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

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

1. [ApiGateway Microservice] Criar branch local dev: **git checkout -b dev**
2. [ApiGateway Microservice] Inserir dependência do Spring Cloud Config Client
3. [ApiGateway Microservice] Renomear o arquivo application.yaml para application-dev.yaml para propriedades exclusivas do ambiente de dev que são diferentes em outros ambientes e inserir as configurações do configserver para dev:

```yaml
spring:
  config:
    import: 'configserver:'
  cloud:
    config:
      discovery:
        serviceId: ead-config-server
      username: configserver
      password: 'ead123cs'
```

4. [ApiGateway Microservice] Criar outro arquivo application.yaml apenas com configurações de spring.application.name, recortar essa propriedade de application-dev.yaml;
5. [ApiGateway Microservice] No arquivo application.yaml criado no passo 1.4 acima, incluir spring.profiles.active=dev;
6. [ApiGateway Microservice] Fazer o commit das alterações;
7. [ConfigServer Repositório Git] Criar o arquivo ead-api-gateway-dev.yaml e inserir configurações de port e rotas do api-gateway:

```yaml
server:
  port: 8080

spring:
  application:
    name: ead-api-gateway
  cloud:
    gateway:
      routes:
        - id: authuser-service
          uri: lb://EAD-AUTHUSER-SERVICE
          predicates:
            - Path=/ead-authuser/**
        - id: course-service
          uri: lb://EAD-COURSE-SERVICE
          predicates:
            - Path=/ead-course/**
        - id: notification-service
          uri: lb://EAD-NOTIFICATION-SERVICE
          predicates:
            - Path=/ead-notification/**
```

8. [ConfigServer Repositório Git] Fazer commit das alterações e subir para o Github.

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

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

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

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

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

spring:
  application:
    name: ead-api-gateway
  cloud:
    gateway:
      routes:
        - id: authuser-service
          uri: lb://EAD-AUTHUSER-SERVICE
          predicates:
            - Path=/ead-authuser/**
        - id: course-service
          uri: lb://EAD-COURSE-SERVICE
          predicates:
            - Path=/ead-course/**
        - id: notification-service
          uri: lb://EAD-NOTIFICATION-SERVICE
          predicates:
            - Path=/ead-notification/**
```

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

3. **Deploy** de ApiGateway 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)

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