

RELATÓRIO FINAL

ARQUITETURA MVC

PROJETO: FOCUSLIST

Professor: André Felix

Disciplina: Arquitetura de Software

GAMA – DF



(61) 3035-3900



SIGA Área Especial para Indústria nº 02
Setor Leste - Gama - DF
CEP: 72445-020



Introdução

Durante as aulas, foi proposto que os grupos desenvolvessem uma atividade prática, consistindo na criação de um sistema utilizando o padrão de arquitetura MVC (Model-View-Controller). O professor apresentou alguns exemplos de sistemas simples que poderiam ser utilizados como base, permitindo também que os grupos optassem por outro tema, caso desejassem.

Nosso grupo entrou em conformidade de mesclar dois dos temas sugeridos: um gerenciador de tarefas e um sistema de login com autenticação simples. A partir disso, desenvolvemos uma aplicação na qual o usuário pode realizar o cadastro, efetuar o login e, posteriormente, utilizar o serviço de gerenciamento de tarefas.

Cada lista de tarefas é vinculada exclusivamente ao respectivo usuário, garantindo que apenas o proprietário tenha acesso às suas informações.

Tecnologias utilizadas

Esta aplicação foi desenvolvida pelas seguintes tecnologias:

Back-End:

1. Java – Linguagem de programação utilizada para o desenvolvimento da lógica da aplicação.
2. Spring framework e Spring Boot – Ferramentas que facilitam e agilizam o desenvolvimento de aplicações web e microsserviços com base no ecossistema Spring.

Banco de dados:

1. PostgreSQL - Sistema gerenciador de banco de dados relacional utilizado para armazenar e gerenciar as informações da aplicação.

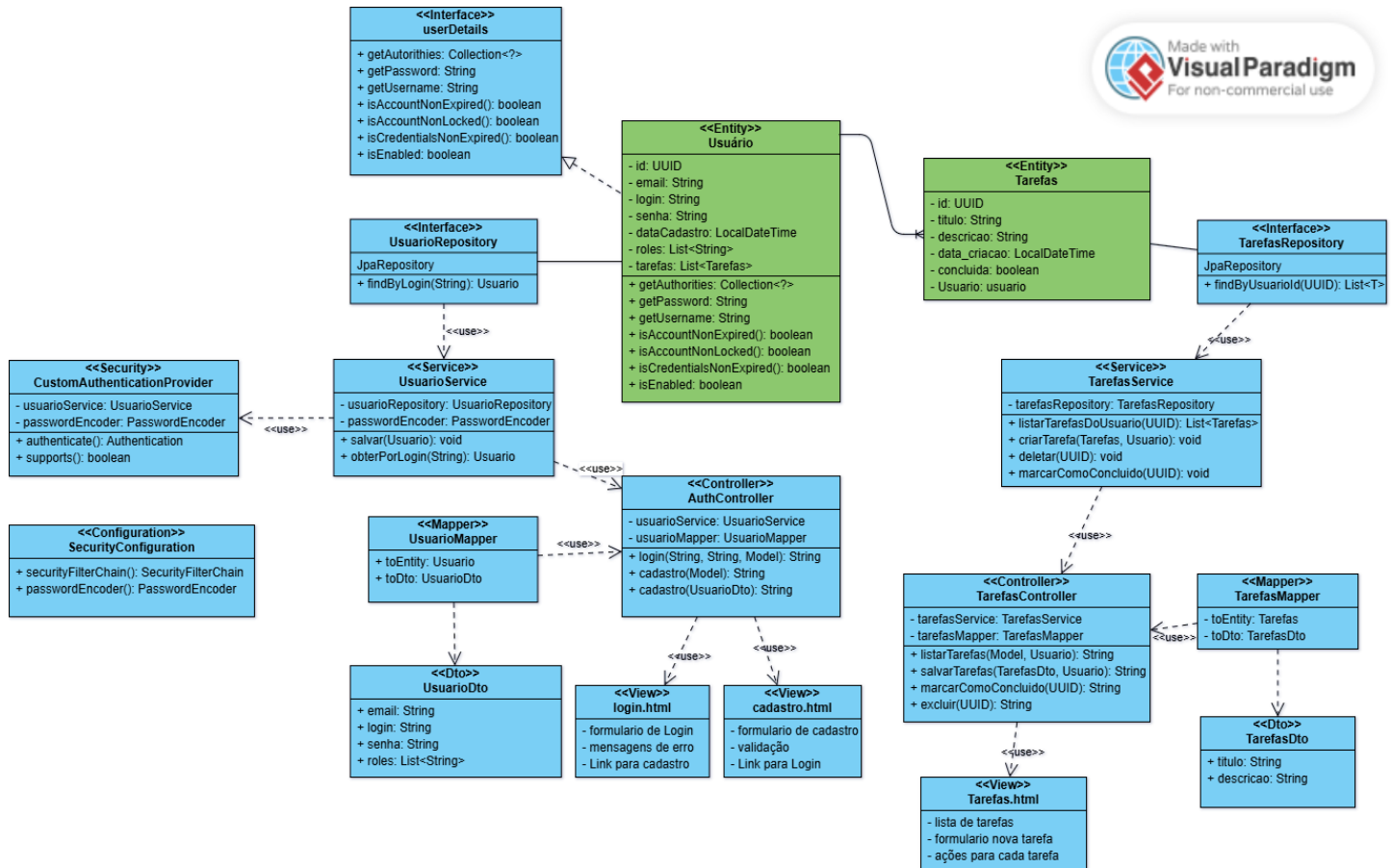
Front-End:

1. Html e css – Linguagens de marcação e estilização utilizadas na construção das páginas da interface do usuário.
2. Bootstrap – Framework CSS utilizado para o desenvolvimento de interfaces responsivas e com design consistente.
3. Thymeleaf – Biblioteca de templates para Java utilizada na renderização dinâmica das páginas HTML no lado do servidor.



Diagrama de classes

Segue abaixo o diagrama de classes, representando as camadas da aplicação:



Decisões de implementação

As decisões de implementação foram tomadas após uma análise de como seria o funcionamento da aplicação, o fluxo de usuário, o tratamento dos dados, como as rotas seriam controladas, e quais seriam as lógicas de negócio.

A partir destas decisões, começamos a desenvolver a aplicação começando pelo modelo de negócio, as entidades usuários e tarefas foram definidas com os seguintes atributos:

Usuário: id, email, login, senha e roles e uma lista de tarefas. (Usuário implementa a interface UserDetails do SpringSecurity onde sobrescreve os métodos para fornecer detalhes de autenticação e autorização).

Tarefas: id, título, descrição, concluída e seu respectivo usuário.

E temos o relacionamento **muitos pra um** das tarefas para o usuário, ou seja, uma lista de tarefas para cada usuário.

Ainda na camada de modelo temos o repositório de tarefas e usuários que estende o JPA (Java persistence API) ORM (Object-relational-mapping) em Java. A JPA especifica uma interface que os frameworks ORM como o hibernate, podem implementar para mapear objetos Java para tabelas em bancos de dados relacionais.

E Logo após ainda demos continuidade no service, onde foi aplicada toda a lógica de negócio da aplicação, para usuários(salvar e criptografar a senha antes de ser salva no banco) e para tarefas(salvar tarefa, marcar como concluída, listar por usuários, e excluir).

Na camada de controle, definimos o AuthController e o TarefasController, onde estas duas classes controladoras estão cuidando apenas da requisição enquanto o service cuida da lógica. No AuthController estão os endpoints de cadastro de usuário e login. E no TarefasController está os endpoints que cuidam das tarefas, como salvar, excluir, marcar como concluído e listar as tarefas por usuário.

E para tornar a aplicação mais robusta decidimos usar dtos (Data transfer object) e criamos dois Records (tipo de classe simplificado, projetado para representar dados imutáveis, ideais para dtos), UsuarioDto e TarefasDto, e utilizamos dtos para transferir apenas os dados necessários, deixando os dados sensíveis e campos de auditoria de fora.



Juntamente ao dto, também optamos por utilizar uma biblioteca de mapeamento, para facilitar o mapeamento do dto para a entidade. A biblioteca é o MapStruct, “é um gerador de código que simplifica a implementação de mapeamentos entre tipos de Java bean com base em uma abordagem de convenção sobre configuração”.

Fonte: <https://mapstruct.org/>

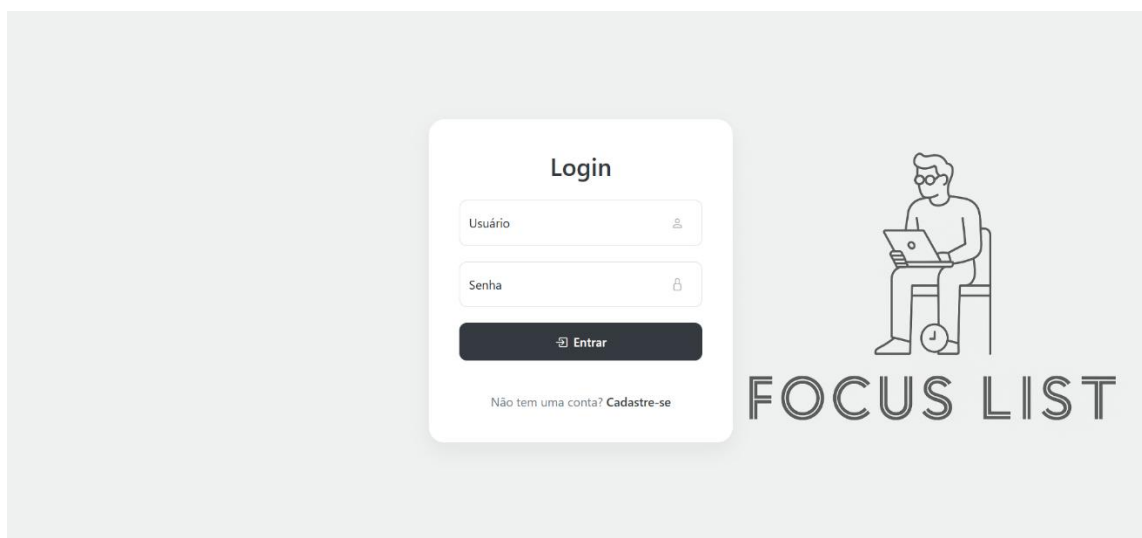
E agora sobre a segurança, utilizamos o SpringSecurity. Criamos uma configuration onde criamos dois Beans, o SecurityFilterChain que serve como um filtro onde tem todas as configurações necessárias para a segurança do sistema e o outro bean PasswordEncoder, que serve para criptografar a senha dentro do banco de dados.

E para a autenticação de usuário criamos a classe CustomAuthenticationProvider, onde implementamos a interface AuthenticationProvider e sobrescrevemos os dois métodos desta interface (Authenticate e supports) e criamos nossa lógica de autenticação.

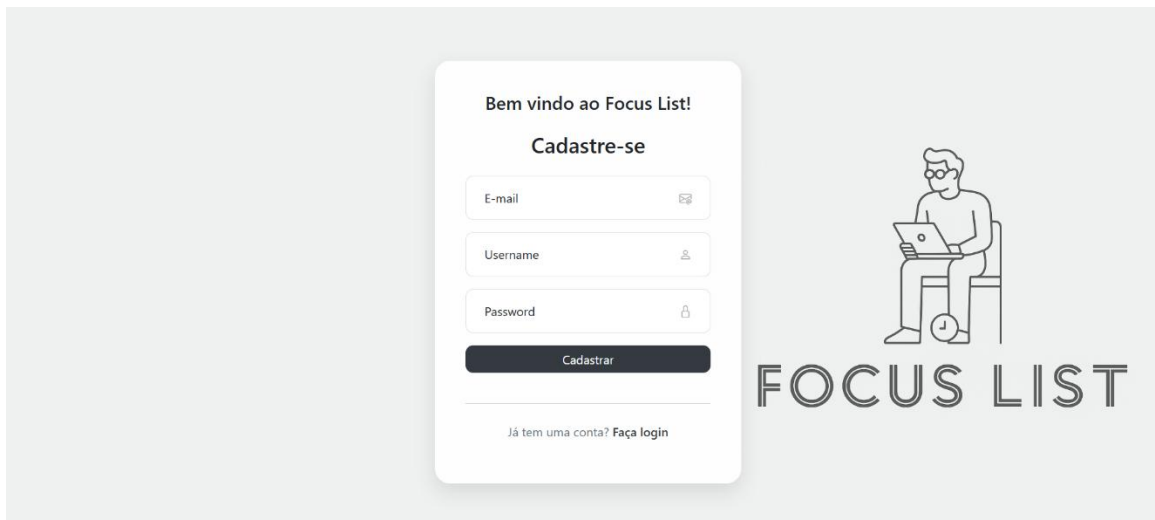
Evidências visuais da aplicação

Neste tópico será introduzido as evidências visuais da aplicação em funcionamento.

1. Tela de Login





2. Tela de cadastro




Bem vindo ao Focus List!

Cadastre-se


E-mail 

Username 

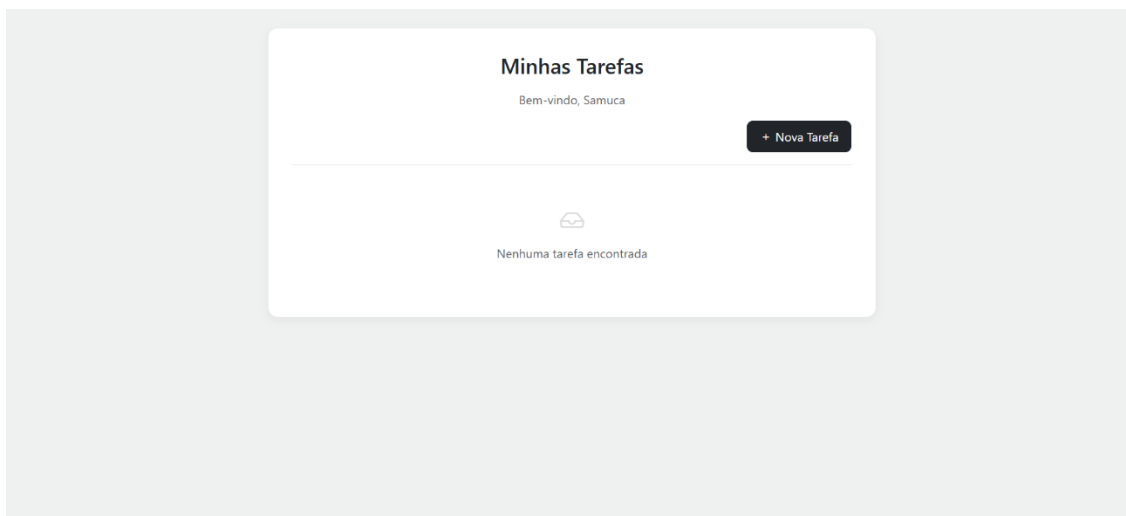
Password 

Cadastrar

Já tem uma conta? [Faça login](#)


FOCUS LIST


3. Tela principal onde estão localizadas as tarefas



Minhas Tarefas

Bem-vindo, Samuca

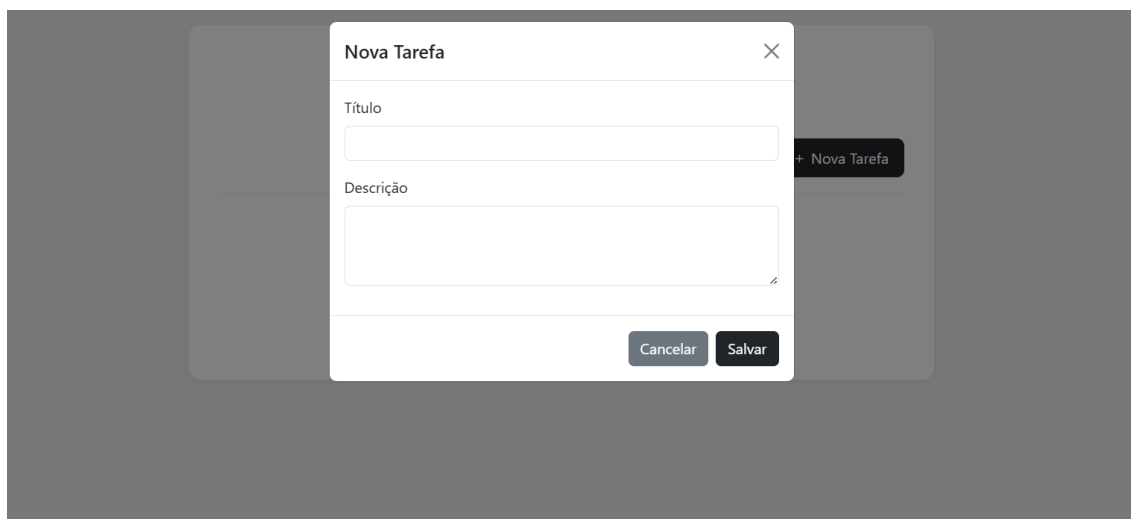
+ Nova Tarefa



Nenhuma tarefa encontrada

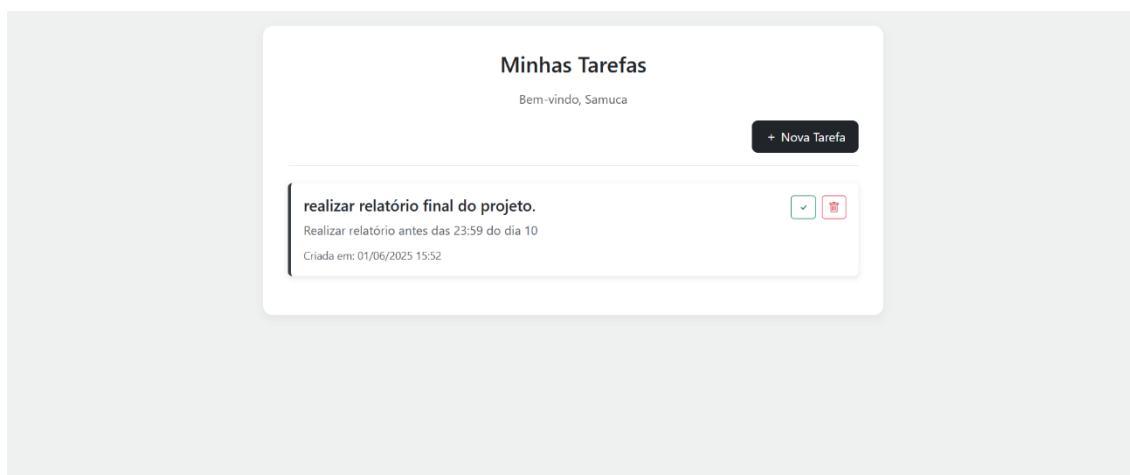


4. Adicionar tarefa



The screenshot shows a modal window titled "Nova Tarefa" with a close button (X) in the top right corner. Inside the modal, there are two input fields: "Título" (Title) and "Descrição" (Description). Below the "Descrição" field, there are two buttons: "Cancelar" (Cancel) and "Salvar" (Save). In the background, a dark button labeled "+ Nova Tarefa" is visible.

5. Tarefa adicionada



The screenshot shows a card titled "Minhas Tarefas" with a greeting "Bem-vindo, Samuca". A button "+ Nova Tarefa" is in the top right. The task list contains one item: "realizar relatório final do projeto." with a green checkmark icon and a trash icon. Below the task name, it says "Realizar relatório antes das 23:59 do dia 10" and "Criada em: 01/06/2025 15:52".

6. Marcada como concluída



The screenshot shows the "Minhas Tarefas" card with the same greeting and "+ Nova Tarefa" button. The task "realizar relatório final do projeto." is now marked as completed, indicated by a green background and a green checkmark icon. The trash icon is still present. The due time and creation date remain the same: "Realizar relatório antes das 23:59 do dia 10" and "Criada em: 01/06/2025 15:52".



Anexo do repositório do projeto

Conforme solicitado o anexo do código fonte, ele foi disponibilizado no github do criador e o arquivo.zip será anexado dentro da plataforma.

GITHUB: <https://github.com/smfmo/Focus-List>

