

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA
SOUZA**
FATEC PRAIA GRANDE

Desenvolvimento de Software Multiplataforma

ALAN RODRIGUES MONTEIRO MATIAS

GABRIEL RIGONI MARTINS

VITOR MARVULLE DE ALMEIDA

**AGILIDADE E CONECTIVIDADE: EXPLORANDO O USO DO PIX NO
SISTEMA DE TRANSPORTE PÚBLICO**

Praia Grande/SP

2025

ALAN RODRIGUES MONTEIRO MATIAS

GABRIEL RIGONI MARTINS

VITOR MARVULLE DE ALMEIDA

**AGILIDADE E CONECTIVIDADE: EXPLORANDO O USO DO PIX NO
SISTEMA DE TRANSPORTE PÚBLICO**

Projeto Interdisciplinar apresentado na disciplina de Laboratório de Desenvolvimento Web, em cumprimento aos requisitos para a conclusão do 4º módulo do curso de Tecnologia em Desenvolvimento de Software Multiplataforma.

Orientadora: Prof.^a Me. Eulaliane Aparecida Gonçalves.

Praia Grande/SP

2025

SUMÁRIO

1 INTRODUÇÃO.....	3
1.1 Tema.....	3
1.2 Objetivos.....	3
1.2.1 Objetivo Geral.....	3
1.2.2 Objetivo Específico.....	4
1.3 Problematização.....	4
1.4 Justificativa.....	4
1.5 Metodologia.....	4
1.6 Organização do Trabalho.....	5
2 DESCRIÇÃO GERAL DO SISTEMA.....	6
2.1 Descrição do problema.....	6
2.2 Principais Envolvidos e suas características.....	6
2.2.1 Usuários do Sistema.....	6
2.2.2 Mapa da empatia.....	7
2.2.3 Desenvolvedores do Sistema.....	10
2.3 Regras de Negócio.....	11
3 REQUISITOS DO SISTEMA.....	13
3.1 Requisitos Funcionais (RF).....	13
3.2 Requisitos Não Funcionais (RNF).....	16
Fluxo: Clicando no botão “Cadastrar Motorista” na tela de Dashboard. Permite voltar para a página Dashboard através do botão superior direito. Descrição: Permite cadastrar novos motoristas inserindo ‘nome’, ‘CPF’ e ‘senha’	24
3.3.1. Diagrama de Navegação.....	32
4 ARQUITETURA DO SISTEMA.....	33
4.1 Componentes Principais.....	33
4.2 Diagrama de Arquitetura.....	34
4.3 Segurança.....	35
4.4 Escalabilidade.....	35
4.5 Manutenção.....	35
4.6 Integração.....	35
4.7 Limitações.....	35
5 ARQUITETURA DE DADOS.....	36
5.1 Modelo Lógico da Base de Dados.....	36
5.2 Criação Física do Modelo de Dados.....	38
5.3. Dicionário de Dados.....	39
6 PROJETO DE SOFTWARE.....	41
6.1 Design dos Componentes.....	41
6.3 Diagrama de Sequência ou Fluxo de Interações.....	43
6.4 Regras de Negócio Detalhadas.....	43
6.5 Estratégias de Tratamento de Erros.....	44

7 IMPLEMENTAÇÃO.....	46
7.1 Estrutura Geral do Projeto.....	46
7.2 Modelos.....	47
7.2.1 Modelo Usuario.....	47
7.2.2 Modelo Funcionario.....	47
7.2.3 Modelo Linha.....	48
7.2.4 Modelo Passagem.....	48
7.2.5 Modelo Historico.....	48
7.3 Views.....	49
7.3.1 View Home.....	49
7.3.2 View Quantidade.....	50
7.3.3 View Formulario.....	50
7.3.4 View Pagamento.....	50
7.3.6 View Bilhete.....	50
7.3.7 View Login.....	51
7.3.8 View Dashboard.....	51
7.3.9 View add_linha e editar_linha.....	51
7.3.10 View excluir_linha.....	51
7.3.11 View add_motorista e lista_motorista.....	51
7.3.12 View linha_motorista e home_motorista.....	52
7.3.13 View validar_bilhete.....	52
7.4 Templates.....	52
7.4.1 Template base_page e mensagem.html.....	52
7.4.2 Template home.....	53
7.4.3 Template quantidade.....	53
7.4.4 Template formulario.....	53
7.4.5 Template pagamento.....	53
7.4.6 Template formulario.....	54
8 REFERÊNCIAS.....	56

1 INTRODUÇÃO

O transporte público municipal da cidade de Praia Grande representa uma das principais formas de locomoção para grande parte da população, especialmente trabalhadores e estudantes. Embora o sistema conte com tecnologias modernas, como veículos equipados com ar-condicionado e opções de pagamento por meio de dinheiro e cartão transporte, ainda há limitações quanto à diversidade de formas de pagamento disponíveis. Essas limitações impactam diretamente a experiência dos usuários, principalmente em situações em que não possuem saldo no cartão transporte ou dinheiro em espécie.

Considerando esse cenário, o presente projeto propõe o desenvolvimento de uma aplicação web que permita o pagamento de passagens de ônibus por meio do PIX. Essa solução visa proporcionar maior comodidade aos usuários e ampliar a acessibilidade ao transporte público, especialmente em situações emergenciais ou de dificuldade de acesso aos meios de pagamento convencionais.

1.1 Tema

Desenvolvimento de uma aplicação web para facilitar o pagamento no sistema de transporte público da cidade de Praia Grande/SP, utilizando o método de pagamento instantâneo PIX.

1.2 Objetivos

1.2.1 Objetivo Geral

Disponibilizar uma alternativa ao modelo convencional de pagamento de passagens no transporte público municipal de Praia Grande, de modo a atender usuários que não possuam dinheiro em espécie ou crédito disponível no cartão transporte.

1.2.2 Objetivo Específico

Desenvolver e implementar uma aplicação web para facilitar o pagamento de passagens no transporte público de Praia Grande por meio de PIX, oferecendo uma alternativa prática e acessível, garantindo facilidade de uso e acessibilidade para todos os perfis de usuário.

1.3 Problematização

O atual sistema de pagamento do transporte público municipal apresenta limitações que podem causar situações de desconforto e transtorno aos usuários, especialmente quando estes não possuem nenhuma das formas de pagamento atualmente aceitas. A ausência de meios para consultar o saldo do cartão antes do embarque, aliada à crescente redução no uso de dinheiro em espécie, agrava essas dificuldades e compromete a agilidade e eficiência do serviço de transporte urbano.

1.4 Justificativa

A proposta deste projeto foi motivada pelos resultados obtidos por meio de uma pesquisa realizada com moradores do município de Praia Grande. Através de um formulário online, foi possível identificar as principais dificuldades enfrentadas pelos usuários do transporte público local, destacando-se a falta de opções de pagamento e o desconhecimento prévio do saldo disponível no cartão transporte. Os dados coletados evidenciam a necessidade de uma solução tecnológica que amplie a acessibilidade e facilite o uso do transporte público.

1.5 Metodologia

Para o desenvolvimento da aplicação, serão adotadas metodologias ágeis, com destaque para as abordagens Scrum e Kanban, amplamente utilizadas no setor de tecnologia da informação para otimizar a organização e execução de tarefas. O uso do Scrum permitirá a realização de entregas incrementais por meio de ciclos de desenvolvimento (sprints), com reuniões periódicas para alinhamento e avaliação contínua do progresso. O Kanban, por sua vez, será utilizado para a visualização do

fluxo de trabalho, contribuindo para a identificação de gargalos e a melhoria contínua do processo.

No que tange à modelagem e desenvolvimento da aplicação, serão aplicados os princípios da Programação Orientada a Objetos (POO), bem como a modelagem de dados voltada para bancos relacionais. A utilização da POO contribuirá para a organização, reutilização e manutenção do código-fonte, enquanto a modelagem relacional garantirá a integridade e eficiência no armazenamento e recuperação dos dados da aplicação.

1.6 Organização do Trabalho

Este trabalho está estruturado em capítulos e seções que seguem uma lógica sequencial e coerente. O Capítulo 1 apresenta a introdução ao tema, incluindo os objetivos da pesquisa, a problematização, a justificativa e a metodologia utilizada. O Capítulo 2, intitulado “Descrição Geral do Sistema”, aborda o problema a ser resolvido, descreve os principais atores envolvidos no desenvolvimento da aplicação — como usuários e desenvolvedores — e mapeia as principais características e necessidades desses perfis. Além disso, são apresentadas as regras de negócio que regem o funcionamento da aplicação proposta. Essa organização visa garantir uma compreensão clara e gradual do contexto e das soluções propostas, preparando o leitor para as etapas seguintes de análise, desenvolvimento e validação do sistema.

2 DESCRIÇÃO GERAL DO SISTEMA

2.1 Descrição do problema

Muitos usuários do transporte público enfrentam problemas com as opções de pagamento do transporte público. Acontece que se o usuário estiver sem saldo no cartão, não tiver cartão, ou dinheiro em cédulas mesmo tendo dinheiro em sua conta ele não consegue pegar o transporte público municipal de forma fácil.

O Sistema propõe solucionar os problemas dos usuários de transporte público municipal de Praia Grande, trazendo uma opção de pagamento moderna, rápida e segura.

Diante disso, esse sistema se baseia numa solução de pagamento via Pix para o transporte público da região.

2.2 Principais Envolvidos e suas características

2.2.1 Usuários do Sistema

O sistema será utilizado por três perfis principais de usuários: os passageiros do transporte público, que realizarão a compra e validação das passagens; os motoristas e/ou cobradores, responsáveis pelo controle de acesso dos passageiros nos veículos; e o administrador da empresa de transporte, que vai gerenciar as operações, realizará os devidos cadastros, acessar relatórios e monitorar o fluxo de pagamentos.

2.2.2 Mapa da empatia

Análise dos Perfis de Usuário

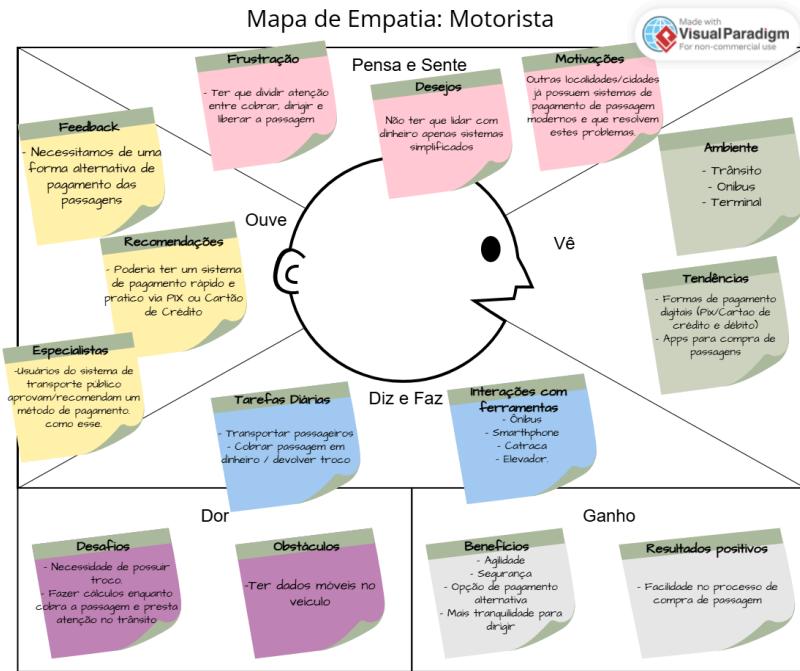
O desenvolvimento do sistema de pagamento digital para o transporte público foi fundamentado na análise empática de três perfis de usuário: Motorista, Administrador e Passageiro. Por meio da construção de Mapas de Empatia, identificamos aspectos comportamentais, emocionais e contextuais que guiaram as decisões de projeto e implementação do sistema.

Perfil: Motorista

O motorista enfrenta frustrações ao precisar dividir sua atenção entre dirigir, cobrar a passagem e liberar a catraca, além de lidar constantemente com troco e manuseio de dinheiro. Seus desejos incluem um sistema simplificado, que elimine a necessidade de lidar com dinheiro em espécie. O ambiente em que atua envolve trânsito intenso e locais de embarque, como ônibus e terminais, o que exige foco e agilidade.

Ouvimos motoristas solicitarem uma forma alternativa de pagamento, preferencialmente digital (como PIX ou cartão de crédito). Eles recomendam soluções modernas, semelhantes às adotadas em outras localidades, que melhorem a segurança e a agilidade no processo. Os principais desafios enfrentados incluem a necessidade de manter o troco disponível e realizar cálculos durante o trajeto. A ausência de dados móveis no veículo é um obstáculo recorrente.

Ao adotar um sistema digital, os motoristas ganham em tranquilidade, segurança e praticidade, reduzindo a carga cognitiva durante o trabalho e promovendo um atendimento mais fluido ao passageiro.

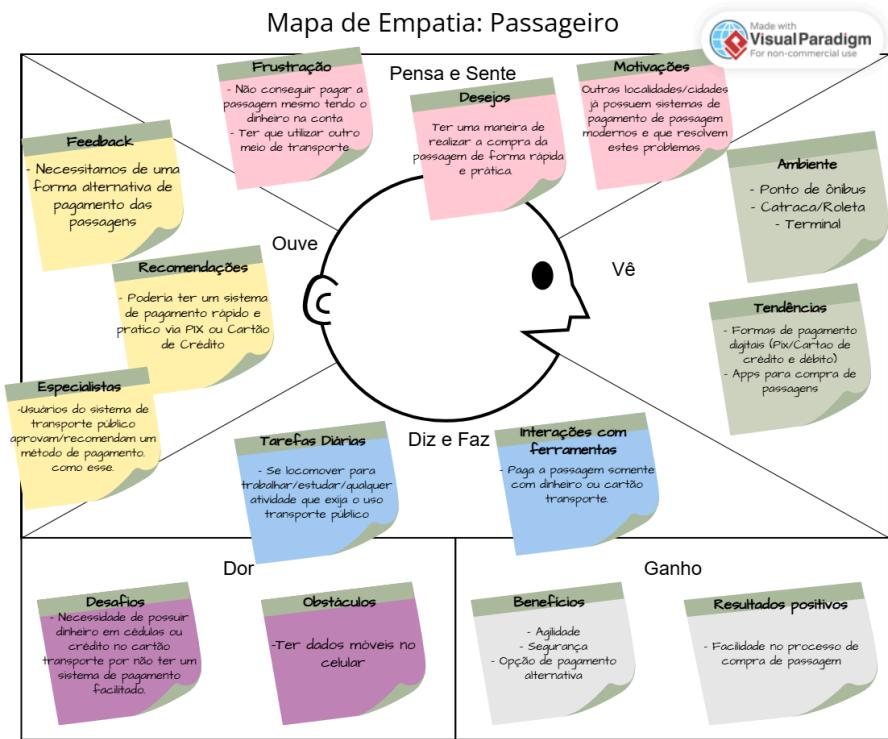


Perfil: Passageiro

O passageiro relata frustração ao não conseguir pagar a passagem, mesmo possuindo saldo em conta, sendo forçado a usar outro meio de transporte. Seu desejo é contar com uma forma rápida e prática de comprar passagens, dispensando o uso de dinheiro físico. O ambiente em que se movimenta inclui pontos de ônibus, roletas e terminais.

Ouvimos passageiros expressando claramente a necessidade de um sistema alternativo de pagamento. Eles recomendam soluções práticas, como PIX e cartões de crédito, e reconhecem que a experiência em outras cidades serve como referência. Suas tarefas diárias incluem deslocar-se para trabalho, estudo ou outras atividades, e o pagamento é feito atualmente apenas em dinheiro ou cartão de transporte. A falta de dados móveis no celular é um obstáculo importante para a adoção dessas soluções digitais.

Com a nova solução, o passageiro ganha agilidade, segurança e praticidade, tendo à disposição métodos de pagamento que acompanham as tendências atuais. O resultado esperado é a melhoria da experiência de compra e maior inclusão de usuários no sistema de transporte público.

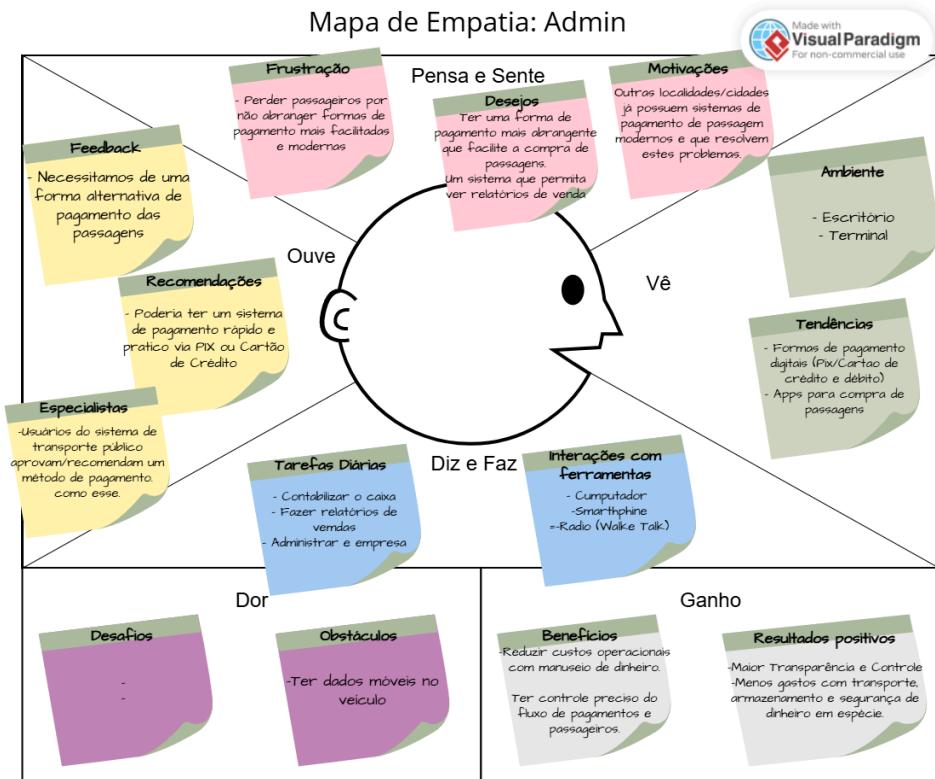


Perfil: Administrador

O administrador manifesta frustração com a perda de passageiros causada pela ausência de métodos modernos de pagamento. Seu principal desejo é dispor de um sistema abrangente, que facilite a compra de passagens, e permita o controle por meio de relatórios detalhados. Seu ambiente de atuação é predominantemente administrativo, composto por escritórios e terminais.

O feedback do administrador também enfatiza a necessidade de modernização nos métodos de pagamento. Recomendam sistemas rápidos e práticos via PIX ou cartões, e reconhecem que usuários do transporte aprovam essas soluções. Suas tarefas diárias envolvem controle financeiro, geração de relatórios e administração da operação. O principal obstáculo técnico identificado é a indisponibilidade de conexão de dados móveis em campo.

A implementação de um sistema digital proporciona ao administrador maior controle, transparência e redução de custos operacionais com dinheiro físico, além de facilitar a auditoria e o planejamento estratégico da empresa.



2.2.3 Desenvolvedores do Sistema

No desenvolvimento desse projeto a equipe foi dividida tentando alinhar as afinidades e habilidades de cada um para o eficiente desenvolvimento do projeto. Foi definido que os três membros da equipe de desenvolvimento Alan Matias, Gabriel Rigoni e Vitor Marvulle ficariam responsáveis por toda a alimentação da documentação.

O desenvolvedor Alan Matias ficou responsável por aplicar as metodologias ágeis, realizar o planejamento e divisão das tarefas para a equipe e pela modelagem e implementação do banco de dados.

Responsável Front-End ficou o desenvolvedor Vitor Marvulle, encarregado de planejar as cores, design, artes, logo e planejamento UX da aplicação web visando entregar beleza, responsividade e qualidade.

No Back-End responsável por integrar as API's , criar as principais funções, métodos, garantir a que a lógica da programação está correta e funcional, o responsável será o desenvolvedor Gabriel Rigoni.

2.3 Regras de Negócio

As regras de negócio são essenciais para garantir que o sistema de pagamento via PIX para o transporte público de Praia Grande funcione de maneira eficiente e eficaz, atendendo tanto aos requisitos operacionais quanto às expectativas dos usuários e das partes envolvidas.

Validação de Transações

- O sistema deve validar a transação em tempo real com a instituição financeira do usuário via PIX para garantir que o pagamento foi concluído antes de permitir o embarque do passageiro no transporte público.
- O motorista não poderá permitir embarque sem que a transação tenha sido concluída com sucesso.

Limites de Pagamento

- O sistema de pagamento via PIX será capaz de processar transações de qualquer valor dentro do limite do transporte público municipal. Caso o valor da passagem exceda o limite aceito pelo sistema, a transação será rejeitada, e o usuário será notificado para tentar outro método de pagamento.

Segurança das Transações

- As transações realizadas pelo sistema devem ser protegidas por criptografia de ponta a ponta para garantir que os dados financeiros e pessoais dos usuários estejam seguros.

Gestão de Erros e Tolerância a Falhas

- Em caso de falha na transação, como erro no processamento do pagamento ou problemas de conexão com a instituição financeira, o sistema deve exibir mensagens claras e orientações para o usuário sobre como proceder, oferecendo uma segunda tentativa.

- O sistema deverá ser tolerante a falhas, implementando estratégias como o registro de falhas temporárias e a recuperação automática de dados em caso de erros inesperados.

Armazenamento de Informações

- O sistema deverá armazenar informações mínimas necessárias para a realização do pagamento, como histórico de transações e identificação do usuário (motorista/admin). As transações serão registradas em um banco de dados relacional, garantindo a integridade e segurança dos dados.

Autenticação e Autorização de Pagamentos

- Usuários deverão fornecer um email válido para envio do bilhete.
- Somente usuários autenticados e com permissão (motorista/admin) terão acesso à plataforma “Para Empresas”.
- O sistema deverá permitir que motoristas verifiquem se o pagamento foi realizado antes de liberar o acesso ao veículo.

Feedback ao Usuário

- O sistema deverá fornecer feedback imediato ao usuário, indicando se o pagamento foi realizado com sucesso ou se ocorreu algum erro. Em caso de erro, instruções claras sobre como resolver a situação devem ser fornecidas.
- O usuário receberá o bilhete no email após a transação ser confirmada.

3 REQUISITOS DO SISTEMA

Este capítulo tem como objetivo descrever os requisitos do sistema. Os requisitos de sistema são especificações detalhadas sobre as funcionalidades, características e condições que um software deve atender para ser eficaz e atender às expectativas dos usuários e stakeholders. Eles são a base para o desenvolvimento, garantindo que o produto final esteja alinhado com as necessidades do cliente e funcione corretamente no ambiente desejado. Os requisitos podem ser divididos em requisitos funcionais, que descrevem o comportamento e as funcionalidades do sistema, e requisitos não funcionais, que detalham aspectos como desempenho, segurança e usabilidade.

3.1 Requisitos Funcionais (RF)

Descreve os requisitos funcionais que especificam ações que um sistema deve ser capaz de executar, ou seja, as funções do sistema.

A seguir a tabela mostra os requisitos funcionais do sistema com um identificador, uma breve descrição da funcionalidade, os critérios de aceitação - que são os parâmetros que devem ser cumpridos para que a funcionalidade seja considerada pronta - e a prioridade que é definida em ALTA - para requisitos extremamente essenciais para o correto funcionamento do sistema, MÉDIA - para requisitos importantes para a qualidade e segurança do sistema e BAIXA - Requisitos importantes mas que não são vitais para o funcionamento do sistema.

Identificador	Descrição	Critérios de Aceitação	Prioridade
RF-01	O sistema deve permitir ao usuário comprar uma passagem.	O usuário fornece um email válido e faz o pagamento via pix. Se o pagamento for validado pela API de pagamento, o sistema envia o bilhete para o email e disponibiliza para download, se houver algum problema no pagamento o sistema deve exibir uma mensagem de erro	ALTA

Identificador	Descrição	Critérios de Aceitação	Prioridade
RF-02	O sistema deve permitir ao Admin o cadastro de motoristas.	Devem ser preenchidos nome, data de nascimento e CPF. O sistema deve verificar se o cadastro já existe. Se não ele cria um novo registro no banco de dados	ALTA
RF-03	O sistema deve permitir ao Motorista /Admin realizar o login.	O Motorista/Admin deve poder preencher um formulário com código (gerado automaticamente no cadastro) e senha. O sistema deve validar a existência do código, se não houver exibir mensagem de código ou senha inválidos. Quando válidos se o tipo de cadastro for motorista abre a tela motorista, se for admin abre o dashboard.	ALTA
RF-04	O sistema deve permitir o Admin Cadastrar Linha de Ônibus.	Devem ser preenchidos o número da linha e nome. Será feita a verificação se o número da linha já está cadastrado.	ALTA
RF-05	O sistema deve permitir ao admin Editar/Excluir motoristas.	A ação excluir deve ter uma mensagem de confirmação. O sistema deve armazenar em um banco os dados do motorista excluído. Editar o motorista exibirá um formulário com os dados atuais que ao preencher o campo atualiza os dados no banco.	MÉDIA
RF-06	O sistema deve permitir ao admin Editar/Excluir linhas.	A ação excluir deve ter uma tela de confirmação. O sistema deve armazenar em um banco os dados da linha excluída. Editar a linha exibirá um formulário com os dados atuais que ao preencher o campo atualiza os dados no banco.	MÉDIA

Identificador	Descrição	Critérios de Aceitação	Prioridade
RF-07	O sistema deve permitir o Admin Ver relatório de vendas.	Clicar no botão Relatório de Vendas exibirá uma lista com todas as vendas registradas tanto pagas no ônibus quanto na plataforma.Com os campos codigo, valor, linha, datahora, motorista	MÉDIA
RF-08	O sistema deve permitir o Admin filtrar os relatórios por motoristas e linhas.	O Admin deve selecionar como quer ver o relatório de vendas	MÉDIA
RF-09	O sistema deve gerar um QR Code para o bilhete.	O sistema deve gerar um registro de bilhete no banco de dados com código da passagem, valor,requisitante,data da compra,data do uso,estado (ativo/usado). Com isso usar o código da passagem para gerar a imagem do QR Code.	ALTA
RF-10	O sistema deve gerar um email com o bilhete da passagem.	O sistema deve verificar se o formato de email é válido, e se for pegar o valor e gerar um email com o arquivo em pdf do bilhete.	ALTA
RF-11	O sistema deve ter a opção de baixar o bilhete.	Ao clicar no botão salvar o sistema deve iniciar o download do bilhete gerado no formato pdf.	ALTA
RF-12	Na tela do administrador, o sistema deve permitir escanear o bilhete ou inserir o código do bilhete.	O sistema deve permitir a abertura da câmera para a leitura do QR Code, verificar se existe se está ativo no sistema para então liberar a passagem uma uma mensagem e som. Ao liberar o sistema deve setar a passagem como usada. Se uma estiver com estado usado deve exibir uma mensagem que essa passagem foi usada.	ALTA
RF-13	Na tela do administrador,	Para poder vender a passagem, o sistema deve estar logado e	ALTA

Identificador	Descrição	Critérios de Aceitação	Prioridade
	deve haver uma opção de selecionar a linha do ônibus.	selecionar a linha que está trabalhando no momento. Esse seleção será feita pelo fiscal para evitar possíveis erros. O sistema não pode permitir vender ou escanear passagem sem estar com a linha selecionada.	

3.2 Requisitos Não Funcionais (RNF)

Descreve os requisitos não funcionais, que especificam restrições sobre os serviços ou funções disponíveis no sistema.

A seguir, a tabela apresenta os requisitos não funcionais do sistema, cada um identificado por um código e acompanhado de uma breve descrição. Também são definidos os critérios de aceitação, que representam os parâmetros mínimos que devem ser atendidos para que o requisito seja considerado satisfatório, além da prioridade atribuída a cada um. A prioridade é classificada em três níveis: alta, para requisitos essenciais relacionados ao desempenho, segurança, usabilidade ou confiabilidade do sistema; média, para requisitos importantes que impactam a experiência do usuário ou a manutenção do sistema, mas que não comprometem seu funcionamento principal; e baixa, para requisitos desejáveis que agregam valor, mas que não são indispensáveis ao funcionamento ou à qualidade geral do sistema.

Identificador	Descrição	Critérios de Aceitação	Prioridade
RNF-01	O sistema deve criptografar dados sensíveis	Campos como identificadores , senha e números de documentos pessoais devem ser criptografados no banco de dados.	ALTA
RNF-02	O sistema deve ter o layout responsivo	O sistema deve ajustar a aplicação web tanto para uso desktop, tablet, smartphone.	ALTA

Identificador	Descrição	Critérios de Aceitação	Prioridade
RNF-03	A interface do usuário deve ser prática e intuitiva	Devem ser dados no máximo 4 cliques para se comprar uma passagem	MÉDIA
RNF-04	O sistema deve estar disponível durante o horários de circulação dos ônibus (4:00 - 2:00 hrs)	O sistema deve estar online durante 99,9% do tempo de circulação dos ônibus. Caso ocorra erros ser restaurado em até 2h	MÉDIA
RNF-05	O sistema deve ser compatível com os principais navegadores	Deve ser compatível com pelo menos 3 das últimas versões dos navegadores mais utilizados.(Chrome, Firefox, Safari, Edge)	MÉDIA
RNF-06	O código deve seguir padrões de boas práticas	O código deve se basear em boas práticas, MVC e Clean Code para garantir fácil entendimento e futuras manutenções e atualizações.	BAIXA
RNF-07	As telas devem ser protegidas e não podem ser acessadas sem o usuário estar autenticado	O usuário não deve poder acessar a tela do motorista ou o dashboard sem estar autenticado no sistema	ALTA
RNF-08	O sistema deve manter um tempo de sessão para Admin e para Motoristas	O tempo de sessão dos motoristas será de no máximo 8h e do Admin no máximo de 1h, sendo encerrada a sessão e levando novamente a página de login.	MÉDIA
RNF-09	O sistema poderá guardar cookies para facilitar preenchimento de dados repetitivos	Os cookies armazenados serão apenas email e código de login.	BAIXA
RNF-10	Chaves de API não devem estar diretamente no código.	As chaves de API usadas na aplicação não poderão estar explícitas no código e não deverão ser versionadas. Devem estar salvas em variáveis de ambiente (.env)	ALTA

3.3 Protótipo

Esta seção consiste no projeto de interface do software. Recurso que detalha e modela os requisitos do sistema, sendo assim, deve estar em conformidade com os requisitos funcionais e não funcionais previstos nas seções anteriores.

Cada tela deve possuir uma descrição detalhada do seu funcionamento, como: objetivo da tela, como chegar até ela, quais regras ela deve seguir (tamanho de campo, tipo de dados que aceita, valor default, campos obrigatórios, etc) e que usuários podem acessar.

A descrição detalhada do protótipo deve registrar informações que possam ser consultadas para facilitar, agilizar e minimizar erros de implementação e execução de testes do software.

Figura 1 - Tela Home



Fluxo: Primeira tela/acesso ao site. Descrição: Permite iniciar o processo de compra de passagens pelos usuários através do botão “compre agora”. Permite acessar o login da empresa para gerenciar motoristas e login de motoristas.

Figura 2 - Tela Escolher Quantidade



Fluxo: Após clicar no botão “compre agora”. Permite voltar para a home através do botão superior esquerdo. Descrição: Permite ao usuário escolher a quantidade de passageiros que deseja, com limite máximo de 5 passageiros.

Figura 3 - Tela Insira email ou telefone



Fluxo: Após clicar no botão “Comprar”. Permite voltar para a página anterior através do botão superior esquerdo. Descrição: Permite ao usuário escolher uma forma de contato para envio do seu bilhete após a etapa final de pagamento.

Figura 4 - Tela Pagamento



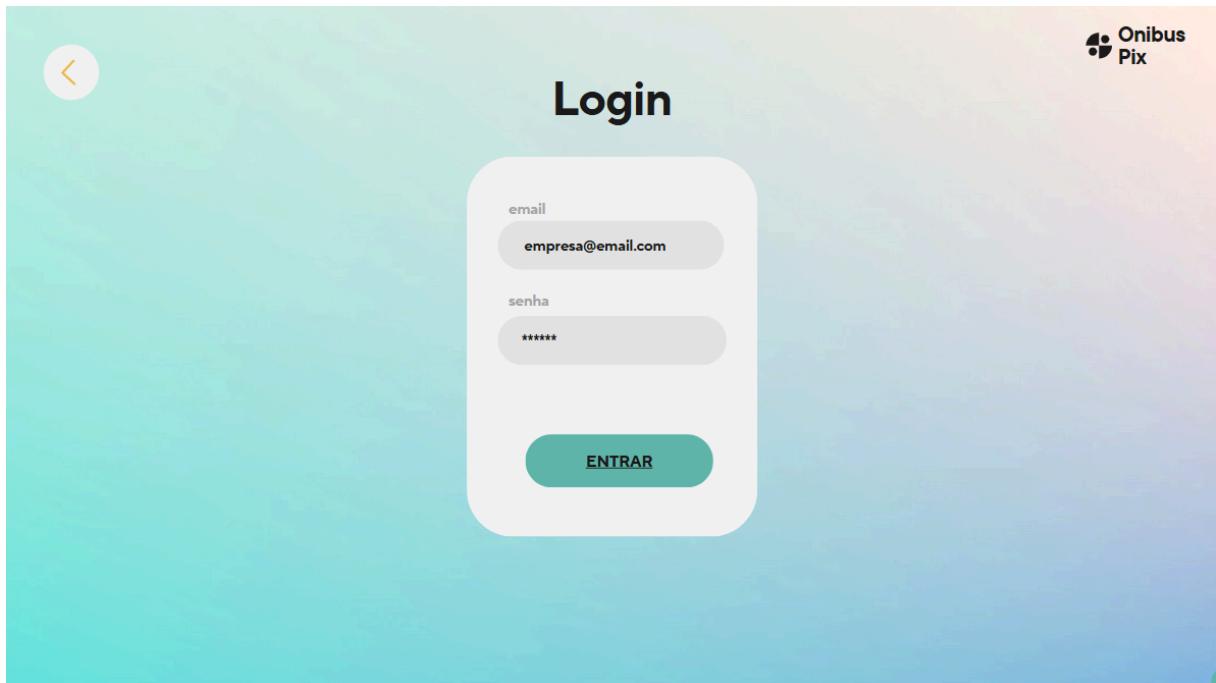
Fluxo: Após clicar no botão “Comprar”. Permite voltar para a página anterior através do botão superior esquerdo. Descrição: Permite ao usuário salvar o código PIX ou acessar via QR Code, para pagamento de seu bilhete em seu banco de preferência.

Figura 5 - Tela Confirmação



Fluxo: Após confirmado o pagamento. Permite voltar para a página Home através do botão superior esquerdo. Descrição: Permite ao usuário salvar a imagem de seu bilhete pago ou mostrar o QR Code ao motorista para validação.

Figura 6 - Tela Login Empresa



Fluxo: Clicando no botão “Para Empresas” na tela Home. Permite voltar para a página Home através do botão superior esquerdo. Descrição: Permite acesso do administrador (empresa) e motoristas.

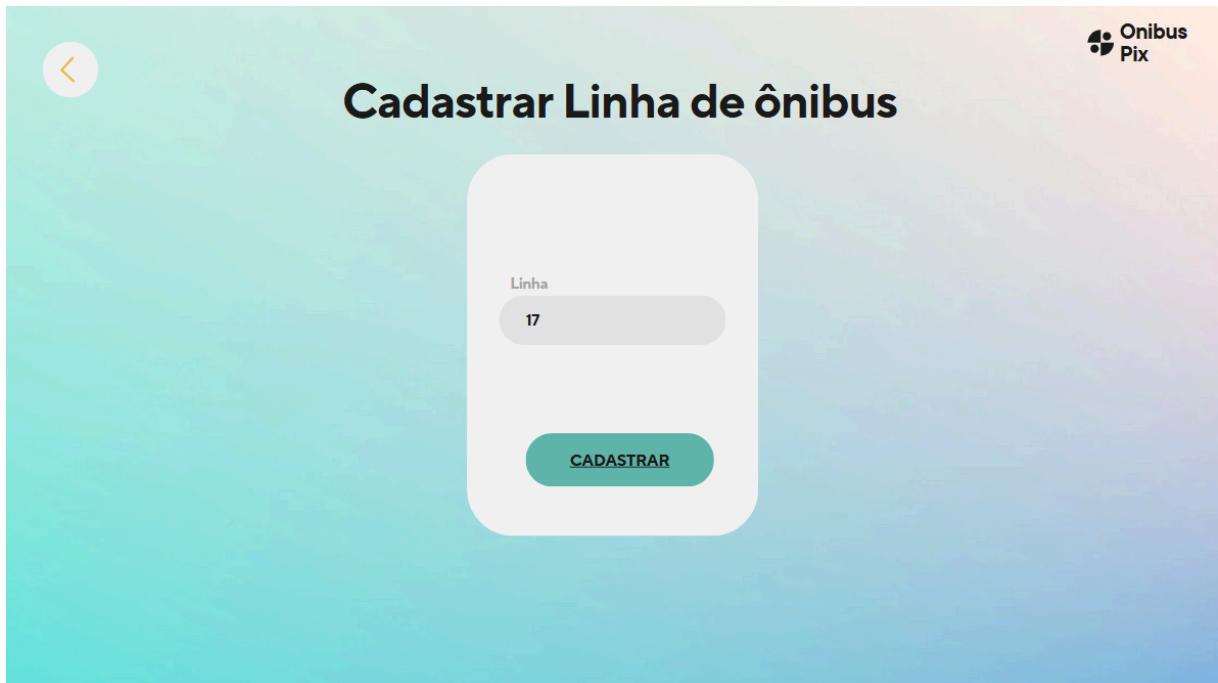
Figura 7 - Tela Dashboard

Nome	Linha	Código	Data	Hora
João da Costa e Silva	17	0123	01/01/2025	08:52:42
João da Costa e Silva	17	9234	01/01/2025	10:45:12
Ana Maria Braga	33	6502	02/01/2025	21:43:08

Fluxo: Clicando no botão “Entrar” na tela de Login, acesso exclusivo da EMPRESA. Permite voltar para a página Home através do botão superior direito. Descrição: Permite visualizar informações dos motoristas e das autenticações de QR Code dos

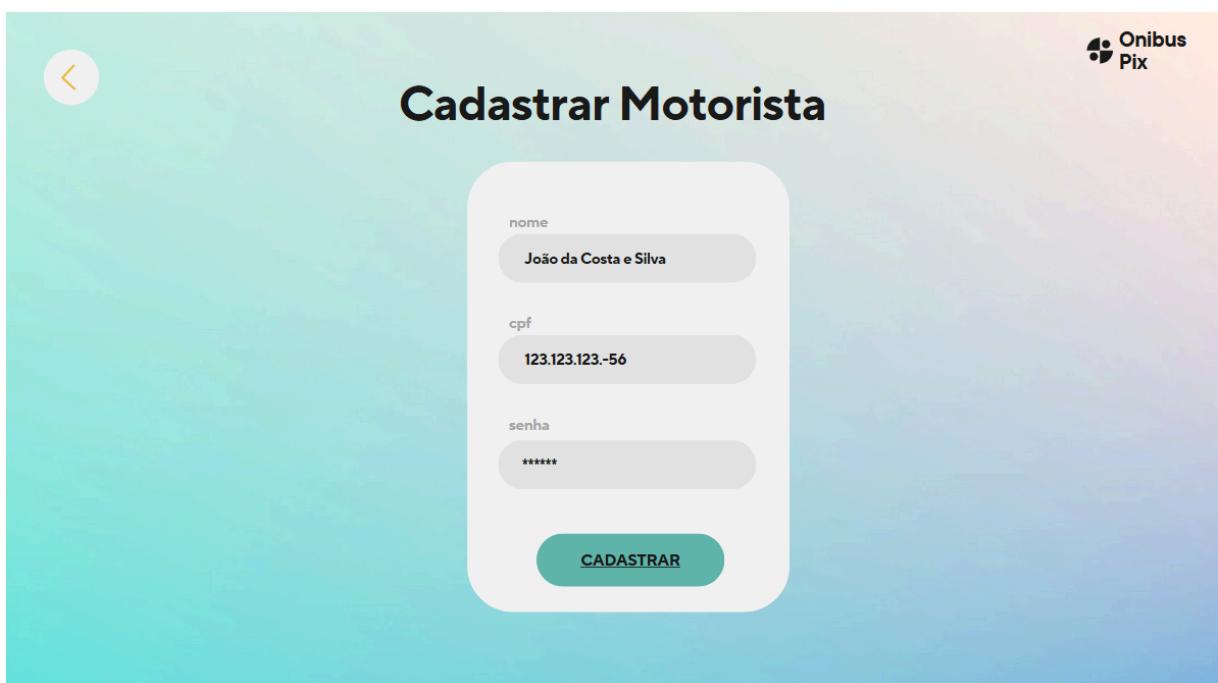
bilhetes, além de cadastrar novos motoristas, cadastrar linha de ônibus e visualizar lista de usuários.

Figura 8 - Tela Cadastrar Linha de ônibus



Fluxo: Clicando no botão “Cadastrar Linha de Ônibus” na tela de Dashboard. Permite voltar para a página Dashboard através do botão superior direito. Descrição: Permite cadastrar novas linhas de ônibus.

Figura 9 - Tela Cadastrar Motorista



Fluxo: Clicando no botão “Cadastrar Motorista” na tela de Dashboard. Permite voltar para a página Dashboard através do botão superior direito. Descrição: Permite cadastrar novos motoristas inserindo ‘nome’, ‘CPF’ e ‘senha’.

Figura 10 - Tela Lista de usuários



Fluxo: Clicando no botão “Lista de Usuários” na tela de Dashboard. Permite voltar para a página Dashboard através do botão superior ‘Dashboard’. Descrição: Permite visualizar os motoristas cadastrados e seus respectivos CPFs e Login de acesso.

Figura 11 - Tela Home (Mobile)



Fluxo: Primeira tela/acesso ao site. Descrição: Permite iniciar o processo de compra de passagens pelos usuários através do botão “compre agora”. Permite acessar o login da empresa para gerenciar motoristas e login de motoristas.

Figura 12 - Tela Escolher Quantidade (Mobile)



Fluxo: Após clicar no botão “compre agora”. Permite voltar para a home através do botão superior esquerdo. Descrição: Permite ao usuário escolher a quantidade de passageiros que deseja, com limite máximo de 5 passageiros.

Figura 13 - Tela Insira email ou Telefone (Mobile)



Fluxo: Após clicar no botão “Comprar”. Permite voltar para a página anterior através do botão superior esquerdo. Descrição: Permite ao usuário escolher uma forma de contato para envio do seu bilhete após a etapa final de pagamento.

Figura 14 - Tela Pagamento (Mobile)



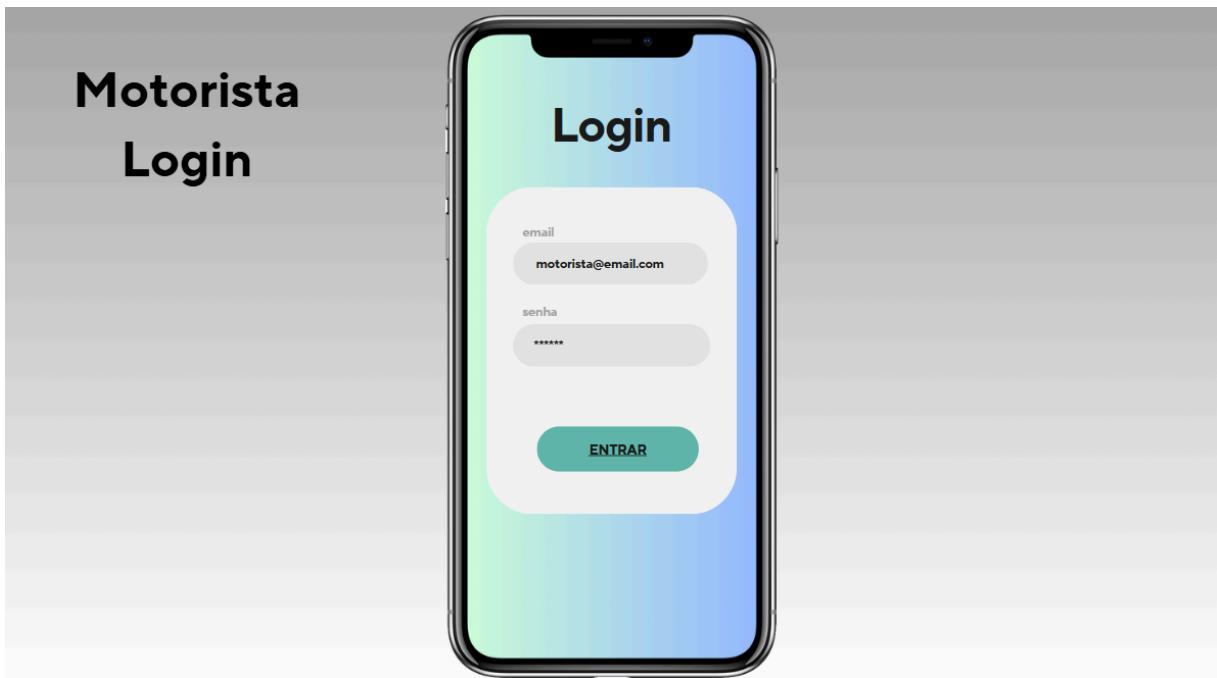
Fluxo: Após clicar no botão “Comprar”. Permite voltar para a página anterior através do botão superior esquerdo. Descrição: Permite ao usuário salvar o código PIX ou acessar via QR Code, para pagamento de seu bilhete em seu banco de preferência.

Figura 15 - Tela Confirmação (Mobile)



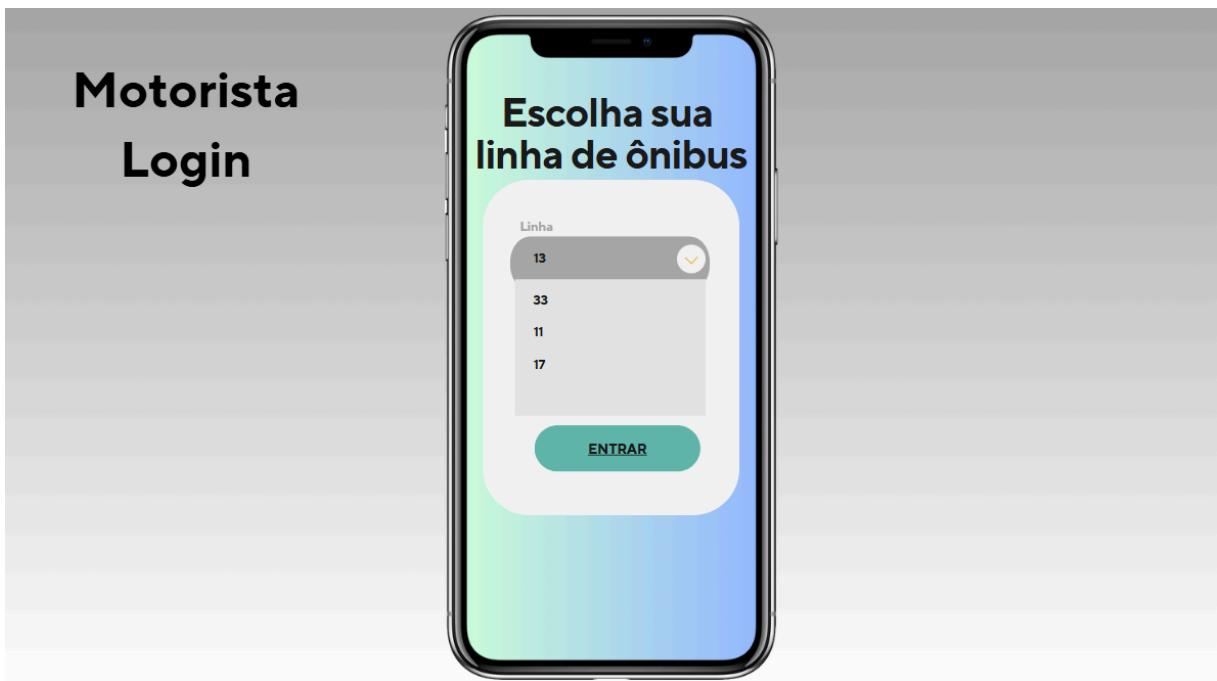
Fluxo: Após confirmado o pagamento. Permite voltar para a página Home através do botão superior esquerdo. Descrição: Permite ao usuário salvar a imagem de seu bilhete pago ou mostrar o QR Code ao motorista para validação.

Figura 16 - Tela Login Motorista (Mobile)



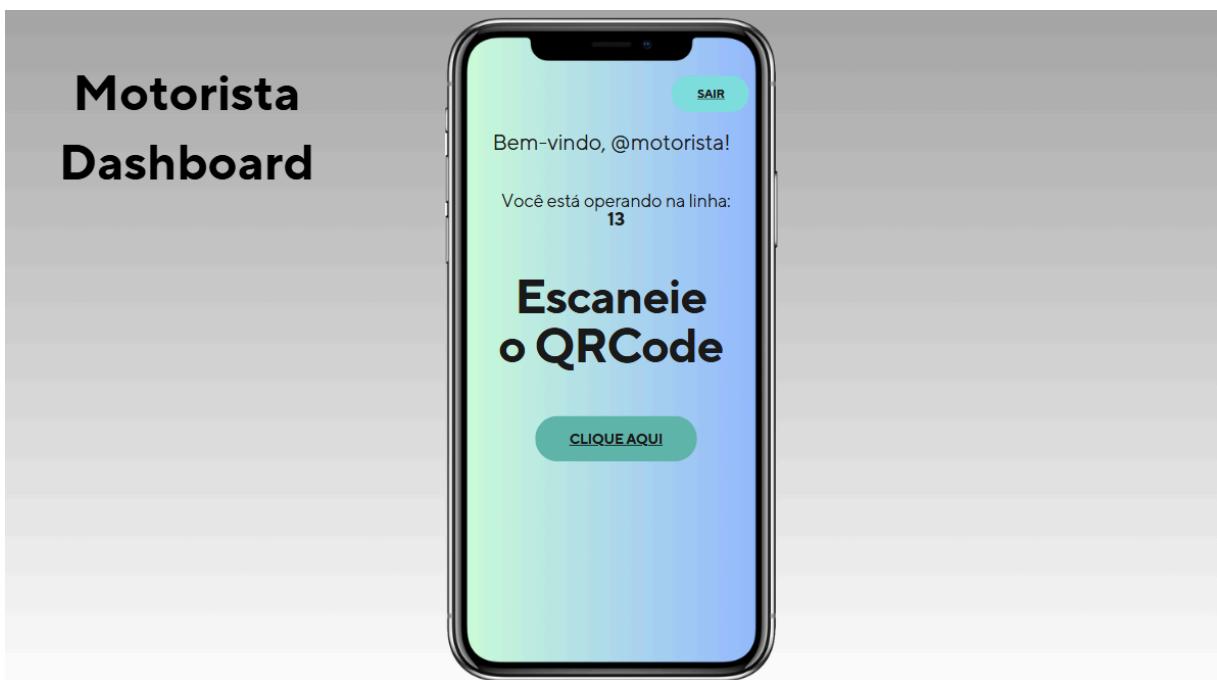
Fluxo: Clicando no botão “Para Empresas” na tela Home. Permite voltar para a página Home através do botão superior esquerdo. Descrição: Permite acesso de motoristas.

Figura 17 - Tela Login Motorista 2 (Mobile)



Fluxo: Após login do motorista autenticado. Descrição: Permite escolher qual linha o motorista vai operar.

Figura 18 - Tela Motorista Dashboard (Mobile)



Fluxo: Após login do motorista, e a linha de operação ser escolhida, permite sair e retornar à tela Home através do botão superior direito. Descrição: Permite escanear o QR Code do passageiro e validar.

Figura 19 - Tela Motorista Confirmação Sucesso (Mobile)



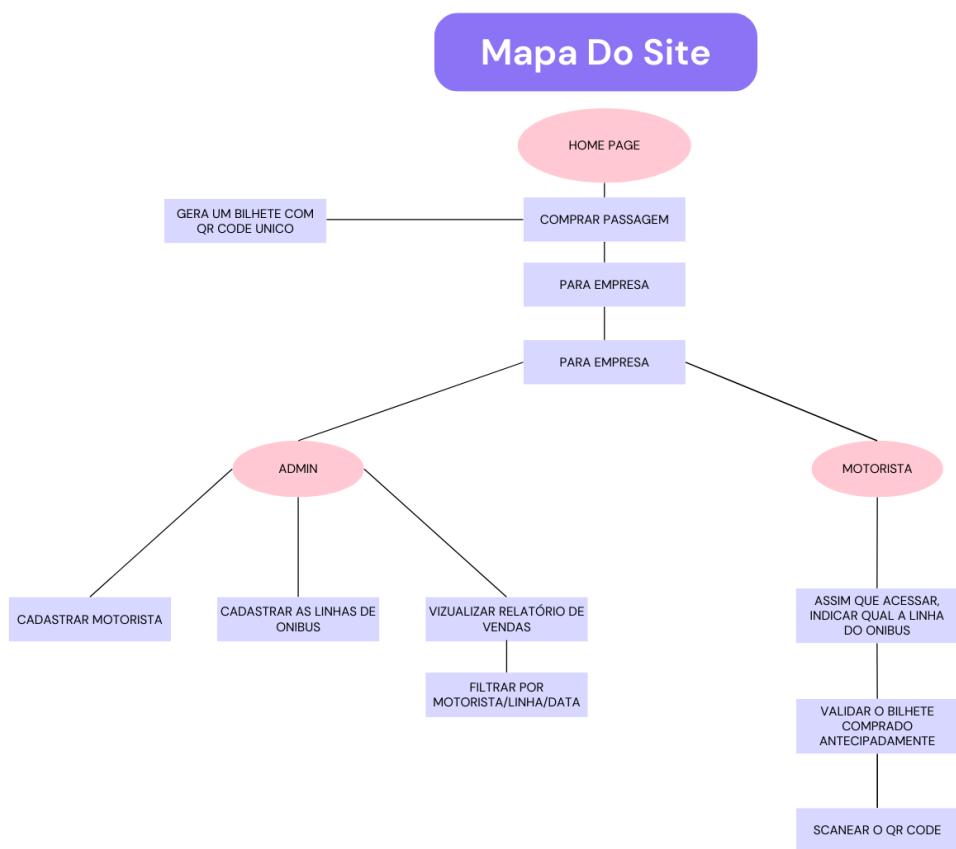
Fluxo: Após clicar no botão “clique aqui” é validado o bilhete com sucesso. Permite retornar a tela Dashboard Motorista através do botão Voltar. Descrição: Validação do bilhete com sucesso.

Figura 20 - Tela Motorista Confirmação Erro(Mobile)



Fluxo: Após clicar no botão “clique aqui” é validado o bilhete com erro. Permite retornar a tela Dashboard Motorista através do botão Voltar. Descrição: Validação do bilhete com erro.

3.3.1. Diagrama de Navegação



4 ARQUITETURA DO SISTEMA

Este capítulo tem como objetivo descrever e justificar de maneira sucinta a arquitetura escolhida para o desenvolvimento do software. Além de conter os subitens a seguir:

4.1 Componentes Principais

Front-End: Aplicação web mobile desenvolvida com HTML, CSS e TailwindCSS, responsáveis pela interface do usuário.

Back-End: Aplicação de compra de passagens via PIX construída em Python com o framework Django, que gerencia a lógica de negócios.

Banco de Dados: SQLite, utilizado para armazenar informações sobre tarefas e usuários.

APIs: API de pagamento do Mercado Pago, que será responsável pelo recebimento e autenticação dos pagamentos da passagem via PIX.

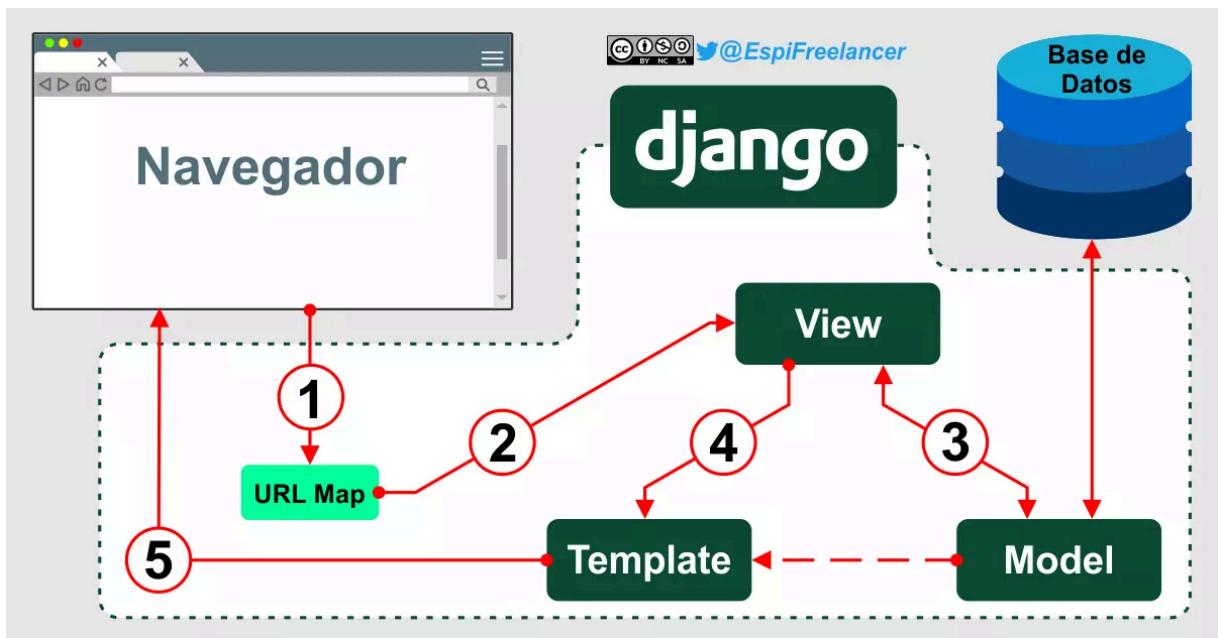
4.2 Diagrama de Arquitetura

A arquitetura usada no desenvolvimento desse projeto é a **MVT** (Model-View-Template) é um padrão de design utilizado no framework Django, que é uma variação do padrão **MVC** (Model-View-Controller). O padrão MVT separa a aplicação em três componentes principais: **Model**, **View** e **Template**, cada um com uma função específica na construção de uma aplicação web.

Model: O Model é responsável pela lógica de dados da aplicação. Ele define a estrutura das tabelas do banco de dados (normalmente representadas por classes Python) e gerencia a interação com o banco de dados.

View: A View no Django é responsável pela lógica de controle da aplicação. Ela recebe as requisições HTTP do usuário, processa as informações (geralmente interagindo com os modelos) e retorna uma resposta, que pode ser uma página HTML, um arquivo JSON, etc.

Template: O Template é responsável pela apresentação, ou seja, ele define a estrutura da interface do usuário (UI) da aplicação. Ele utiliza o sistema de templating do Django para combinar o HTML com os dados passados pela view.



4.3 Segurança

- Utilização de Login, Logout e Session para autenticação e tempo de sessão do usuário.
- Criptografia de senhas usando Hashers do Django como (make_password e check_password)
- Utilização do csrf_token nativo do Django que impede SQLInjection nas entradas de formulário.

4.4 Escalabilidade

O sistema pode ser escalado verticalmente pois será hospedado em nuvem na sua implantação, o que permite a adição de hardware para melhoria de desempenho caso necessário.

4.5 Manutenção

A estrutura do Django com MVT permite que a manutenção e atualização do código seja feita de forma mais organizada, e além disso, colocando em conjunto a orientação a objetos (utilizada nas Models e nas funções), permitindo a facilidade na manutenção.

4.6 Integração

- Integração com serviços externos, como mensagens via Whatsapp, e recebimento do bilhete no e-mail.
- Possibilidade de baixar o bilhete no formato .jpg

4.7 Limitações

- Falta de internet para autenticação das passagens com banco de dados.
- Necessidade de utilização de equipamentos como smartphones e tablets para utilização da aplicação.

5 ARQUITETURA DE DADOS

Este capítulo tem como objetivo descrever os fundamentos e diretrizes para a estruturação e gerenciamento dos dados dentro de um software.

5.1 Modelo Lógico da Base de Dados

O Modelo Lógico é uma representação abstrata dos dados que descreve como os dados estão organizados e como as relações entre eles são definidas, sem considerar detalhes físicos. Ele se concentra na estrutura de dados e nas relações entre eles. Os modelos lógicos podem ser representados através de diagramas, como o Diagrama Entidade Relacionamento (DER) ou Diagrama de Classe.

Diagrama de Classe

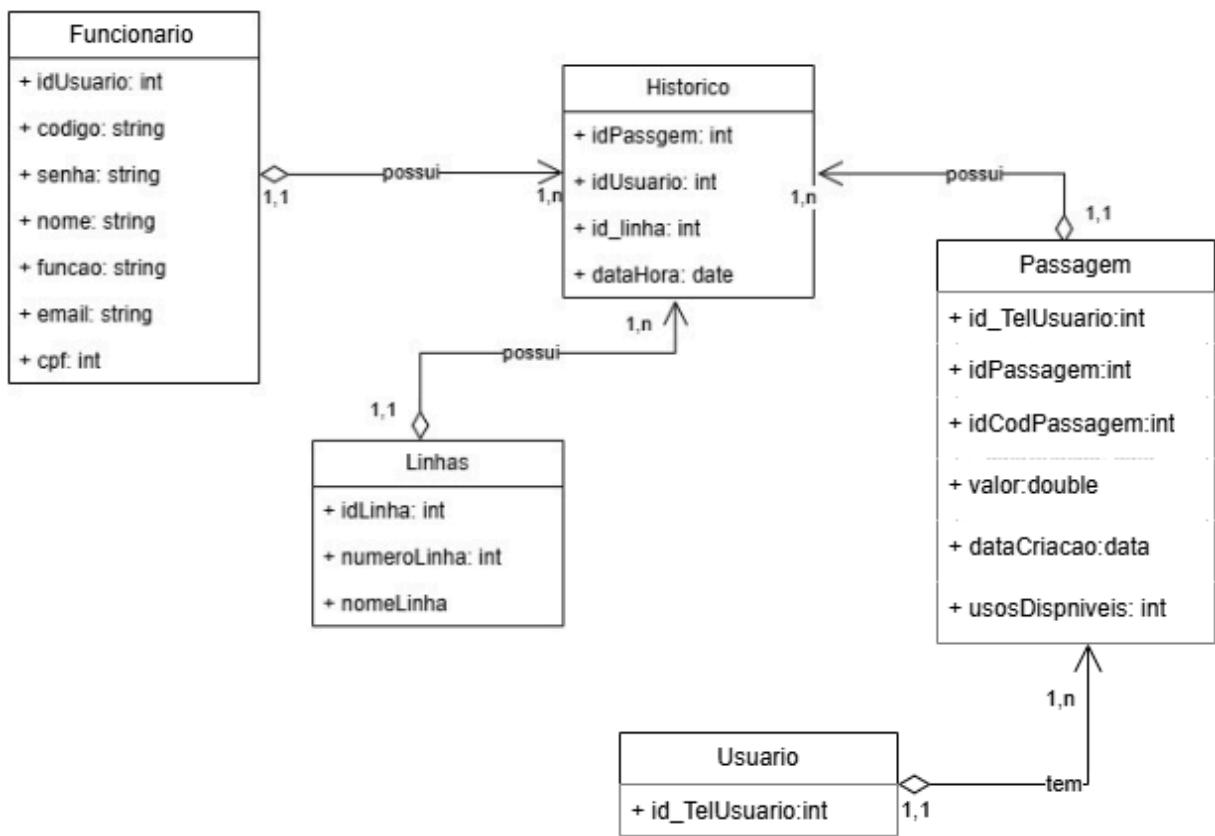
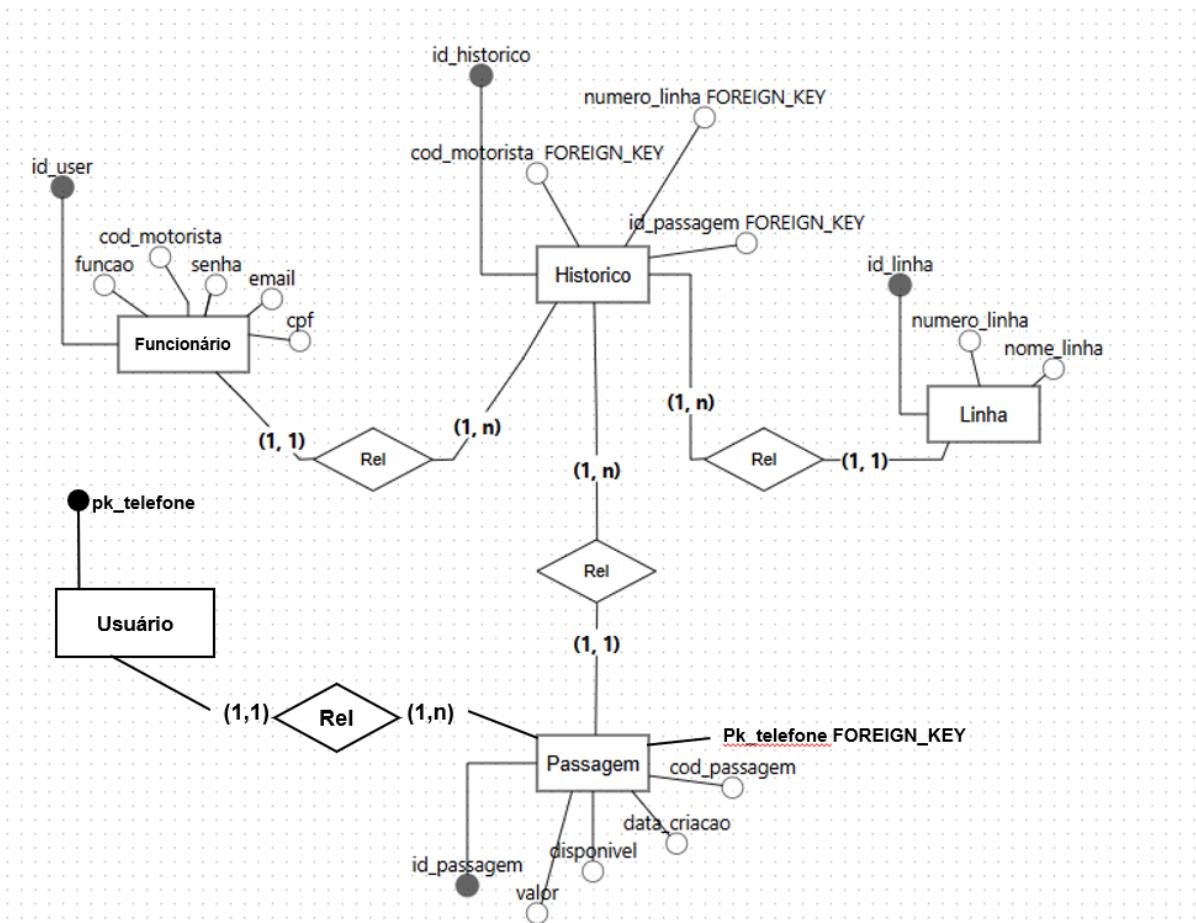


Diagrama Entidade Relacionamento



5.2 Criação Física do Modelo de Dados

A Criação Física do Modelo de Dados descreve, por meio de alguma linguagem, a implementação concreta do modelo lógico em um SGBD específico. Nessa fase, é definido como os dados serão armazenados fisicamente, considerando aspectos como tipos de dados, índices, partições e configuração do hardware.

```

1 ✓ CREATE TABLE Usuario (
2     id_usuario INTEGER PRIMARY KEY AUTOINCREMENT,
3     nome TEXT NOT NULL,
4     cpf TEXT UNIQUE NOT NULL,
5     senha TEXT NOT NULL,
6     email TEXT,
7     funcao TEXT CHECK(funcao IN ('ADM', 'MTR')) NOT NULL
8 );
9
10 ✓ CREATE TABLE Linha (
11     id_linha INTEGER PRIMARY KEY AUTOINCREMENT,
12     numero TEXT UNIQUE NOT NULL,
13     nome TEXT NOT NULL
14 );
15
16 ✓ CREATE TABLE Passagem (
17     id_passagem INTEGER PRIMARY KEY AUTOINCREMENT,
18     valor REAL NOT NULL,
19     data_criacao DATETIME DEFAULT CURRENT_TIMESTAMP,
20     usos_disponiveis INTEGER DEFAULT 1 CHECK(usos_disponiveis BETWEEN 1 AND 10),
21     status TEXT CHECK(status IN ('ativo', 'usado')) DEFAULT 'ativo',
22     codigo TEXT UNIQUE NOT NULL
23 );
24
25 ✓ CREATE TABLE Historico_Validacao (
26     id_historico INTEGER PRIMARY KEY AUTOINCREMENT,
27     datahora DATETIME DEFAULT CURRENT_TIMESTAMP,
28     id_passagem INTEGER NOT NULL,
29     id_motorista INTEGER NOT NULL,
30     id_linha INTEGER NOT NULL,
31     FOREIGN KEY(id_passagem) REFERENCES Passagem(id_passagem),
32     FOREIGN KEY(id_motorista) REFERENCES Usuario(id_usuario),
33     FOREIGN KEY(id_linha) REFERENCES Linha(id_linha)
34 );
35

```

5.3. Dicionário de Dados

O Dicionário de Dados é um repositório que contém informações detalhadas sobre os dados na base de dados, incluindo definições de dados, tipos, restrições e relacionamentos. Ele serve como uma referência para os desenvolvedores e usuários do sistema.

Tabela Usuario

Nome da Coluna	Tipo de Dado	Restrições	Descrição
id_user	INTEGER(5)	NOT NULL, AUTO_INCREMENT, PRIMARY KEY	Identificador único do usuário
cod_motorista	VARCHAR(10)	UNIQUE	Identificador para login dos motoristas
senha	VARCHAR(10)		Senha para acesso dos motoristas e acesso empresa
funcao	VARCHAR(3)	NOT NULL	Identificador se o usuário é a empresa ou motorista (ADM/MTR)
email	VARCHAR(50)		Para acesso dos usuários 'empresa'
cpf	VARCHAR(11)	NOT NULL, UNIQUE	Identificador do usuário no sistema

Tabela Passagem

Nome da Coluna	Tipo de Dado	Restrições	Descrição
id_passagem	INTEGER(5)	NOT NULL, AUTO_INCREMENT, PRIMARY KEY	Identificador único da passagem
cod_passagem	INTEGER(5)	NOT NULL, UNIQUE	Código do bilhete comprado
valor	INTEGER(10)	NOT NULL	Valor das passagens
data_criacao	TIMESTAMP()	NOT NULL	Data em que a passagem foi comprada pelo usuário
disponivel	INTEGER(3)	NOT NULL	Quantidade de passagens disponíveis

Tabela Linha

Nome da Coluna	Tipo de Dado	Restrições	Descrição
id_linha	INTEGER(5)	NOT NULL, AUTO_INCREMENT, PRIMARY KEY	Identificador único da linha
numero_linha	INTEGER(5)	NOT NULL, UNIQUE	Número da linha de ônibus
nome_linha	VARCHAR(100)	NOT NULL	Nome das linhas de ônibus

Tabela Historico

Nome da Coluna	Tipo de Dado	Restrições	Descrição
id_historico	INTEGER(5)	NOT NULL, AUTO_INCREMENT, PRIMARY KEY	Identificador único da transação
id_passagem	INTEGER(5)	NOT NULL, FOREIGN KEY	Chave estrangeira da passagem
cod_motorista	VARCHAR(10)	NOT NULL, FOREIGN KEY	Chave estrangeira do motorista
numero_linha	INTEGER(5)	NOT NULL, FOREIGN KEY	Chave estrangeira do número da linha

6 PROJETO DE SOFTWARE

Este capítulo tem como objetivo detalhar o design interno dos componentes do sistema com base na arquitetura já definida, preparando a equipe para a fase de implementação. Detalhando como as partes do sistema interagem e quais responsabilidades cada componente tem e como a lógica será distribuída.

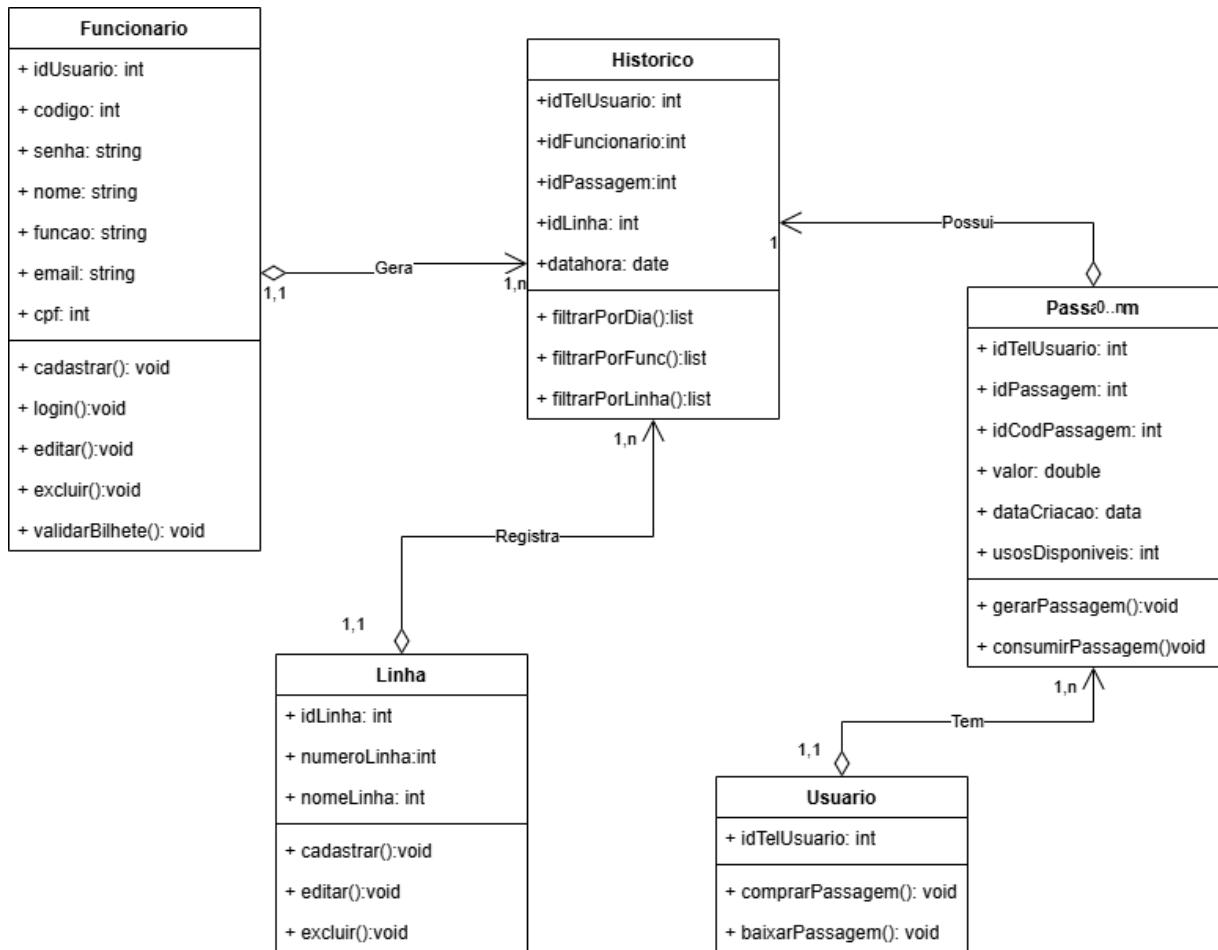
6.1 Design dos Componentes

O projeto Django possui em sua estrutura, dois *apps*. Sendo eles:

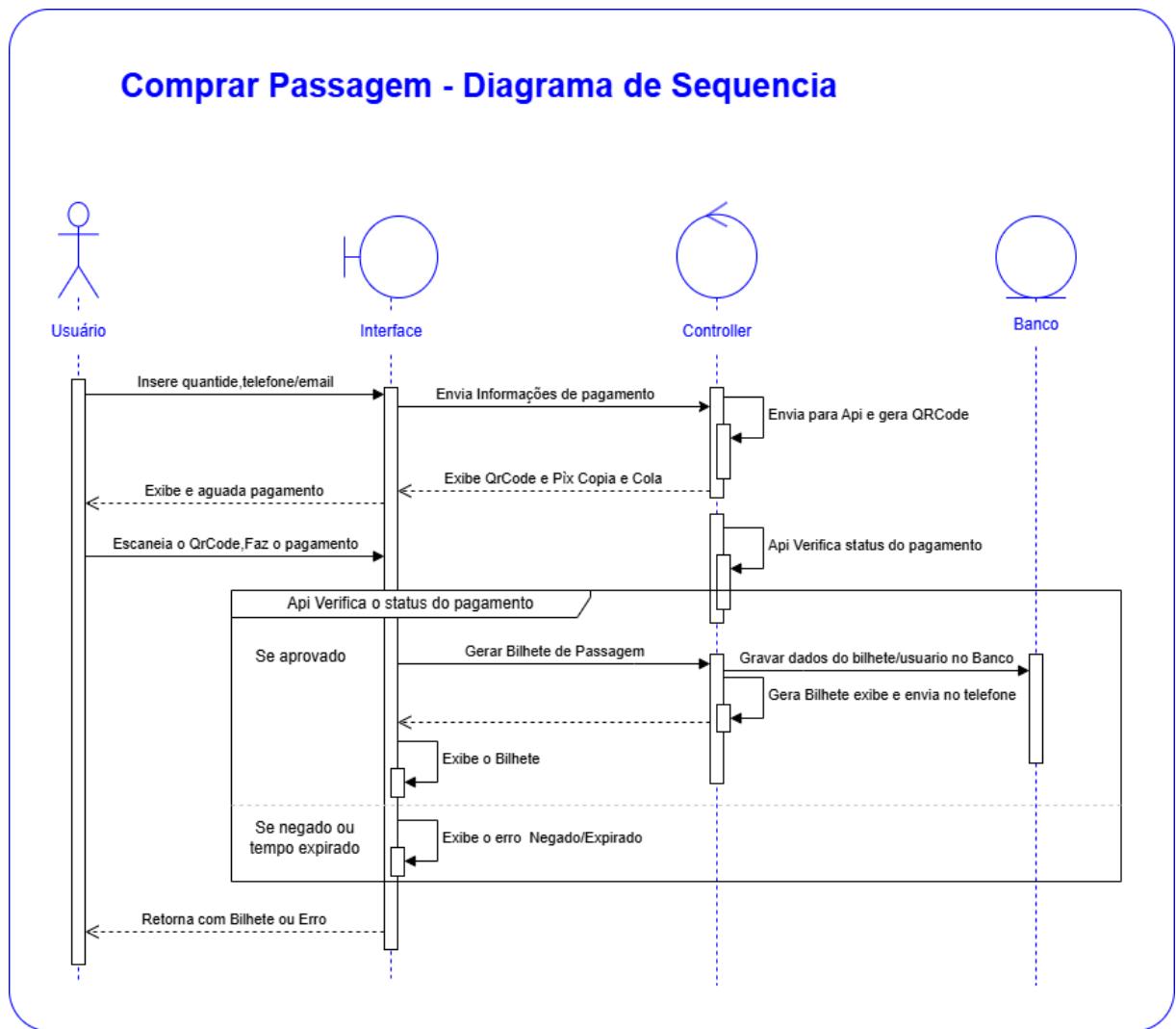
- **app**: para as funcionalidades e fluxos de compra, atribuídos ao usuário (passageiro) e seus respectivos templates, views.py e urls.py
- **app_empresa**: para as funcionalidades e fluxos de gerenciamento da empresa/login dos motoristas e seus respectivos templates, views.py e urls.py.

Os apps possuem dependência e se relacionam através da resposta da API de pagamento, gerando a autenticação e confirmação do pagamento realizado pelo usuário. O **app_empresa** possui as models responsáveis por armazenar os dados de transação entre as validações de bilhetes.

6.2 Diagrama de Classes



6.3 Diagrama de Sequência ou Fluxo de Interações



6.4 Regras de Negócio Detalhadas

- **Validação de Transações:**
- O sistema deve validar a transação PIX em tempo real com a instituição financeira do usuário para garantir que o pagamento foi concluído antes de gerar a passagem/bilhete.
- O motorista não poderá permitir o embarque sem que a passagem/bilhete tenha sido validada com sucesso.

- **Gestão de Erros e Tolerância a Falhas:**

Em caso de falha na transação, o sistema deve exibir mensagens claras de erro na requisição.

- **Autenticação e Autorização de Pagamentos:**

- Usuários deverão fornecer um telefone válido para envio do bilhete via whatsapp.
- Apenas usuários autenticados (motorista/admin) terão acesso à plataforma "Para Empresas".
- O sistema deverá permitir que motoristas verifiquem se o bilhete foi validado, antes de liberar o acesso ao veículo.

- **Feedback ao Usuário:**

- O usuário receberá o bilhete no Whatsapp após a transação ser confirmada.

Essas regras são cruciais porque abordam os aspectos centrais do sistema: a efetivação do pagamento, o controle de acesso e a comunicação clara sobre o status da transação. Sem elas, a funcionalidade principal do sistema e a confiança dos usuários seriam comprometidas.

6.5 Estratégias de Tratamento de Erros

Os possíveis erros ou diversidades que os usuários do sistema pode enfrentar, se baseando nos seguintes fluxos de navegação:

Passageiro - Comprar a passagem:

- No momento do preenchimento de dados, haverá verificações de validação da escrita do e-mail, por exemplo: terminar com “@dominio.com”, e caso esteja inválido, o usuário deverá inserir um e-mail.
- No momento da aquisição, haverá um tempo de expiração de 5 minutos para efetuar o pagamento, caso não seja feito, haverá um erro de “compra não

realizada” devido ao tempo expirado. Também caso o pagamento falhe por algum outro motivo, ele continuará inválido, e cairá no tempo de expiração.

Motorista - Logar no sistema

- No momento do login da tela de motorista, caso o código e senha seja inválido, haverá um erro de “credenciais inválidas no login”.

Motorista - Escanear QR Code do passageiro

- No momento do embarque do passageiro, o mesmo deve apresentar o QR Code do bilhete gerado no website no lugar indicado, caso ocorra algum erro no processo,

Empresa - Cadastrar Motorista e Linha

- No momento nesses cadastros, ambos haverá erro de dados duplicados, por exemplo: Cadastrar motorista com o mesmo CPF, e cadastrar a mesma linha de ônibus mais de uma vez.

7 IMPLEMENTAÇÃO

Este capítulo tem como objetivo apresentar a implementação prática das funcionalidades desenvolvidas para a aplicação web de pagamento de passagens via PIX. A aplicação foi construída utilizando o framework Django, que oferece recursos robustos para desenvolvimento backend, integrando o uso de Programação Orientada a Objetos (POO), banco de dados relacional e um ORM (Object-Relational Mapping) eficiente.

Serão detalhadas a estrutura geral do projeto, os modelos implementados, suas respectivas funcionalidades e interações, destacando aspectos como modularidade, reutilização de código e manutenibilidade do sistema.

7.1 Estrutura Geral do Projeto

A arquitetura do sistema segue o padrão de projetos Django, com separação por aplicações independentes, cada uma responsável por uma parte específica da lógica de negócio. A estrutura do projeto favorece a escalabilidade e facilita a manutenção e evolução do sistema. A seguir, será apresentada a estrutura hierárquica do projeto.

7.2 Modelos

Os modelos são responsáveis pela definição das entidades do sistema e sua persistência no banco de dados. Com o uso do ORM do Django, cada modelo (classe) representa uma tabela no banco relacional, e suas instâncias representam os registros dessas tabelas. As relações entre as entidades são definidas por meio de chaves primárias e estrangeiras, conforme a necessidade do domínio do sistema.

7.2.1 Modelo Usuario

A classe Usuario representa o passageiro, ou seja, aquele que irá comprar a passagem e utiliza-lá no ônibus, a única informação que o identifica como chave primária é o número do celular (adicionado durante o processo de compra do bilhete)

7.2.2 Modelo Funcionario

A classe Funcionario representa tanto os motoristas quanto as empresas, se for motorista, será necessário o campo Código preenchido, porém com o campo email vazio, e o contrário se refere para as empresas.

Campos principais:

- idFuncionario: identificador primário do funcionário;
- codigo: código único de identificação gerado automaticamente;
- senha: senha de acesso, gerada automaticamente se não fornecida;
- nome, funcao, email, cpf: dados pessoais e funcionais.

Métodos relevantes:

- gerarCodigoUnico(): cria um código exclusivo baseado em abreviação da função, nome e CPF;
- gerarSenhaAleatoria(): gera senhas randômicas para acesso ao sistema.

7.2.3 Modelo Linha

O modelo Linha representa as linhas de ônibus cadastradas no sistema, com campos básicos como o identificador e o nome da linha. Ele é utilizado na associação com o histórico de validação de passagens.

Campos principais:

- idLinha: identificador da linha;
- nomeLinha: nome descritivo da linha.

7.2.4 Modelo Passagem

A classe Passagem define os atributos relacionados ao bilhete digital que o usuário adquire. Cada passagem possui um identificador gerado automaticamente, valor, data de criação e o número de usos disponíveis.

Campos principais:

- idPassagem: identificador único da passagem, gerado automaticamente;
- usuario: relacionamento com o usuário proprietário da passagem;
- valor: valor monetário da passagem;
- dataCriacao: data e hora de criação do bilhete;
- usosDisponiveis: quantidade de vezes que a passagem pode ser utilizada.

Método relevante:

- save(): sobrescrito para garantir a geração de um código único (idPassagem) antes da persistência no banco de dados.

7.2.5 Modelo Historico

O modelo Historico armazena o registro de uso das passagens, associando o usuário, o funcionário que realizou a validação, a linha do ônibus e a passagem utilizada, além da data e hora do evento.

Campos principais:

- usuario: referência ao usuário que utilizou a passagem;
- funcionario: funcionário responsável pela validação;
- passagem: bilhete utilizado;
- linha: linha em que a passagem foi validada;
- datahora: data e horário da validação.

Objetivo: Permitir o rastreamento completo das passagens validadas, para fins de auditoria, controle de uso e análise de dados.

7.3 Views

As views no framework Django são responsáveis por receber, processar e responder às requisições HTTP, atuando como uma ponte entre os modelos (dados) e os templates (interface com o usuário). Elas definem a lógica de controle da aplicação e coordenam as ações que ocorrem durante a interação do usuário com o sistema.

No projeto desenvolvido, as views foram implementadas para conduzir o fluxo de compra de passagens, desde a seleção da quantidade até a geração do QR Code do bilhete digital, passando pela integração com o sistema de pagamentos via PIX.

7.3.1 View Home

Responsável por renderizar a página inicial do sistema, esta view fornece os primeiros direcionamentos da aplicação, como o início da compra ou o redirecionamento para a área administrativa de empresas

7.3.2 View Quantidade

Esta view apresenta a interface onde o usuário informa a quantidade de passagens desejada. Os dados são recuperados através da requisição e encaminhados para a próxima etapa do fluxo

7.3.3 View Formulario

Nesta etapa, o sistema recebe a quantidade de passagens selecionada e exibe um formulário para o preenchimento do telefone do usuário. Essa informação é necessária para associar o pagamento e posterior geração da passagem.

7.3.4 View Pagamento

A view pagamento é responsável por processar os dados do formulário, gerar o valor total da compra, integrar com a API do Mercado Pago e criar o QR Code correspondente ao pagamento via PIX. Caso o pagamento seja aprovado, o usuário é redirecionado para a página de visualização do bilhete. Essa view utiliza a biblioteca qrcode para gerar a imagem do código QR a partir do payload de pagamento, retornando essa informação codificada em base64 para ser renderizada no front-end.

7.3.5 View check_payment_status

Essa view é chamada periodicamente via JavaScript para verificar o status do pagamento. Quando um pagamento é aprovado, uma instância de Passagem é criada e vinculada ao número de telefone do usuário, armazenando a quantidade de usos adquirida.

7.3.6 View Bilhete

Após a confirmação do pagamento, esta view exibe ao usuário seu bilhete digital com os dados da transação, como o código do bilhete, valor pago, quantidade de usos e a hora/data da criação. Um novo QR Code é gerado com base no ID da passagem, permitindo sua leitura posterior em dispositivos de validação.

7.3.7 View Login

A view login permite que motoristas e administradores se autentiquem no sistema. Dependendo do tipo de login (código ou email), o usuário é redirecionado para a página correspondente após uma autenticação bem-sucedida. Caso contrário, uma mensagem de erro é exibida.

7.3.8 View Dashboard

A view dashboard é acessada por administradores para visualizar o histórico de transações realizadas, como a criação e validação de bilhetes. O histórico é obtido utilizando a função `select_related` para otimizar as consultas aos modelos relacionados.

7.3.9 View add_linha e editar_linha

As views `add_linha` e `editar_linha` permitem que o administrador adicione novas linhas ao sistema ou edite as existentes. Ambas as views utilizam formulários baseados no modelo Linha para garantir a integridade dos dados.

7.3.10 View excluir_linha

A view `excluir_linha` permite ao administrador excluir uma linha existente do sistema. Ela garante que a linha selecionada seja removida de forma segura.

7.3.11 View add_motorista e lista_motorista

A view `add_motorista` é responsável por registrar novos motoristas no sistema, enquanto a view `lista_motorista` exibe a lista de motoristas cadastrados. Ambos os formulários são baseados no modelo Funcionário e permitem a visualização e o cadastro de motoristas.

7.3.12 View linha_motorista e home_motorista

As views linha_motorista e home_motorista são utilizadas por motoristas para selecionar uma linha de ônibus e visualizar informações relacionadas à sua jornada. A linha_motorista permite ao motorista escolher a linha em que ele está operando, enquanto a home_motorista apresenta dados do motorista e da linha escolhida.

7.3.13 View validar_bilhete

A view validar_bilhete recebe uma requisição POST contendo o ID de um bilhete e valida se ele possui usos disponíveis. Caso o bilhete seja válido, ele é registrado no histórico de validações.

7.4 Templates

Os templates são responsáveis pela camada de apresentação da aplicação, permitindo que as páginas HTML sejam dinamicamente renderizadas com dados provenientes das views. Utilizando a linguagem de templates do Django, elementos como variáveis, estruturas condicionais e de repetição são incorporados por meio de tags específicas, como { % block % }, { % for % } e {{ variavel }}. A seguir, detalha-se a função de cada template principal utilizado no projeto.

7.4.1 Template base_page e mensagem.html

Responsável por renderizar as divs que se permanecem iguais durante o desenrolar da compra de passagem, a base_page sempre mostra os mesmos header e footers, já a mensagem (juntamente com seu Script), fica sempre disponível na página para ser chamada caso desejamos adicionar uma mensagem de notificação para o usuário.

7.4.2 Template home

O template home.html representa a página inicial do sistema, oferecendo uma interface introdutória para os usuários. Ele utiliza a biblioteca de animações AOS (Animate On Scroll) para proporcionar efeitos visuais ao navegar. A estrutura divide a tela em dois blocos: o lado esquerdo apresenta a marca e uma chamada à ação principal, com botão para iniciar a compra; o lado direito exibe uma imagem ilustrativa e acesso ao painel de empresas. O uso do `{% load static %}` e dos caminhos estáticos permite a inclusão de recursos visuais e folhas de estilo.

7.4.3 Template quantidade

O template quantidade.html exibe a interface para o usuário selecionar a quantidade de passagens a serem adquiridas. O layout conta com botões para aumentar ou diminuir o número de bilhetes, limitado entre um e quatro. A interação é manipulada por JavaScript, que atualiza o valor exibido e controla o envio do formulário para a próxima etapa do fluxo de compra.

7.4.4 Template formulario

O template formulario.html apresenta um campo de entrada para que o usuário informe seu número de telefone celular, necessário para prosseguir com o processo de pagamento. O formulário utiliza validação de formato numérico e um botão para envio das informações. O uso da tag `{% csrf_token %}` garante a segurança contra ataques CSRF.

7.4.5 Template pagamento

O template formulario.html apresenta um campo de entrada para que o usuário informe seu número de telefone celular, necessário para prosseguir com o processo de pagamento. O formulário utiliza validação de formato numérico e um botão para envio das informações. O uso da tag `{% csrf_token %}` garante a segurança contra ataques CSRF.

7.4.6 Template formulario

O template bilhete.html é exibido após a confirmação de pagamento. Ele mostra um bilhete gerado com QR Code, número de identificação e data/hora da emissão. Um botão permite ao usuário salvar uma imagem do bilhete localmente, funcionalidade viabilizada com o uso da biblioteca externa html2canvas. A estilização do bilhete busca simular um tíquete real, com foco na clareza visual e facilidade de identificação.

7.4.7 Template login

O template login.html é utilizado para autenticação de usuários, apresentando um formulário de login com campos dinâmicos gerados por meio do objeto form. O formulário oferece opções de seleção por radio buttons, entrada de identificador e senha, além de proteção contra CSRF com a tag `{% csrf_token %}`. A organização visual é feita com blocos centralizados e estilização personalizada, visando facilitar o acesso ao sistema por diferentes perfis de usuários.

7.4.7 Template login

O template login.html é utilizado para autenticação de usuários, apresentando um formulário de login com campos dinâmicos gerados por meio do objeto form. O formulário oferece opções de seleção por radio buttons, entrada de identificador e senha, além de proteção contra CSRF com a tag `{% csrf_token %}`. A organização visual é feita com blocos centralizados e estilização personalizada, visando facilitar o acesso ao sistema por diferentes perfis de usuários.

7.4.8 Template Cadastro de Linha

O template add_linha.html é utilizado tanto para o cadastro quanto para a edição de linhas de ônibus no sistema. O formulário é exibido no lado esquerdo da tela e contém campos para o número e o nome da linha. À direita, é apresentada uma lista de linhas já cadastradas, com botões de ação para edição ou exclusão. A renderização condicional (`{% if form.instance.pk %}`) permite reuso do mesmo template para ambas as operações, tornando a interface mais eficiente e reduzindo redundância no código.

7.4.9 Template de Seleção de Linha para Motoristas

O template linha_motorista.html permite ao motorista selecionar a linha de ônibus à qual está vinculado. Para isso, utiliza um campo de formulário com um menu suspenso estilizado que exibe as linhas disponíveis, carregadas dinamicamente com base na queryset do campo linha. A seleção é registrada em um campo oculto e submetida ao servidor, possibilitando o redirecionamento do motorista para sua interface específica.

7.4.10 Template de Lista de Funcionários

O template lista_motorista.html exibe uma lista de motoristas e funcionários cadastrados pela empresa, acessível por meio do painel administrativo. A interface inclui cabeçalhos como nome, código, função e CPF, organizados em uma tabela responsiva. O cabeçalho superior e a navegação lateral permitem transitar entre outras funcionalidades administrativas como cadastro de motorista, cadastro de linha e retorno ao dashboard.

7.4.11 Template de Dashboard

O template dashboard.html apresenta um painel administrativo voltado à empresa operadora. A interface centraliza o histórico de validação das passagens, exibindo dados como o código da passagem, usuário, valor, linha, motorista e data/hora da validação. A navegação superior oferece acesso rápido a funcionalidades administrativas. A tabela é construída para suportar múltiplos registros e possibilitar uma análise gerencial das operações realizadas.

7.4.12 Template de Confirmação de Validação

O template confirmacao.html é responsável por exibir ao usuário o resultado do processo de validação do bilhete no momento do uso. Ele apresenta duas seções distintas: uma para o caso de sucesso e outra para falha na validação.

Na tela de sucesso, é exibida uma mensagem afirmando que o bilhete foi validado com sucesso, acompanhada de informações como o número do bilhete e a quantidade de usos restantes. Já na tela de erro, uma mensagem de falha é

apresentada, informando que não foi possível concluir a validação. Ambas as telas incluem um botão de retorno para facilitar a navegação.

O controle visual das seções é feito dinamicamente via JavaScript, alterando a visibilidade das áreas de sucesso ou erro conforme o resultado da operação. O design visa oferecer uma resposta clara e imediata ao usuário, reforçando a confiabilidade e a usabilidade do sistema.

8 REFERÊNCIAS

VALIDAPIX. *Validação de pagamento instantâneo*. Disponível em:

<https://www.validapix.com.br/#about>. Acesso em: 19 fev. 2025.

ANDRION, Roseli; YUGE, Claudio (ed.). *Pagamentos online por mobilidade urbana são usados por 80 % dos brasileiros*. Canaltech, 07 fev. 2022. Disponível em: <https://canaltech.com.br/negocios/pagamentos-online-por-mobilidade-urbana-sao-usados-por-80-dos-brasileiros-208624/>. Acesso em: 12 mar. 2025.

SILVA, Solange Cristina Ricardo; FREITAS, Henrique Mello Rodrigues de. *O Sistema de Mobile Payment no transporte público na cidade de São Paulo*. In: *Análise da utilização de pagamentos móveis no contexto brasileiro: percepção de usuários e não usuários*. 2015. Disponível em:

https://www.researchgate.net/publication/347674298_Analise_da_utilizacao_de_pagamentos_moveis_no_contexto_brasileiro_percepcao_de_usuarios_e_nao_usuarios. Acesso em: 12 mar. 2025.

