Formulário de proposta de solução - Spring Boot e Teste: Profiles, Testes e Deploy

Todas as respostas dadas pelos alunos e alunas do programa Orange Talent podem ser acessadas por mentores e programa managers que acompanham as turmas. Mas, como esse endereço reside no mesmo sistema onde os próprios alunos e alunas respondem às avaliações, é necessário controlarmos o acesso. Tanto aluno, como mentor, como program manager fazem login no mesmo sistema mas, por conta do nível de acesso, tem acessos a endereços diferentes.   
  
Além disso, agora, para cada resposta dada por um aluno ou aluna, o sistema deve enviar um email para os(as) program managers avisando do evento. Só que tem um detalhe, em ambiente de desenvolvimento esse envio deve ser apenas de brincadeira, um mero print exibindo o email que deveria ser enviado. Só que em produção, o email deveria ser enviado usando o provedor de envio de emails que a Zup utiliza.

Descreva quais seriam os passos para você, usando o Spring Security, proteger os acessos em função do nível de acesso(roles) de cada tipo de usuário logado no sistema.   
Descreva quais seriam os passos para você implementar o suporte a envios de emails de maneiras diferentes em função do ambiente de execução.

Então você pode querer desabilitar. Eu quero que tenha envio de e-mails só no ambiente de produção; no ambiente de desenvolvimento eu quero desligar essa funcionalidade. Eu não quero ficar fazendo envio de e-mails enquanto estou testando, enquanto estou desenvolvendo meu software. Esse seria outro exemplo de uma funcionalidade que, dependendo do ambiente, está desligada ou ligada.

Para proteger os acessos em função do nível de acesso(roles) de cada tipo de usuário logado no sistema usaria o método hasRole.

Na classe de configuração de segurança no método configure HttpSecurity, que é responsável pela configuração de autorização, usaria o http.authorizeRequests().antMatchers(HttpMethod) declarando o método http que é referente a requisição como no caso para liberar o acesso das respostas respondidas apenas para os mentores seria um get, para trazer as respostas dos alunos, passando o endereço da requisição seguido do método hasRole e recebendo o tipo do perfil que terá o acesso a essa requisição.

Para implementar o suporte a envios de emails de maneiras diferentes em função do ambiente de execução utilizaria o recurso de profile, onde o spring da todo o suporte para cada ambiente ter configurações e coisas habilitadas ou desabilitadas.

Primeiro iria duplicar a classe responsável por enviar os emails e nela apenas deixar uma funcionabilidade de apenas exibir um print do email que deveria ser enviado, nessa classe usaria a anotação @Profile(“desenvolvimento”) indicando que essa será a chamada quando estiver em ambiente de desenvolvimento, na outra classe também vou usar a anotação mais recebendo o parâmetro produção @Profile(“produção”) indicando que essa será a chamada no ambiente de produção.

Para indicar qual profile usar no caso de estar em produção entraria em Run Configurations, na aba Arguments, no campo VM arguments e colocaria as informações Dspring.profiles.active= desenvolvimento, então com isso o Spring já sabe que só vai carregar as classe se o *profile* ativo for o de desenvolvimento.



**Spring Boot e Teste: Profiles, Testes e Deploy**

|  |
| --- |
| **Cenário:** |
| Todas as respostas dadas pelos alunos e alunas do programa Orange Talent podem ser acessadas por mentores e programa managers que acompanham as turmas. Mas, como esse endereço reside no mesmo sistema onde os próprios alunos e alunas respondem às avaliações, é necessário controlarmos o acesso. Tanto aluno, como mentor, como program manager fazem login no mesmo sistema mas, por conta do nível de acesso, tem acessos a endereços diferentes.  Além disso, agora, para cada resposta dada por um aluno ou aluna, o sistema deve enviar um email para os(as) program managers avisando do evento. Só que tem um detalhe, em ambiente de desenvolvimento esse envio deve ser apenas de brincadeira, um mero print exibindo o email que deveria ser enviado. Só que em produção, o email deveria ser enviado usando o provedor de envio de emails que a Zup utiliza.   * Descreva quais seriam os passos para você, usando o Spring Security, proteger os acessos em função do nível de acesso(roles) de cada tipo de usuário logado no sistema. * Descreva quais seriam os passos para você implementar o suporte a envios de emails de maneiras diferentes em função do ambiente de execução. |

|  |
| --- |
| **O que seria bom ver nessa resposta?** |

* **Peso 2:**Definição de role para cada tipo de usuário que teria acesso ao sistema.
* **Peso 2:**Associação das roles para os endereços, de forma a controlar o acesso.
* **Peso 6:**Descrição detalhada do uso dos profiles para definir qual bean seria carregado em função do ambiente.
* **Peso 3:**Utilização da annotation Profile para ter beans carregados diferentes ou uso do próprio Environment para a tomada de decisão.
* **Peso 3:**Passagem do argumento indicando o profile desejado para load da aplicação

|  |
| --- |
| **Resposta do Especialista:** |
| * *Definição de Roles:*Para cada novo usuário do sistema já seria associada uma Role diferente. Exemplo: Aluno, Mentor, Program Manager. Essa Role é uma implementação da interface GrantedAuthority. * *Definição de Roles:*Para cada endereço mapeado para ser protegido, vai ter também agora a associação das roles que tem acesso. No endereço de análise das respostas, as roles que terão acesso são as de Mentor e Program Manager. * *Sobre os profiles:*Criaria uma classe anotada com @Configuration com um método marcado com @Bean que receberia como argumento um Environment. A partir desse argumento o código faria um if para decidir qual implementação de email seria criada. Uma outra opção seria criar dois métodos diferentes anotados com @Profile referenciando os ambientes definidos pelo projeto. * *Sobre os profiles:*Eu deixaria o profile default no arquivo application.properties e passaria como argumento de produção o profile correto a ser utilizado em tal ambiente. Outra opção seria sempre passar o argumento de definição por linha de comando, mesmo em ambiente de dev. As ides suportam isso. |