markfy

Entregável de Java Advanced

O objetivo do projeto é desenvolver um sistema Java Spring Boot que gerencie compras de roupas e colete dados para treinar um modelo de inteligência artificial que futuramente servirá para recomendar roupas de acordo com o perfil da pessoa.

Melhorias do Sistema para Sprint 2

Primeiramente, aplicamos validações de dados utilizando o Bean Validation para garantir a integridade e consistência dos dados recebidos pela API. Em seguida, reorganizamos a documentação, tornando-a mais clara e acessível aos desenvolvedores, além de adicionar a documentação completa da API usando o Swagger, o que facilita a compreensão e o consumo dos endpoints disponíveis.

Refatoramos a camada de login para torná-la mais modular e fácil de manter, enquanto implementamos autenticação e autorização utilizando tokens JWT para garantir a segurança dos dados e dos usuários. Além disso, aplicamos filtros para interceptar requisições, permitindo a execução de ações específicas antes ou depois da execução dos endpoints.

Liberamos rotas do Swagger para autenticação com OAuth e integramos a autenticação diretamente na interface do Swagger, proporcionando uma experiência mais integrada e segura para os desenvolvedores. Também aprimoramos o retorno de exceções relacionadas à autenticação, fornecendo mensagens genéricas para os usuários, garantindo uma melhor experiência de uso.

Implementamos uma rota do Actuator para verificar o status da aplicação, garantindo sua disponibilidade e saúde. Além disso, desenvolvemos funcionalidades de envio de e-mail para recuperação de senha, juntamente com um endpoint para redefinição de senha, aumentando a segurança e a usabilidade da aplicação.

Por fim, integramos a API com o frontend, permitindo uma comunicação eficiente entre os sistemas backend e frontend. Essas melhorias e implementações contribuem significativamente para a qualidade, segurança e usabilidade da nossa aplicação.

Itens do Backlog concluídos nessa Sprint

[x] Bean validation

[x] Reorganizar a documentação

[x] Documentar a API com Swagger

[x] Refatorar camada de login

[x] Autenticação e autorização com JWT

[x] Filtros para interceptar as requisições

[x] Liberar rota do Swagger para oauth e doFilter

[X] Implementar autenticação pela interface do Swagger

[x] Retorno da excpetion do oauth generica para o usuário

[x] Rota de actuator para verificar se a aplicação está no ar

[x] Implementar envio de email para usuário caso ele tenha esquecido a senha

[x] Implementar endpoint para redefinição de senha

[ ] Sistema de roles pra bloquear os endpoint de cadastro de itens para o usuario

[ ] Autenticação com google e github

[ ] Bater em api para buscar dados automaticamento com o cpf

[x] Integração com frontend

[ ] Integração com IA [ ] Implementar backend para quem esqueceu a senha

Equipe

- **Bruno de Paula** (RM552226): Representante/gestor do projeto
- **Kayque Lima** (RM550782): Lider Técnico
- **Gabriel França** (RM551905): Desenvolvedor Frontend/Mobile
- **Edward de Lima** (RM98676): Desenvolvedor/redator

Ferramentas

As seguintes ferramentas serão utilizadas no desenvolvimento do projeto:

- Java
- Spring Boot
- Banco de dados relacional OracleSQL
- Biblioteca de IA
- Python
- Ferramenta de controle de versão: GitHub
- Ferramenta de gerenciamento de projetos: Trello

Cronograma detalhado – Sprint 2

1. Planejamento (20/04 a 01/05)

1.1 Refinamento da Arquitetura do sistema (20/04 a 26/04):

Responsável: Kayque

Atividade: Refinar a arquitetura de microsserviços que será utilizada no Sistema, incluindo o BFF, as aplicações da IA e as bases de dados.

Prazo: 26/04.

1.3 Planejamento de tarefas Sprint 2 (27/03 a 29/03):
Responsável: Kayque e Bruno

Atividade: Dividir o projeto em tarefas menores, estimar o tempo necessário para cada tarefa e definir quem será o responsável por cada uma delas.

Prazo: 26/04.

2. Desenvolvimento (30/03 a 26/04)

2.1 Implementação da camada de segurança no backend (30/04 a 03/05):

Responsável: Kayque e Edward

Atividade: Implementação da camada de segurança utilizando JWT, autenticação pelo Google e outras redes sociais.

Prazo: 15/04

2.2 Implementação do Bean Validation nas camadas de DTO (03/05 a 04/05)

Responsável: Kayque e Edward

Atividade: Implementação de validação dos dados recebidos pela API utilizando a biblioteca Bean Validation.

Prazo: 15/04

2.3 Mapear e implementar demais melhorias (03/05 a 04/05)

Responsável: Kayque e Edward

Atividade: Implementar demais melhorias nas tecnologias usadas, configurações e regras de negócio do backend

Prazo: 20/05

2.4 Implementação do aplicativo em React Native (20/04 a 05/05):

Responsável: Gabriel

Atividade: Implementação do MVP do aplicativo mobile usando React Native.

Prazo: 05/05.

2.5 Integração do mobile com a api do backend (20/04 a 05/05):

Responsável: Gabriel e Kayque

Atividade: Integração do Backend com a aplicação Mobile desenvolvida na matéria de Mobile Application Development

Prazo: 10/05.

2.6 Implementação do modelo de IA (27/04 a 03/05):
Responsável: Bruno e Edward

Atividade: Implementar primeira versão do modelo de inteligência artificial usando Python e outras bibliotecas de IA.

Prazo: 11/05

3. Documentação (08/05 a 11/05)

3.1 Documentação conceitual do Projeto e ideia (08/04 a 09/05):

Responsável: Bruno e Edward

Atividade: Escrever descrição da ideia, explicação de como funcionará e como a ideia pode contribuir com a empresa parceira.

Prazo: 09/05

3.2 Vídeo Pitch (08/04 a 10/05):

Responsável: Bruno

Atividade: Gravar vídeo Pitch falando sobre a ideia para adicionar na documentação.

Prazo: 10/05

3.2 Documentação do Backend implementado em Java (08/04 a 11/05):

Responsável: Kayque e Edward

Atividade: Escrever descrição do sistema, listagem dos endpoints, demonstração de uso do sistema, desenhos de arquitetura e observações sobre o backend Java.

Prazo: 11/05

Arquitetura do sistema

O sistema será desenvolvido usando uma arquitetura de microsserviços baseada em MVC e com um BFF em Java com Spring Boot. Seguindo as boas práticas do conceito de microsserviços, cada sistema terá sua própria base de dados inclusive o sistema responsável por treinar o modelo de IA que vai contar com um banco de dados não relacional (MongoDB). Na parte visual a principio vamos utilizar uma aplicação escrita em React Native para rodar em dispositivos mobile.

Abaixo temos os diagramas que representam o que seria o BFF do nosso sistema.

Diagrama de Entidade e Relacionamento (DER)

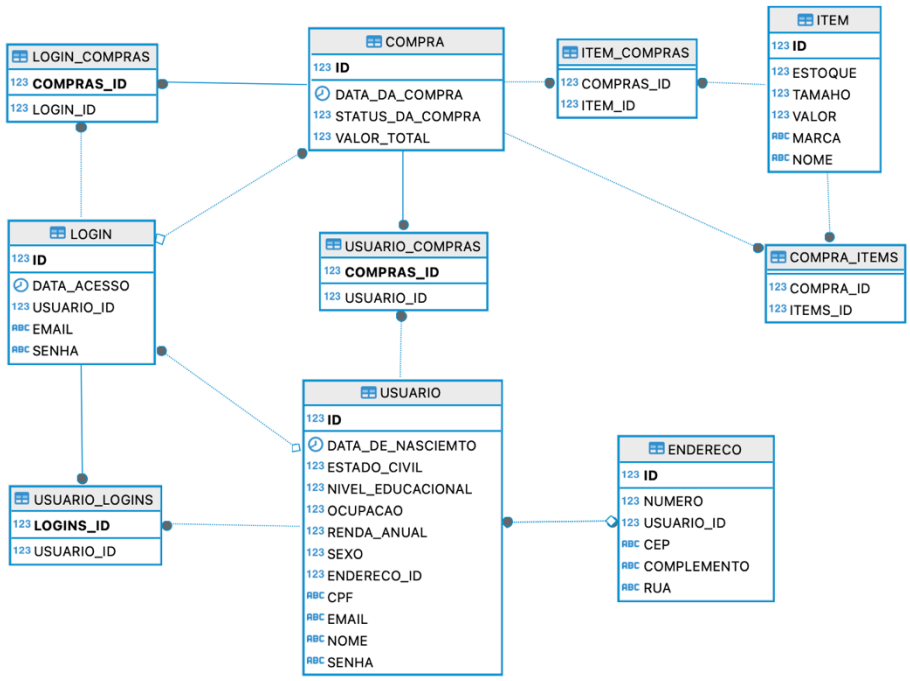

Diagrama de Classes