

**Documentação de um**

**Produto de Software**

**BICHO EXPRESS**

**Nome dos Alunos:**

Maria Eduarda Medeiro Porto- 824144948

Matheus Alves Santana- 824144952

Erick Domingues Soares- 82414486

Wellington de Oliveira Sousa - 825240209

Matheus Henrique da Costa e Silva – 82410661

Eduardo Baptistella Gonçalves- 824147595

Gabriel Prieto Lima- 824142064

**2025**

**RESUMO**

O presente projeto visa o desenvolvimento de uma plataforma digital de mobilidade

especializada para animais de estimação, composta por dois aplicativos móveis

um voltado para os usuários (tutores de pets), responsáveis pela solicitação dos

serviços de transporte, e outro para os motoristas, encarregados de atender às

demandas recebidas. Além disso, contará com um portal administrativo para gestão do serviço.

O sistema tem como propósito oferecer uma solução segura, prática e inovadora para o deslocamento de animais, integrando funcionalidades específicas como cadastro detalhado do pet, monitoramento em tempo real da corrida, comunicação direta entre usuário e motorista e métodos de pagamento digitais. Para os motoristas, a solução amplia o leque de oportunidades de trabalho, permitindo a construção de um portfólio profissional e o acompanhamento de ganhos. Para os tutores, a plataforma oferece comodidade, confiança e centralização na solicitação de serviços, garantindo o bem-estar do animal durante o transporte.

O desenvolvimento do projeto contemplou a definição de requisitos funcionais e não funcionais, elaboração de modelos de estrutura e arquitetura de software baseados em microserviços, etapas de prototipagem, critérios de qualidade, além de fases de testes e planejamento de implantação. O sistema foi projetado para ser escalável, multiplataforma e em conformidade com legislações como a LGPD, priorizando a segurança da informação e a experiência do usuário.

Este trabalho tornou-se viável graças à aplicação dos conceitos apresentados na unidade curricular de Modelos, métodos e técnicas de engenharia de software, aliados a pesquisas sobre soluções similares no mercado de transporte e mobilidade pet-friendly.

**Palavras-chave:** Software, Transporte Pet, Mobilidade, Plataforma Digital, Prestador-Cliente.

**LISTA DE FIGURAS**

Figura 1-

**LISTA DE TABELAS**

Tabela 1-

**ÍNDICE DETALHADO**

**1.** **Introdução** 61.1. Tema 71.2. Objetivos a serem alcançados 71.3. Escopo principal 8**2.** **Definição do Modelo de Processo** 11**3.** **Requisitos do Sistema de Software** 123.1. Requisitos Funcionais **Erro! Indicador não definido.**3.2. Requisitos Não-Funcionais **Erro! Indicador não definido.4.** **Projeto** 114.1. Arquitetura Lógica 234.2. Arquitetura Física 23**5.** **Protótipo de Interface** 28**6.** **Critérios de Qualidade de Software** 29**7. Testes** 307.1. Plano de Testes 307.2. Roteiro de Testes 30**Anexo I** 30

1. **Introdução**

Com o crescimento do uso da tecnologia e a expansão de soluções digitais voltadas para mobilidade, surge a necessidade de desenvolver plataformas que atendam nichos específicos da sociedade, oferecendo praticidade, segurança e inovação. Dentro desse contexto, os serviços pet-friendly vêm ganhando destaque, acompanhando o aumento do número de animais de estimação e a valorização do bem-estar animal nas grandes cidades.

O presente trabalho propõe o desenvolvimento de uma plataforma digital de transporte especializado para animais de estimação, denominada **Bicho Express**. O sistema será composto por dois aplicativos móveis – um voltado para os tutores de pets (usuários) e outro para os motoristas responsáveis pelo transporte – além de um portal administrativo para a gestão do serviço.

A solução contempla funcionalidades como cadastro detalhado do animal, solicitação e agendamento de corridas, cálculo automático de rotas, monitoramento em tempo real, comunicação direta entre usuário e motorista, avaliações pós-serviço e múltiplas formas de pagamento digitais. Com isso, pretende-se garantir a segurança, o conforto e a transparência no deslocamento dos animais.

Do ponto de vista dos motoristas, o sistema oferece novas oportunidades de trabalho, possibilitando ganhos extras, organização de corridas e construção de portfólio profissional. Para os tutores, representa uma alternativa confiável e prática para o transporte de seus pets, centralizando todas as etapas em uma única plataforma digital.

O desenvolvimento do projeto seguirá metodologias ágeis (Scrum), visando entregas incrementais e escalabilidade da solução. Além disso, será considerada a conformidade com legislações como a **LGPD** e normas de segurança da informação, assegurando a proteção dos dados e a confiabilidade da plataforma.

* 1. **Tema**

O projeto visa o desenvolvimento de um sistema de transporte especializado para animais de estimação, que funcione como uma plataforma digital de mobilidade pet-friendly. A solução será composta por um aplicativo móvel para usuários (donos de pets) e motoristas, além de um portal administrativo para gestão do serviço. O sistema deverá contemplar funcionalidades específicas para o transporte seguro e confortável dos animais, como cadastro detalhado do pet, requisitos de segurança, monitoramento em tempo real e comunicação direta entre usuário e motorista.

A escolha de São Paulo como sede é estratégica, pois a cidade possui uma grande concentração de animais de estimação e uma demanda crescente por serviços especializados, além de desafios logísticos típicos de grandes metrópoles, como trânsito intenso e diversidade de perfis de usuários.

A arquitetura do sistema será baseada em microserviços para garantir escalabilidade e facilidade de manutenção, utilizando APIs RESTful para comunicação entre front-end e back-end. A interface será desenvolvida com foco em usabilidade e acessibilidade, seguindo padrões modernos de design responsivo

* 1. **Objetivos a serem alcançados**

O principal objetivo do projeto é criar uma plataforma digital intuitiva e confiável para o transporte de animais de estimação, que atenda às necessidades dos donos e garanta o bem-estar dos pets durante o trajeto. Especificamente, pretende-se:

* Desenvolver um aplicativo móvel multiplataforma (iOS e Android) e um portal web para cadastro, solicitação e gerenciamento de corridas.
* Implementar funcionalidades de geolocalização em tempo real, cálculo automático de rotas e estimativa de preços.
* Garantir a segurança dos animais com requisitos técnicos, como motoristas treinados, veículos adaptados e monitoramento via câmera.
* Automatizar processos de agendamento, pagamento e avaliação dos serviços.
* Oferecer suporte ao cliente integrado via chat e notificações push.
* Atender ao público-alvo composto por donos de animais de estimação residentes em São Paulo e região metropolitana, que buscam uma alternativa prática e segura para transportar seus pets.
* Promover inovação no mercado pet, integrando tecnologias de IoT para monitoramento do bem-estar animal durante o transporte.
  1. **Escopo principal**

A proposta principal do projeto é desenvolver um sistema completo de transporte de animais que englobe as seguintes ações e implementações:

* Cadastro e perfil do usuário e do pet: inclusão de informações detalhadas sobre o animal (raça, porte, necessidades especiais) e do dono.
* Solicitação e agendamento de corridas: interface para o usuário solicitar transporte imediato ou agendado, com opções de tipos de veículos adaptados.
* Geolocalização e roteirização: integração com APIs de mapas para rastreamento em tempo real e otimização de rotas.
* Sistema de pagamento integrado: suporte a múltiplas formas de pagamento digitais, com segurança e transparência.
* Avaliação e feedback: sistema de avaliação para motoristas e usuários, promovendo a qualidade do serviço.
* Painel administrativo: dashboard para gestão de motoristas, corridas, faturamento e suporte.
* Notificações e comunicação: envio de alertas via SMS, e-mail e push para status da corrida e comunicação direta entre partes.
* Segurança e conformidade: implementação de protocolos para garantir a segurança dos animais e conformidade com legislações locais.

O desenvolvimento seguirá metodologias ágeis, com entregas incrementais e foco na experiência do usuário, garantindo que o sistema seja escalável e adaptável às necessidades futuras.

* 1. **Modelo e Objetivo de Negócio**

O **Bicho Express** visa ser uma plataforma digital de mobilidade pet-friendly, composta por dois aplicativos – um voltado para os tutores de animais de estimação (usuários) e outro para os motoristas autorizados – além de um portal administrativo para gestão dos serviços. A solução promove a intermediação entre cliente e prestador, garantindo transporte seguro, ágil e confortável para os animais, a partir de um cadastro detalhado e solicitações diretas via aplicativo.

“Temos como objetivo facilitar a busca por um transporte de qualidade para os tutores, gerar novas oportunidades de renda para motoristas qualificados e assegurar o bem-estar animal em cada viagem, entregando autonomia e confiança para ambos os lados.”

* 1. **Serviço e Mercado do Projeto**

Nosso foco é atingir tutores de animais de estimação por meio de uma plataforma digital **intuitiva, prática e segura**, oferecendo um serviço especializado de transporte pet-friendly. A solução garante o bem-estar dos animais durante o trajeto, proporcionando comodidade para os clientes e uma experiência diferenciada em comparação a transportes comuns.

Em relação ao mercado, observa-se um **crescimento constante do setor pet no Brasil**, especialmente em serviços voltados para mobilidade e cuidados com animais. Muitos tutores enfrentam dificuldades em deslocar seus pets de forma segura, seja para consultas veterinárias, pet shops ou viagens, e carecem de alternativas adequadas. Ao mesmo tempo, existem motoristas interessados em novas oportunidades de renda, mas que ainda não encontram plataformas que conectem de forma estruturada esse nicho de mercado.

Com essas informações, o **Bicho Express** direciona seu foco na demanda crescente por transporte especializado, oferecendo um leque maior de possibilidades de atendimento e consolidando-se como uma plataforma **inovadora, confiável e adaptada às necessidades do setor pet**.

* 1. **Concorrência de Mercado**

O mercado pet no Brasil é um dos que mais cresce no mundo, movimentando mais de R$ 60 bilhões anuais e ocupando a terceira posição mundial em faturamento, ficando atrás apenas dos Estados Unidos e da China. No entanto, a área de mobilidade pet ainda é pouco explorada, com poucas soluções que ofereçam transporte seguro, confortável e dedicado exclusivamente a animais de estimação.

Atualmente, os tutores que precisam deslocar seus pets contam com três principais alternativas:

1. Aplicativos de transporte convencionais (Uber, 99, InDrive): permitem o transporte de pets, mas não possuem estrutura especializada, gerando insegurança para os animais e desconforto para motoristas que não são preparados.
2. Táxis pet e empresas locais de transporte: presentes em grandes cidades, mas com alcance limitado, alto custo e pouca praticidade para agendamento imediato.
3. Serviços informais (redes sociais, grupos de WhatsApp): embora comuns, carecem de profissionalização, segurança e padronização nos preços.

Diante disso, o Bicho Express se diferencia por unir a praticidade de um aplicativo de mobilidade urbana com a especialização em transporte animal, atendendo a uma dor real dos tutores e explorando uma oportunidade de mercado em expansão.

TABELA 1- Comparativo de Concorrentes Diretos e Indiretos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Concorrente / Solução** | **Cobertura** | **Foco no Pet** | **Preço Médio** | **Segurança** | **Diferencial** |
| Uber / 99 (Pet friendly) | Nacional | Baixo | R$ 20-60 | Limitada | Alta capilaridade |
| Táxis Pet locais | Regional | Alto | R$ 60-150 | Alto | Transporte dedicado |
| Serviços informais (WhatsApp) | Regional | Médio | Negociável | Baixo | Flexibilidade |
| **Bicho Express (Proposto)** | Nacional (expansão escalável) | Alto | R$ 30-80 | Alto | Plataforma 100% pet-friendly com rastreamento em tempo real |

*Fonte:* Base de pesquisas dos autores

GRÁFICO 1- Nicho De Mercado Pet-Friendly no Brasil

*Fonte:* Base de pesquisas dos autores

O **Bicho Express** tem como diferencial a **especialização em transporte pet**, aliado à praticidade de aplicativos já consolidados. Seu modelo escalável e digital o coloca em posição de vantagem competitiva, com potencial de conquistar rapidamente a confiança dos tutores e expandir para grandes centros urbanos.

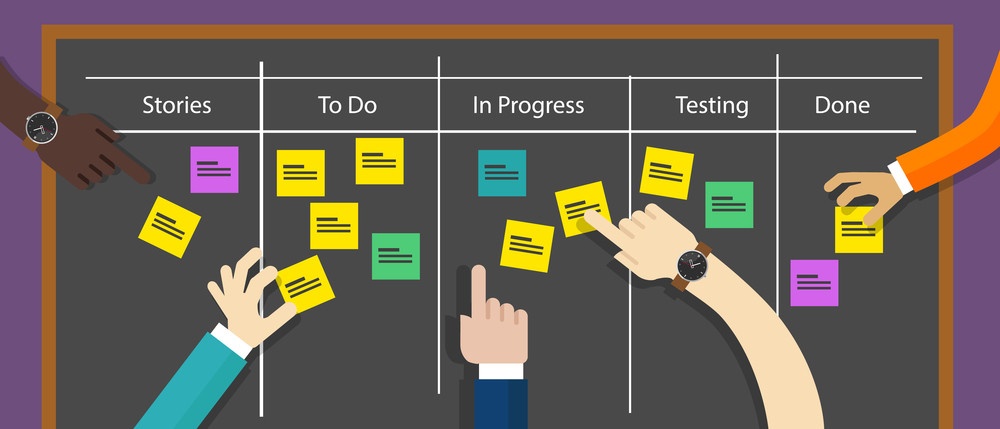
1. **Definição do Modelo de Processo**

A empresa adotou o Scrum como modelo de desenvolvimento para a plataforma de transporte Bicho Express, aproveitando sua flexibilidade e agilidade para projetos complexos e dinâmicos. Com ciclos curtos de trabalho (sprints), estamos entregando funcionalidades rapidamente, permitindo ajustes contínuos baseados no feedback dos usuários e nas demandas do mercado.

Essa metodologia garante foco constante na experiência do usuário, priorizando usabilidade, segurança e inovação. Além disso, facilita a coordenação eficiente entre as equipes multidisciplinares, desde o desenvolvimento mobile até o suporte, assegurando que a arquitetura baseada em microserviços seja implementada de forma escalável e modular.

Com o Scrum, o projeto conta com total transparência no progresso, redução de riscos e entregas contínuas de valor. Isso acelera o lançamento no mercado, garantindo que a solução atenda às expectativas reais dos clientes e fortaleça a posição da empresa no mercado de mobilidade pet.

Optamos por não utilizar modelos tradicionais como Cascata, Iterativo, Prototipação ou Espiral devido às suas limitações em lidar com a complexidade, necessidade de adaptação rápida e constante evolução do projeto. Esses modelos são mais rígidos ou burocráticos, o que poderia dificultar a incorporação de feedbacks e atrasar entregas essenciais, enquanto o Scrum oferece a agilidade e colaboração necessárias para garantir o sucesso dessa iniciativa inovadora.

FIGURA 1- Demonstrativo de Funcionamento Método Scrum

# Desenvolvimento do Software

* 1. **Diagrama de Funcionamento do Aplicativo do Usuário/Tutor de Pets**

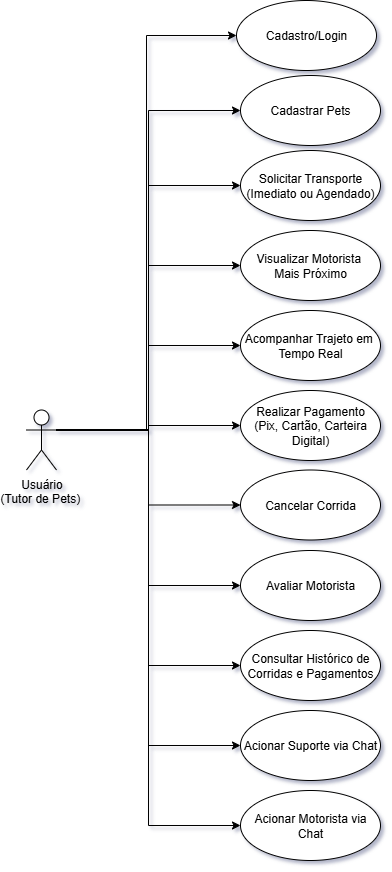
****

FIGURA 2- Diagrama caso de uso do cliente, (Fonte: Autores realizado no Draw.io, 2025)

* 1. **Diagrama de Funcionamento Aplicativo Funcionário/Motorista**

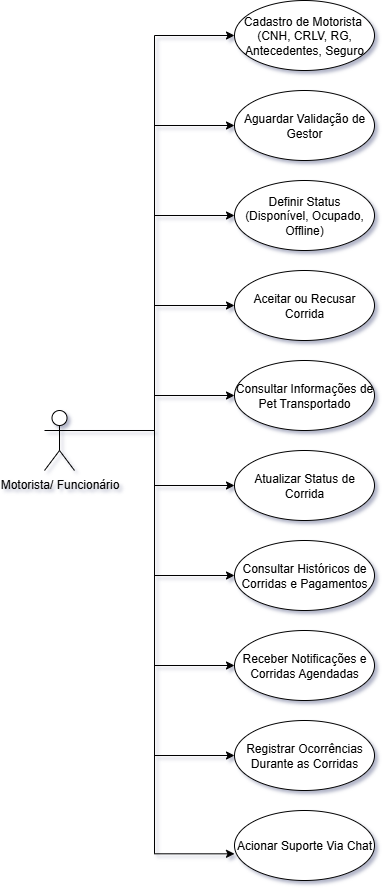
****

FIGURA 3- Diagrama caso de uso do Funcionário/Motorista, (Fonte: Autores realizado no Draw.io, 2025)

* 1. **Diagrama de Funcionamento Gestor do Software**

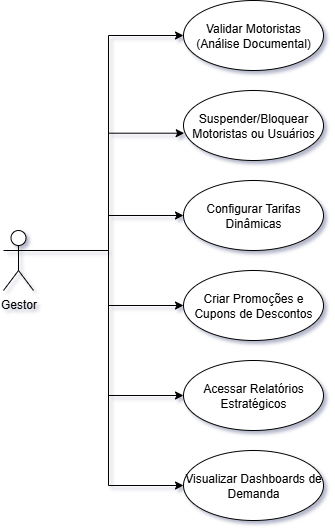
****

FIGURA 4- Diagrama caso de uso do Gestor, (Fonte: Autores realizado no Draw.io, 2025)

* 1. **Diagrama de Funcionamento do Software Setor Administrativo**

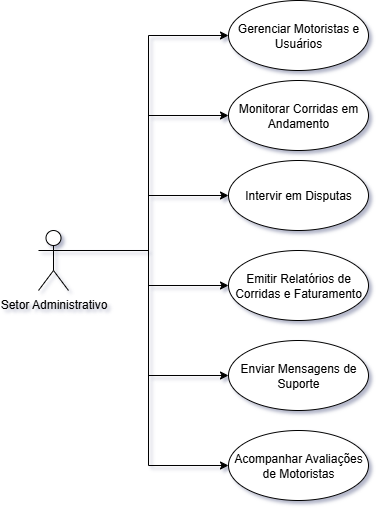
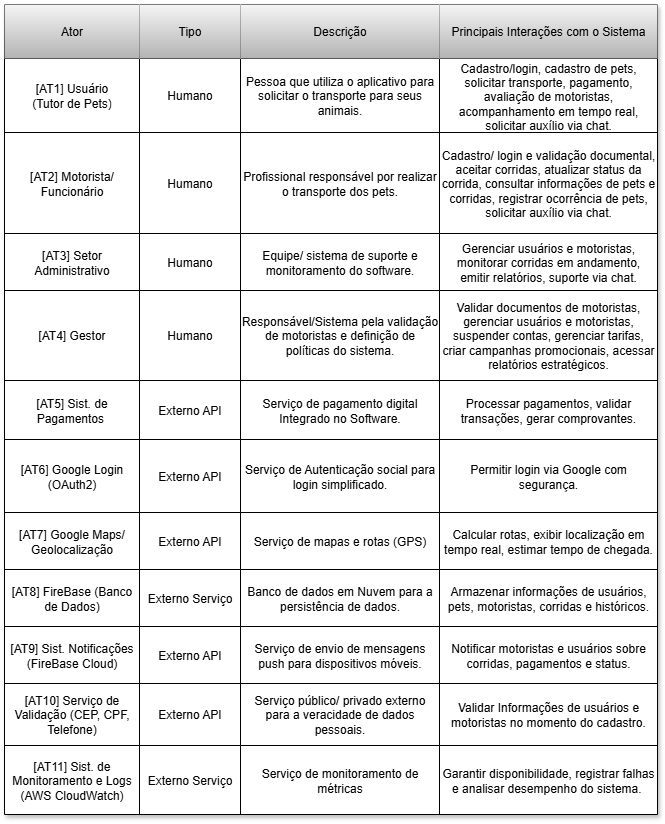


FIGURA 5- Diagrama caso de uso do Setor Administrativo, (Fonte: Autores realizado no Draw.io, 2025)

* 1. **Catálogo de Atores**

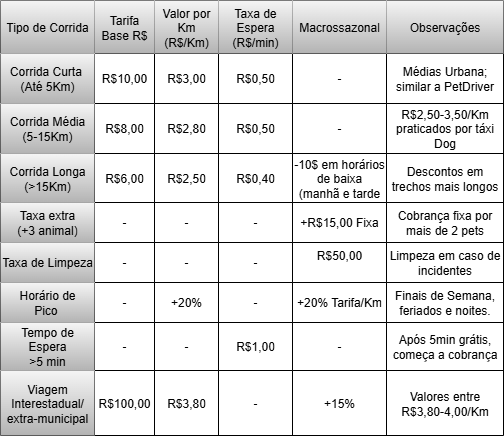
**TABELA 2- Catálogo de Atores do Software**



*Fonte:* Autores, 2025

* 1. **Base de Preços**

**TABELA 3- Base de Preços Calculada Com Softwares concorrentes**

****

*Fonte:* Autores, 2025

* 1. **Requisitos Funcionais**

Os requisitos funcionais definem o comportamento esperado do sistema sob a perspectiva dos usuários, descritos de forma simples para facilitar a compreensão. São organizados por tipos de usuários.

**3.1.1 Requisitos do Usuário (Tutor de Pets)**

RF1 O sistema deve permitir cadastro/login via e-mail, senha, Google e autenticação multifator (MFA).

RF2 O usuário deve poder cadastrar múltiplos pets, com informações detalhadas como espécie, porte, raça, peso, idade, vacinas e necessidades especiais.

RF3 O usuário pode solicitar transporte imediato ou agendado para seus pets.

RF4 Deve exibir o motorista mais próximo, com estimativas de tempo e valor da corrida.

RF5 O tutor pode acompanhar o transporte em tempo real via mapa.

RF6 O sistema deve oferecer múltiplas formas de pagamento, incluindo Pix, cartão de crédito/débito e carteira digital.

RF7 O usuário pode cancelar corridas conforme regras definidas no sistema.

RF8 Deve possibilitar avaliação de motoristas após corrida.

RF9 Usuários podem consultar histórico de corridas e pagamentos realizados.

RF10 Deve permitir acionamento de suporte ao cliente via chat integrado.

**3.1.2 Requisitos do Motorista/Funcionário**

RF11 O sistema deve permitir cadastro de motoristas, com documentos obrigatórios (CNH, antecedentes criminais, CRLV, seguro).

RF12 Apenas motoristas validados por um gestor autorizado podem realizar corridas.

RF13 Motoristas devem indicar seu status: Disponível, Ocupado ou Offline.

RF14 Devem receber solicitações de transporte podendo aceitar ou recusar.

RF15 Motoristas têm acesso a informações do pet transportado, incluindo porte, cuidados e restrições.

RF16 Devem atualizar o status da corrida (a caminho, embarcado, em andamento, concluído).

RF17 Motoristas podem consultar histórico de corridas e seus ganhos.

RF18 Devem receber notificações sobre corridas agendadas.

RF19 Possibilidade de registrar ocorrências durante a corrida.

**3.1.3 Requisitos do Setor Administrativo**

RF20 Deve poder cadastrar, alterar e excluir motoristas e usuários em situações de suporte.

RF21 Monitorar corridas em andamento via painel de controle.

RF22 Intervir em disputas entre usuários e motoristas.

RF23 Emitir relatórios detalhados de corridas, faturamento e cancelamentos.

RF24 Enviar mensagens de suporte via chat interno.

RF25 Acompanhar estatísticas de avaliações de motoristas.

**3.1.4 Requisitos do Gestor**

RF26 Validar motoristas após análise documental.

RF27 Suspender ou bloquear motoristas ou usuários denunciados.

RF28 Configurar tarifas dinâmicas baseadas em preço por km, porte do pet e taxa mínima.

RF29 Criar campanhas promocionais e cupons de desconto.

RF30 Acessar relatórios estratégicos sobre receitas, clientes ativos e cancelamentos.

RF31 Acessar dashboards com previsões de demanda por região e horários de pico.

**3.2 Requisitos Não Funcionais**

Os requisitos não funcionais especificam restrições gerais do sistema, garantindo qualidade e eficiência.

**3.2.1 Requisitos Externos**

**3.2.1.1 Interoperabilidade**

RNF1 Permitir login integrado com Google.

RNF2 Integrar APIs bancárias, CPF, CEP e telefone para validações.

**3.2.1.2 Éticos**

RNF3 Garantir transparência total nos preços antes da corrida.

RNF4 Apenas motoristas aprovados podem realizar transporte de pets.

RNF5 Respeitar princípios de bem-estar animal durante o transporte.

**3.2.1.3 Legais**

RNF6 Estar em conformidade com a Lei Geral de Proteção de Dados (LGPD).

RNF7 Utilizar criptografia para proteger dados sensíveis.

RNF8 Seguir a norma ISO/IEC 27001 para segurança da informação.

**3.2.2 Requisitos Organizacionais**

**3.2.2.1 Entrega**

RNF9 Apenas motoristas validados devem realizar corridas.

RNF10 Filtrar motoristas por localização para otimização das corridas.

RNF11 Permitir pagamento apenas por meios digitais.

RNF12 Bloquear perfis denunciados quando confirmadas irregularidades.

RNF13 Permitir comunicação via chat com criptografia de ponta a ponta.

**3.2.2.2 Implementação**

RNF14 Todas as páginas devem carregar em até 2 segundos.

RNF15 Notificações devem ser entregues em até 5 segundos.

RNF16 Campos de texto devem acionar teclado automaticamente na interface móvel.

RNF17 Textos devem usar fonte padrão, com títulos destacados em negrito.

**3.2.2.3 Padrões**

RNF18 Todas as telas devem seguir design consistente.

RNF19 Botões e campos interativos devem ter comportamentos previsíveis.

RNF20 Código do sistema deve seguir boas práticas de desenvolvimento seguro.

**3.2.3 Requisitos de Facilidade de Uso**

RNF21 A interface deve ser intuitiva e acessível a pessoas com diferentes perfis e necessidades.

RNF22 Cadastro e login devem ser concluídos em até 30 segundos.

RNF23 A busca por motoristas deve permitir filtros por localização, porte do animal, preço e avaliações.

RNF24 Sistema deve oferecer acessibilidade, incluindo alto contraste, leitores de tela e fontes ajustáveis.

**3.2.4 Requisitos de Confiabilidade**

RNF25 Garantir disponibilidade do sistema em 99,9% do tempo.

RNF26 Realizar backups diários com retenção mínima de 12 meses.

RNF27 Tempo máximo para recuperação do sistema após falhas críticas é de 15 minutos.

RNF28 Notificar automaticamente equipe técnica em caso de falhas graves.

**3.2.5 Requisitos de Portabilidade**

RNF29 Aplicativo deve ser compatível com Android 8.0+ e iOS 12+.

RNF30 Aplicativo não deve ultrapassar 100MB e deve consumir pouca bateria.

RNF31 Desenvolver em framework multiplataforma (React Native ou Flutter) para facilitar manutenção e escalabilidade.

**3.2.6 Requisitos de Eficiência**

RNF32 Consultas e operações devem responder em até 2 segundos.

RNF33 Suportar até 100 mil usuários simultâneos.

RNF34 O desempenho do sistema deve ser monitorado continuamente para identificação e mitigação de gargalos.

**3.2.7 Requisitos de Espaço**

RNF35 O sistema deve suportar crescimento da base de dados de corridas, pets e motoristas.

RNF36 Possibilitar escalabilidade horizontal, adicionando servidores conforme demanda.

RNF37 Compatibilidade com armazenamento em nuvem híbrida para garantir disponibilidade e segurança de dados.

**3.3. Fluxo de Requisitos do Software**

**3.3.1. Aplicativo 1- BichoExpress (Usuário/Tutor de Pets)**

**3.3.3.1.1. RF01- Cadastro (Login)**

Este requisito do Cliente na plataforma. O ator pode fazer seu cadastro pelo Google ou pela plataforma inserindo seus dados manualmente.

Fluxo de Requisitos:

1. Após o Cliente instalar e abrir o aplicativo ele é direcionado para a página inicial.
2. Nesta página existem três opções: "Continue pelo Google", fazer o login já possuindo conta, sendo solicitado o e-mail e a senha já cadastrados no momento da criação da conta, e a opção "registrar-se", para cadastrar um novo usuário.
3. Crie seu cadastro direciona o cliente a uma nova página com as seguintes solicitações de campos pessoais do tutor do pet a serem preenchidos:
4. E-mail;
5. Data de nascimento (sendo o mínimo de 18 anos)
6. Nome completo;
7. CPF;
8. Telefone
9. Senha
10. Confirmação de senha
11. Um botão de "Próximo" finaliza o processo
12. O aplicativo aceita somente uma inscrição por CPF
13. O CPF, telefone e e-mail são validados pelo banco de dados correspondentes (APIs)
14. Após a validação o sistema salva todas as informações do ator cliente
15. Com os dados salvos a mensagem "Cadastro Concluído" é imprimido na tela
16. Adicionado a página 'perfil" do aplicativo
17. O cliente pode consultar os seus dados
18. O cliente pode escolher sair da sua conta a partir do botão "sair" no seu perfil
19. O cliente não pode excluir diretamente os seus dados
20. O cliente pode excluir seu cadastro encerrando sua participação no aplicativo

**3.3.3.1.1.1. Subfluxos [SUB1]: Login pelo Google**

1. O cliente tem a possibilidade de criar seu cadastro através da plataforma do Google
2. Um botão localizado na parte inferior da tela escrito "continuar com o Google"
3. Após ser selecionado o botão o cliente deve ser direcionado para uma aba do Google  de autorização de cadastro na plataforma
4. Cliente autoriza
5. Google transfere as informações do cliente para o banco de dados da plataforma
6. Após as informações serem transferidas inserir a mensagem de "cadastro concluído"
7. O cliente é direcionado para a nova página "perfil" do aplicativo

**3.3.3.1.1.2. Subfluxo [SUB2]: Cliente já possui cadastro**

1. O cliente tem a possibilidade de inserir o cadastro já criado em um acesso anterior
2. As caixas de texto no centro da tela direcionam o usuário a inserir seu e-mail e senha já cadastrados anteriormente
3. Após a validação do cliente, o sistema salva todas as informações do ator cliente
4. Após as informações transferidas, inserir a mensagem "Login concluído"
5. O cliente é transferido diretamente para a aba principal de chamada de nova corrida.

**3.3.3.1.1.3. Subfluxo [SUB3]: Telefone**

1. O campo telefone e CPF são obrigatórios de serem preenchidos
2. O número de telefone e CPF são vinculados ao inserido no cadastro e na conta Google
3. Par atualizar o telefone é necessário clicar no campo "telefone", apagar o antigo e inserir um novo
4. Não é permitido alterar o CPF de uma conta já existente
5. É obrigatório pelo menos um telefone por conta e usuário
6. Quando o novo telefone for escrito o cliente deve pressionar o botão ao lado "salvar as informações"
7. Após vinculado, uma nova deverá ser impressa "Novo telefone cadastrado com sucesso!"

**3.3.3.1.1.4. Subfluxo [SUB4]: Saída do aplicativo**

1. Ao entrar na aba "perfil" do aplicativo, o cliente tem a possibilidade de pressionar o botão "sair"
2. Ao pressionar este botão uma mensagem de confirmação aparece
3. Confirmando a sua saída, o cliente tem a possibilidade de se cadastrar novamente com a mesma conta, ou logar com um registro diferente

**3.3.3.1.1.5. Subfluxo[SUB5]: Exclusão de cadastro**

1. Possível apenas depois de concluída a etapa de criação de perfil
2. O cliente seleciona a opção de "excluir conta"
3. Ao pressionar este botão uma mensagem de confirmação aparece, onde o administrador também alerta sobre as consequências causadas ao aceitar essa exclusão
4. Após a validação, o sistema exclui o perfil assim como o solicitado
5. Com os dados deletados a mensagem "cadastro deletado" é imprimida na tela

**3.3.1.2. Fluxos alternativos [FA]:**

**3.3.1.2.1. Fluxo alternativo [FA1]: Restrição de idade**

1. Ocorre quando o usuário tenta realizar a criação de um cadastro , registrando a sua idade como sendo de alguém com menos de 18 anos
2. Ao tentar, uma mensagem automática aparece:" cadastro não permitido para menores de 18 anos"
3. O cadastro só poderá ser concluído se a data de nascimento informada atender ao requisito de maioridade

**3.3.1.2.2. Fluxo Alternativo [FA2]: Nome incompatível**

1. Este fluxo é acionado quando o nome informado não coincide com o registrado no CPF
2. Ao tentar, uma menagem automática aparece. "inserir nome registrado no CPF"
3. O cadastro só poderá ser concluído se o nome informado for o mesmo encontrado no CPF

**3.3.1.2.3. Fluxo Alternativo [FA3]: Desenvolvimento de senha**

1. Ocorre quando o cliente não procede com as regras delimitadas para a construção de uma senha fortificada
2. Enquanto não obedecer esses itens, uma mensagem de alerta será emitida, avisando que o login não pode ser concluído  até que essa etapa seja concluída
3. Quando as regras forem seguidas e as outras informações forem preenchidas, é possível concluir o cadastro

**3.3.1.2.4. Fluxo alternativo [FA4] : Confirmação de senha incorreta**

1. O cliente insere uma senha no campo "confirmação de senha", a qual não está correlacionada com a adicionada no campo "senha"
2. Enquanto as senhas descritas não forem compatíveis, uma mensagem de alerta será emitida, alertando o erro e solicitando a inserção das informações corretamente
3. Quando as regras forem seguidas e as outras informações forem preenchidas, é possível concluir o cadastro

**3.3.1.2.5. Fluxo alternativo [FA5]: Login ou senha incorretos**

1. Ao realizar a tentativa de inserir um cadastro já criado previamente e o mesmo não for localizado no banco de dados, uma mensagem será emitida avisando o preenchimento incorreto do login e ou senha
2. O cadastro só poderá ser concluído quando as informações forem inseridas corretamente

**3.3.1.2.6. Fluxo Alternativo [FA6]: Recuperar senha**

1. Caso o cliente esqueça a senha utilizada em sua conta, é possível que ele realize a recuperação da mesma
2. Um e-mail de recuperação será enviado onde o usuário poderá criar uma nova senha ao clicar no botão "criar nova senha"
3. Esse botão levará a uma página onde pode ser inserida e aprovada para poder ser utilizada
4. O login deve ser realizado novamente no aplicativo

**3.3.3.1.2. RF02- Cadastro de Pets**: Este requisito é voltado par a a criação do perfil do(s) Pet(s) que serão transportados pelo serviço de aplicativo

Fluxo de Requisitos

1. Após o cadastro do tutor concluído com sucesso, o usuário é direcionada para uma página onde poderá realizar o cadastro de um ou mais pets
2. O usuário tem a opção de visualizar, editar e adicionar os campos
   1. Nome do animal
   2. Espécie
   3. Raça
   4. Porte
   5. Peso
   6. Idade
   7. Se é vacinado, esterilizado/castrado
   8. Condições médicas
   9. Outras necessidade especiais
3. Esta etapa deve ser concluída para a solicitação de uma nova corrida
4. Avaliações e comentários recebidos ficarão expostos nesta página
5. Perfil do usuário e do animal ficará visível ao prestador de serviços/motorista, quando a corrida estiver sendo solicitada
6. Após a edição completa do perfil do pet, o usuário será direcionado a aba de solicitar nova corrida
7. O perfil poderá ser acessado a qualquer momento pelo usuário par a edição e consulta

**3.3.3.1.2.1. Subfluxo [SUB1]:**

Campos não inseridos

1. Ocorre quando algum dos campos obrigatórios(nome, estado de vacinação, porte, espécie e peso) do cadastro do pet está incompleto, inválido ou não preenchido
2. O sistema deve imprimir uma mensagem de "dados incompletos" até que os campos sejam plenamente inseridos
3. O sistema após os dados inseridos corretamente deve concluir o cadastro do pet e direcionar o usuário a aba de solicitação de nova corrida

3.**3.3.1.2.2. Subfluxo [SUB2]:**

Visibilidade ao Prestador

1. Quando o usuário solicitar um serviço de corrida através do formulário de solicitação, quando o motorista receber a solicitação de corrida, o perfil do usuário e do pet será visível
2. O prestador poderá acessar o perfil do usuário e do pet com mais detalhes clicando no botão "mais detalhes"

**3.3.3.1.3. RF03 — Solicitar Transporte (Imediato ou Agendado)**

O usuário pode solicitar transporte para um ou mais pets, escolhendo entre corrida imediata (ao vivo) ou corrida agendada (data/hora futura). Deve informar origem, destino, pet(s) e opções (veículo adaptado, necessidade especial).

**Pré-condições**

* Usuário cadastrado e logado (RF01).
* Pelo menos 1 pet cadastrado (RF02) antes de solicitar corrida.
* Localização (GPS) habilitada no dispositivo ou endereço digitado.

**Fluxo de Requisitos**

1. Usuário abre app e seleciona “Solicitar Corrida”.
2. App exibe tela com opções: **Imediato** / **Agendado**.
3. Usuário escolhe pet(s) a transportar (checkbox), origem (GPS ou endereço), destino (endereço), e pode incluir observações/necessidades especiais.
4. Usuário seleciona tipo de veículo, se disponível (ex.: veículo para animais grandes, caixa de transporte).
5. App calcula **estimativa de preço** (Tarifa base + R$/km + adicionais por porte/animal extra) e **ETA** (tempo estimado de chegada).
6. Usuário confirma solicitação; o sistema envia solicitação a motoristas compatíveis (por localização e requisitos do pet).
7. Motorista aceita a corrida (RF14) — sistema notifica usuário com dados do motorista (nome, avaliação, veículo, tempo estimado).
8. App passa para fluxo de acompanhamento (RF05) até conclusão.
9. Ao finalizar, app direciona para pagamento (RF06) e avaliação (RF08).

**3.3.3.1.3.1. Subfluxos**

* **SUB1 — Agendamento com Google Calendar:** se usuário optar, o agendamento pode ser sincronizado com Google Calendar (com permissão).
* **SUB2 — Seleção de múltiplos pets:** usuário marca mais de um pet; sistema recalcula preço e verifica capacidade do veículo.
* **SUB3 — Escolha de método de pagamento pré-aprovado:** se o usuário tiver carteira cadastrada, pode confirmar pagamento automático.

**3.3.3.1.3.2. Fluxos Alternativos**

* **FA1 — Nenhum motorista disponível:** caso não haja motoristas compatíveis/online, app informa “Sem motoristas disponíveis” e sugere agendamento posterior ou notificação quando houver disponibilidade.
* **FA2 — Aceite recusado / timeout:** se nenhum motorista aceitar em X segundos (configurável), sistema reprova a busca e notifica o usuário.
* **FA3 — Endereço inválido:** se o endereço informado for inconsistente/fora de área atendida, app exibe “Endereço fora da área de cobertura”.
* **FA4 — Requisitos especiais não compatíveis:** se o pet tem necessidades que exigem veículo/treinamento específico e não há motoristas compatíveis, exibe mensagem e opções (reprogramar / contatar suporte).

**Pós-condições**

* Solicitação criada no banco de dados com status: PENDENTE → ACEITA → A CAMINHO → EMBARCADO → CONCLUÍDO / CANCELADO.
* Notificações enviadas ao motorista e usuário.
* Registro de estimativa de preço e detalhe do pet.

**Regras de negócio**

* Máximo de X pets por corrida (definir política).
* Taxa extra por animal a partir do 3º pet.
* Multa de cancelamento se cancelado após motorista sair para coleta (ver política do gestor — RF28).

**3.3.3.1.4. RF04 — Exibir Motorista Mais Próximo / Estimativa (Driver Matching)**

Mostrar ao usuário os motoristas disponíveis mais próximos, estimativa de chegada (ETA) e preço estimado antes da confirmação.

**Pré-condições**

* GPS/serviço de localização ativo.
* Motoristas online com status “Disponível” (RF13).

**Fluxo de Requisitos**

1. Após inserir origem/destino, app chama serviço de **matching** (API) que busca motoristas por proximidade e compatibilidade (porte do pet, veículo).
2. Service retorna lista ordenada por distância/tempo/nota.
3. App exibe o motorista sugerido (ou múltiplas opções se houver) com ETA e preço estimado.
4. Usuário pode visualizar perfil do motorista (avaliação, histórico) antes de confirmar.

**3.3.3.1.4.1. Subfluxos**

* **SUB1 — Filtrar por requisitos:** usuário solicita somente motoristas com veículos adaptados ou experiência com pets agressivos; sistema aplica filtro no matching.
* **SUB2 — Mostrar alternativos:** botão “Ver outras opções” lista mais motoristas com ETAs e preços.

**3.3.3.1.4.2. Fluxos Alternativos**

* **FA1 — Motorista indisponível no instante do clique:** sistema atualiza lista em tempo real; se motorista perder disponibilidade, sugere próxima opção.
* **FA2 — Diferença entre estimativa e preço final:** se percurso real divergir do estimado (ex.: desvio), o preço final será recalculado no final da corrida (detalhar na política).

**Pós-condições**

* Motorista indicado e requisitado; tela de espera enquanto aguarda aceitação.

**Regras de negócio**

* Priorizar motoristas por ETA e por avaliação média (configurável).
* Limite de tentativas de chamada a motoristas (p.ex. 3 motoristas em sequência).

**3.3.3.1.4. RF05 — Acompanhar Transporte em Tempo Real (Tracking)**

Mostrar em mapa a localização do motorista, trajeto, status da corrida e tempo estimado de chegada.

**Pré-condições**

* Corrida aceita por motorista (RF14).
* Permissão de localização concedida pelo motorista e usuário (para rastreamento).

**Fluxo de Requisitos**

1. Ao aceitar a corrida, motoristas compartilham localização em tempo real via WebSocket / service de localização.
2. App do usuário exibe mapa com:
   * Posição atual do motorista;
   * Rota prevista e ETA;
   * Status textual (A caminho / Em trânsito / Concluído).
3. Usuário pode abrir painel com detalhes do motorista e pet (RF02 visível).
4. Durante transporte, o app atualiza estimativas e status.
5. Ao final da corrida, status muda para “Concluído” e segue fluxo de pagamento/avaliação.

**3.3.3.1.4.1. Subfluxos**

* **SUB1 — Compartilhar localização temporária para suporte:** usuário pode permitir link de rastreamento por tempo limitado para outra pessoa.
* **SUB2 — Notificações automáticas:** push quando motorista a X minutos de chegada, quando embarcou, quando concluiu.

**3.3.3.1.4.2. Fluxos Alternativos**

* **FA1 — Perda temporária de sinal GPS:** app exibe última posição conhecida e notifica o usuário; tenta reconexão automática.
* **FA2 — Desvio de rota:** se motorista fizer desvio maior que Y%, sistema solicita justificativa ao motorista e notifica usuário.

**Pós-condições**

* Histórico de rastreamento armazenado (logs) para auditoria.

**Regras de negócio**

* Políticas de privacidade sobre armazenamento do trajeto (retention conforme LGPD).
* Tempo máximo de exposição do tracking para terceiros (link temporário).

**3.3.3.1.5. RF06 — Pagamento (Pix, Cartão, Carteira Digital)**

Permitir que o usuário efetue pagamento da corrida através de meios digitais integrados (Pix, cartão, carteira digital). Deve suportar pagamento pré-autorizado ou no final da corrida.

**Pré-condições**

* Usuário com método de pagamento cadastrado ou pronto para inserir no momento.
* Integração com gateway de pagamento ativa.

**Fluxo de Requisitos**

1. Ao confirmar corrida, usuário escolhe método de pagamento:
   * **Pagamento automático (pré-aprovado):** cartão salvo / wallet;
   * **Pagamento no app (ao final):** selecionar Pix, Cartão ou Carteira.
2. Ao término da corrida, app calcula valor final (tarifa base + km + tempo + adicionais).
3. Se método automático: gateway captura o valor; usuário recebe comprovante.
4. Se Pix: app gera QR / chave e confirma recebimento; ou redireciona para app de banco.
5. Se carteira: processa via API correspondente.
6. App gera recibo e atualiza histórico (RF09). Motorista recebe confirmação de pagamento na sua interface.

**3.3.3.1.5.1. Subfluxos**

* **SUB1 — Pagamento dividido/vale:** possibilidade futura de dividir pagamento (não obrigatório no MVP).
* **SUB2 — Promoções/cupons:** aplicar cupom antes do pagamento (RF29 configurado pelo gestor).

**3.3.3.1.5.2. Fluxos Alternativos**

* **FA1 — Falha na captura do cartão:** app solicita método alternativo; usuário pode tentar novamente.
* **FA2 — Chargeback / disputa:** caso de disputa, transações retidas e suporte notificado (RF20).
* **FA3 — Cancelamento com pagamento pré-aprovado:** política de reembolso conforme RF7 e regras do gestor (RF28).

**Pós-condições**

* Transação registrada, recibo gerado, extrato do usuário atualizado.

**Regras de negócio**

* Compliance PCI-DSS para tratamento de cartão (guardar somente token, não dados completos).
* Taxa de serviço da plataforma aplicada automaticamente (definida por RF28).

**3.3.3.1.6. RF07 — Cancelamento de Corrida**

O usuário pode cancelar uma corrida, com regras de cancelamento que podem gerar multa caso o cancelamento ocorra após determinado evento (ex.: após motorista iniciar deslocamento).

**Pré-condições**

* Corrida solicitada e em status PENDENTE/ACEITA/A CAMINHO.

**Fluxo de Requisitos**

1. Usuário acessa a corrida em andamento e pressiona “Cancelar”.
2. App exibe política de cancelamento (tempo grátis, multa aplicável).
3. Usuário confirma cancelamento.
4. Sistema:
   * Se dentro do período sem multa: atualiza status para CANCELADO sem cobrança.
   * Se multa aplicável: aplica taxa e processa cobrança (RF06), notifica motorista e setor administrativo.
5. Motorista recebe notificação do cancelamento e solicitação disponibilizada a outros motoristas.

**3.3.3.1.6.1. Subfluxos**

* **SUB1 — Cancelamento pelo motorista:** similar para motorista, com regras para o motorista (suspensão por recusas repetidas).

**3.3.3.1.6.2. Fluxos Alternativos**

* **FA1 — Usuário contesta multa:** inicia-se processo de disputa via suporte (RF20/24).
* **FA2 — Corrida em agendamento cancelada com prazo X antes:** sem multa.

**Pós-condições**

* Status da corrida atualizado e registro de multa (se houver).

**Regras de negócio**

* Definir janela grátis (ex.: cancelamento até 5 minutos sem custo).
* Política diferenciada para corridas agendadas.

**3.3.3.1.7. RF08 — Avaliação de Motoristas**

Usuário pode avaliar e deixar comentário após finalização da corrida (nota 1–5 e comentário opcional). Avaliação impacta perfis dos motoristas.

**Pré-condições**

* Corrida concluída (RF05 finalizada e pagamento realizado).

**Fluxo de Requisitos**

1. Após conclusão, app exibe tela de avaliação.
2. Usuário escolhe nota (1 a 5) e escreve comentário opcional.
3. Submete avaliação; o sistema atualiza média do motorista e armazena comentário.
4. Motorista pode visualizar avaliação em seu app (RF17) e setor admin pode monitorar (RF25).

**3.3.3.1.7.1. Subfluxos**

* **SUB1 — Avaliação anônima:** política de privacidade define se comentário mostra nome do usuário.
* **SUB2 — Denúncia associada:** opção de marcar incidentes graves que acionam RF20 (suporte).

**3.3.3.1.7.2. Fluxos Alternativos**

* **FA1 — Usuário não avalia:** é possível pular a avaliação; sistema pode enviar lembrete após X horas.

**Pós-condições**

* Nota e comentário salvos; média do motorista recalculada.

**Regras de negócio**

* Sistema de pesagem de avaliações (ex.: avaliações recentes têm peso maior).
* Mecanismo de moderação para comentários ofensivos.

**3.3.3.1.8. RF09 — Histórico de Corridas e Pagamentos**

Usuário pode consultar o histórico completo de corridas, recibos, detalhes do trajeto, data, valor e motorista atendente.

**Pré-condições**

* Histórico populado após corridas concluídas.

**Fluxo principal**

1. Usuário acessa menu “Histórico”.
2. App lista corridas por data, mostrando resumo (origem, destino, valor, status).
3. Ao clicar em um registro, usuário visualiza detalhes: mapa da rota, tempo, taxa aplicada, recibo (opção de exportar PDF/email).
4. Histórico financeiro exibe totais mensais e filtros (por período).

**3.3.3.1.8.1. Subfluxos**

* **SUB1 — Exportar recibo:** gerar PDF e enviar por e-mail.
* **SUB2 — Solicitar estorno / disputa:** iniciar processo de disputa para cobrança indevida (encaminha para RF20).

**3.3.3.1.8.2. Fluxos Alternativos**

* **FA1 — Registro ausente:** se corrida não aparece, usuário contata suporte; log de auditoria verificado.

**Pós-condições**

* Acesso ao registro e possibilidade de acionar suporte.

**Regras de negócio**

* Retenção de registros conforme LGPD e políticas (ex.: 12 meses mínimo, anonimização posterior conforme regras).

**3.3.3.1.8. RF10 — Suporte via Chat Integrado**

Serviço de suporte ao cliente via chat in-app, para dúvidas, denúncias, problemas técnicos ou disputas de pagamento.

**Pré-condições**

* Usuário logado (pode também permitir acesso limitado para não-logados com dados básicos).

**Fluxo principal**

1. Usuário acessa “Suporte” / “Chat”.
2. Seleciona tópico (Ex.: Problema com cadastro, pagamento, corrida, denúncia).
3. Chat abre com atendente ou bot (triagem automática).
4. Para problemas simples, bot responde; para casos complexos, chat é encaminhado a atendente humano (Setor Administrativo — RF24).
5. Atendente visualiza contexto da corrida/histórico e pode solicitar ações (reembolsos, investigações).
6. Conversa arquivada no histórico do usuário e setor admin.

**3.3.3.1.8.1. Subfluxos**

* **SUB1 — Anexar evidências:** usuário envia foto, vídeo, prints (limite de tamanho).
* **SUB2 — Escalar para gestor:** para casos críticos, agente escalona para gestor (RF26/RF27).

**3.3.3.1.8.2 Fluxos Alternativos**

* **FA1 — Off-hours:** fora do horário de atendimento humano, bot registra ticket e informa tempo estimado de retorno.
* **FA2 — Spam / abuso:** sistema de moderação trata mensagens ofensivas, aplicando sanções se necessário.

**Pós-condições**

* Ticket de suporte criado, solução registrada, métricas de atendimento atualizadas.

**Regras de negócio**

* SLA de resposta definido (ex.: resposta inicial em 2 horas; resolução em X dias).
* Armazenamento das conversas conforme LGPD e política de retenção.

**3.3.2. Aplicativo 1- Requisitos do Motorista / Funcionário**

**3.3.2.1. RF11 — Cadastro de Motorista (CNH, CRLV, antecedentes, seguro)**

Permitir que um motorista se cadastre na plataforma fornecendo dados pessoais e documentos obrigatórios para verificação.

**Pré-condições**

* Motorista possui dispositivo móvel com o app instalado.
* Motorista possui documentos digitais (fotos/scan) prontos.

**Fluxo de Requisitos**

1. Motorista abre app e seleciona “Quero ser motorista / cadastrar”.
2. Preenche dados pessoais: nome completo, data de nascimento, CPF, telefone, e-mail.
3. Anexa documentos exigidos: CNH (foto frente/verso), CRLV do veículo, comprovante de seguro, comprovante de antecedentes criminais.
4. Informa dados do veículo: marca, modelo, placa, capacidade (kg/pets), se possui itens de segurança para pets (cintos, caixas).
5. Envia e confirma cadastro.
6. Sistema realiza validações automáticas básicas (formato CPF, telefone) e envia documentos para fila de análise do gestor (RF26).
7. Motorista recebe tela com status: **Em análise** e notificação por push/email.

**3.3.2.1.1.Subfluxos**

* **SUB1 — Upload de documentos via câmera**: captura direta da câmera com validação de borda (imagem legível).
* **SUB2 — Salvamento temporário**: possibilidade de salvar rascunho do cadastro e continuar depois.

**3.3.2.1.2. Fluxos Alternativos**

* **FA1 — Documento ilegível/incompleto**: sistema solicita novo envio com mensagem explicativa.
* **FA2 — Dados inválidos (ex.: CPF já cadastrado)**: alerta e direcionamento para recuperar conta ou suporte.

**Pós-condições**

* Cadastro criado com status **PENDENTE\_VALIDACAO**.
* Notificação enviada ao gestor para análise documental.

**Regras de negócio**

* Motorista com CPF já vinculado a conta não pode se cadastrar novamente.
* Documentos expirados (ex.: seguro vencido) não são aceitos — sistema requer upload válido.

**3.3.2.2. RF12 — Validação por Gestor (liberação para realizar corridas)**

Fluxo de validação documental e aprovação por gestor para liberação do motorista na plataforma.

**Pré-condições**

* Cadastro do motorista concluído (RF11) e documentos anexados.

**Fluxo de Requisitos**

1. Gestor recebe notificação/posto de trabalho com lista de cadastros pendentes.
2. Gestor acessa painel de verificação documental (visualiza CNH, CRLV, seguros e antecedentes).
3. Gestor marca **Aprovar** ou **Rejeitar** com justificativa.
4. Se **Aprovar**: status do motorista atualiza para **Ativo** e motorista recebe notificação de liberação.
5. Se **Rejeitar**: motorista recebe notificação com justificativa e instruções para reenvio.

**3.3.2.2.1. Subfluxos**

* **SUB1 — Solicitar documentação adicional**: gestor pede documentos extras via chat interno; motorista envia e processo reavalia.
* **SUB2 — Aprovação condicional**: gestor aprova com restrição (ex.: só pode atender pets pequenos) — tag de restrição aplicada ao perfil.

**3.3.2.2.2. Fluxos Alternativos**

* **FA1 — Divergência nos documentos**: gestor abre ticket para investigação com setor administrativo (RF20).

**Pós-condições**

* Motorista fica **Ativo** (pode receber corridas) ou **Rejeitado** (não pode operar até regularização).

**Regras de negócio**

* Aprovação manual é obrigatória antes de aceitar corridas.
* Histórico da decisão (quem aprovou, data) deve ser registrado para auditoria.

**3.3.2.3. RF13 — Indicar Status (Disponível / Ocupado / Offline)**

Motorista deve gerenciar seu status operacional no app.

**Pré-condições**

* Motorista logado e perfil ativo.

**Fluxo principal**

1. Motorista abre app e acessa painel de controle.
2. Seleciona um dos status: **Disponível**, **Ocupado**, **Offline**.
3. App atualiza status no backend e transmite alteração via websocket para o sistema de matching.
4. Usuários e sistema visualizam disponibilidade atualizada.

**3.3.2.3.1. Subfluxos**

* **SUB1 — Status automático**: ao aceitar corrida, status muda automaticamente para **Ocupado**; ao concluir, volta para **Disponível**.
* **SUB2 — Horário de trabalho**: motoristas podem definir janelas de disponibilidade futura.

**3.3.2.3.2. Fluxos Alternativos**

* **FA1 — Conexão perdida**: se app perde conexão, status define como **Offline** após timeout.

**Pós-condições**

* Motorista passa a ser elegível (ou não) para receber solicitações.

**Regras de negócio**

* Motoristas que ficam repetidamente **Offline** após aceitar corridas podem ser penalizados (configurável).

**3.3.3.4. RF14 — Receber/aceitar/recusar solicitações de transporte**

Motorista recebe solicitações compatíveis e decide aceitar ou recusar.

**Pré-condições**

* Motorista com status **Disponível**.

**Fluxo de Requisitos**

1. Motorista recebe notificação push com detalhes resumidos da corrida (origem, destino, tipo de pet, distância estimada, preço estimado).
2. No app, botão **Aceitar** e **Recusar** são apresentados.
3. Motorista tem X segundos para responder (ex.: 30s).
4. Ao **Aceitar**, app altera status para **A caminho** e direciona para o fluxo de navegação; usuário recebe confirmação com dados do motorista.
5. Ao **Recusar**, sistema tenta chamar próximo motorista elegível.

**3.3.3.4.1. Subfluxos**

* **SUB1 — Aceite automático para corridas agendadas**: para corridas agendadas, motorista pode aceitar com antecedência.
* **SUB2 — Motivo da recusa**: opção de informar motivo (ex.: veículo incompatível) para métricas.

**3.3.3.4.2. Fluxos Alternativos**

* **FA1 — Timeout**: se o motorista não responde em X segundos, é considerado **Recusar** e próximo motorista é acionado.
* **FA2 — Aceitar e depois cancelar**: se motorista cancela após aceitar, regras de penalidade aplicam-se.

**Pós-condições**

* Corrida atualizada para **Aceita** com link de navegação e dados trocados.

**Regras de negócio**

* Limite de recusas por período; excesso resulta em advertência/suspensão temporária.

**3.3.3.5. RF15 — Acesso a informações do pet**

Motorista pode visualizar, antes de aceitar (resumo) e após aceitar (detalhe), informações do pet (porte, necessidades, restrições).

**Pré-condições**

* Usuário cadastrou pet (RF02).
* Privacidade e consentimento respeitados (usuario autorizou compartilhar).

**Fluxo de Requisitos**

1. Ao notificação de corrida, motorista vê resumo do pet (espécie, porte, restrições).
2. Ao aceitar, motorista pode abrir perfil completo do pet (vacinação, condições médicas, observações).
3. Motorista confirma que possui condições para realizar o transporte (opcional checklist).

**3.3.3.5.1. Subfluxos**

* **SUB1 — Solicitar esclarecimento ao usuário**: botão para enviar mensagem antes de aceitar (ex.: confirmar se o pet está com guia).

**3.3.3.5.2. Fluxos Alternativos**

* **FA1 — Informação sensível faltante**: se dados críticos não estiverem preenchidos (ex.: risco de agressividade), motorista pode recusar sem penalidade.

**Pós-condições**

* Motorista tem ciência das exigências e pode preparar veículo/equipamentos.

**Regras de negócio**

* Motoristas não podem negar transporte por critério discriminatório; motivos para recusa devem ser registrados.

**3.3.3.6. RF16 — Atualizar status da corrida (a caminho, embarcado, em andamento, concluído)**

Motorista atualiza o ciclo de vida da corrida através de status.

**Pré-condições**

* Corrida aceita.

**Fluxo de Requisitos**

1. Motorista inicia navegação e pressiona **A caminho**.
2. Ao chegar ao ponto de coleta, pressiona **Cheguei / Embarcando**.
3. Após pet embarcado, atualiza para **Em transporte**.
4. Ao chegar no destino e finalizar, pressiona **Concluído**.
5. Cada alteração dispara notificações push para o usuário e atualiza o histórico.

**3.3.3.6.1. Subfluxos**

* **SUB1 — Fotos de embarque/desembarque**: opção de registro fotográfico (para disputas).
* **SUB2 — Confirmação via código**: usuário confirma recebimento com código PIN para segurança.

**3.3.3.6.2. Fluxos Alternativos**

* **FA1 — Ocorrência durante transporte**: motorista registra ocorrência (RF19) e altera status para **Em ocorrência**; suporte e usuário são notificados.

**Pós-condições**

* Registro de timestamps de cada status armazenado.

**Regras de negócio**

* Ordem de status deve ser respeitada (não pode ir de PENDENTE → CONCLUÍDO sem passar por A CAMINHO).
* Falta de atualização de status sem justificativa pode gerar penalidade.

**3.3.3.7. RF17 — Histórico de corridas e ganhos**

Motorista pode consultar histórico de corridas, valores recebidos, taxas e extratos.

**Pré-condições**

* Motorista ativo com corridas realizadas.

**Fluxo de Requisitos**

1. Motorista acessa menu **Histórico / Ganhos**.
2. Sistema lista corridas por data e exibe detalhes (valor bruto, taxa da plataforma, valor líquido, data).
3. Motorista pode filtrar por período (semana, mês) e exportar extrato em PDF/CSV.
4. Motorista visualiza saldo disponível e histórico de pagamentos (transferências efetuadas).

**3.3.3.7.1. Subfluxos**

* **SUB1 — Solicitar pagamento**: botão para solicitar saque/transferência para conta bancária cadastrada.
* **SUB2 — Verificar disputas**: opção de visualizar corridas com disputas/chargebacks.

**3.3.3.7.2. Fluxos Alternativos**

* **FA1 — Divergência no valor**: iniciar disputa para revisão do cálculo (RF20).

**Pós-condições**

* Extratos gerados e solicitações de saque registradas.

**Regras de negócio**

* Período de pagamento (ex.: semanal/mensal) e taxas de transferência devem estar explícitos.

**3.3.3.8. RF18 — Notificações sobre corridas agendadas**

Motorista recebe notificações para corridas futuras agendadas.

**Pré-condições**

* Motorista aceitou/agendou corrida com antecedência.

**Fluxo de Requisitos**

1. Sistema agenda notificações: 24h antes, 1h antes e 10 min antes (configurável).
2. Motorista recebe push / email com resumo e opção de confirmar presença.
3. Se motorista confirmar, sistema marca como **Confirmado**; se recusar, reabre busca.

**3.3.3.8.1. Subfluxos**

* **SUB1 — Remarcar**: motorista pode propor novo horário (sujeito à aceitação do usuário).

**3.3.3.8.2. Fluxos Alternativos**

* **FA1 — Não confirmação**: se motorista não confirmar até X horas antes, sistema tenta substituir.

**Pós-condições**

* Notificações e confirmações registradas.

**Regras de negócio**

* Política de substituição e penalidades por faltas em corridas agendadas.

**3.3.3.9. RF19 — Registrar ocorrências durante a corrida**

Motorista pode registrar eventos/ocorrências (acidentes, comportamento do pet, danos) durante ou após corrida.

**Pré-condições**

* Corrida em andamento ou finalizada.

**Fluxo principal**

1. Motorista acessa opção **Registrar Ocorrência** no app da corrida.
2. Seleciona tipo de ocorrência (Saúde, Agressividade, Acidente, Danos ao veículo, Limpeza).
3. Anexa fotos/descrição e indicação de urgência.
4. Submete; sistema cria ticket e notifica setor administrativo e usuário simultaneamente.
5. Caso necessário, aciona políticas (reembolso, taxa de limpeza) e inicia fluxo de suporte (RF24).

**3.3.3.9.2. Subfluxos**

* **SUB1 — Acionamento de emergência**: opção para ligar para serviços de emergência/veterinário (se necessário).

**3.3.3.9.2. Fluxos Alternativos**

* **FA1 — Ocorrência falsa / disputa**: inicia investigação com logs, fotos e declarações.

**Pós-condições**

* Ticket criado e rastreado no painel administrativo.

**Regras de negócio**

* Todos os tickets devem ter SLA de resposta; ocorrências graves podem levar à suspensão provisória do motorista até investigação.

**3.3.4 Aplicativo 1- Requisitos do Setor Administrativo**

**3.3.4.1. RF20 — Gerenciar (cadastrar, alterar, excluir) motoristas e usuários**

Painel administrativo para suporte com CRUD completo sobre perfis de motoristas e usuários (com logs e justificativas).

**Pré-condições**

* Usuário administrativo autenticado com privilégios adequados.

**Fluxo de Requisitos**

1. Admin acessa painel e busca usuário/motorista por filtros (nome, CPF, status).
2. Admin pode editar dados cadastrais, anexar documentos, atualizar status ou excluir (com justificativa e registro).
3. Exclusão lógica (soft delete) por padrão — para conformidade LGPD, opção para exclusão definitiva mediante solicitação e auditoria.
4. Alterações geram logs de auditoria (quem, quando, o quê).

**3.3.4.1.1Subfluxos**

* **SUB1 — Reativar perfil**: restaurar contas suspensas mediante análise.
* **SUB2 — Forçar validação**: solicitar nova documentação.

**3.3.4.1.2. Fluxos Alternativos**

* **FA1 — Tentativa de exclusão sem justificativa**: sistema nega até que justificativa seja informada.

**Pós-condições**

* Alterações persistidas e logadas.

**Regras de negócio**

* Alterações críticas (ex.: exclusão) requerem duplo fator de autenticação e justificativa.

**3.3.4.2. RF21 — Monitorar corridas em andamento via painel**

Painel em tempo real com mapa e lista de corridas para monitoramento operacional.

**Pré-condições**

* Admin autenticado; sistemas de rastreamento funcionando.

**Fluxo de Requisitos**

1. Admin abre painel “Operações” e visualiza mapa com pins das corridas ativas.
2. Seleciona corrida para visualizar detalhes (usuário, motorista, rota, status, histórico de mensagens).
3. Pode enviar mensagens para motorista/usuário via chat interno e intervir (ex.: reatribuir corrida).

**3.3.4.2.1. Subfluxos**

* **SUB1 — Filtros operacionais**: filtrar por região, status, tipo de ocorrência.

**3.3.4.2.2. Fluxos Alternativos**

* **FA1 — Perda de telemetria**: se perda de sinal, painel mostra última posição e alerta equipe.

**Pós-condições**

* Ações tomadas e registradas (mensagens, reassign).

**Regras de negócio**

* Logs de intervenção são obrigatórios para auditoria.

**3.3.4.3. RF22 — Intervir em disputas entre usuário e motorista**

Ferramentas para investigação e resolução de conflitos (reembolsos, advertências, bloqueios temporários).

**Pré-condições**

* Ticket de disputa criado (RF19 ou RF10).

**Fluxo de Requisitos**

1. Admin acessa ticket com evidências (fotos, logs, avaliações).
2. Avalia contexto e decide: Reembolso total/ parcial, advertência ao motorista, suspensão temporária, bloqueio definitivo.
3. Aplica ação no sistema (processa reembolso via gateway, altera status do motorista) e notifica partes.
4. Registra decisão e motivo.

**3.3.4.3.1. Subfluxos**

* **SUB1 — Pedido de informações**: solicitar provas adicionais ao motorista/usuário.

**3.3.4.3.2. Fluxos Alternativos**

* **FA1 — Caso não resolvido**: escalar para gestor (RF26) com resumo.

**Pós-condições**

* Disputa encerrada; ações aplicadas; registro de auditoria.

**Regras de negócio**

* Reembolsos seguem política financeira e devem passar por aprovação gerencial dependendo do valor.

**3.3.4.4.RF23 — Emitir relatórios detalhados**

Gerar relatórios de corridas, faturamento, cancelamentos e KPIs para análise.

**Pré-condições**

* Admin autenticado com permissão de relatório.

**Fluxo de Requisitos**

1. Admin acessa módulo de relatórios e define período/ filtros (região, motorista).
2. Gera relatório (tabela/CSV/PDF) com métricas: número de corridas, receita bruta, cancelamentos, avaliação média.
3. Pode agendar envio automático para e-mails executivos.

**3.3.4.4.1. Subfluxos**

* **SUB1 — Relatórios customizados**: combinar múltiplos filtros e métricas.

**3.3.4.4.2. Fluxos Alternativos**

* **FA1 — Dados incompletos**: alertar sobre ausência de dados e sugerir janela diferente.

**Pós-condições**

* Relatório exportado e armazenado.

**Regras de negócio**

* Relatórios financeiros sensíveis devem ter controle de acesso e logs.

**3.3.4.5. RF24 — Enviar mensagens de suporte via chat interno**

Ferramenta de chat interno para atendimento ao cliente e comunicação com motoristas.

**Pré-condições**

* Admin autenticado; integração de chat funcionando.

**Fluxo de Requisitos**

1. Admin abre chat do ticket ou conversa direta com motorista/usuário.
2. Envia mensagens, solicita evidências e aplica ações.
3. Encerrado o atendimento, registra resolução e tempo de atendimento.

**3.3.4.5.1. Subfluxos**

* **SUB1 — Templates de resposta**: respostas rápidas para casos comuns.

**3.3.4.5.2. Fluxos Alternativos**

* **FA1 — Escala para gestor**: caso crítico, chat é marcado e escalado.

**Pós-condições**

* Conversa arquivada e vinculada ao ticket.

**Regras de negócio**

* SLA de atendimento deve ser medido e mantido.

**3.3.4.6. RF25 — Acompanhar estatísticas de avaliações**

Painel com métricas de avaliação de motoristas (média, tendências, flags).

**Pré-condições**

* Dados de avaliações disponíveis.

**Fluxo de Requisitos**

1. Admin acessa dashboard de avaliações.
2. Visualiza motoristas com média baixa, histórico de reclamações e volume de avaliações.
3. Pode aplicar ações (treinamento, advertência, bloqueio).

**3.3.4.6.1. Subfluxos**

* **SUB1 — Alertas automáticos**: motoristas com média < X por período geram alerta automático.

**3.3.4.6.2. Fluxos Alternativos**

* **FA1 — Contestação de avaliação**: motoristas podem solicitar revisão; admin avalia.

**Pós-condições**

* Ações tomadas e registradas.

**Regras de negócio**

* Sistema de peso para avaliações (mais recentes podem ter maior peso).

**3.3.4 Aplicativo 1 - Requisitos do Gestor**

**3.3.4.1. RF26 — Validar motoristas (Análise documental)**

Gestor realiza análise documental profunda e decide aprovação/suspensão.

**Pré-condições**

* Cadastro enviado (RF11).

**Fluxo de Requisitos**

1. Gestor abre fila de cadastros pendentes.
2. Realiza checagens externas se necessário (ex.: checar antecedentes em base pública).
3. Aprova ou rejeita com justificativa.
4. Decide restrições (ex.: operar somente em determinadas regiões) se aplicável.

**3.3.4.1.1. Subfluxos**

* **SUB1 — Solicitar auditoria**: em caso de inconsistência, pedir suporte jurídico.

**3.3.4.2.2. Fluxos Alternativos**

* **FA1 — Aprovação por exceção**: gestor pode aprovar temporariamente sob condição (ex.: reavaliação em 30 dias).

**Pós-condições**

* Decisão gravada e comunicado ao motorista.

**Regras de negócio**

* Registrar data e responsável pela aprovação (compliance).

**3.3.4.2. RF27 — Suspender / bloquear motoristas ou usuários denunciados**

Poder de ação para suspender/ bloquer contas conforme investigações e políticas.

**Pré-condições**

* Ticket de denúncia ou investigação concluída.

**Fluxo de Requisitos**

1. Gestor acessa caso e determina sanção (advertência, suspensão temporária, bloqueio permanente).
2. Executa ação no sistema; notifica partes com justificativa.
3. Define prazo de suspensão ou condições de reativação.

**3.3.4.2.1. Subfluxos**

* **SUB1 — Suspensão administrativa provisória**: bloqueio enquanto investiga.

**3.3.4.2.2. Fluxos Alternativos**

* **FA1 — Apelação**: usuário/motorista pode solicitar revisão; gestor reavalia.

**Pós-condições**

* Conta atualizada e ações registradas.

**Regras de negócio**

* Sanções baseadas em política de conduta e documentação do caso.

**3.3.4.3. RF28 — Configurar tarifas dinâmicas**

Gestor define regras de tarifação (preço por km, taxa mínima, adições por porte, horário de pico).

**Pré-condições**

* Gestor autenticado com permissão de preço.

**Fluxo de Requisitos**

1. Gestor acessa módulo de tarifas.
2. Define parâmetros: tarifa base, R$/km, taxa por porte, taxa mínima, multiplicador de pico.
3. Salva regras; sistema aplica em tempo real para cálculo de estimativas.
4. Versões anteriores armazenadas (audit trail).

**3.3.4.3.1. Subfluxos**

* **SUB1 — Tarifas por região**: definir zonas com preços diferenciados.
* **SUB2 — Promoções temporárias**: ligar/desligar promoções (RF29).

**3.3.4.3.2. Fluxos Alternativos**

* **FA1 — Teste de tarifa**: simular cenários antes de aplicar.

**Pós-condições**

* Novas regras ativadas e aplicadas em cálculos.

**Regras de negócio**

* Mudanças de tarifa comunicadas com antecedência e registradas.

**3.3.4.4. RF29 — Criar campanhas prom. e cupons de desconto**

Gestor cria e gerencia campanhas promocionais para atrair usuários.

**Pré-condições**

* Gestor com permissão de marketing.

**Fluxo de Requisitos**

1. Gestor acessa módulo de promoções.
2. Define cupom (código, desconto fixo/percentual, validade, regras de elegibilidade).
3. Define público alvo (ex.: novos usuários, região X).
4. Ativa campanha; sistema aplica desconto em cálculo de pagamento.

**3.3.4.4.1. Subfluxos**

* **SUB1 — Relatório de campanha**: métricas de uso e ROI.

**3.3.4.4.2. Fluxos Alternativos**

* **FA1 — Cancelar campanha**: encerrar antes da validade.

**Pós-condições**

* Campanha ativa e métricas sendo coletadas.

**Regras de negócio**

* Limites de uso por usuário e políticas de cumulatividade.

**3.3.4.5. RF30 — Acessar relatórios estratégicos**

Gestor obtém relatórios estratégicos (receita, churn, KPIs regionais).

**Pré-condições**

* Gestor autenticado.

**Fluxo de Requisitos**

1. Gestor seleciona métricas e período.
2. Sistema gera dashboards interativos (gráficos, tabelas).
3. Possibilidade de exportar e agendar envios.

**3.3.4.5.1. Subfluxos**

* **SUB1 — Drill-down**: detalhamento por região/motorista.

**3.3.4.5.2. Fluxos Alternativos**

* **FA1 — Dados inconsistentes**: alertar equipe de dados.

**Pós-condições**

* Insights obtidos e decisões tomadas.

**Regras de negócio**

* Controle de acesso e logs; relatórios financeiros possuem camada extra de segurança.

**3.3.4.6. RF31 — Acessar dashboards com previsões de demanda**

Dashboards com previsões (machine learning / heurísticas) por região e horário.

**Pré-condições**

* Modelos de previsão treinados e integrados ao sistema.

**Fluxo de Requisitos**

1. Gestor abre painel de previsão.
2. Visualiza heatmaps por região e previsão de demanda por hora/dia.
3. Utiliza dados para ajustar tarifas, ofertas e alocação de motoristas.

**3.3.4.6.1. Subfluxos**

* **SUB1 — Simulação de cenários**: testar impacto de ajuste de tarifas.

**3.3.4.6.2. Fluxos Alternativos**

* **FA1 — Modelo indisponível**: fallback para heurística simples (média histórica).

**Pós-condições**

* Ajustes operacionais realizados com base nas previsões.

**Regras de negócio**

* Periodicidade de retraining do modelo e validação de acurácia.

1. **Projeto**
   1. **Arquitetura Lógica**

Este capítulo especifica a arquitetura do sistema de transporte especializado para animais de estimação, detalhando sua composição lógica e física, além das justificativas para a escolha dos componentes, garantindo escalabilidade, manutenção facilitada e atendimento às necessidades do usuário.

## 4.1.1 Arquitetura de Software (Lógica)

### A arquitetura do sistema Bicho Express é baseada no modelo de microserviços, que divide o sistema em componentes independentes, cada um responsável por funcionalidades específicas, facilitando a manutenção, escalabilidade e a incorporação de novas funcionalidades futuras. Essa abordagem modular permite uma comunicação eficiente entre os componentes via APIs RESTful, favorecendo a interoperabilidade entre o aplicativo móvel (usuário e motorista), portal administrativo e serviços de terceiros (como APIs de mapas, pagamentos e notificações).

### Cada microserviço é responsável por um domínio do sistema, o que garante que o impacto de mudanças em uma parte do sistema seja isolado. Isso facilita atualizações contínuas adotando a metodologia ágil Scrum, com entregas incrementais.

### 4.1.1 Layers do Sistema

As camadas do sistema foram organizadas para distribuir funcionalidades específicas de acordo com o tipo de usuário, garantindo clareza lógica e eficiência na implementação:

**Login e Autenticação: Controle do cadastro, login via e-mail, Google e autenticação multifator, além da gestão de sessões e segurança.**

**Cadastro e Perfil: Gestão de perfis dos usuários e pets, incluindo detalhes como raça, porte, vacinas e necessidades especiais.**

**Solicitação e Agendamento: Interface para solicitar corridas imediatas ou agendadas, tipo de veículo adaptado, cálculo e otimização de rotas.**

### ****Geolocalização e Rastreamento:** Integração com APIs de mapas para localização em tempo real de motoristas e acompanhamento da corrida.**

### ****Pagamento:** Processamento de pagamentos via Pix, cartões e carteira digital, assegurando transparência e segurança.**

### ****Avaliações e Feedback:** Registro e exibição de avaliações entre usuários e motoristas para estimular confiança e qualidade.**

### ****Histórico de Corridas:** Gerenciamento do status das corridas, pagamentos e cancelamentos.**

### ****Painel Administrativo:** Gestão de motoristas, corridas, faturamento, suporte e relatórios para o setor administrativo.**

### ****Notificações e Comunicação:** Sistema de mensagens push, chat criptografado e alertas para manter usuários informados.**

### ****Segurança e Conformidade:** Tratamento seguro dos dados, conformidade com LGPD e normas ISO/IEC 27001, além de protocolos de bem-estar animal.**

### 4.1.2 Justificativa da Arquitetura

A escolha da arquitetura baseada em microserviços permite:

* Escalabilidade horizontal, com adição de serviços conforme crescimento do sistema;
* Independência entre módulos, facilitando atualizações e manutenção sem impactar toda a aplicação;
* Implementação ágil, alinhada à metodologia Scrum, com entregas incrementais e feedback contínuo;
* Integração com APIs externas (pagamento, mapas, notificações) sem comprometer a estabilidade do sistema;
* Modularidade, permitindo futuras implementações de IoT ou novos serviços relacionados ao transporte pet.

### 4.1.3 Tier

O **Tier** é o modelo de arquitetura que realiza a separação do aplicativo em partes, combinando conceitos físicos e lógicos. Essa divisão permite que diferentes equipes trabalhem de forma independente em cada camada, garantindo que alterações em uma camada não causem impacto nas demais.

A estrutura física e lógica do sistema segue o modelo **N-Tier**, permitindo divisão e administração independentes das camadas, aumentando a flexibilidade e eficiência:

**Apresentação:** camada visível ao usuário, responsável por coletar informações e interagir diretamente com o aplicativo.

**Aplicativo:** camada responsável por processar as informações recebidas da camada de apresentação, realizando lógica de negócio e regras de transporte.

**Banco de dados:** camada onde os dados são armazenados em um servidor NoSQL (Firebase), garantindo persistência e segurança das informações.

**Cache:** banco de memória temporário usado para armazenar informações acessadas com frequência, acelerando o acesso sem necessidade de consultar o banco de dados principal. Deve ser utilizado para itens que não requerem atualizações constantes, como avaliações, portfólios profissionais, lista de serviços, mensagens automáticas e FAQ.

**API Gateway:** camada que gerencia a comunicação entre o aplicativo principal e outras plataformas, controlando o tráfego de informações externas.

**4.1.4. Armazenamento Cloud Computer**

Para garantir alta disponibilidade, segurança e escalabilidade, o sistema utilizará a infraestrutura de **cloud computing AWS (Amazon Web Services)**, escolhida pela sua robustez e serviços inovadores:

Utilização do **AWS EC2**, com instância configurada para alta capacidade de memória e processamento, capaz de suportar apps móveis e backend, além do banco de dados Firebase.

A instância EC2 oferece recursos para hospedagem dos microserviços, com capacidade para múltiplas máquinas virtuais independentes, permitindo isolamento e escalabilidade planejada.

Configuração de alta velocidade de rede (clusters em zona de disponibilidade) para baixa latência e alta velocidade de acesso ao banco de dados e APIs.

Backups automáticos e políticas de recuperação acelerada em caso de falhas para garantir a segurança dos dados.

**4.1.5. Linguagem de Programação**

Para acelerar o desenvolvimento e garantir compatibilidade multiplataforma, o sistema utilizará principalmente:

**JavaScript/TypeScript** para desenvolvimento do backend (Node.js) e frontend (React Native para apps móveis), garantindo código unificado e fácil manutenção.

Frameworks modernos que suportam desenvolvimento rápido, com vastas bibliotecas para integração com APIs externas e atendimento a requisitos de performance, segurança e usabilidade.

Uso de **React Native** para desenvolvimento multiplataforma (iOS e Android) para oferecer uma experiência de usuário consistente e otimizada.

* 1. **Arquitetura Física**
     1. **Configurações de Hardware**

O sistema Bicho Express será executado em uma infraestrutura de cloud computing, para garantir escalabilidade, alta disponibilidade e segurança para operações. A seguir está a configuração proposta:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.Tabela 1- Amazon EC2- t3.large

*OBS:* A infraestrutura poderá ser dimensionada para instâncias maiores ou adicionais conforme o crescimento do sistema.

* + 1. Gráfico, Gráfico de cascata

       O conteúdo gerado por IA pode estar incorreto.**Diagrama de Funcionamento de Hardware**

(Figura 1 - Diagrama de funcionamento do Hardware, Fonte:Autores realizado no

Lucidchart,2025)

* + 1. **Configurações de Rede**

Para garantir a segurança e eficiência da rede usada pelo sistema, as seguintes configurações serão adotadas:

**AWS Virtual Private Cloud (VPC):** Para isolar a rede do sistema do tráfego público, garantindo segurança e controle do tráfego interno.

**Grupos de segurança (Security Groups):** Definição restrita de portas e protocolos permitidos para comunicação, minimizando riscos externos.

**Balanceador de Carga (Load Balancer):** Distribui as requisições dos usuários entre as instâncias de backend automaticamente, assegurando alta disponibilidade e escalabilidade.

**Cluster de Posicionamento:** Para despesas críticas, como banco de dados, serviços que demandam alta velocidade e baixa latência utilizam clusters na mesma zona de disponibilidade, reduzindo latência e aumentando a largura de banda interna da AWS.

**Protocolos Seguros**: Comunicação entre componentes do sistema e usuários garante HTTPS com TLS/SSL, protegendo a troca de dados.

**Redes de Distribuição de Conteúdo (CDN):** Para entrega rápida de conteúdo estático, como imagens, ícones e recursos do aplicativo no front-end.

**4.2.4. Topologia lógica (resumo do fluxo)**

* Usuário (app móvel iOS/Android) → HTTPS → CloudFront (opcional CDN) → DNS (Route 53) → ALB / API Gateway → API (microserviços em containers/EC2) → Firebase (dados), S3 (arquivos), Redis/ElastiCache (cache/sessões) → Serviços externos (Payments, Google Maps, FCM).
* Administração / Painel Web → VPN / Bastion → Console/Admin UI → Mesma API interna.
* Logs & monitoramento: CloudWatch / OpenSearch / S3. Backups e snapshots automáticos.

**4.2.5. Inventário de componentes e configuração recomendada**

|  |  |  |  |
| --- | --- | --- | --- |
| **Componente** | **Função** | **Recomendação / Especificação** | **Observações** |
| **VPC** | Rede privada | VPC com 3 sub-redes públicas + 6 sub-redes privadas (3 AZs ideal) | NACLs, roteamento entre subnets |
| **Subnets** | Isolamento | Public (ALB, NAT, Bastion) + Private (microserviços, cache, jobs) | Alta disponibilidade via múltiplas AZs |
| **ALB** | Balanceamento HTTP(S) | Application Load Balancer (HTTPS 443) com ACM (certificado) | Terminação TLS no ALB |
| **Auto Scaling** | Escalabilidade | ASG mínimo 2 instâncias t3.large (MVP), escalonamento baseado em CPU/latência | Min 2 (AZ redundancy), Max sugerido 50 (veja cálculo) |
| **EC2 (app)** | Compute | **t3.large** (2 vCPU, 8GB) — para MVP; recomenda-se migrar para **ECS Fargate** ou **EKS** para containers | Em produção, prefira instâncias de família M ou C se CPU-bound |
| **Containers** | Isolamento | Docker + Node.js/Typescript microservices | Use ECR para imagens |
| **Firebase** | Banco de Dados | Firestore / Realtime Database (NoSQL) | Conexão segura via service account; exportações regulares p/ S3 |
| **S3** | Armazenamento objetos | 100 GB+ (arquivos, imagens, backup exports) | Ciclo de vida, versioning e cross-region replication (se DR) |
| **EBS gp3** | Root / storage local | 100 GB gp3 por EC2 | IOPS ajustável conforme necessidade |
| **EFS (opcional)** | Arquivos compartilhados | NFS compartilhado entre instâncias | Se necessário para uploads/processamento |
| **ElastiCache (Redis)** | Cache / sessão | 2 nós (multi-AZ) — para rastreamento em tempo real / sessão / rate limiting | Diminui latência / carga no DB |
| **NAT Gateway** | Acesso saída | 1 por AZ (redundância) | Custo por hora + GB |
| **Bastion / SSM** | Acesso admin | SSM Session Manager (recomendado) + bastion para emergency | Não abrir SSH ao mundo |
| **CloudFront** | CDN (opcional) | Distribuição para frontend estático e mapas cache | Reduz latência / custos da API |
| **Route53** | DNS | DNS primário, health checks | Failover e routing policies |
| **ACM** | Certificados TLS | Cert manager para ALB / CloudFront | TLS 1.2+ |
| **WAF + Shield** | Segurança | Proteção contra OWASP / DDoS | Regras de IP blocking |
| **CloudWatch & OpenSearch** | Logs / métricas | Métricas + alarms + dashboard + OpenSearch para logs | Retenção 90 dias (logs) |
| **Backup** | Snapshot / export | EBS snapshots, S3 backups, Firestore export diário | Retenção 12 meses (RNF) |
| **IAM / KMS** | Segurança | IAM roles least privilege + KMS para at-rest encryption | Rotação de chaves, logs de uso |
| **CI/CD** | Deploy | GitHub Actions / CodePipeline + ECR + Terraform | Deploy automatizado, rollback |
| **Monitoring / APM** | Observability | Datadog / NewRelic / AWS X-Ray (opcional) | Tracing de requests |

1. **Protótipo de Interface**

Neste item deve ser apresentado o protótipo do projeto. O protótipo é um recurso que deve ser adotado como estratégia para levantamento, detalhamento, validação de requisitos e modelagem de interface com o usuário (usabilidade).

Referência: UC Modelos, métodos e técnicas da engenharia de software

1. **Critérios de Qualidade de Software**

Neste item devem ser listados e descritos os critérios de garantia da qualidade do processo que serão considerados no Projeto.

Referência: UC Gestão e Qualidade de software

**7. Testes**

**7.1. Plano de Testes**

Neste item deve ser criado o plano de testes do sistema, permitindo a validação do sistema por parte do desenvolvedor, através da verificação dos requisitos do sistema desenvolvido.

Referência: UC Gestão e Qualidade de software

**7.2. Roteiro de Testes**

Neste item devem ser registrados os testes realizados no sistema tendo como base o Plano de Testes do Sistema. O roteiro de testes deve ser elaborado com base nos casos de uso ou user stories (cartão de histórias) de forma manual ou automatizada.

Referência: UC Gestão e Qualidade de software

**Anexo I**

Neste item deve ser anexado o roteiro de entrevista ou questionário respondido, caso tenha sido aplicado.

Referência: UC Modelos, métodos e técnicas da engenharia de software