

Check Point 1/2º semestre – 2024

Profª. Aparecida F. Castello Rosa

**Cenário**

A ACR Financeira é uma empresa que atua no ramo financeiro concedendo empréstimos para pessoas físicas.

Para uma pessoa solicitar um empréstimo é necessário realizar o cadastro para a análise da proposta de crédito solicitada.

Nesta fase de desenvolvimento do microsserviço está sendo realizado somente o cadastro do cliente e da solicitação do empréstimo. A análise da proposta será desenvolvida posteriormente.

Considere a tela de cadastro para solicitação de análise de crédito conforme apresentado na Figura 1.

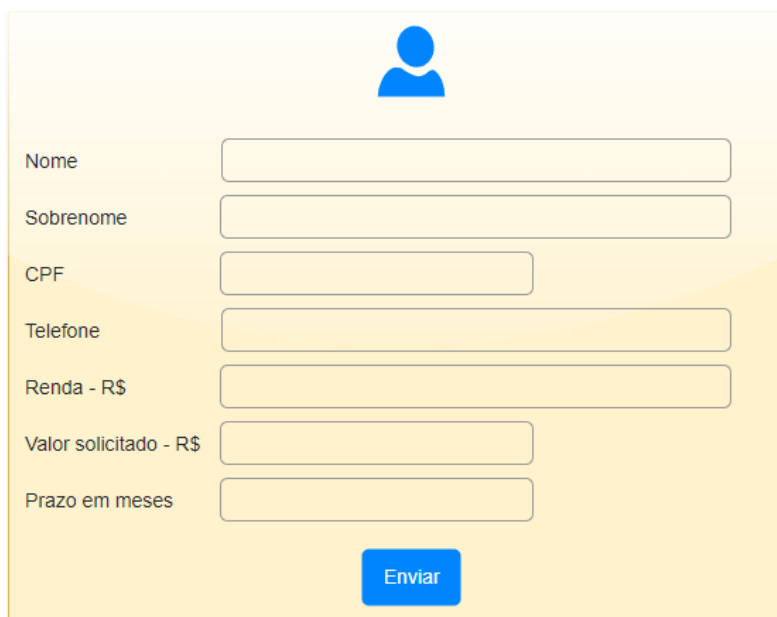
A imagem mostra uma interface de usuário para o cadastro de um cliente e a solicitação de um empréstimo. No topo, há um ícone de perfil de usuário em azul. Abaixo, há sete campos de entrada de texto, cada um com uma label à esquerda: 'Nome', 'Sobrenome', 'CPF', 'Telefone', 'Renda - R\$', 'Valor solicitado - R\$' e 'Prazo em meses'. Os campos para 'CPF' e 'Prazo em meses' são mais curtos que os outros. No canto inferior direito, há um botão azul com o texto 'Enviar' em branco.

Figura 1 – Tela de cadastro para solicitação de análise de crédito

O campo prazo para pagamento refere-se ao número de meses, como, por exemplo, 24 meses para pagamento do empréstimo.

Inicialmente, quando o cadastro é criado, o campo aprovado deve ser falso. Somente depois de passar pela análise de crédito é que esse campo poderá ser alterado para verdadeiro caso o crédito seja aprovado.

O sistema deverá persistir os dados no banco de dados e realizar as operações CRUD conforme especificações a seguir.

De acordo com o contexto, desenvolver um microserviço para gerenciamento de Proposta de Análise de Crédito. O diagrama UML é apresentado na Figura 2.

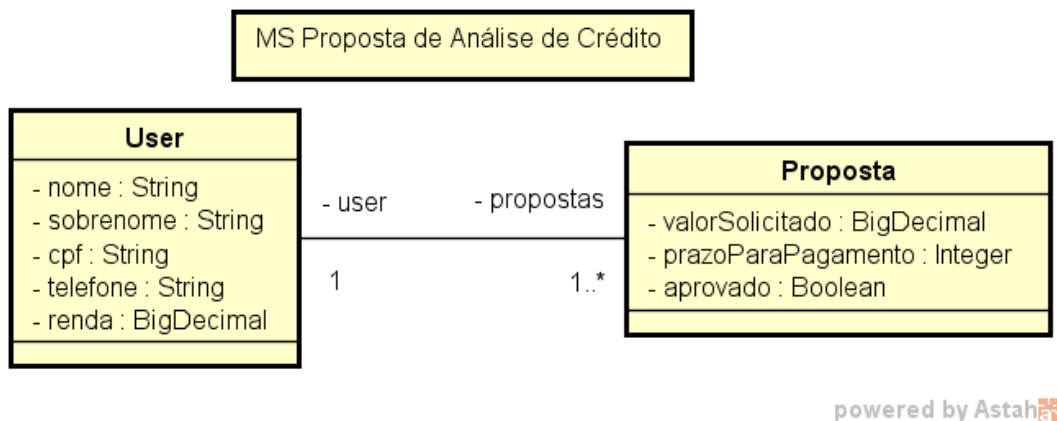


Figura 2 – Diagrama UML

## 1. Entregável Check point 1

### 1.1. Configurações do projeto

- Projeto: Maven
- Language: Java
- Spring Boot 3.2.X ou superior; ou Spring Boot 3.3.X ou superior
- Project Metadata
  - Group: seu domínio do GitHub. Caso seja feito em dupla, escolher o domínio de um dos integrantes.
  - Artifact: ms-proposta
  - Packaging: Jar
  - Java: versão 17
- Dependências:
  - Adicionar as dependências necessárias para criação do microserviço com Banco de Dados H2. As dependências Spring Boot Dev Tools, Lombok e ModelMapper são opcionais e poderão ser utilizadas.
- **Propriedades da Aplicação**
  - A aplicação deverá ter os seguintes arquivos de configurações:
    - application.properties: definições.
    - application-test.properties: perfil de teste com o banco de dados H2.

### 1.2. API Back-end

- Entidade **USER**: implementar **CRUD (findById, findAll, insert, update, delete)**.

- Entidade **PROPOSTA**: implementar **INSERT** de acordo com a tela da aplicação apresentada na Figura 1. Para **findAll** e **findById** deverá apresentar os dados de Proposta conforme exemplo do JSON apresentado na Figura 3.

```
{
  "id": 2,
  "valorSolicitado": 8250.00,
  "prazoParaPagamento": 36,
  "aprovado": false,
  "userId": 2
},
```

Figura 3 – Exemplo do JSON de Proposta para *findAll* e *findById*

- Controllers de USER e PROPOSTA com a utilização correta dos métodos HTTP.  
Estruturar o projeto com as camadas: Model, DTO, Repository, Service e Controller.
  - a) Validação (0,5 ponto)**  
**Entidade User:** todos os campos são requeridos.  
**Entidade Proposta:** todos os campos são requeridos exceto o campo aprovado que deverá ser falso quando a proposta é criada.
  - b) Arquivo README (0,5 ponto)**  
Deverá ser elaborado um arquivo README.md com nome(s) do(s) integrante(s), especificações do projeto, como, por exemplo, as *stacks* utilizadas, qual a finalidade da aplicação etc. Esse arquivo deverá ficar na raiz do projeto.
  - c) Arquivo JSON (0,5 ponto)**  
Deverá ser elaborado um arquivo JSON para a realização do teste para inserir um registro de proposta conforme apresentado na Figura 1. Esse arquivo deve ficar na raiz do projeto.
  - d) Camada MODEL (1,5 ponto)**  
Classes do domínio da aplicação.
  - e) Camada DTO (1,5 ponto)**  
Desenvolver os DTO's para transferência de dados.
  - f) Camada REPOSITORY (0,5 ponto)**  
Criar a interface *repository* para as Entidades.
  - g) Carga do banco de dados (1,0 ponto)**  
Criar o seed do DB com registros distintos. Nesse arquivo deverá ter o registro de 3 usuários distintos e para cada um deles duas propostas distintas.
  - h) Camada SERVICE (2,0 pontos)**  
Criar as classes de serviços para User e Proposta.
  - i) Camada CONTROLLER (2,0 pontos)**  
Criar as classes Controladoras para User e Proposta.

## 2. Instruções Gerais

### 2.1. Entrega em equipe

O Check point 1 poderá ser realizada de forma **individual ou em dupla**. O aluno é responsável por montar a sua dupla, caso não deseje fazer de forma individual.

### 2.2. Data da entrega

**Até dia 02/09/2024 até às 23:55h.**

**Não deixe para a última hora. Imprevistos acontecem e a data não será prorrogada.**

**Faça o upload até o prazo máximo que está na área de entregas, ou seja, o prazo acima.**

### 2.3. ENTREGA:

**Obs:** É importante que seu programa rode, caso tenha algum erro, inutilize a linha com o erro utilizando comentário "//", dessa forma consigo avaliar os outros itens que estão funcionando.

Criar uma pasta com nome do responsável **Nome+RM**. Salve o projeto descompactado nessa pasta e abra o projeto que está nessa pasta.

### 2.4. Local de Entrega

**Upload:** Ao finalizar o trabalho compacte todos os arquivos e faça upload na área de entregas de trabalhos. É necessário que apenas um aluno faça o upload pela equipe ou caso escolha fazer individualmente, no entanto, deve ser informado o(s) nome(s) de cada integrante, bem como o RM(s) **no arquivo JSON e na classe principal de cada entidade.**

**Faça o upload até o prazo máximo que está na área de entregas.**

**Importante:** Fork de código, uso de projeto pronto, empréstimo ou cópia de código de outro aluno o CP será zerado para ambos.

**ATENÇÃO: Verifique se o arquivo que será enviado é o correto.**

**OBS: Compactar os arquivos para fazer o upload. Não deixe a entrega para o último minuto.**

Bom checkpoint 😊