**FIAP  
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**GUSTAVO CARVALHO – 552466**

**RONALD DE OLIVEIRA – 552364**

**NICOLY OLIVEIRA – 552410**

**NICOLLY DE ALMEIDA - 552579**

**VITOR TEIXEIRA – 552228**

**Nome do grupo: UniDevs**

**Nome da solução: GuinFlex**

**SPRINT 4  
DOMAIN DRIVEN DESIGN**

**SÃO PAULO**

**2023**

**SUMÁRIO**

1. **JUSTIFICATIVA.........................................................................**
2. **OBJETIVO.................................................................................**
3. **DESCRIÇÃO DAS FUNCIONALIDADES..................................**
4. **ENDPOINTS...............................................................................**
5. **DIAGRAMA DE CLASSES........................................................**
6. **MODELO DO BANCO DE DADOS...........................................**
7. **PROTÓTIPO..............................................................................**

**JUSTIFICATIVA**

Após a análise, pesquisa e investigação do desafio proposto pela empresa Porto, compreendemos que o principal problema não é a aplicação de uma plataforma que conecte o cliente a porto, e sim, sobre a eficiência em acertar a escolha do melhor modal para resgatar os veículos dos clientes, sem necessitar totalmente da intervenção humana e retrabalhos. O foco para a solução está em como podemos melhorar este processo e torná-lo mais assertivo do ponto de vista da Porto e favorável ao cliente.

Em primeira instancia, temos como foco a criação de uma aplicação web, focada primeiramente no desempenho durante o uso de navegadores mobile. Isso se deve ao fato, de que em sua grande maioria, os clientes abririam o sinistro via dispositivos moveis, pois os mesmos, facilitam a coleta de informações sobre ocorrido por possuírem câmeras de fácil acesso.

Na aplicação, queremos que o usuário consiga realizar sua jornada de abertura do chamado, login, sinistro, acompanhamento, verificação, informações cadastradas, cancelamento e afins totalmente via website sem que se torne algo incompreensível e ineficaz para experiência.

**OBJETIVOS**

O ponto do projeto que demanda atenção redobrada é que o problema ocorre durante o processo de abertura de sinistro, demandando em torno de 60% dos casos intervenção humana e mais 10% a 15% de retrabalho do prestador de serviço. Chegando a ter casos de encaminhar o guincho incorreto para o local e precisar solicitar outro. Para evitar essas situações temos como objetivos;

* Nossa aplicação deve ser leve, intuitiva e agradável de se utilizar.   
  É preciso que o cliente se sinta confortável e seguro para não gerar a necessidade de contato por outros canais de atendimento, como e-mails, telefonemas...
* A coleta de informações e sua qualidade tem extrema importância, pois a falta delas ou erros podem ocasionar a necessidade de intervenção humana e retrabalhos dos funcionários.

**Descrição das funcionalidades**

A solução permite com que o cliente entre com as informações referente ao ocorrido durante a abertura do chamado, com base nessas informações é decidido o guincho indicado para a situação e então feita a abertura do chamado. Para a realização desse processo o cliente faz o login na aplicação, uma vez logado ele pode navegar na aplicação, verificando assim, suas informações de cadastro e tendo a opção de abrir o chamado, verificar o histórico dele e acompanhar o chamado. No que tange a parte de autenticar o cliente na hora do login e recuperar dados como endereços, veículos, dados pessoas, plano do seguro e afins, a API em Java faz a recuperação dessas informações e com base nas credenciais do cliente, devolve somente as informações relacionadas a ele (para cuidar da parte da autenticação, foi feita com Basic Auth, recebendo as credenciais através do header da requisição para que se possa saber as informações relacionadas a qual cliente pode ser retornada, caso não seja passada as credenciais do cliente não é possível utilizar os endpoints que contém informações pertencentes a algum cliente), a API em Java também fica responsável por garantir que o frontend tenha as informações necessárias para que o chamado seja aberto vinculado ao cliente. Já a parte de abertura do chamado, atualização dele e retorno de alguma informação relacionada ao chamado, é feita por em Python, uma vez que o próprio modelo que recebe as informações e faz a escolha do guincho também se encontra em Python.

**Endpoints**

**Verbo:** GET

**Endpoint:** http://localhost:8080/PortoApi/api/guinchos

**Descrição:** Retorna uma lista com todos os guinchos disponíveis no cadastro do banco de dados.

**Precisa de credenciais (email, senha):** Não.

**Verbo:** GET

**Endpoint:** http://localhost:8080/PortoApi/api/guinchos/{id}

**Descrição:** Retorna um guincho específico conforme id fornecido.

**Precisa de credenciais (email, senha):** Não.

**Verbo:** GET

**Endpoint:** http://localhost:8080/PortoApi/api/cliente

**Descrição:** Com base nas credenciais fornecidas, é localizado o cliente no banco de dados e retornado o mesmo com as suas informações básicas e os ids referente a endereço e plano de seguro caso para que se possa buscar essas informações no banco caso necessário.

**Precisa de credenciais (email, senha):** Sim.

**Verbo:** GET

**Endpoint:** http://localhost:8080/PortoApi/api/endereco/{id}

**Descrição:** Irá buscar o endereço com base no id informado na url, e irá utilizar as credenciais para verificar se o endereço que está sendo solicitado pertence ou não ao cliente que tem essas credenciais.

**Precisa de credenciais (email, senha):** Sim.

**Verbo:** GET

**Endpoint:** http://localhost:8080/PortoApi/api/plano/{id}

**Descrição:** Irá buscar o plano com base no id informado na url, e irá utilizar as credenciais para verificar se o plano que está sendo solicitado pertence ou não ao cliente que tem essas credenciais.

**Precisa de credenciais (email, senha):** Sim.

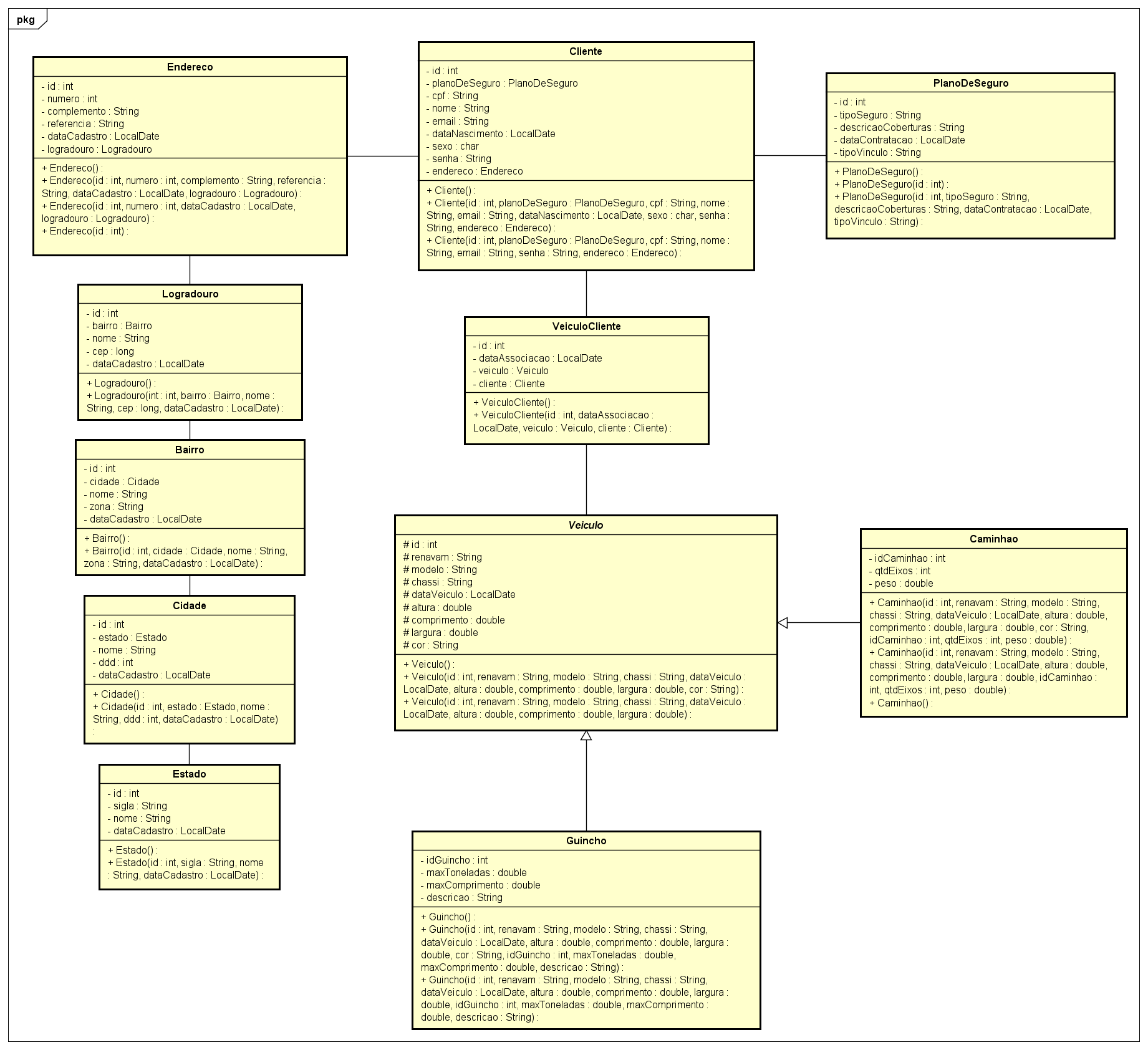
**Verbo:** GET

**Endpoint:** http://localhost:8080/PortoApi/api/caminhao

**Descrição:** Com base nas credenciais informadas, irá buscar todos os veículos que estão vinculados a esse cliente e retornar uma lista contendo-os.

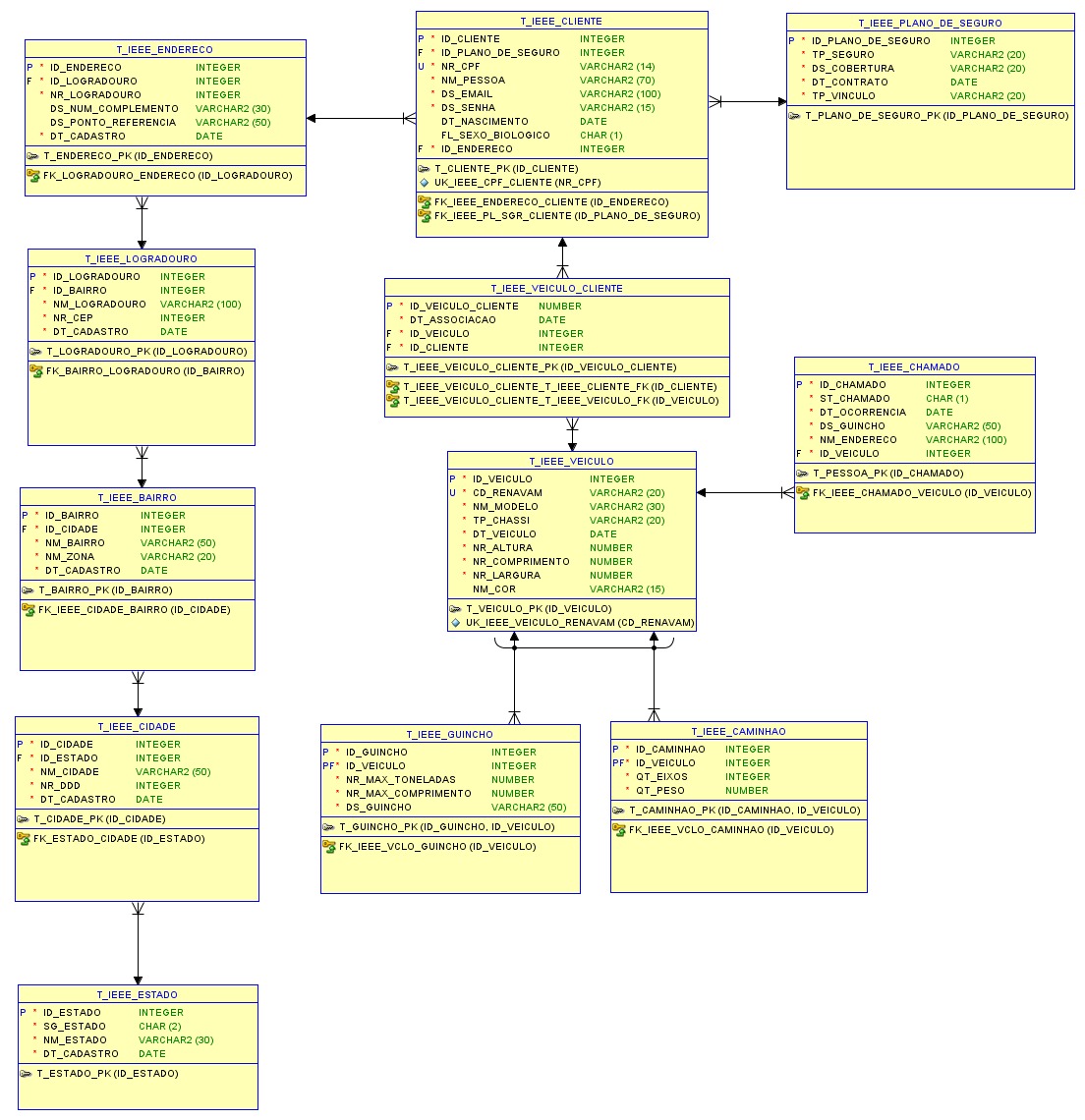
**Precisa de credenciais (email, senha):** Sim.

**Diagrama de Classes**

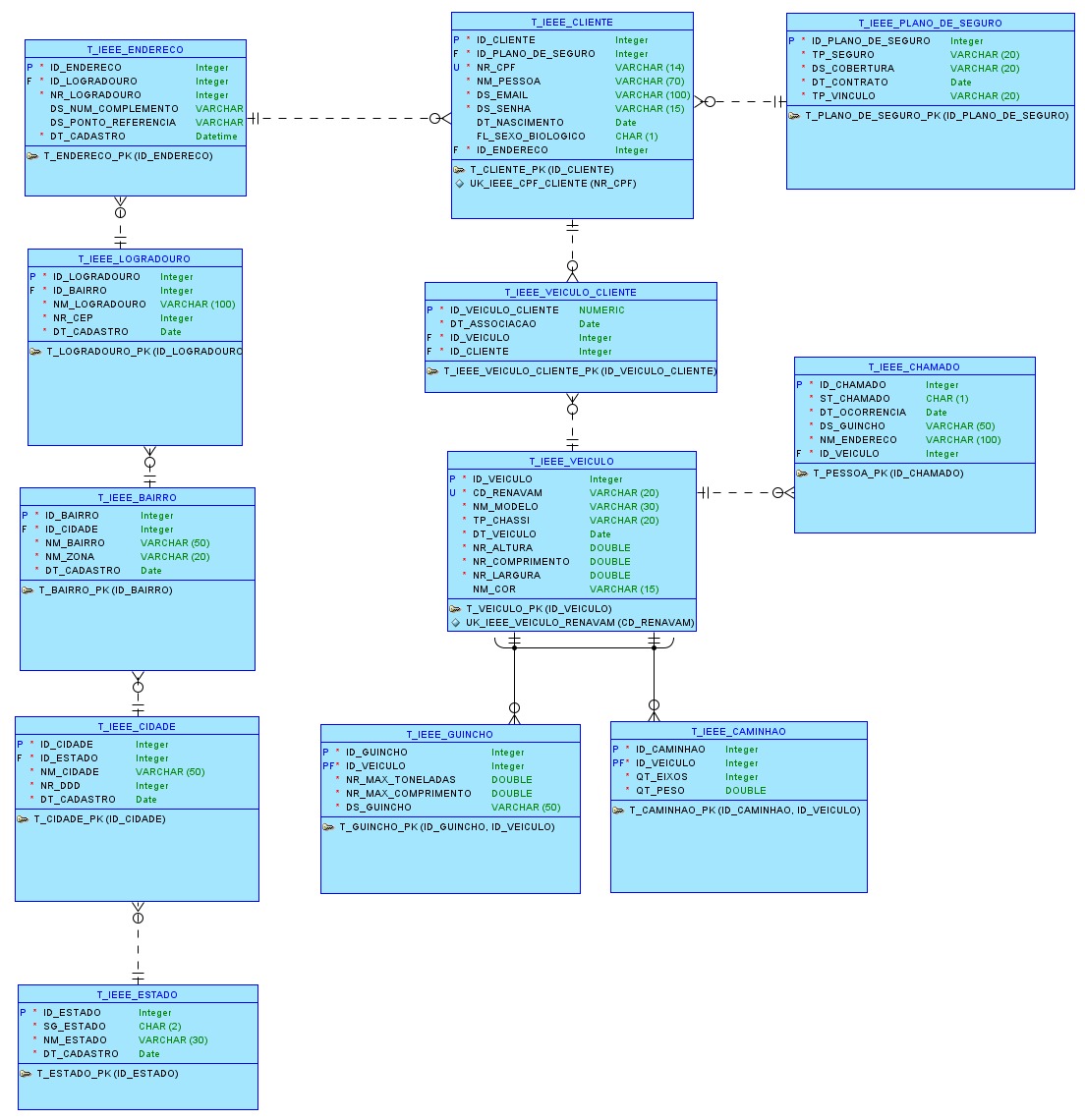
****

**Modelo do banco de dados**

**Relacional**

****

**Lógico**

****

**Protótipo**

