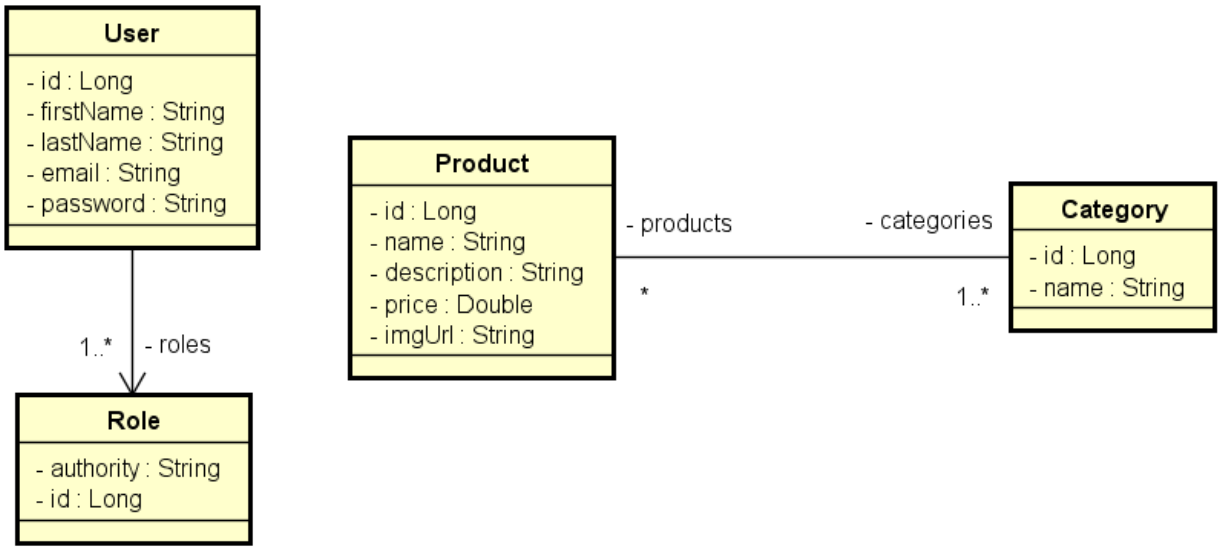
Bootcamp Spring React 3.0 - Cap. 03

Validação e segurança

Competências

- Modelo de dados de usuários e perfis
- Validação com Bean Validation
  - Annotations
  - Customizando a resposta HTTP
  - Validações personalizadas com acesso a banco
- Autenticação e autorização
  - Spring Security
  - OAuth 2.0
  - Token JWT
  - Autorização de rotas por perfil
- Dicas para Postman
- Variáveis de ambiente no projeto com coalescência

Modelo conceitual do DSCatalog



Referências sobre Bean Validation

https://beanvalidation.org/

https://docs.jboss.org/hibernate/beanvalidation/spec/2.0/api/overview-summary.html

https://docs.jboss.org/hibernate/beanvalidation/spec/2.0/api/javax/validation/constraints/package-summary.html

https://www.baeldung.com/java-bean-validation-not-null-empty-blank

https://www.baeldung.com/spring-custom-validation-message-source

https://pt.stackoverflow.com/questions/133691/formatar-campo-cpf-ou-cnpj-usando-regex

https://regexlib.com/

https://regexr.com/

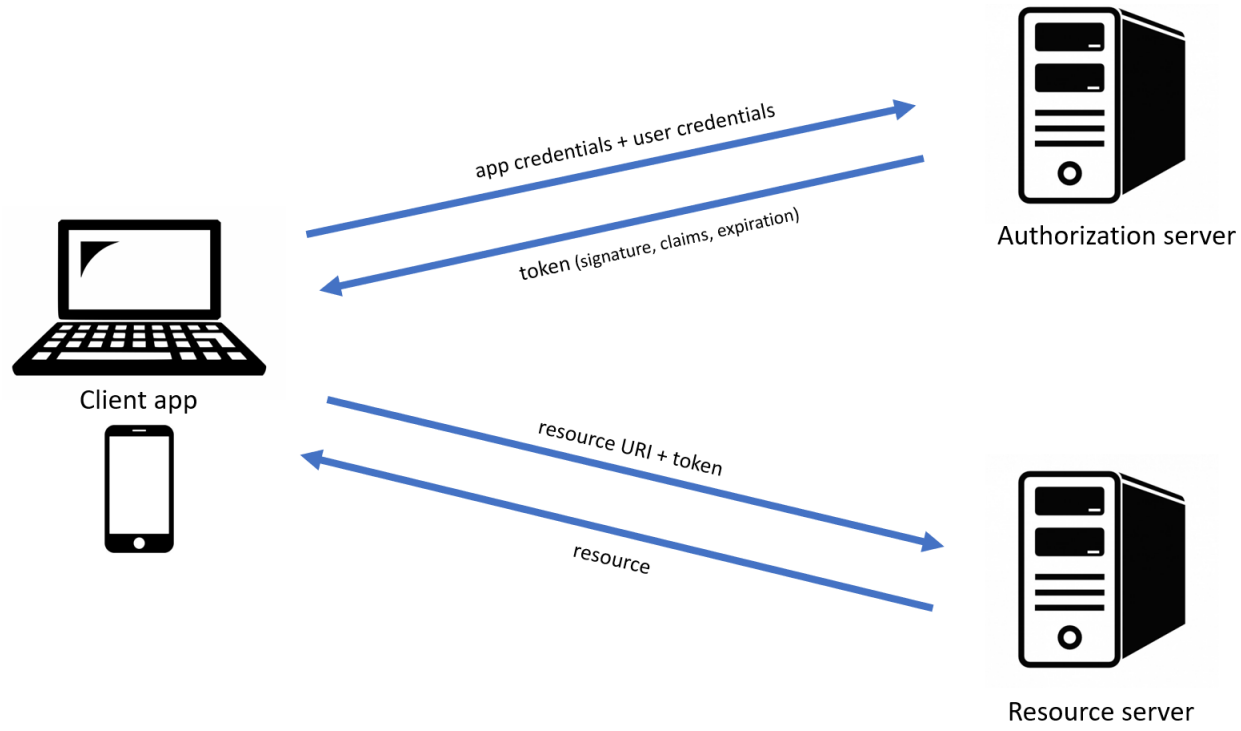
Referências token JWT, autenticação e autorização

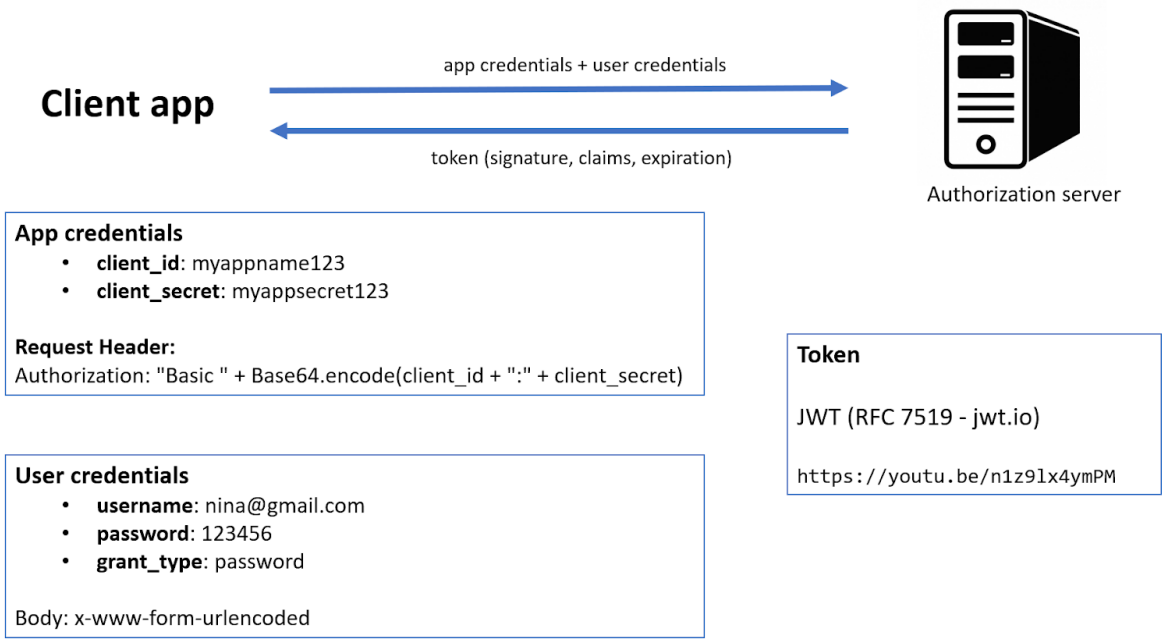
https://jwt.io

https://www.youtube.com/watch?v=n1z9lx4ymPM

OAuth 2.0

https://oauth.net/2/





Spring Security

Interfaces que devem ser implementadas

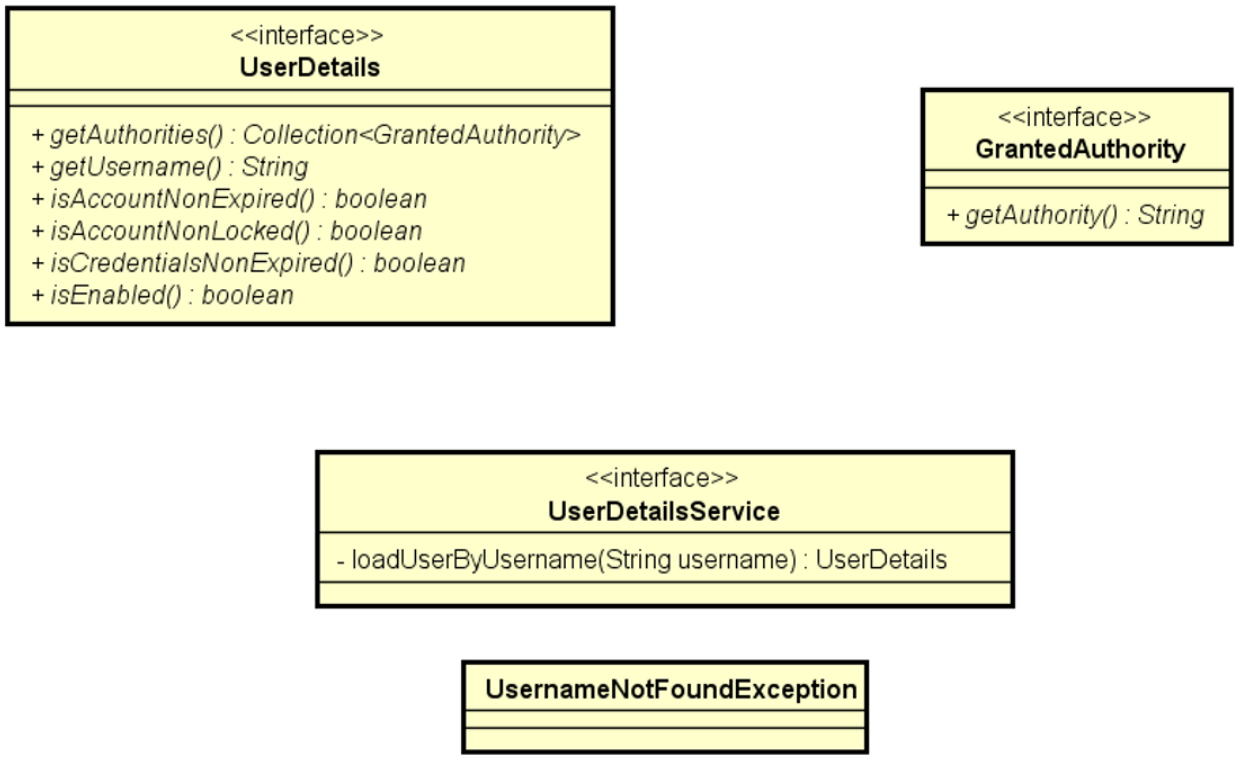
UserDetails
UserDetailsService

Classe para configuração de segurança web

WebSecurityConfigurerAdapter

Bean para efetuar autenticação

AuthenticationManager



Spring Cloud OAuth2

Classe de configuração para Authorization Server

AuthorizationServerConfigurerAdapter

Classe de configuração para Resource Server

ResourceServerConfigurerAdapter

Beans para implementar o padrão JWT

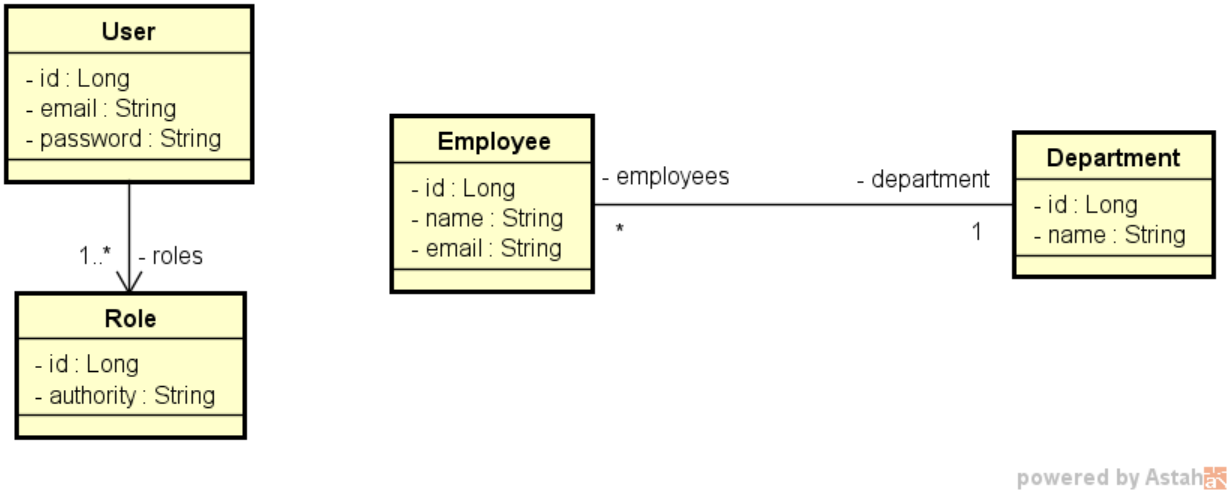
JwtAccessTokenConverter
JwtTokenStore

Desafio resolvido

Implemente as funcionalidades necessárias para que os testes do projeto abaixo passem:
https://github.com/devsuperior/bds03

Collection do Postman:
https://www.getpostman.com/collections/9c19a0ad21eb8a7d864a

Este é um sistema de funcionários e departamentos com uma relação N-1 entre eles:



Neste sistema, **todas** as rotas são protegidas. Usuários **ADMIN** podem ler e alterar recursos, enquanto que usuários **OPERATOR** podem apenas ler.

Validações de Employee:

- Nome não pode ser vazio
- Email deve ser válido
- Departamento não pode ser nulo

Passos para resolver:

- Modelo de dados User-Role
- Incluir infraestrutura de validação ao projeto
- Incluir infraestrutura de segurança ao projeto
- Implementar as funcionalidades

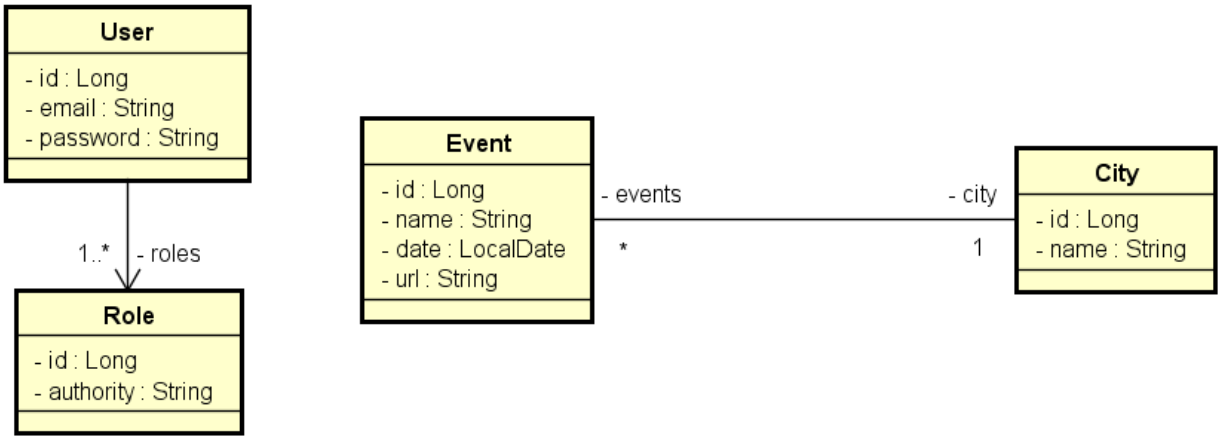
Desafio para entregar

TAREFA: Validação e Segurança

Implemente as funcionalidades necessárias para que os testes do projeto abaixo passem:
https://github.com/devsuperior/bds04

Collection do Postman:
https://www.getpostman.com/collections/e1f59c905aeca84c1ebc

Este é um sistema de eventos e cidades com uma relação N-1 entre eles:



Neste sistema, somente as rotas de leitura (GET) de eventos e cidades são **públicas** (não precisa de login). Usuários **CLIENT** podem também inserir (POST) novos eventos. Os demais acessos são permitidos apenas a usuários **ADMIN**.

> Caso tenha dúvidas nas regras de autorização do ResourceServerConfig, colocamos uma sugestão em linguagem natural na próxima página.

Validações de City:

- Nome não pode ser vazio

Validações de Event:

- Nome não pode ser vazio
- Data não pode ser passada
- Cidade não pode ser nula

Mínimo para aprovação: 9/12

Regras de autorização do ResourceServerConfig descritas em linguagem natural.

1) Os endpoints de login e do H2 devem ser públicos
2) Os endpoints GET para /cities e /events devem ser públicos
3) O endpoint POST de /events devem requerer login de ADMIN ou CLIENT
4) Todos demais endpoints devem requerer login de ADMIN

Spoiler: outros tópicos sobre autenticação e autorização veremos adiante no projeto DSLearn

- Autorização customizada em nível de serviço. Exemplo:
  - Usuário pode acessar user/{id} somente se for o id dele
  - Admin pode acessar user/{id} de todos usuários
- Conteúdo customizado para usuário logado. Exemplo:
  - Ao acessar /notifications, devem ser retornadas somente as notificações do próprio usuário logado
- Refresh token
- Pré-autorizando métodos por perfil de usuário