Projeto Vem Ser DBC - 9ª Edição
Descrição de trabalho final

RetroCards DBC

Resumo

Seu grupo deve criar um sistema para gerenciar reuniões de retrospectiva e também de Kudo Box/Cards

Perfis

O sistema terá apenas um tipo de usuário, que poderá desempenhar dois papéis:

- Facilitador (pessoa que cria sprints e que pode criar reuniões de retro e kudo boxes)
- Membro do time (pessoa que pode criar itens de retrô e kudo cards)

Lista de requisitos

1. Página inicial
    a. A página inicial do sistema, caso o usuário não esteja logado, é a de login (item 16)
    b. Caso o usuário esteja logado, a tela inicial do sistema exibirá:
        i. Botão para criar novo sprint (item 4)
        ii. Lista de sprints (item 2)
        iii. Link para a retrospectiva mais recente em aberto, caso exista (um sprint pode ter várias retrospectivas, mas somente uma reunião de retrospectiva em aberto por vez)
        iv. Listagem de kudo cards (item 3)
2. Lista de sprints
    a. Lista ordenada por data de conclusão do sprint desc. Ou seja, o sprint mais recente vem antes.
    b. Cada item da lista deve conter as seguintes informações do sprint:
        i. ID
        ii. Título
        iii. Data de conclusão
    c. Ao clicar em um sprint, deverá ser exibida a lista de retrospectivas. Além da listagem, deverá ser colocado, em algum ponto, um botão para criar nova retrospectiva, de acordo com item 5. Abaixo seguem as colunas da listagem de retrospectivas:
        i. ID da retrospectiva
        ii. Título
        iii. Data em que ocorreu / ocorrerá a reunião
        iv. Status (Criada, Em andamento, Concluída)
        v. Quantidade de itens apontados (0 caso nenhum, óbvio)

vi. Caso não haja nenhuma retrospectiva EM ANDAMENTO no sprint: exibir botão “Iniciar” e chamar a ação do item 6.
vii. Caso seja a retrospectiva em andamento no sprint: exibir botão “Concluir” e chamar a ação do item 9.

3. Listagem de Kudo Cards
    a. Deverá ser agrupada por sprint e de forma descendente (ou seja, sprints mais recentes antes). Na listagem de kudos do sprint, a ordenação é por data de criação do kudo card ascendente. Ou seja, mais antigo primeiro.
    b. Ao clicar no kudo card, deve ser exibido o detalhe dele (item 15).
    c. Colunas da listagem:
        i. ID
        ii. Título
        iii. Data de criação
        iv. De (usuário que escreveu)
        v. Para (usuário que recebeu)

    Ex:

    Sprint 10

    - 97 - Parabéns pelo esforço no frontend! - 10/04/2019 13:45 - Fulano - Ciclana
    - 108 - Obrigado pela ajuda nos testes - 12/04/2019 17:45 - Fulano - Beltrano
    - 119 - Mandou muito no Spring! - 14/04/2019 13:45 - Ciclana - Fulano

    Sprint 9

    - 81 - Parabéns pela certificação! - 04/04/2019 13:45 - Fulano - Ciclana
    - 85 - Você foi essencial essa semana! - 05/04/2019 13:45 - Fulano - Ciclana

4. Criar Sprint
    a. Tela para criar um novo sprint no sistema. É a partir do sprint que criamos as retrospectivas (podemos ter mais de uma por sprint, somente uma em aberto) e também os kudo boxes (podemos ter mais de um por sprint, mas somente um aberto). Campos da tela (todos obrigatórios):
        i. Título
        ii. Data de início
        iii. Data de conclusão (não pode ser menor que a data de início)
5. Criar retrospectiva
    a. Tela para criar uma retrospectiva, dentro de um sprint. Campos da tela (todos obrigatórios):
        i. Título
        ii. Data da reunião
    b. Ao ser criada, uma retrospectiva obtém o status CRIADA.

6. Iniciar retrospectiva
    a. Caso não exista nenhuma outra retrospectiva EM ANDAMENTO para o sprint, altera o status para EM ANDAMENTO e navega até o detalhe da retrospectiva, conforme item 13.
    b. Caso contrário, informar essa situação de erro ao usuário.
7. Criar item de retrospectiva
    a. O usuário deve preencher os seguintes campos (todos obrigatórios):
        i. Tipo (O que funcionou bem? / O que pode ser melhorado ? / O que faremos no próximo sprint para melhorar?)
            1. Este campo pode ser dispensado se o detalhamento da retrospectiva for feito em estilo “Trello”, onde as colunas podem ser os tipos e quando o usuário clicar no “+” de cada coluna, já sabe qual é o tipo
        ii. Título
        iii. Descrição
8. Apagar item de retrospectiva
    a. Qualquer pessoa pode apagar itens criados em uma retrospectiva, desde que ela esteja EM ANDAMENTO e mediante confirmação da UI.
9. Concluir retrospectiva

Ao clicar neste botão, o sistema deve pedir uma lista de emails destinatários para enviar o relatório da retrospectiva. Após preenchimento da lista e confirmação de ação, o sistema envia um email no seguinte template para a lista de destinatários:

> Destinatário: <lista-de-destinatarios>
> Assunto: [RetroCards - Retrospectiva concluída!] <ID> - <TITULO>
> Corpo:
>
> > Olá,
> >
> > A Retrospectiva <ID> - <TITULO> foi concluída! Veja os itens apontados pelo time:
> >
> > O que funcionou bem?
> > <LISTA-DO-QUE-FOI-BOM>
> >
> > O que pode ser melhorado?
> > <LISTA-DO-QUE-PODE-SER-MELHORADO>
> >
> > O que faremos no próximo sprint para melhorar?
> > <LISTA-DOS-COMPROMISSOS-PARA-PROXIMO-SPRINT>
> >
> > Clique aqui<LINK PARA DETALHE DA RETROSPECTIVA> para abri-la

10. Criar kudo box

    a. Tela para criar um kudo box, dentro de um sprint. Campos da tela (todos obrigatórios):
        i. Título
        ii. Data de leitura dos cartões (encerramento dos cadastros)
    b. Ao ser criado, um kudo box obtém o status CRIADO.
11. Criar kudo card
    a. O usuário deve preencher os seguintes campos:
        i. Título (Obrigatório)
        ii. Descrição (Obrigatório)
        iii. De (Opcional)
        iv. Para (Obrigatório)
12. Apagar kudo card
    a. O criador de um kudo card pode apagá-lo, desde que o status do kudo box esteja EM ANDAMENTO e mediante confirmação da UI.
13. Detalhe de uma retrospectiva
    a. Principal tela da retrospectiva, nela deve ser pensada uma forma de exibir os itens cadastrados até o momento e como criá-los / deletá-los.
14. Detalhe de um kudo box
    a. Principal tela do kudo box, nela deve ser pensada uma forma de exibir os itens cadastrados até o momento e como criá-los / deletá-los.
15. Detalhe do kudo card
    a. A tela de detalhe do kudo card exibirá os seguintes campos:
        i. ID
        ii. Título
        iii. Data de criação
        iv. Descrição
        v. De
        vi. Para
16. Login
    a. Deve ser implementada uma tela de login com campos usuário e senha (ambos obrigatórios)
    b. A tela de login deve ser acessada sempre que um usuário não estiver com uma sessão / token válido e tentar acessar alguma página
    c. Ao realizar login com sucesso, o usuário deve ser redirecionado para a tela inicial, descrita no item 1.
    d. Não é necessário fazer uma tela de cadastro de usuários. Insira eles direto na base.

Observações gerais

1. Frontend deve ser desenvolvido utilizando React e Context ou Redux (componentizado), podendo ou não utilizar Bootstrap.
2. Backend deve ser desenvolvido utilizando Spring Boot, com RestController, Service, Repository, Entity, respeitando a responsabilidade de cada classe. Deve ser utilizada autenticação com Token JWT.

3. Testes automatizados de Front-end devem ser desenvolvidos utilizando a ferramenta Selenium com Java ou Cypress com JavaScript.
4. Testes automatizados de API devem ser desenvolvidos utilizando a ferramenta REST-Assured com Java ou Cypress com JavaScript.
5. Os testes automatizados devem cobrir no mínimo 90% da aplicação.
6. Os usuários não podem ter acesso a dados de outros usuários, nem mesmo forçando por requisições no postman, a autenticação tem que validar os dados enviados com os dados de autenticação.
7. A aplicação deverá ser publicada no Heroku / Vercel
8. Deve ser criado um repositório no github
9. A data limite para commits do grupo no repositório é **23:59:59 do dia 22/08/2022**

Cronograma

- 12/08: início dos trabalhos
- 12/08: até 22/08: desenvolvimento
- 23/08: apresentações dos resultados para colegas, diretoria e staff
- 24/08: avaliação dos trabalhos pelos instrutores
- 25/08: banca técnica

Checklists

Para ajudar na avaliação dos trabalhos, utilizaremos dois checklists: um técnico backend, frontend e outro geral. Seguem os itens a serem avaliados e seus respectivos pesos.

Checklist técnico backend

Item	Peso
Complexidade de código (duplicação, princípios SOLID, performance, etc)	2
Legibilidade do código (nomes de variáveis, nomes de métodos, comentários úteis)	2
Estrutura do código (padrão de projeto MVC, RestController, Service, Repository, Entity, Pagination)	2
Modelagem de banco	2
Cobertura de testes unitários de todas as services	2

Checklist técnico frontend

Item	Peso
Complexidade de código (duplicação, princípios SOLID, performance, etc)	2
Legibilidade do código (nomes de variáveis, nomes de métodos, comentários úteis)	1
Estrutura do código React (componentização e boas práticas JavaScript)	3
Qualidade do HTML (legibilidade e semântica)	1,5
Qualidade do CSS (sem duplicação e com seletores otimizados)	1
Performance na utilização dos recursos web (CSS e JavaScripts desnecessários, imagens muito pesadas, requests HTTP muito pesadas)	1,5

Checklist de Qualidade de Software

Item	Peso
Plano de testes	2.0
Checklist Testes Automatizados API	
Item	**Peso**
Automação dos cenários positivos e negativos das funcionalidades	2.3
Organização do projeto - pastas, classes etc.	0.7
Títulos dos cenários com objetivos bem definidos	0.5
Reports de bugs	0.5
Checklist Testes Automatizados Front-End	
Item	**Peso**
Automação dos cenários positivos e negativos das funcionalidades	2.3
Organização do projeto - pastas, classes etc.	0.7
Títulos dos cenários com objetivos bem definidos	0.5
Reports de bugs	0.5

Checklist geral

Item	Peso
Funcionamento do software (quantidade de bugs, funcionalidades implementadas)	5

Trabalho em equipe (organização, comunicação, planejamento)	3
Engajamento dos participantes da equipe	2

Links úteis

https://trello.com/ - Micro gerenciador de tarefas
https://lucid.app/ - Lucid, modelagem e diagramação
https://app.diagrams.net/ - Drawio, modelagem e diagramação
https://balsamiq.com/ - Mockups e prototipação
https://proto.io/ - Mockups e prototipação
https://coverr.co/ - Vídeos gratuitos para utilizar de fundo
https://www.npmjs.com/package/react-password-strength - exemplo de senha forte
https://www.callicoder.com/spring-boot-file-upload-download-rest-api-example/ - upload de arquivos

Exemplo de upload de arquivo spring boot:
https://www.codejava.net/frameworks/spring-boot/spring-boot-file-upload-tutorial
https://mkyong.com/spring-boot/spring-boot-file-upload-example

Exemplo de upload de arquivo spring react:
https://www.pluralsight.com/guides/uploading-files-with-reactjs

Dicas

- Antes de commitar seu código, teste ele bem
- Após baixar o código dos colegas, teste a aplicação antes de começar a trabalhar (para evitar começar a trabalhar com problemas gerados por outros códigos)
- Estabeleçam metas diárias para concluir as coisas
- Não deixem para commitar as coisas muito em cima do deadline (limite de horário). Tentem estabelecer um horário de congelamento de código (ex: no último dia, após 12hs, ninguém mais commita, apenas se testa e faz correções urgentes)