Documentação de Projeto

para o sistema

AlugaCar

Versão 1.0

Projeto de sistema elaborado pelo(s) aluno(s) Luis Henrique Fantini, Arthur Candian, Rafael Lopes
como parte da disciplina **Projeto de Software**.

16/11/2025

Tabela de Conteúdo

1. Introdução

Este documento agrega: 1) a elaboração e revisão de modelos de domínio e 2) modelos de projeto para o sistema AlugaCar. A referência principal para a descrição geral do problema, domínio e requisitos do sistema é o documento de especificação que descreve a visão de domínio do sistema.

2. Modelos de Usuário e Requisitos

2.1 Descrição de Atores

Nesta subseção é apresentado descrição de cada um dos atores que interagem com o sistema.

Cliente

- Pessoa que deseja alugar um automóvel.
- Pode realizar cadastro, login, criar pedidos, consultar status, modificar ou cancelar pedidos.

Usuário (genérico)

- Representa qualquer indivíduo que acessa o sistema e pode realizar cadastro e login.
- Engloba tanto clientes quanto agentes.

Agente

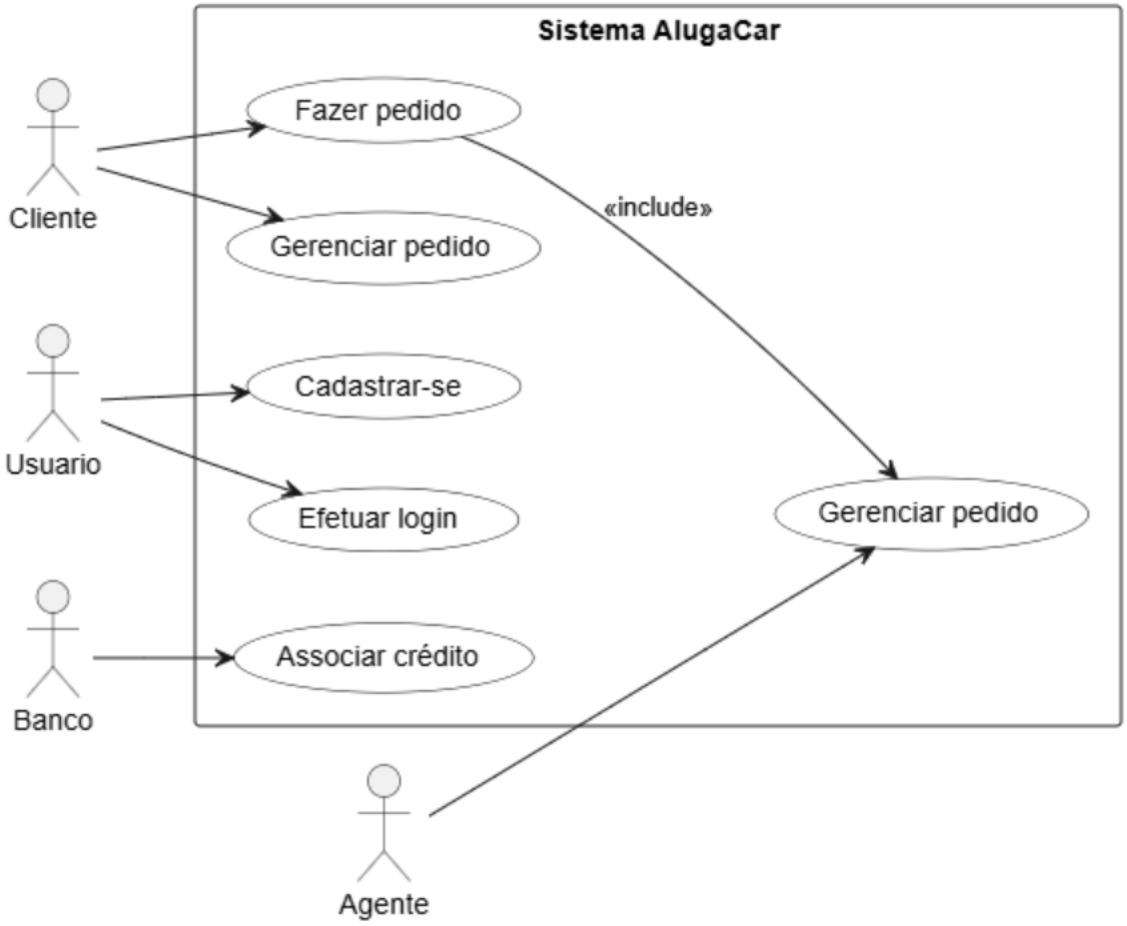
- Funcionário responsável por analisar pedidos realizados pelos clientes.
- Avalia capacidade financeira, aprova ou reprova pedidos e formaliza o processo.

Banco

- Entidade financeira integrada ao sistema.
- Associa crédito ao pedido, permitindo contratos de financiamento para o aluguel.

2.2 Modelo de Casos de Uso

Nesta subseção é apresentado o diagrama de casos de uso do sistema. UC-01 para o Caso de Uso Sistema AlugaCar.



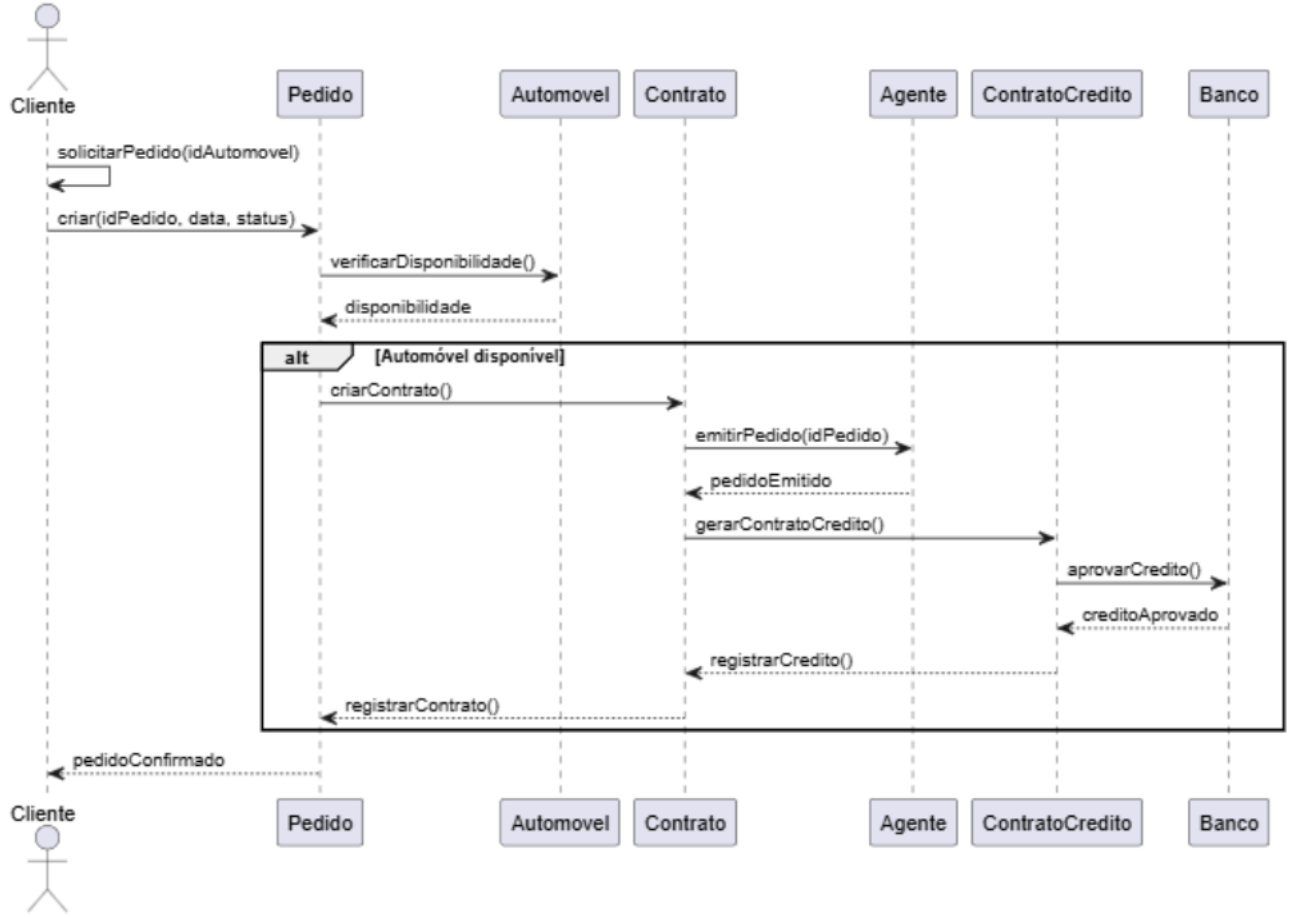
Historias de Usuário:

- **Como cliente**, quero me cadastrar no sistema para poder realizar pedidos de aluguel.
- **Como cliente**, quero consultar meus pedidos para acompanhar o status de cada um.
- **Como cliente**, quero modificar ou cancelar um pedido para ajustar minhas necessidades.
- **Como agente**, quero avaliar os pedidos financeiramente para verificar se o cliente tem condições de pagamento.
- **Como agente**, quero aprovar ou reprovar pedidos para garantir a viabilidade do contrato.
- **Como sistema**, quero gerar automaticamente contratos de aluguel para formalizar o processo.
- **Como sistema**, quero registrar dados dos automóveis para vinculá-los aos contratos.
- **Como sistema**, quero associar contratos de crédito fornecidos pelos bancos para viabilizar o financiamento.

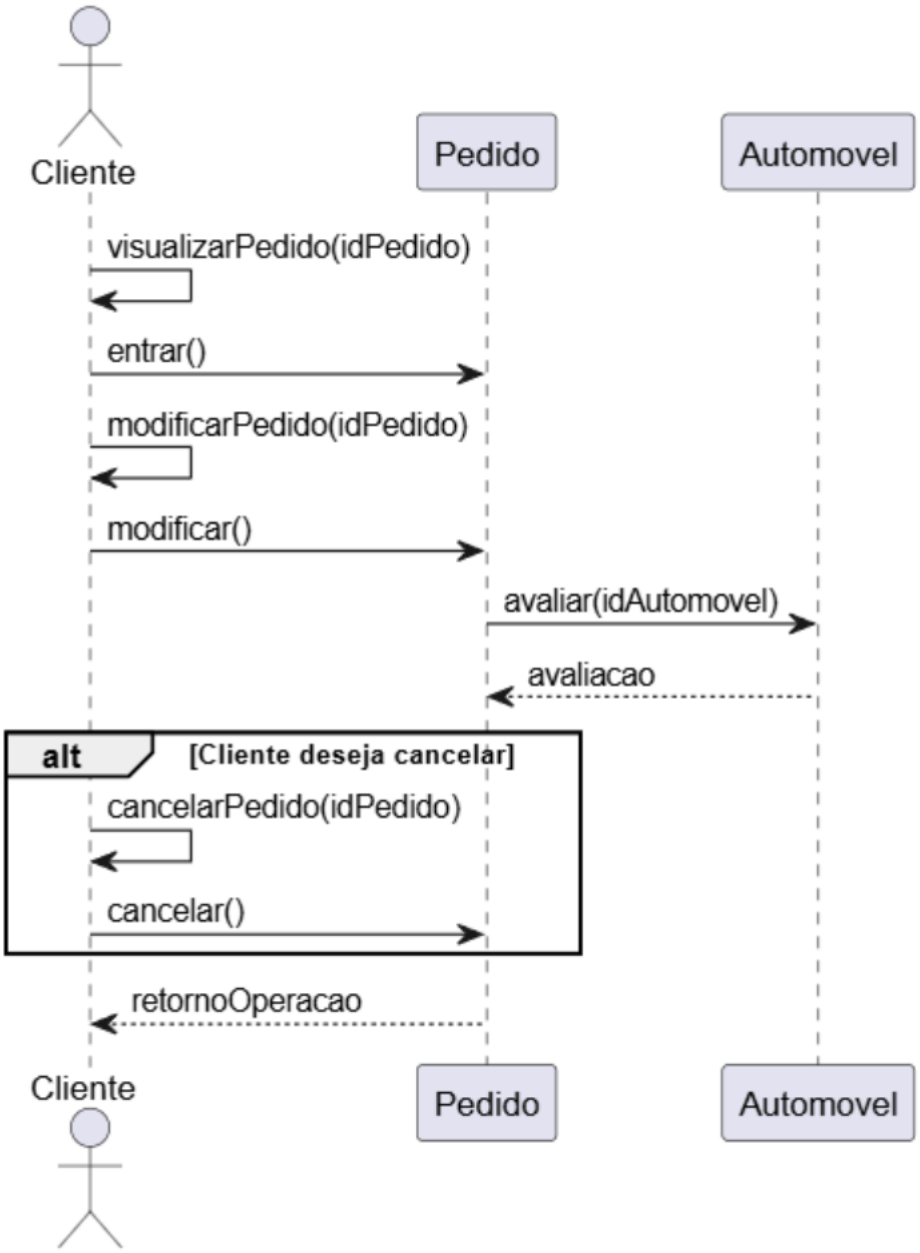
2.3 Diagrama de Sequência do Sistema

Nesta subseção é apresentado o diagrama de sequência do sistema.

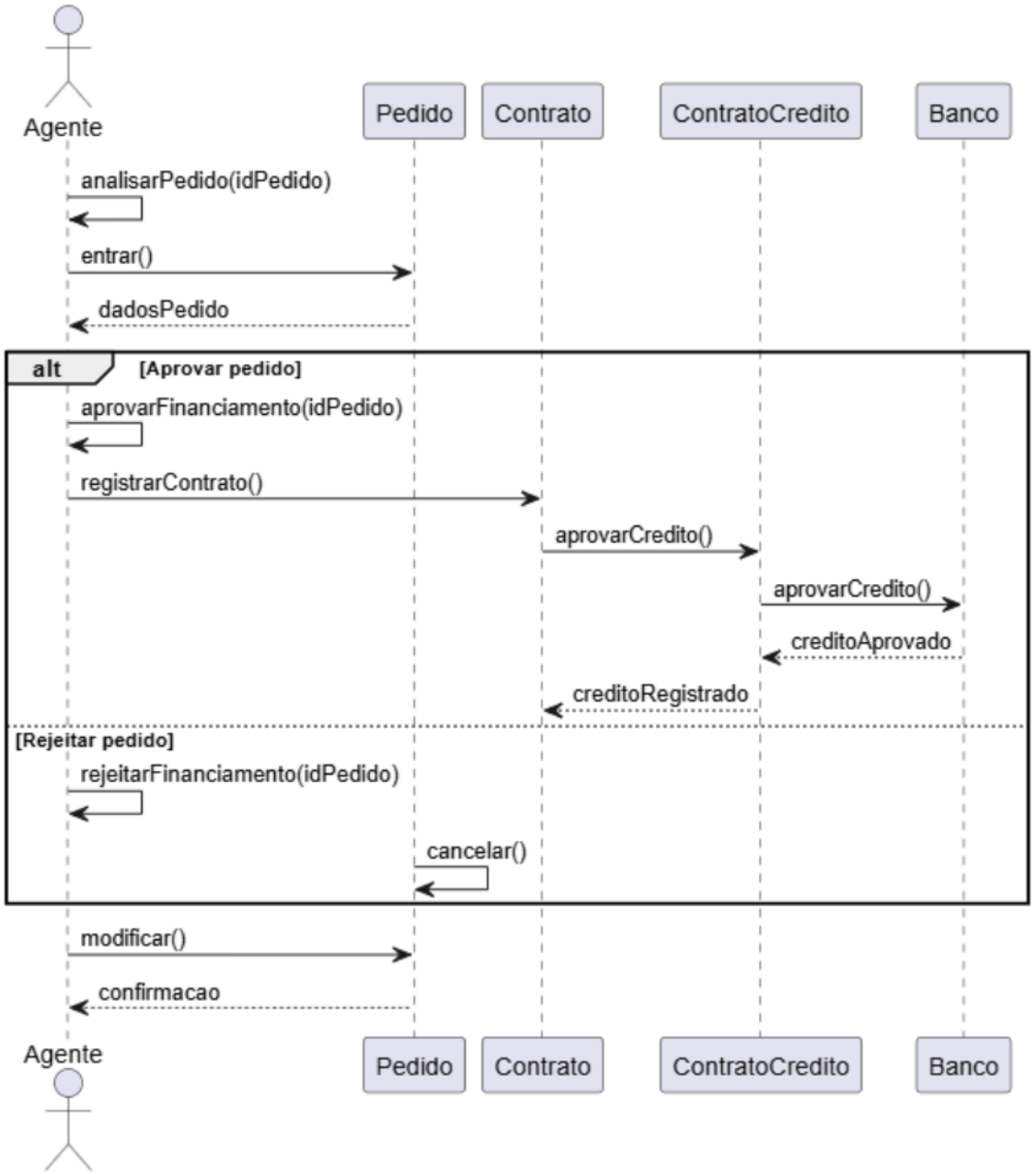
Fazer Pedido:



Gerenciar Pedido:


Cliente
Pedido
Automovel
visualizarPedido(idPedido)
entrar()
modificarPedido(idPedido)
modificar()
avaliar(idAutomovel)
avaliacao
alt
[Cliente deseja cancelar]
cancelarPedido(idPedido)
cancelar()
retornoOperacao
Cliente
Pedido
Automovel

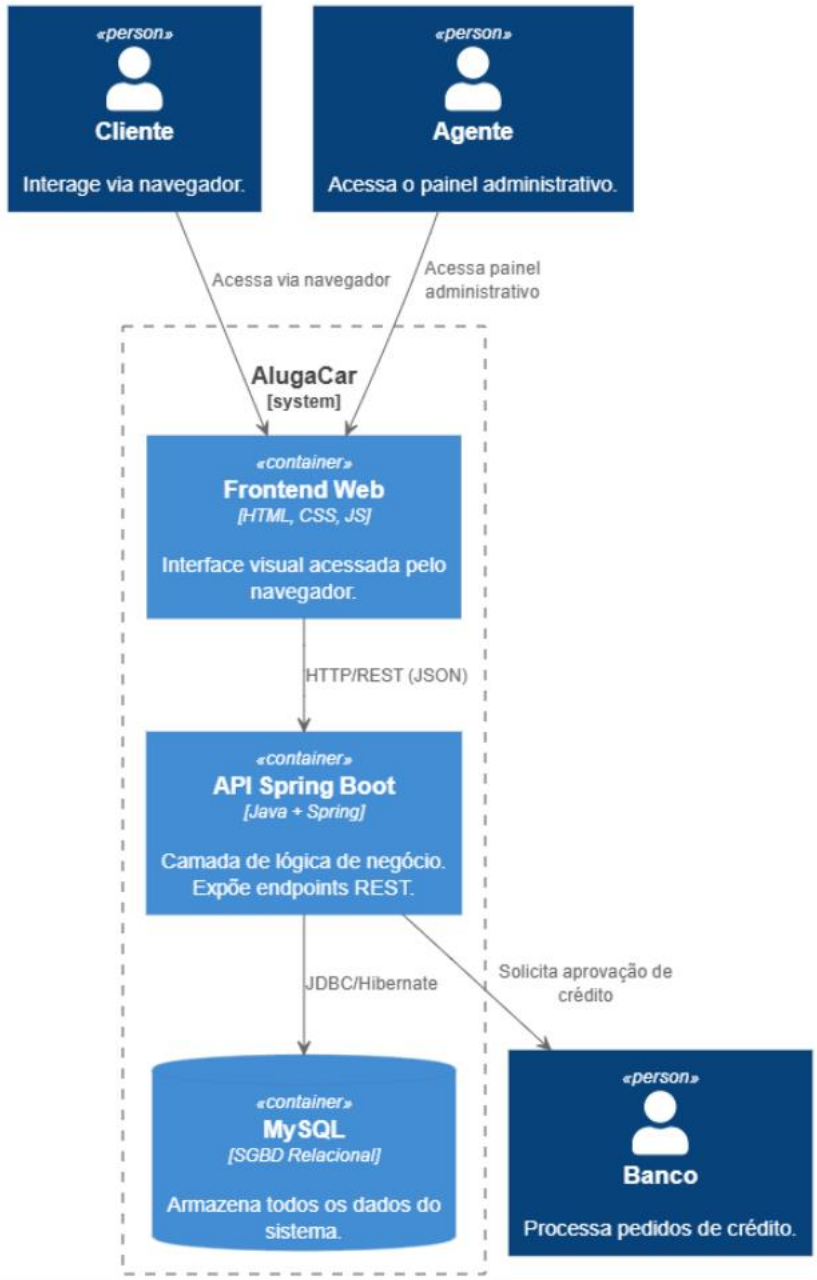
Analisar Pedido:



3. Modelos de Projeto

3.1 Arquitetura

Arquiterura UML:



A arquitetura do sistema **AlugaCar** segue um modelo estruturado em três camadas principais, representadas em containers conforme o **C4 Model – Nível 2**, garantindo organização, escalabilidade e separação clara de responsabilidades.

1. Frontend (Aplicação Web – Interface do Usuário)

- **Local:** pasta `frontend/`
- **Tecnologias:** HTML, CSS, JavaScript (ou framework utilizado), Fetch API
- **Funções principais:**

- Exibir a interface visual acessada pelos clientes e agentes.
- Coletar ações do usuário (login, criação de pedidos, consultas, avaliações etc.).
- Consumir a API REST do backend via requisições **HTTP/HTTPS** no formato **JSON**.
- Renderizar páginas como:
  - Formulário de cadastro e login
  - Lista de automóveis disponíveis
  - Histórico de pedidos do cliente
  - Painel do agente com pedidos pendentes
- Enviar e receber dados do backend, suportando operações de consulta, criação, modificação e cancelamento de pedidos.

O frontend funciona como a **camada de apresentação**, responsável por toda a interação visual com o usuário.

2. Backend (API REST – Spring Boot)

• **Local:** pasta `backend/`
• **Tecnologias:**

- **Java + Spring Boot**
- **Spring Web (@RestController)**
- **Spring Data JPA (@Repository)**
- **Hibernate (ORM)**
- **MySQL Connector/J**
- **DTOs, Services, Exception Handler**

O backend implementa toda a lógica central do sistema e é dividido em três camadas internas (MVC):

Componentes principais:

• *Controllers (`controller/`)*

- Exposição dos endpoints REST.
- Entrada de requisições HTTP do frontend.
- Conversão entre DTOs e entidades do domínio.
- Controle do fluxo básico da operação.
  Exemplos de endpoints:
  `POST /login`
  `POST /pedidos`
  `PUT /pedidos/{id}`
  `GET /pedidos/cliente/{idCliente}`
  `GET /pedidos/pendentes`
  `PUT /pedidos/{id}/aprovar`

• *Services (`service/`)*

- Núcleo da **lógica de negócio**.
- Regras como:
  - validação de dados
  - análise de crédito
  - verificação de disponibilidade do automóvel
  - emissão, alteração ou cancelamento de pedidos
- Orquestra operações entre controller, repository e integrações externas.

• *Repositories (`repository/`)*

- Componentes responsáveis por acessar o banco MySQL.
- Implementam operações CRUD via **Spring Data JPA**.
- Utilizam o **Hibernate** como provedor ORM.

• *Models (`model/`)*

- Representam entidades do domínio:
  - Cliente
  - Agente
  - Automóvel
  - Pedido
  - Contrato
  - Banco
  - ContratoCredito

• *DTOs (`dto/`)*

- Objetos de transferência de dados entre frontend ↔ backend.
- Evitam exposição direta das entidades internas.

• *Exception Handler (`exception/`)*

- Lida com erros e validações.
- Retorna respostas padronizadas para o cliente.

3. Banco de Dados (MySQL)

• **Tecnologia:** MySQL Server
• **Acesso via:** JPA/Hibernate + MySQL Connector
• **Local:** servidor de banco de dados separado (container físico próprio)

Função:

- Persistir todas as entidades do sistema:
  - Usuários
  - Automóveis
  - Pedidos
  - Contratos
  - Registros de crédito
  - Agentes e Clientes
- Garantir integridade referencial entre Pedido → Cliente → Automóvel → Contrato → ContratoCredito.

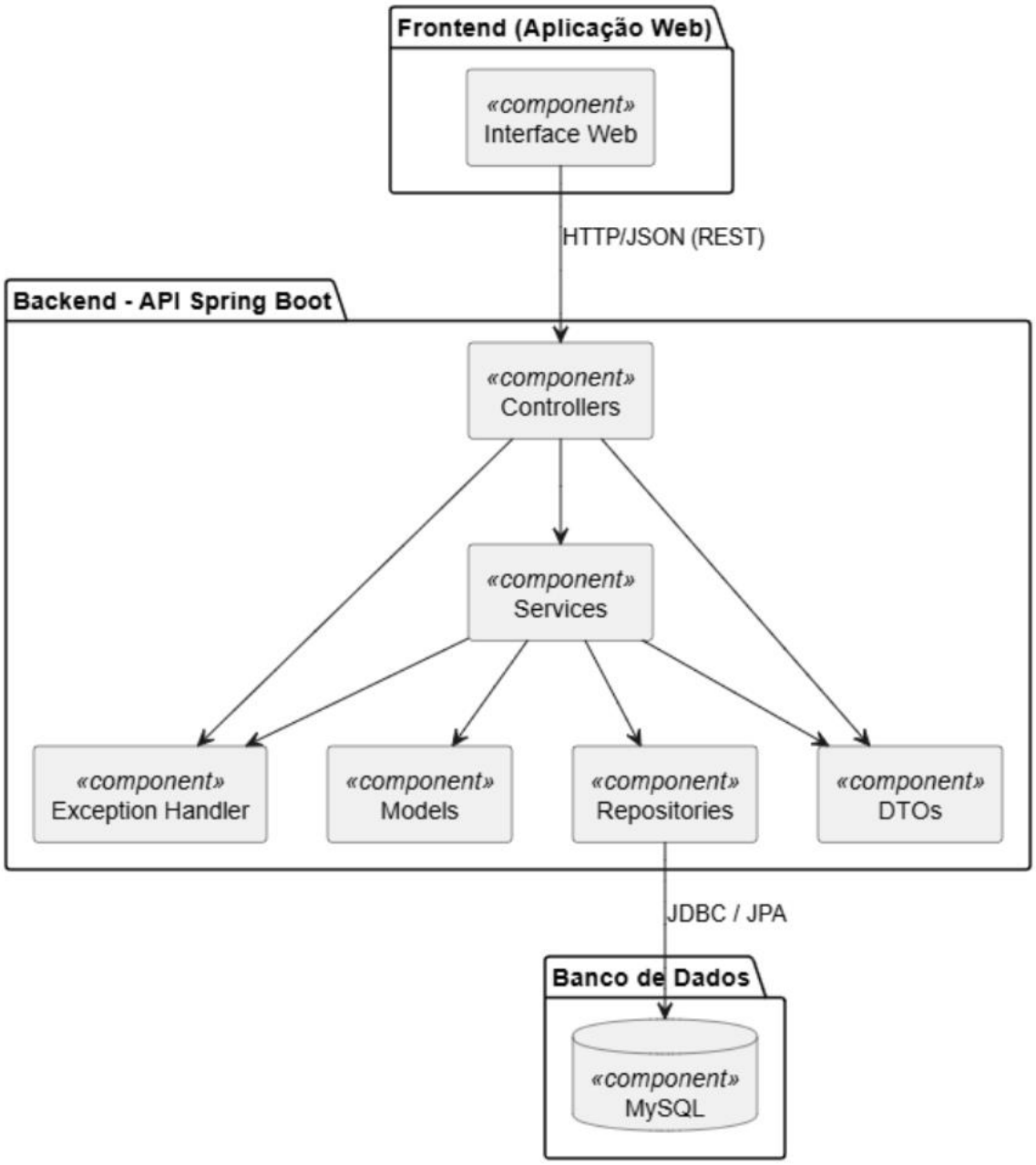
Mecanismos adicionais:

- Migrações controladas pelo Spring (schema auto-update).
- Índices e chaves estrangeiras mantendo coerência dos dados.

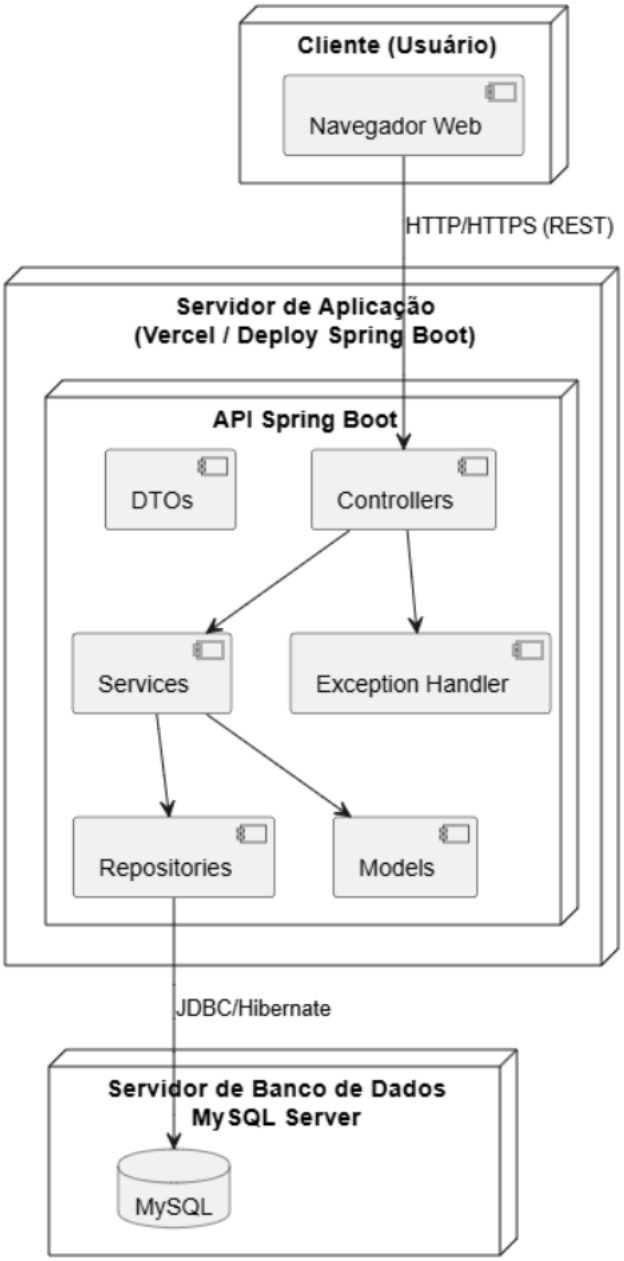
3.2 Diagrama de Componentes e Implantação.

Diagramas de componentes e implantação do sistema.

Componentes:

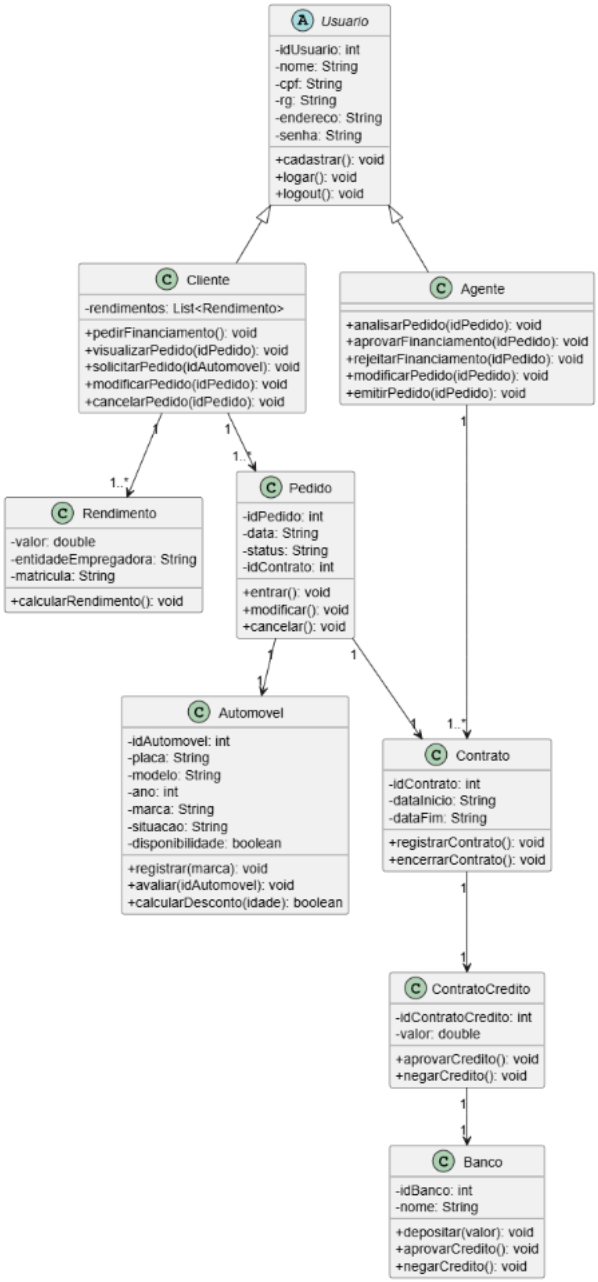


Implantação:



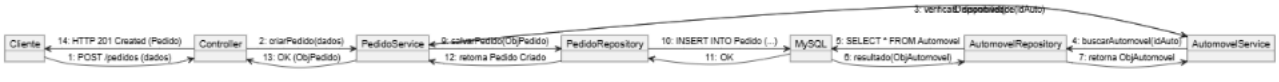
3.3 Diagrama de Classes

Diagrama de classes do sistema.



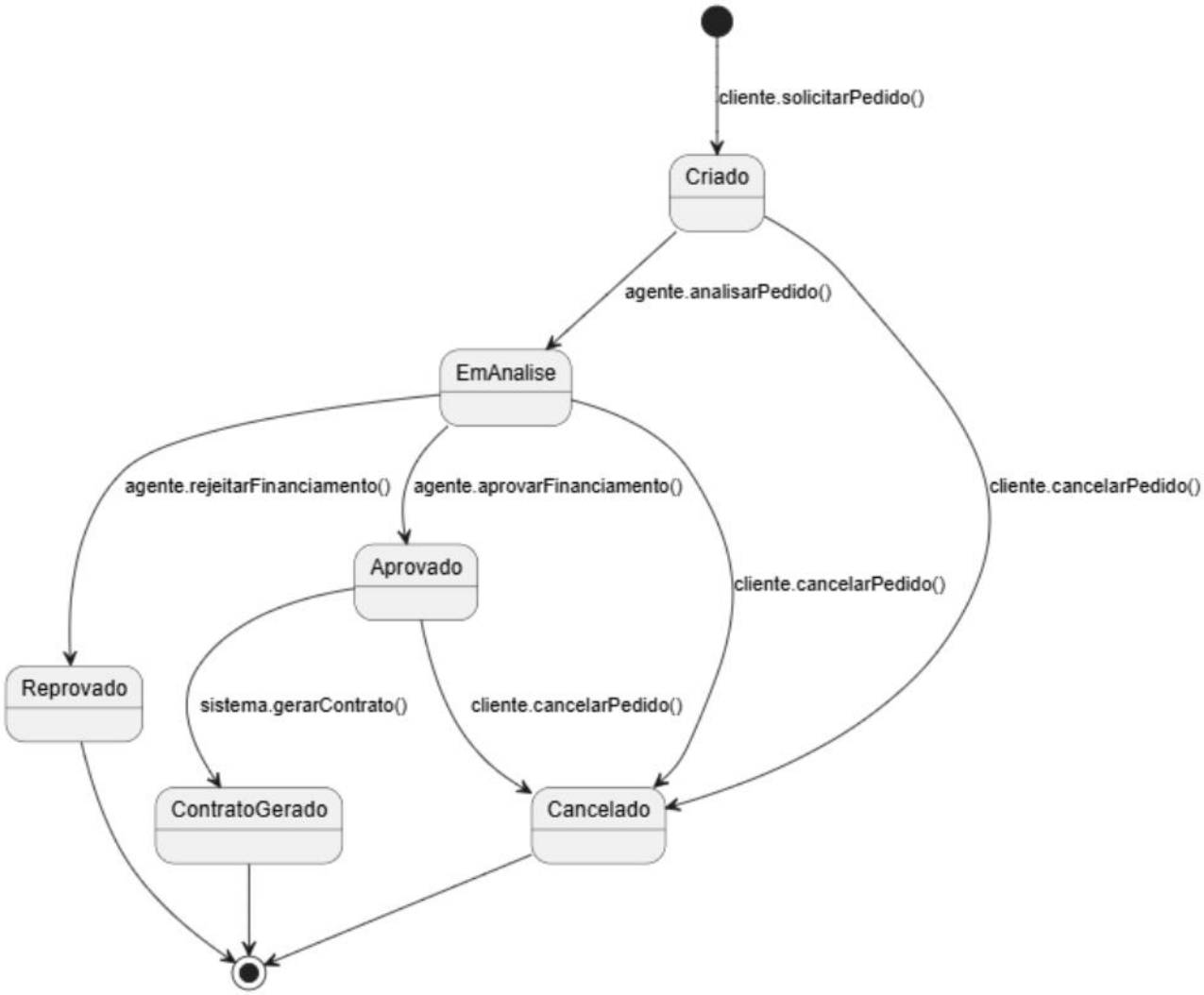
3.4 Diagramas de Comunicação

Diagramas de comunicação para realização de casos de uso.



3.5 Diagramas de Estados

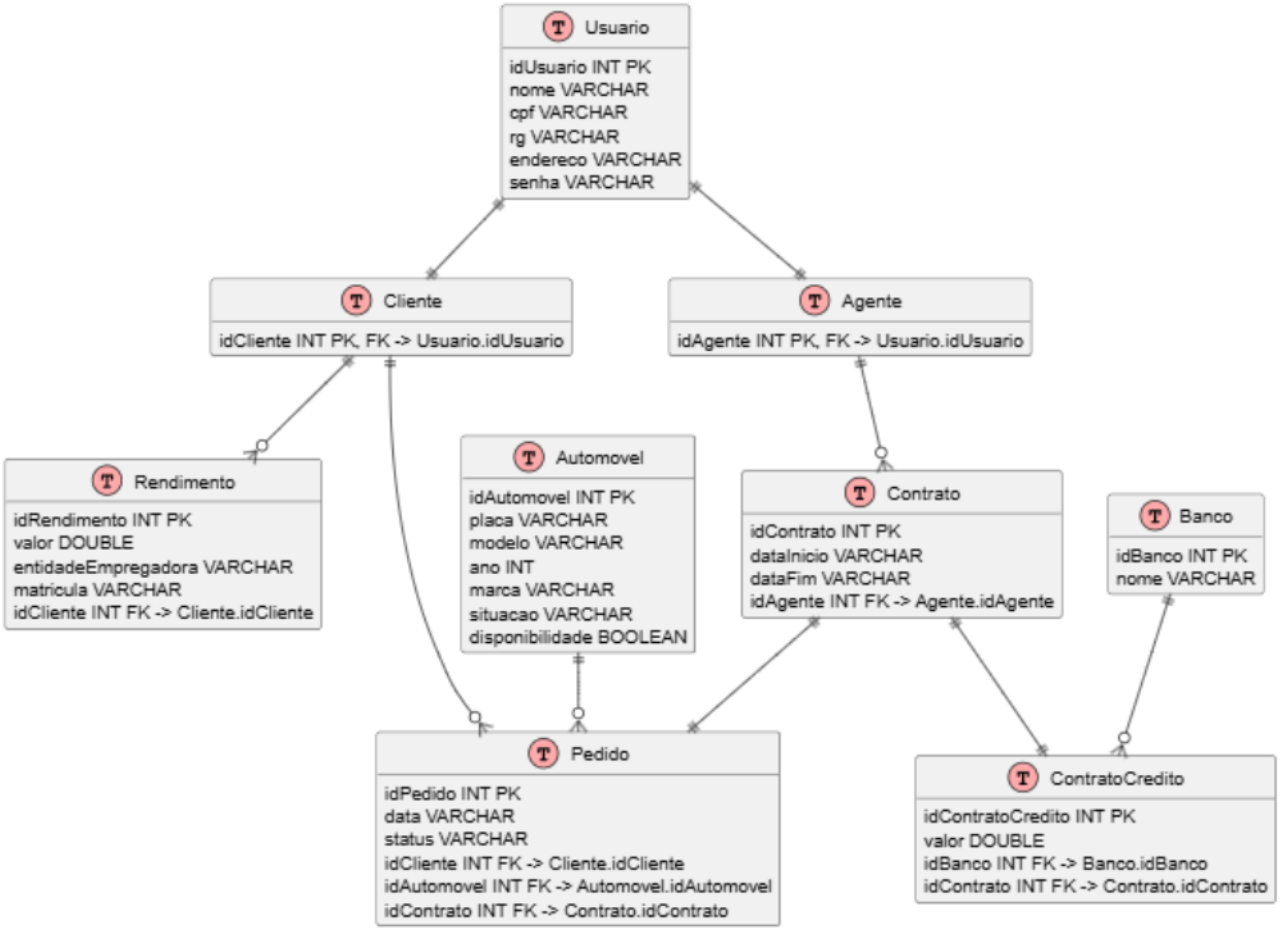
Diagramas de estados do sistema.



4. Modelos de Dados

Esta seção detalha os modelos de dados desenvolvidos, composto pelo Diagrama Relacional.

Diagrama Relacional:


Usuario
idUsuario INT PK
nome VARCHAR
cpf VARCHAR
rg VARCHAR
endereco VARCHAR
senha VARCHAR
Cliente
idCliente INT PK, FK -> Usuario.idUsuario
Agente
idAgente INT PK, FK -> Usuario.idUsuario
Rendimento
idRendimento INT PK
valor DOUBLE
entidadeEmpregadora VARCHAR
matricula VARCHAR
idCliente INT FK -> Cliente.idCliente
Automovel
idAutomovel INT PK
placa VARCHAR
modelo VARCHAR
ano INT
marca VARCHAR
situacao VARCHAR
disponibilidade BOOLEAN
Contrato
idContrato INT PK
dataInicio VARCHAR
dataFim VARCHAR
idAgente INT FK -> Agente.idAgente
Banco
idBanco INT PK
nome VARCHAR
Pedido
idPedido INT PK
data VARCHAR
status VARCHAR
idCliente INT FK -> Cliente.idCliente
idAutomovel INT FK -> Automovel.idAutomovel
idContrato INT FK -> Contrato.idContrato
ContratoCredito
idContratoCredito INT PK
valor DOUBLE
idBanco INT FK -> Banco.idBanco
idContrato INT FK -> Contrato.idContrato