Para verificar o endereço do repositório remoto git da app dentro do heroku que foi criada utilizar **git remote -v** ou visualizar a url em heroku -> Settings (Opcional)
    d. Inserir Add-on CLOUDAMQP plano free utilizado o seguinte command line via terminal: **heroku addons:create cloudamqp:lemur**
    e. No Heroku, em Resources verificar Add-on inserido e variáveis de ambientes criadas

3. No Heroku, em Settings -> Reveal Config Vars do app criar variáveis de ambiente necessárias (key-value):

    a. APP_DOMAIN_NAME : <app-name>.herokuapp.com
    b. CONFIG_SERVER_URL : https://username:password@<app-name>.herokuapp.com

4. Realizar o deploy enviando o código-fonte para o repositório remoto Git do Heroku criado no passo 3.2(a) utilizando o seguinte comando: **git push heroku-prod prod:master**
5. Para verificar logs pode-se utilizar o comando: **heroku logs –tail** (opcional)
6. Pode-se verificar logs no dashboard do Heroku também clicando no botão More -> View logs
7. Verificar em Resourses o Add-on criado para Database Postgres
8. Obter Heroku CLI nas credenciais da database criada e acessar remotamente via terminal -> verificar tabelas criadas com o comando \d
9. Executar scripts para criar Roles:

    insert into tb_roles values ('c4555c56-76ab-474f-aeae-f2a414ab97e3', 'ROLE_ADMIN');
    insert into tb_roles values ('83989363-03e2-4160-82ac-6e0a10fd2248', 'ROLE_INSTRUCTOR');
    insert into tb_roles values ('24d85f67-acd5-4ee5-a38a-29eeb5c2edfe', 'ROLE_STUDENT');
    insert into tb_roles values ('e96f43d9-22c5-4a6f-b65e-0aa7c9dfa952', 'ROLE_USER');

10. Verificar registro do authuser no ServiceRegistry acessando o dashboard Eureka
11. Voltar em Resources e alterar dynos para o plano Hobby Dev de 7 dólares (opcional).