Módulo Edgar

Funcionalidades

Login

- URL de acesso: http://localhost:8080/desafio/login
- Componentes:
  - **Formulário de login**:
    - `Usuário`: campo do formulário no qual o usuário informará seu nome.
    - `Senha`: campo do formulário no qual o usuário informará sua senha.
    - `Acessar`: botão do formulário que efetuará a autenticação do usuário.
- Regras de interface:
  - Se o usuário for autenticado com sucesso, redirecioná-lo a página de consulta de pessoas.
- Mensagens de alerta:
  - `Usuário e senha inválidos.`: Mensagem exibida caso o usuário e senha não estejam corretos.

Cadastro de pessoa

- URL de acesso:
  - Inclusão: http://localhost:8080/desafio/pessoa

    - Alteração: http://localhost:8080/desafio/pessoa/id (na qual `id` é uma variável representando a chave primária da pessoa sendo alterada)
- Componentes:
    - **Formulário de pessoa**: formulário para cadastro de pessoa.
        - `Nome`: campo do formulário no qual o usuário informará o nome da pessoa.
        - `CPF`: campo do formulário no qual o usuário informará o CPF da pessoa.
        - `Nascimento`: campo de formulário no qual o usuário informará a data de nascimento da pessoa.
        - `Currículo`: campo do formulário no qual o usuário informará os dados do currículo da pessoa, sendo um texto sem limites de tamanho.
        - `Salvar`: botão do formulário que salvará os dados da pessoa no banco de dados.
    - **Formulário de endereço**: formulário para cadastro de endereço.
        - `CEP`: campo do formulário no qual o usuário informará o CEP do endereço.
        - `Número`: campo do formulário no qual o usuário informará o número do endereço.
        - `Adicionar`: botão do formulário que associará o endereço à pessoa.
    - **Tabela de endereços**: lista com os endereços da pessoa.
        - `Principal`: coluna com um seletor pelo qual o usuário especificará o endereço principal da pessoa.
        - `Logradouro`: coluna com o logradouro do endereço.
        - `Número`: coluna com o número do endereço.
        - `Complemento`: coluna com o complemento do endereço.
        - `Bairro`: coluna com o bairro do endereço.
        - `Cidade`: coluna com a cidade do endereço.
        - `UF`: coluna com a UF do endereço.
        - `CEP`: coluna com o CEP do endereço.
        - `Ações`: coluna contendo opções que o usuário poderá executar.
            - `Excluir`: botão que desassociará o endereço da pessoa.
- Regras de interface:
    - O campo `CPF` deve ser preenchido com máscara (999.999.999-99).
    - O campo `CEP` deve ser preenchido com máscara (99.999-99).
    - Se os dados da pessoa forem salvas com sucesso, redirecionar para a página de consulta de pessoas.
- Mensagens:
    - `CPF já existe.`: mensagem exibida quando o CPF já existir no banco de dados.
    - `CPF inválido.`: mensagem exibida quando o CPF for inválido.
    - `CEP inválido.`: mensagem exibida quando o CEP for inválido.
    - `CEP não encontrado.`: mensagem exibida quando o CEP não for encontrado no *ViaCEP*.
    - `Campo XXX é obrigatório.`: mensagem exibida quando faltar algum campo obrigatório na qual XXX é um variável que representa o nome do campo.
    - `Data de nascimento inválida.`: mensagem exibida quando a data de nascimento não for válida.
    - `Informe um endereço principal.`: mensagem exibida quando não há endereço principal associado.

- Regras de negócio:
  - Todos os campos são obrigatórios exceto pelo currículo da pessoa.
  - A data de nascimento de ser a data atual ou uma data anterior.
  - Uma pessoa pode ter vários endereços e um endereço pode pertencer a várias pessoas.
  - O sistema deve pesquisar o CEP e o número do endereço no banco da dados.
    - Caso exista, o endereço cadastrado deve ser associado à pessoa.
    - Caso não exista, os dados do endereço devem ser buscados utilizando o serviço *ViaCEP*, inseridos no banco de dados e o endereço deve ser associado à pessoa.
  - Não é necessário validar o dígito verificador do CPF, apenas se é formado por 11 números.
  - O CEP e o número do endereço devem ser únicos no banco de dados.
  - O CPF deve ser único no banco de dados.
  - Uma deve deve possuir um endereço principal.

Consulta de pessoa

- URL de acesso: http://localhost:8080/desafio/pessoas
- Componentes:
  - **Filtro**: campo de formulário no qual o usuário informará o nome a ser filtrado.
  - **Tabela de pessoas**:
    - `Nome`: coluna com o nome da pessoa.
    - `CPF`: coluna com o CPF da pessoa.do sistema
    - `Nascimento`: coluna com a data de nascimento da pessoa.
    - `Endereço`: coluna contendo os seguintes dados do endereço principal separados por hífen: logradouro, número, bairro, cidade e UF.
    - `Ações`: coluna contendo opções que o usuário poderá executar.
      - `Alterar`: botão que carregará os dados da pessoa na tela de cadastro de pessoa para alteração.
      - `Excluir`: botão que excluirá os dados de uma pessoa do banco de dados.
  - **Paginação**: componente que faz a paginação da tabela de pessoas, caso essa estoure o limite de 10 registros.
- Regras de interface:
  - O CPF deve ser exibido com máscara (999.999.999-99).
  - A tabela de pessoas deve ser ocultada caso esteja vazia.
  - Deverá aparecer uma *popup* de confirmação quando o usuário clicar no botão de exclusão.
- Mensagens:
  - `Nenhum registro encontrado.`: mensagem exibida caso a tabela de pessoas esteja vazia.
  - `Confirmar a exclusão da pessoa?`: mensagem exibida ao clicar no botão de exclusão.

Requisitos de segurança

- O usuário pode ter um de dois perfis: `administrador` ou `relator`.
- Caso o perfil seja relator, o usuário poderá acessar todas as telas e dados mas não poderá realizar nenhuma alteração neles.
- O usuário só poderá acessar a tela de login (http://localhost:8080/desafio/login) caso ainda não esteja autenticado.
- Guarde a senha encriptada no banco de dados.

Requisitos técnicos

- O projeto deve ser configurado com *Maven*.
  - O pacote `jar` do projeto deve ser construído com o comando `mvn clean package` e deve ter o nome `edgar.jar`.
- O projeto deve utilizar o *H2 Database* como banco de dados.
  - A *URL* do *JDBC* deve ser `jdbc:h2:tcp://localhost:9092/~/.h2/pde/edgar`.
  - O projeto, ao ser inicializado, deve criar o banco de dados e realizar uma carga inicial com ao menos 15 pessoas e 3 endereços.
  - O banco de dados pode ser destruído na inicialização ou no desligamento da aplicação.
  - O banco de dados deve ser carregado com os seguinte usuários:
    - Usuário: `adm` | Senha: `123` | Perfil: `administrador`
    - Usuário: `rel` | Senha: `123` | Perfil: `relator`
- O projeto deve utilizar os seguintes *frameworks*:
  - Angular 8
  - Spring Boot 2.1
  - *Hibernate 5*
  - *Bootstrap 4*
- Você pode adicionar bibliotecas utilitárias que julgar necessárias, desde que justifique o seu uso na entrega do projeto.
- O pacote `jar` gerado pelo *Maven* deve rodar através do comando `java -jar edgar.jar`

Requisitos de arquitetura

- A aplicação deve possuir uma barra de menu superior possibilitando acesso às paginas de cadastro de pessoa e, consulta de pessoa e realização de logout.
- O menu superior deve ser construído como *template* das páginas de cadastro e consulta.
- O sistema deve ser responsivo, i.e., deve se adaptar a telas pequenas *(mobile)*.
- Cidade e UF não precisam estar normalizadas no banco de dados.
- Procure economizar ao máximo o tráfego de dados com o banco, buscando apenas os dados necessários à funcionalidade específica.
- Não utilize *SQL* nativo, utilize *JPQL* ou *Criteria* ou Spring Data.
- No acesso ao endereço do contexto (http://localhost:8080/desafio), redirecione para a tela de login (caso não esteja autenticado) ou para a tela de consulta de pessoas (caso contrário).
- De preferência, mostre mensagens de erro próximas ao campo referente usando o *Bootstrap Form Validation*.