Faculdades Integradas de Taquara - Faccat
Av. Oscar Martins Rangel, 4.500
Taquara, RS, CEP 95600-000

*Sistemas de Informação*

# SISTEMA PARA PAGAMENTO DE BOLETOS ATRAVÉS DE CARTÕES DE CRÉDITO

Guilherme Aruí Appel[1]

Francisco Assis Moreira Do Nascimento[2]

## RESUMO

Este artigo apresenta o desenvolvimento de um software web para pagamento de boletos através de cartões de crédito, chamado Zero Boleto. Essa aplicação busca resolver alguns dos principais problemas dos sistemas para pagamento de boletos através de cartões de crédito, que são as taxas altas e a impossibilidade de utilizar mais de um cartão de crédito para pagar um boleto. Explica também os problemas enfrentados, o processo de análise e desenvolvimento e seu resultado final utilizando a metodologia Scrum Solo para desenvolvimento de um sistema web com integração com a API do gateway de pagamentos da Pagar.me.

**Palavras-chave:** Pagamento. Boleto. Web. Cartão de Crédito.

## ABSTRACT

This article presents the results of a research, which aimed at the development of a web software for payment of bank tickets through credit cards called Zero Boleto. This application seeks to solve some of the main problems of the softwares for payment of bank tickets through credit card, which are the high taxes and the impossibility to use more than one credit card for the bank ticket payment. It also explains the problems faced, the analysis and development process and its final result using the Scrum Solo methodology of a web system with an integration with the API of the Pagar.me payment gateway.

**Keywords:** Payment. Bank Ticket. Web. Credit Card.

[1] Acadêmico do curso de Sistemas de Informação das Faculdades Integradas de Taquara – Faccat/RS. *E-mail*: guilherme.appel@sou.faccat.br.
[2] Professor orientador das Faculdades Integradas de Taquara – Faccat/RS. *E-mail*: assis@faccat.br

# 1 INTRODUÇÃO

Mais da metade dos brasileiros recorre a empréstimos para terminar o mês de maneira positiva e a maioria afirma que o dinheiro geralmente é para impostos e contas recorrentes (BERTÃO, 2021). Neste caso, para que não precisem recorrer a esse tipo de estratégia, caso tenham cartão de crédito, as pessoas podem se beneficiar do mesmo. Podendo, por exemplo, parcelar alguma conta muito alta que veio em um mês, como uma conta de luz e/ou de água, para que não seja necessário comprometer o seu orçamento mensal. Ou até mesmo para poder pagar à vista uma outra conta que custaria juros maiores em caso de atraso ou parcelamento.

De acordo com a pesquisa PEIC[3] realizada em julho de 2021, cerca de 71,4% das famílias brasileiras possuem algum tipo de dívida (CNC, 2021). A ideia do trabalho aqui apresentado é, através de taxas mais baixas e com a opção de parcelamento da dívida, que evite-se da mesma ficar ainda maior.

Este artigo está estruturado em seções, da seguinte forma: a seção 2 apresenta o estado da arte, a qual descreve as ferramentas e tecnologias adotadas. A seção 3 trata acerca da metodologia adotada para o desenvolvimento do projeto. Já na seção 4 é apresentada as principais telas e funcionalidades do sistema. A seção 5 traz para fins de comparação, trabalhos correlatos com o presente projeto. E por fim, na seção 6 elabora-se em torno das considerações finais sobre esse projeto de pesquisa.

# 2 ESTADO DA ARTE

Nesta seção, é apresentado o estado da arte em fintechs, sistemas de pagamento de boletos e uso das tecnologias adotadas.

## 2.1 Fintechs e Sistemas de Pagamento de Boletos

O termo Fintech surgiu a partir da combinação das seguintes palavras nativas do inglês: financial[4] e technology[5] (DINIZ, 2020). Porém, o termo não se atém

[3] Pesquisa de Endividamento e Inadimplência do Consumidor
[4] Financeiro
[5] Tecnologia

apenas à questão de “tecnologia financeira”. As fintechs podem entregar soluções bastante diversificadas, como cartão de crédito, conta digital, cartão de débito, empréstimos, seguros, entre outros (NUBANK, 2021b).

As fintechs, de maneira geral, são conhecidas no mercado por oferecer soluções financeiras muito mais intuitivas de serem usadas, além de serem menos burocráticas e de também oferecer custos baixíssimos, ou até mesmo inexistentes, para os usuários (NUBANK, 2021b).

Conforme dito por Marques (2018):

> “Há muito se sabe que o conceito de confiança se relaciona diretamente ao mercado financeiro, é através deste conceito que somos capazes de realizar trocas por produtos ou serviços, virtuais ou físicos. Sendo assim, o homem precisou criar uma ferramenta que contabilizasse de maneira simbólica, a quantia que possuía e que poderia trocar o dinheiro. A partir do surgimento da moeda, os meios de troca por onde pudessem ser efetuadas transações de maneira confiável se tornaram cada vez mais exigíveis e fundamentais para a operação eficaz da economia de forma geral. Desta forma, as instituições bancárias, os meios de pagamento e a tecnologia, desempenham papéis importantíssimos nesse contexto, se estabelecendo ainda como fato delicado para que a economia alcance êxito em toda sua complexidade.”

Marques (2018) também diz que os bancos digitais são considerados assim, não apenas por possuírem aplicações que viabilizem transações financeiras, mas também por fornecerem transformação online em serviços feitos tradicionalmente off-line e por agregar valor na vida das pessoas.

De acordo com BCB (2021), no Brasil existem várias categorias de fintechs e dentre elas, encontram-se modelos de fornecimento de crédito, de pagamento, gestão financeira, empréstimo, investimento, financiamento, seguro, negociação de dívidas, câmbio, e multisserviços.

No país, estão autorizadas a operar dois tipos de fintechs de crédito – para intermediação entre credores e devedores por meio de negociações transacionadas através de meios eletrônicos: a Sociedade de Crédito Direto (SCD) e a Sociedade de Empréstimo entre Pessoas (SEP), cujas operações estarão vinculadas ao SCR Sistema de Informações de Créditos (BCB, 2021).

A SEP tem função de intermediar os contratos realizados entre os credores e os tomadores de crédito, enquanto a SCD assume a responsabilidade por realizar operações de crédito, bem como disponibilizar os serviços de análise e cobrança de crédito de terceiros, além de permitir a distribuição de seguro relacionado com as operações concedidas por ela mesma, através de sua plataforma eletrônica, com emissão de moeda eletrônica (BCB, 2021).

Neste trabalho, foi desenvolvido um sistema para realizar o serviço de pagamento parcelado de boletos, utilizando cartão de crédito, que pode ser oferecido por fintechs do ramo financeiro.

## 2.2 Sistemas web voltados para fintechs

Em pesquisa realizada para este projeto, dentre as principais soluções encontradas tem-se Nubank (NUBANK, 2021a) e PicPay (PICPAY, 2021) por exemplo, as quais disponibilizam apenas a versão mobile. Alguns outros disponibilizam internet banking, como Banco do Brasil por exemplo, onde é possível pagar utilizando caixa eletrônico, aplicativo mobile ou o navegador. BB (2021) informa que o pagamento de boleto parcelado está disponível acessando a conta no aplicativo do banco, no autoatendimento pelo computador, ou no caixa eletrônico.

O sistema desenvolvido neste trabalho encontra-se no meio dos dois mundos, entre fintechs de crédito e os bancos tradicionais, pois utiliza cartões de créditos de outras instituições como forma de pagamento, porém fornecendo taxas menores que as das próprias instituições.

## 2.3 Tecnologias Adotadas

No desenvolvimento do sistema de pagamento de boletos foram adotadas as tecnologias mais atuais disponíveis no mercado. A seguir são descritas cada uma delas.

### 2.3.1 Frameworks, bibliotecas e algoritmos

No back-end foi utilizada a linguagem de programação C# em conjunto com o framework ASP.NET Core. O ASP.NET Core é um framework open-source, leve,

rápido, modular e multiplataforma, criado pela comunidade e liderado pela Microsoft e funciona em conjunto com o .NET Core. Permite a criação de serviços RESTful, também conhecidos como APIs Web, mais performáticos que outros web frameworks populares (LOCK, 2021), enquanto o C# é uma linguagem de programação moderna, orientada a objeto e type-safe. O C# permite que os desenvolvedores criem muitos tipos de aplicativos seguros e robustos que são executados no framework .NET (DEJONGHE, 2015). Ainda conforme Dejonghe (2021), os recursos do C# ajudam a criar aplicativos robustos e duráveis.

Esta estrutura ainda conta com um modelo de API REST (Representational State Transfer)[6], que segundo Fielding e Taylor (2000) é um conjunto coordenado de restrições arquitetônicas que tentam minimizar a latência e comunicação de rede enquanto, ao mesmo tempo, maximizar a independência e escalabilidade das implementações de componentes. REST permite o armazenamento em cache e reutilização de interações, substituibilidade dinâmica de componentes e processamento de ações por intermediários, atendendo assim às necessidades de um sistema hipermídia distribuído em escala de internet.

Os programas clientes usam interfaces de programação de aplicativos (APIs) para se comunicar com serviços da web. O estilo de arquitetura REST é comumente aplicado ao design de APIs para serviços da web modernos. Uma API da Web em conformidade com o estilo de arquitetura REST é uma API REST (MASSE, 2011).

E para tratar da autenticação no servidor, foi utilizado o JSON Web Token (JWT) token de codificação para segurança, baseado em JSON (JavaScript Object Notation) [RFC 7159], que permite que informações de identidade e segurança sejam compartilhadas em domínios de segurança (BLOKDYK, 2018). Um token de segurança geralmente é emitido por um Provedor de Identidade e consumido por uma Parte Confiante que depende de seu conteúdo para identificar o assunto do token para questões relacionadas à finalidades de segurança (JONES et al., 2015).

Entre os benefícios de utilizar JSON Web Tokens, estão: mais compactos, mais comum em múltiplas linguagens de programação, mais seguro e fácil de processar. Pois são menos verbosos que XML (eXtensible Markup Language), além de utilizarem par de chaves na forma de um certificado X.509, assinado simetricamente utilizando o algoritmo HMAC (Hash-based Message Authentication Code) e serem utilizados na escala da internet (Auth0, 2021a).

[6] Transferência de Estado Representativo

Ainda na questão de segurança, foi utilizado também a biblioteca Argon2, a qual cria um hash de senha, ou seja, um processo unilateral de proteger a senha de texto simples criando uma string de bits de tamanho fixo chamada hash usando a função hash criptográfica (PENARD e WERKHOVEN, 2008). Funções de hash criptográficas projetadas para serem uma função unilateral, ou seja, uma função que é inviável de inverter (AUTH0, 2021b). Álvarez et. al., 2018, afirma que:

> "As funções de derivação de chave são empregadas para obter uma ou mais chaves de um segredo mestre. Isso é especialmente útil no caso de senhas de usuário, que podem ter comprimento arbitrário e não são adequadas para serem usadas diretamente como chaves de criptografia de tamanho fixo, portanto, deve haver um processo para converter senhas em chaves secretas. Esse processo é executado por funções de derivação de chave baseadas em senha (PBKDFs). Os PBKDFs também são chamados de funções de hashing de senha e são comumente empregados na autenticação do usuário, eles têm certas vantagens sobre outros métodos de processamento de senha: eles são capazes de aceitar um salt, evitando ataques a tabelas pré-calculadas; elas são funções unilaterais (assim como funções hash criptográficas comuns), portanto, o banco de dados de senha com hash não pode ser revertido se for roubado; e geralmente podem ser parametrizados em termos de custo temporal e de memória para evitar ataques baseados em hardware massivamente paralelo, como unidades de processamento gráfico de uso geral (GPGPU) ou hardware personalizado."

Argon2 é um algoritmo de hash criptográfico, mais recomendado para hash de senha, sendo uma família de esquemas de hash de senha que foi declarada vencedora do Password Hashing Competition 2015, a qual visava selecionar um ou mais esquemas de hash de senha, que correspondesse aos requerimentos modernos, determinando assim o melhor (WETZELS, 2015).

E para garantir que toda esta aplicação de back-end seja consistente, foram aplicados testes unitários utilizando a biblioteca xUnit.net, a qual é uma ferramenta de teste unitários gratuita, de código aberto e focada na comunidade de desenvolvedores que atuam com o .NET Framework (XUNIT, 2022).

Para a aplicação front-end foi trabalhado com o ReactJS, que é uma biblioteca JavaScript, utilizada para desenvolver componentes de interface de usuário modulares e reutilizáveis (BANKS e PORCELLO, 2017). Ele é usado para lidar com a camada de visualização em aplicativos de página única e

desenvolvimento de aplicativos móveis. É mantido pelo Facebook, Instagram e uma comunidade de desenvolvedores e corporações. Se esforça para fornecer velocidade, simplicidade e escalabilidade. Algumas de suas características mais notáveis são JSX (JavaScript XML), componentes com estado, modelo de objeto de documento virtual (Khuat, 2018).

### 2.3.2 Infraestrutura e ferramentas de desenvolvimento

Para o desenvolvimento de toda a aplicação, foi utilizado o Visual Studio, uma ferramenta que permite que você conclua todo o ciclo de desenvolvimento em um único lugar. Ou seja, você pode editar, depurar, testar, controlar a versão e implantar na nuvem (GARCIA e ROJAS, 2022).

Em relação a hospedagem da aplicação, foi utilizado o Heroku, o qual trata-se de uma PaaS (Platform as a Service)[7] que possibilita a configuração, hospedagem, homologação e publicação de projetos virtuais na nuvem. Entre outras funções, ele atua como um facilitador no trabalho dos desenvolvedores na questão que corresponde a configuração da infraestrutura para a publicação, ou seja, a implantação das aplicações (MIDDLETON e SCHNEEMAN, 2013).

No banco de dados, utilizou-se o PostgreSQL, um servidor avançado de banco de dados SQL (Standard Query Language), open source e disponível em diversas plataformas (RIGGS et. al., 2017), o qual é integrado no Entity Framework Core, uma versão leve, extensível, de código aberto e cross-platform da popular tecnologia de acesso a dados Entity Framework (MICROSOFT, 2021). EF Core pode servir como um mapeador relacional de objetos (O/RM), que permite que os desenvolvedores .NET trabalhem com um banco de dados usando objetos .NET. Isso elimina a necessidade da maior parte do código de acesso a dados que normalmente precisa ser escrito (MICROSOFT, 2021).

Em um banco de dados relacional, uma coleção de tabelas, todas com nomes únicos, compõem a base de dados, podendo estar relacionada a uma ou mais tabelas, ou seja, conceitos como integridade referencial de dados – que garantem que um dado referenciado em uma tabela esteja presente na tabela que está sendo referenciada – e chaves primárias estão presentes e garantem que um conjunto de

[7] Plataforma como um Serviço

informações possa ser representado de maneira consistente, independente da forma de acesso (BOSCARIOLI, et. al., 2021).

Nesse conjunto, também foi necessário utilizar o NGINX para fazer a conexão da aplicação com o Heroku. O NGINX é um servidor HTTP (HyperText Transfer Protocol) e proxy reverso gratuito, de código aberto e de alto desempenho. É conhecido principalmente por seu alto desempenho, mas também por sua grande estabilidade, seu rico conjunto de recursos, com configuração simples e baixo consumo de recursos (DEJONGHE, 2022).

Para controle de versão de código, foi utilizado o GitHub, um serviço em nuvem responsável por hospedar o sistema controlador de versão Git. Ele permite que os desenvolvedores colaborem e façam alterações em projetos compartilhados, mantendo registros detalhados de seu progresso (TSITOARA, 2019).

# 3 METODOLOGIA DE DESENVOLVIMENTO

Nesta seção, é apresentada a metodologia de desenvolvimento utilizada para o desenvolvimento da aplicação.

## 3.1 Proposta do Sistema

Neste trabalho, o sistema desenvolvido visa criar uma solução que permite utilizar mais de um cartão de crédito, que pode estar vinculado a qualquer instituição financeira e com taxas menores que as disponibilizadas nos bancos tradicionais, ajudando os clientes a quitar suas dívidas sem atraso.

Além disso, o sistema traz mais facilidade para os clientes, através de uma plataforma web, que possibilita o acesso à plataforma tanto através de um computador desktop, quanto através de um tablet ou smartphone.

Nas soluções atuais, é necessário análise de cadastro, a qual depende de aprovação, podendo demorar alguns dias ou até mesmo ter o cadastro negado, o que pode gerar um atraso para o cliente. Segundo Nubank (2022), caso o cadastro seja negado, é necessário esperar 3 meses para fazer uma nova solicitação. PicPay (2021) informa que caso seja feito fora do horário comercial ou em finais de semana e feriados, é necessário aguardar o horário de funcionamento do sistema para a

chave ser cadastrada, que no caso é entre 9h e 18h dos dias úteis da semana. O software desenvolvido neste trabalho disponibiliza fácil cadastro e aprovação imediata.

O sistema desenvolvido inclui uma API REST, a qual é responsável por gerenciar os pagamentos e cadastros de usuários e também conta com uma interface visual que serve tanto para web quanto para mobile (dispositivos móveis).

Ao acessar o sistema de pagamentos, o usuário que possuir cadastro pode acessar com seu login e senha e caso não tenha, pode cadastrar-se na hora e já sair utilizando a aplicação, sem necessidade de alguma avaliação de cadastro. O cadastro e login interagem com uma criptografia de senha, utilizando o algoritmo de criptografia Argon2 para guardar a senha no banco de dados e também para a validação do login na plataforma. Além de utilizar Json Web Token durante a autenticação, o que potencializa ainda mais a segurança durante a fase de autenticação do usuário. Dentro do sistema, ainda podem ser feitos ajustes nos dados de perfil do usuário, e claro, pagamentos e listagem de pagamentos, com informações de status. O pagamento como é gerenciado pela API do Pagar.me, possui antifraude próprio do serviço, o qual garante a segurança da transação. O antifraude é um sistema que melhora a proteção de transações online contra fraudes no cartão de crédito. Por meio de inteligência artificial, a mesma analisa os dados das transações, identificando e impedindo fraudes e golpes (PAGAR.ME, 2022). A plataforma de pagamento de boletos também conta ainda com funcionalidades para o processo de recuperação de senha, o qual também possui uma forma de segurança através de criptografia assimétrica, a qual é alterada a cada 1 hora. Segundo Carts (2001), uma mensagem encriptada com uma chave pública só pode ser decriptada por quem criou a mensagem, porque somente esta tem conhecimento sobre a chave privada para poder ler a mensagem original.

### 3.2 Desenvolvimento do Sistema

Para o desenvolvimento da aplicação, foi utilizado o processo ágil de desenvolvimento de software, Scrum. Aproveitando o método Kanban para gerenciar as etapas do projeto.

O Scrum foi criado por Jeff Sutherland e Ken Schwaber, onde definem que Scrum é um projeto em ciclos, divididos e chamados de sprints. Tudo que é

desenvolvido, é mantido em um backlog[8] e no início de cada sprint algumas destas tarefas são alocadas para desenvolvimento. Estas deverão ser entregues prontas no final da sprint. Termo este, que se refere a maneira como um time trabalha em conjunto para “fazer com que a bola avance no campo” (SUTHERLAND, 2016).

> No scrum tradicional, estão incluídos product owner[9], scrum master[10] e development team[11]. Então, para adaptá-lo para uma única pessoa, é necessário que esta mesma pessoa desempenhe e assuma a responsabilidade de cada papel. No scrum para uma pessoa, onde esta assume o papel de seu próprio product owner, é necessário definir claramente seu produto final, identificando recursos e requisitos e registrando-os para referência posterior. Já para assumir o papel de scrum master, é necessário ter em mente, que a pessoa será sua própria desimpedidora, posicionando-se como um legítimo scrum master, criando motivação, removendo impedimentos e se livrando de distrações). Para aplicar o scrum para uma pessoa, onde esta torna-se o seu próprio time de desenvolvimento, basta ter em mente que um time de desenvolvimento no método ágil scrum, é auto-gerenciável e auto-motivacional para que a sprint seja concluída com sucesso. Em relação a cerimônias, é necessário que se reserve um tempo diariamente para executar as dailys, onde se analisará o dia anterior e a partir desta análise, serão tomadas decisões e então planejar-se o que fazer no dia atual. Sobre as etapas de revisão e planejamento no scrum para uma pessoa, esta precisará agir diante destas situações, como seu próprio product owner e scrum master, revisando a sprint anterior e planejando a sprint atual com base nas revisões que foram realizadas (LUCIDCHART, 2021).

A chave para o desenvolvimento iterativo é frequentemente produzir versões do sistema final que funcionem, sendo que cada versão deve ser totalmente testada e integrada como se fosse a entrega do produto final (SBROCCO; MACEDO, 2012).

### 3.2.1 Análise

Nesta etapa do projeto, foi estudado sobre as principais soluções de pagamento disponíveis no mercado, bem como as taxas oferecidas por cada um.

---

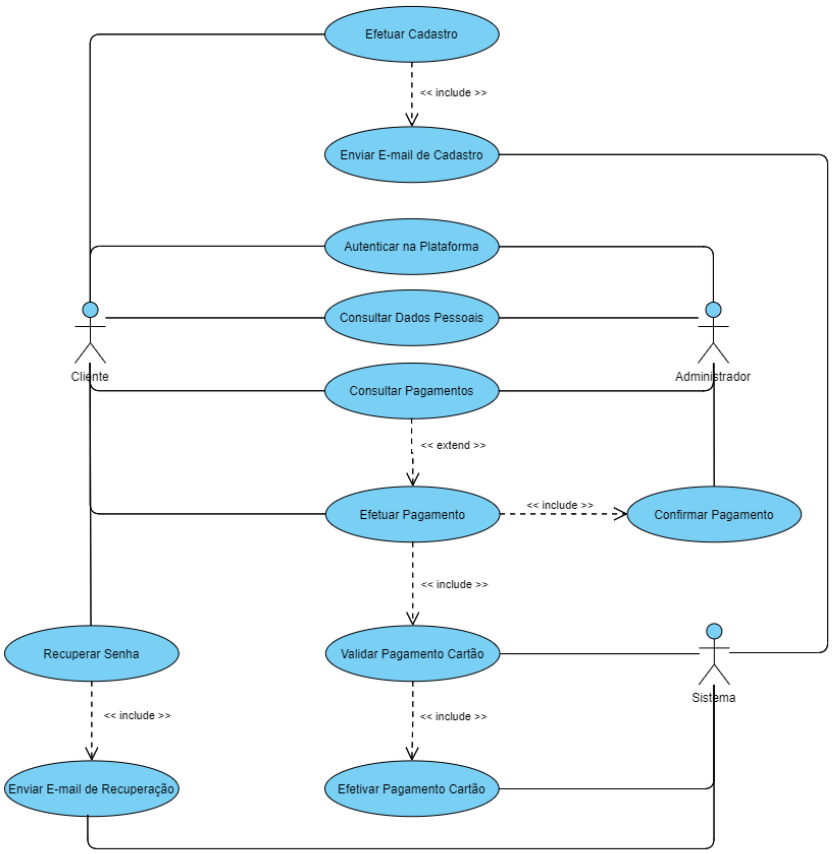
[8] Lista de funcionalidades
[9] Dono do Produto
[10] Mestre do Scrum
[11] Time de Desenvolvimento

Também foi feito orçamento em relação a servidores VPS (Virtual Private Server)[12], comparando preço e desempenho fornecido. Com isso, foi então iniciada a análise de requisitos e definições de regras de negócio, os quais foram os norteadores para o desenvolvimento da aplicação.

O trabalho realizado na fase de análise envolveu a ampliação da visão descrita na solicitação do sistema em um entendimento completo e detalhado de exatamente o que o novo sistema precisa fazer (DENNIS, et. al., 2014) .

Esta foi a etapa em que os requisitos foram levantados e começa-se a ter uma melhor visão das funcionalidades e das características do sistema. Tendo estes requisitos bem definidos e entendidos pelo time de desenvolvimento, pode-se então criar um conjunto de cenários que identifique um roteiro de uso do software. Estes cenários são conhecidos também como “casos de uso” (PRESSMAN, 2011).

Figura 1 – Diagrama de casos de uso - Zero Boleto




Fonte: Autor

[12] Servidor Virtual Privado é uma máquina virtual que permite controle total no lado do próprio servidor (GATTI, 2014).

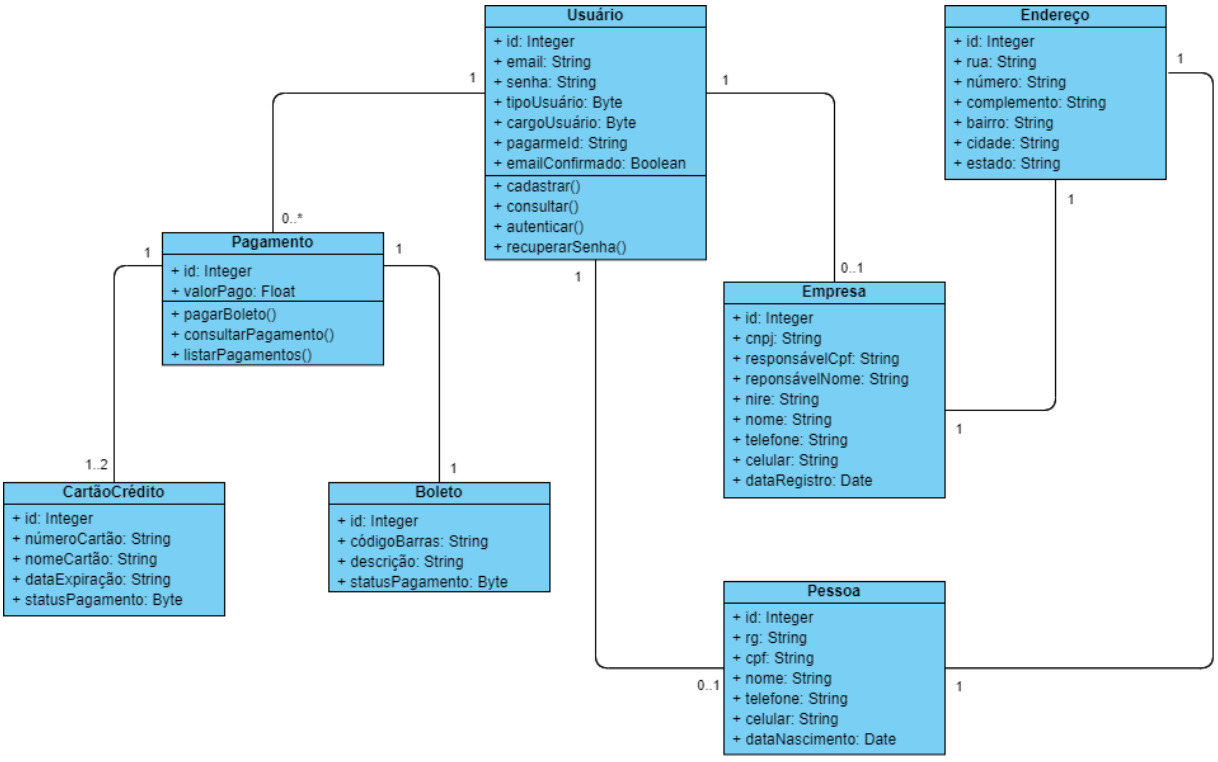
É possível visualizar através da Figura 1, de maneira generalizada, todas as maneiras existentes de interagir com o sistema, atrelados a cada agente existente.

### 3.2.2 Projeto

Após concluir a análise e obter as definições necessárias para a execução do projeto, foram desenvolvidos os documentos com a estrutura da aplicação. Esta documentação contém diagramas UML, que serviram de base para a criação do sistema, descrevendo detalhadamente todas as especificações e funcionalidades que compõem a aplicação.

Um diagrama de classes também foi modelado para ajudar a visualizar todas as entidades necessárias para construir a arquitetura da aplicação. Aqui, todas as classes relacionadas a dados são mapeadas, juntamente com possíveis relacionamentos entre cada entidade. Essa etapa auxilia no processo de definição de quais dados necessitam ser armazenados no banco de dados.

Figura 2 – Diagrama de classes - Zero Boleto

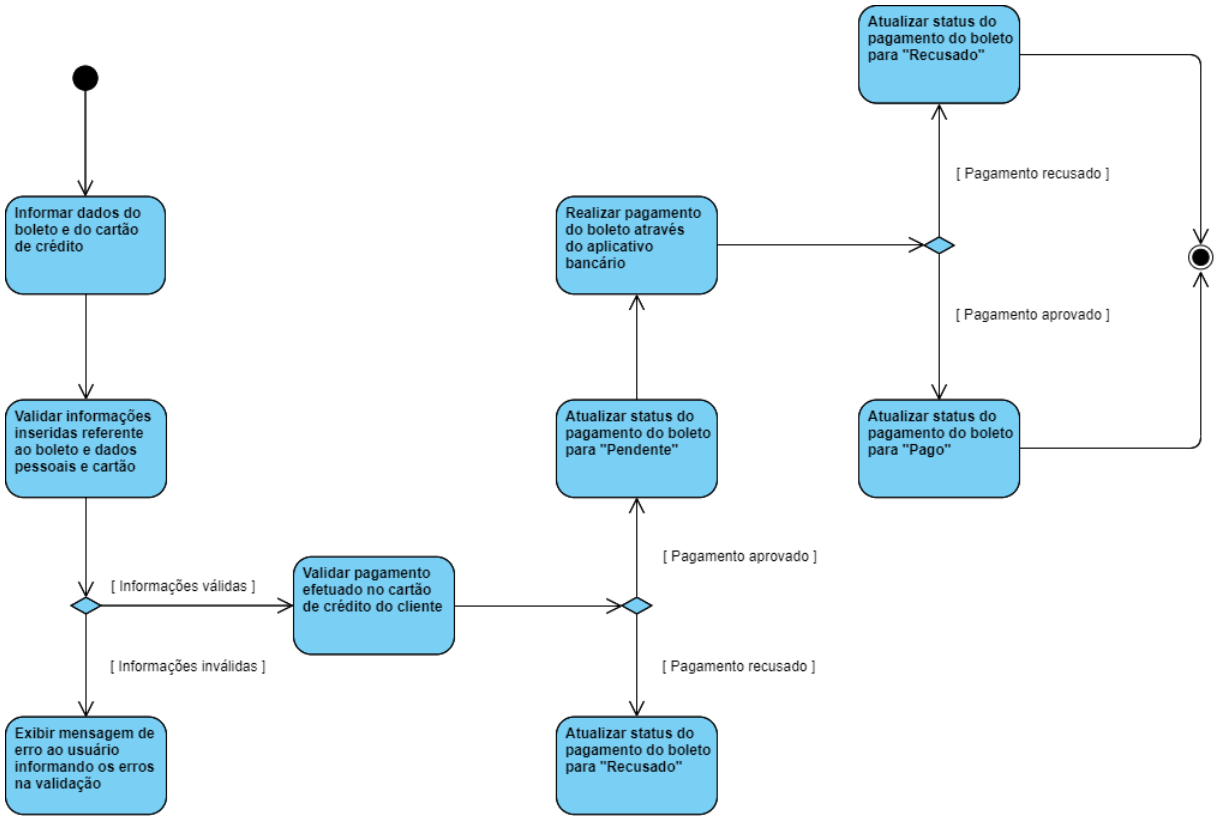



Fonte: Autor

Na Figura 2, pode ser visto com mais detalhes, sobre como a estrutura do sistema foi mapeada, conforme suas classes, atributos, operações e relações entre objetos. No caso, o diagrama ficou estruturado de forma que o usuário é o objeto principal, responsável por todas as relações. Sendo assim, um usuário pode ser uma empresa (CNPJ) ou uma pessoa (CPF) e estes ficam responsáveis pela relação entre o usuário e o endereço. Todo usuário pode ter nenhum ou muitos pagamentos, onde o pagamento pode ter um até dois cartões de créditos e apenas um único boleto atrelado a si próprio.

Também foi desenvolvido um diagrama de atividade para facilitar o entendimento da principal funcionalidade do sistema, que no caso é a de pagar o boleto propriamente.

Figura 3 – Diagrama de atividade - Zero Boleto




Fonte: Autor

A Figura 3 então permite visualizar de maneira clara a atividade de pagamento de um boleto, desde a sua fase inicial até a fase final, onde o pagamento

é aceito ou recusado de acordo com as políticas do cartão de crédito do cliente. Partindo do princípio que o usuário já esteja logado na plataforma, dá-se início a validação do fluxo de pagamento. Onde o usuário inicia informando os dados do boleto e do cartão de crédito, para que sejam validados na plataforma e também na API do Pagar.me. Após validado, caso obtenha sucesso, o boleto ficará com status pendente até que o administrador revise o pagamento e aprove o mesmo. Caso contrário, retornam-se mensagens de informações ao usuário relatando cada problema ocorrido durante a tentativa de realização do pagamento.

### 3.2.3 Codificação

Com as fases de análise e de projeto prontas, foi iniciada a fase de codificação do sistema, ou seja, tudo que foi analisado e projetado foi transformado em código. Com este, pode ser validado através de testes, todas as funcionalidades.

Para o desenvolvimento do sistema, foram utilizadas tecnologias como por exemplo C# .NET Core para controlar a API, onde a mesma conecta-se com um banco de dados PostgreSQL para armazenamento dos dados de usuários e logs de pagamentos. Ainda no backend da aplicação, foram utilizadas algumas técnicas de segurança, como criptografia hash para senhas armazenadas no banco de dados, utilizando o algoritmo Argon2 e também o método de autenticação JWT, conforme citados anteriormente. Na interface visual do usuário, foi utilizado Reactjs e na parte de infraestrutura, tem-se uma VPS Linux para hospedar as aplicações, utilizando o web server NGINX aplicando o protocolo HTTPS (Hyper Text Transfer Protocol Secure)[13]. Além da integração com a API terceira, responsável pela parte de pagamentos.

[13] Protocolo de Transferência de Hipertexto Seguro

Figura 4 - Trecho de código - Zero Boleto

```
5 references
public async Task<BaseResult> SaveTransactionAsync(PaymentTransactionInputModel payment)
{
    try
    {
        if (!VerifyBoletoIsValid(payment))
        {
            return BaseResult;
        }

        var user = await _userRepository.GetByIdAsync(
            id: payment.UserId,
            include: source => source
                .Include(p => p.Person)
                .ThenInclude(u => u.Address)
                .Include(p => p.Company)
                .ThenInclude(u => u.Address));

        PagarMeService.DefaultApiKey = _appSettings.CurrentValue.PagarMeApiKey;
        PagarMeService.DefaultEncryptionKey = _appSettings.CurrentValue.PagarMeEncryptionApiKey;

        var amount = payment.Amount.SumInterestRate() / payment.CreditCards.Count();

        var transactionsIds = new List<int>();
        var groupId = Guid.NewGuid();

        foreach (var creditCard in payment.CreditCards)
        {
            var newTransaction = new Transaction...;

            newTransaction.Save();

            var actualTransaction = PagarMeService.GetDefaultService().Transactions.Find(newTransaction.Id);

            var newPayment = new Payment()...;

            await _paymentRepository.AddAsync(newPayment);

            transactionsIds.Add(newPayment.Id);
        }

        BaseResult.Response = transactionsIds;
    }
    catch (PagarMeException ex)
    {
        ex.Error.Errors.ToList().ForEach(error => AddError(error.Parameter, error.Message));
    }
    catch (Exception ex)
    {
        AddError(string.Empty, ex.Message);
    }

    return BaseResult;
}
```

Fonte: Autor

Na Figura 4, tem-se um pouco de noção da principal funcionalidade do sistema, onde esta é responsável por fazer a cobrança do boleto no cartão de crédito do cliente e armazenar no banco de dados do lado do Zero Boleto. No trecho de código exibido, pode-se observar que a mesma recebe um objeto com todas as propriedades necessárias para concluir o pagamento. O método inicia fazendo algumas validações em relação ao boleto, como por exemplo, em relação a validade do mesmo ou se já é um boleto que está quitado. Caso esteja tudo ok, o método

avança e vai para a parte de pagamento através do cartão de crédito com a API do Pagar.me. Após tudo ocorrer bem, então os dados de pagamento são salvos no banco de dados interno do sistema e uma resposta de sucesso é devolvida para a requisição recebida.

### 3.2.4 Testes

Conforme citado na seção de codificação, após concluída cada iteração da fase de desenvolvimento, aplicava-se uma fase de testes, onde eram executados testes unitários e testes exploratórios. Em relação aos testes exploratórios, foram criados pelo menos dois testes para cada funcionalidade do sistema, onde um buscava validar se o método havia sido executado com sucesso e a outra se o método havia erros e se os erros eram os esperados. Por exemplo, na execução de submissão de um formulário, onde sabe-se todos os erros esperados para cada campo validado. Em relação aos testes exploratórios, neste momento realizava-se a validação dos requisitos propostos durante a análise do projeto, onde buscava-se validar a usabilidade da aplicação como um todo, bem como se todos os requisitos levantados e se os mesmos estavam sendo atendidos. Todas as funções devem ser testadas com entradas corretas e incorretas no local de entrada do usuário (SOMMERVILLE, 2019). Ainda segundo Sommerville (2019), o teste é um processo que almeja alcançar a confiabilidade do software.

Durante o desenvolvimento da aplicação, também foram desenvolvidos testes unitários e testes de integração, utilizando a ferramenta xUnit disponível através do Visual Studio, IDE (Integrated Development Environment)[14] que foi utilizada para o desenvolvimento da aplicação.

A cada etapa de desenvolvimento foram realizados testes de usabilidade, com a finalidade de verificar se a experiência e percepção de usuário são positivas ao utilizar o software e testes exploratórios, acessando diversas telas através de diversos fluxos da aplicação, na intenção de garantir em termos de usabilidade, o funcionamento desejado referente aos requisitos.

[14] Ambiente de Desenvolvimento Integrado

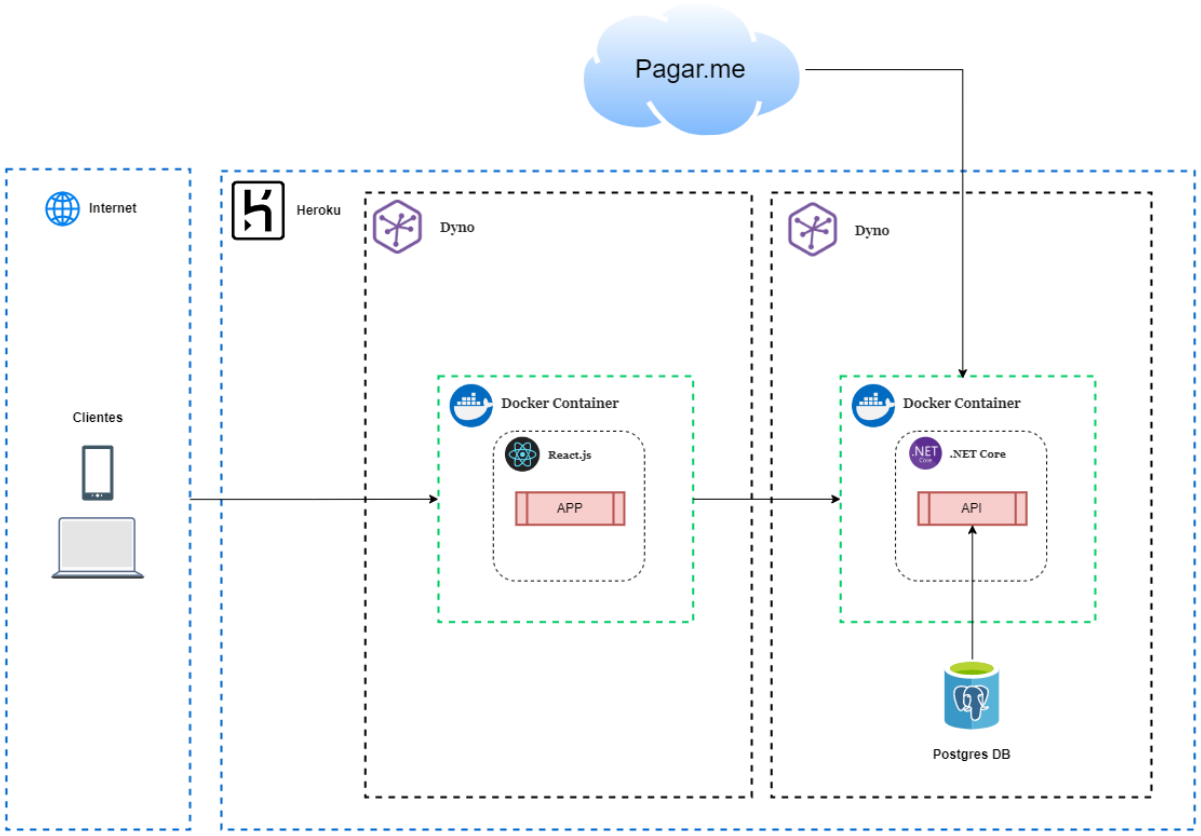
### 3.2.5 Implantação

Tendo os testes sido concluídos com sucesso, os quais foram executados na etapa anterior, então foi gerada uma versão de homologação para ser hospedada no servidor, para pudessem ser feitas validações em um ambiente o mais similar possível ao de ambiente de produção.

No decorrer do projeto, as fases citadas acima, foram realizadas ao longo de cada sprint, de acordo com o escopo de tarefas estimadas para cada uma destas, desenvolvendo cada conjunto de funcionalidades alocadas para cada sprint.

A implantação foi realizada utilizando o serviço de hospedagem na nuvem, Heroku.

Figura 5 - Diagrama de arquitetura - Zero Boleto




Fonte: Autor

A Figura 5 possibilita entender melhor em detalhes como se dispõe a arquitetura do sistema, demonstrando algumas das tecnologias adotadas para cada parte da publicação da aplicação em um servidor na nuvem.

Dessa maneira, pode-se perceber que a aplicação está hospedada no servidor do Heroku, através de dois containers docker, considerados Dynos na plataforma e que são responsáveis por executarem as aplicações tanto de front-end quanto de back-end. Sendo assim, ao acessar o serviço através de smartphone ou computador, a aplicação front-end se comunica com a aplicação de back-end, que por sua vez tem as informações necessárias para realizar o processo entre a API em nuvem do Pagar.me e com o banco de dados PostgreSQL que fica sob domínio direto do Heroku.

### 3.2.6 Manutenção

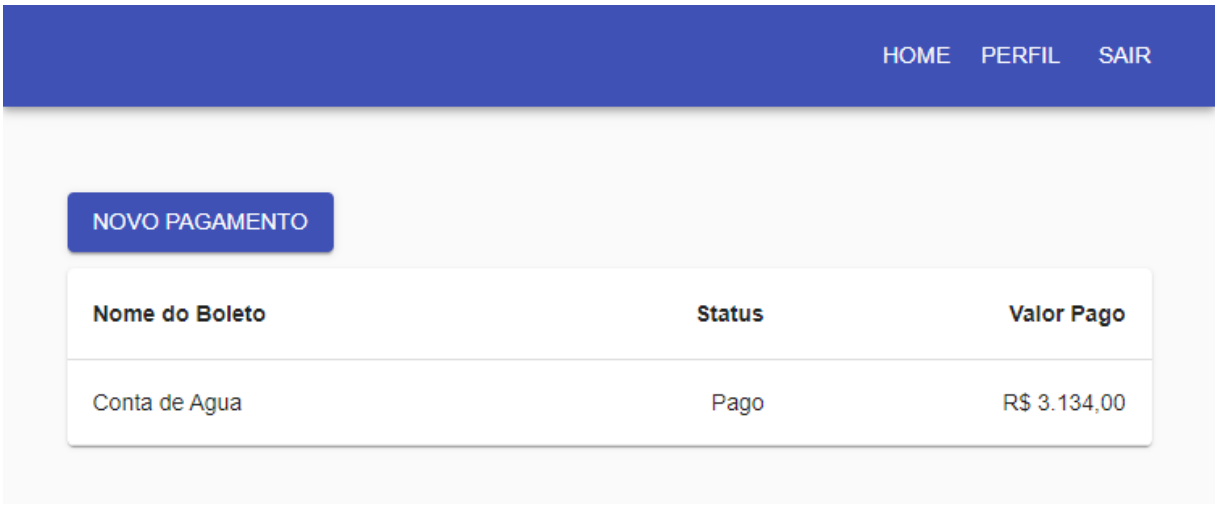
A aplicação está completamente integrada com o GitHub e o Heroku. Desta maneira, sempre que há uma atualização no projeto, a publicação da versão atualizada do software ocorre de forma automática e automatizada.

## 4 ZERO BOLETO: UM SISTEMA DE PAGAMENTOS

Nesta seção, são apresentadas as principais e mais importantes telas do sistema, que engloba toda a questão em relação ao que se refere ao pagamento dos boletos, bem como as descrições detalhadas de suas finalidades.

Partindo da premissa que o usuário já está cadastrado no sistema, ao fazer o acesso através de suas credenciais, o mesmo é direcionado para a tela inicial do sistema, a qual pode ser vista a partir da Figura 6.

Figura 6 - Tela Inicial - Zero Boleto

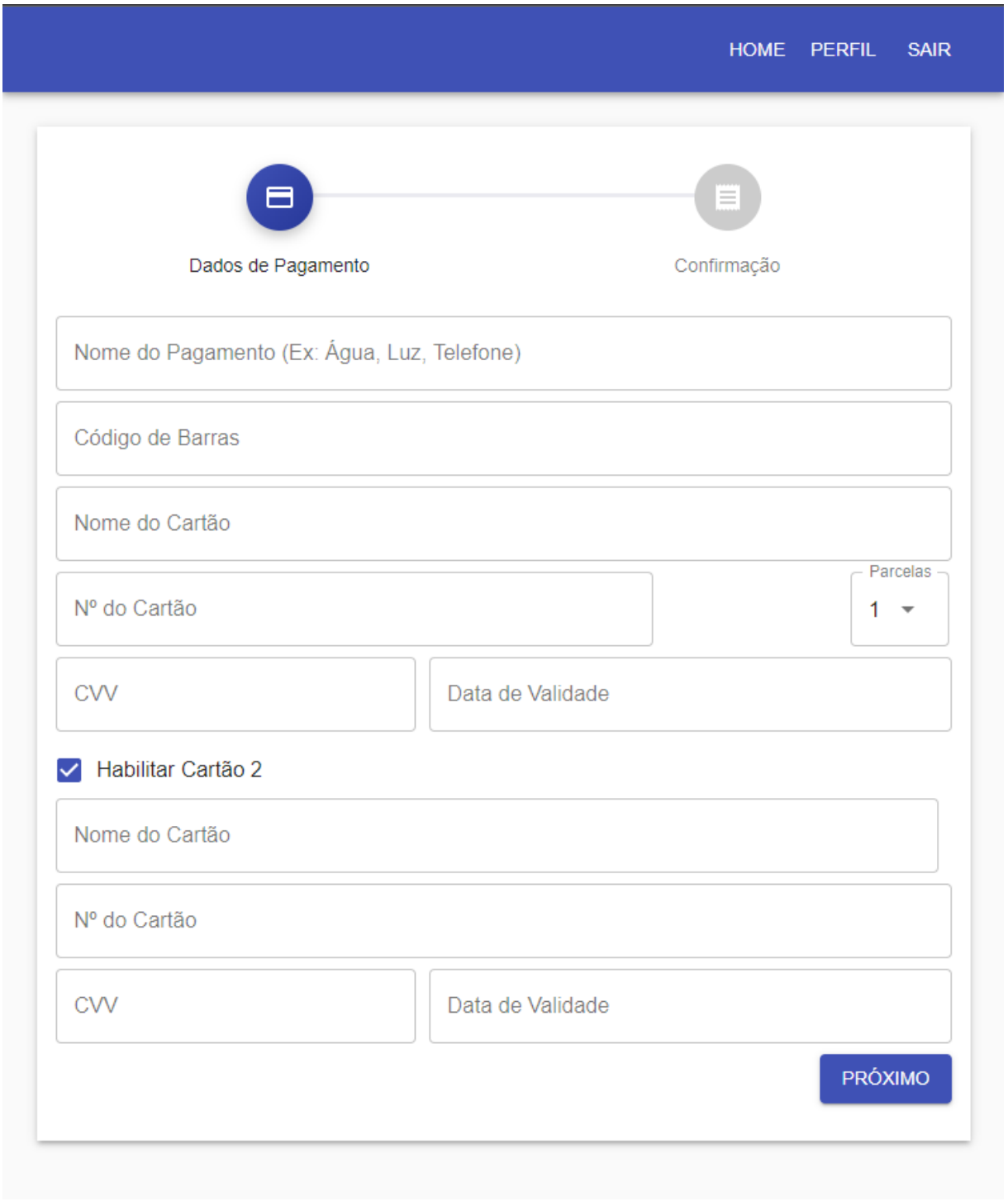



Fonte: Autor

Conforme pode ser visto na Figura 6, a partir da tela inicial o usuário dispõe de acesso ao menu Perfil e ao menu Novo Pagamento, além de ter um extrato detalhado de seus pagamentos através da tabela exibida na tela inicial.

Na Figura 7, é possível visualizar o formulário padrão para a realização de um novo pagamento de boleto utilizando até dois cartões de crédito, a qual pode ser acessada através da tela inicial do sistema.

Figura 7 - Tela de Dados de Pagamento - Zero Boleto

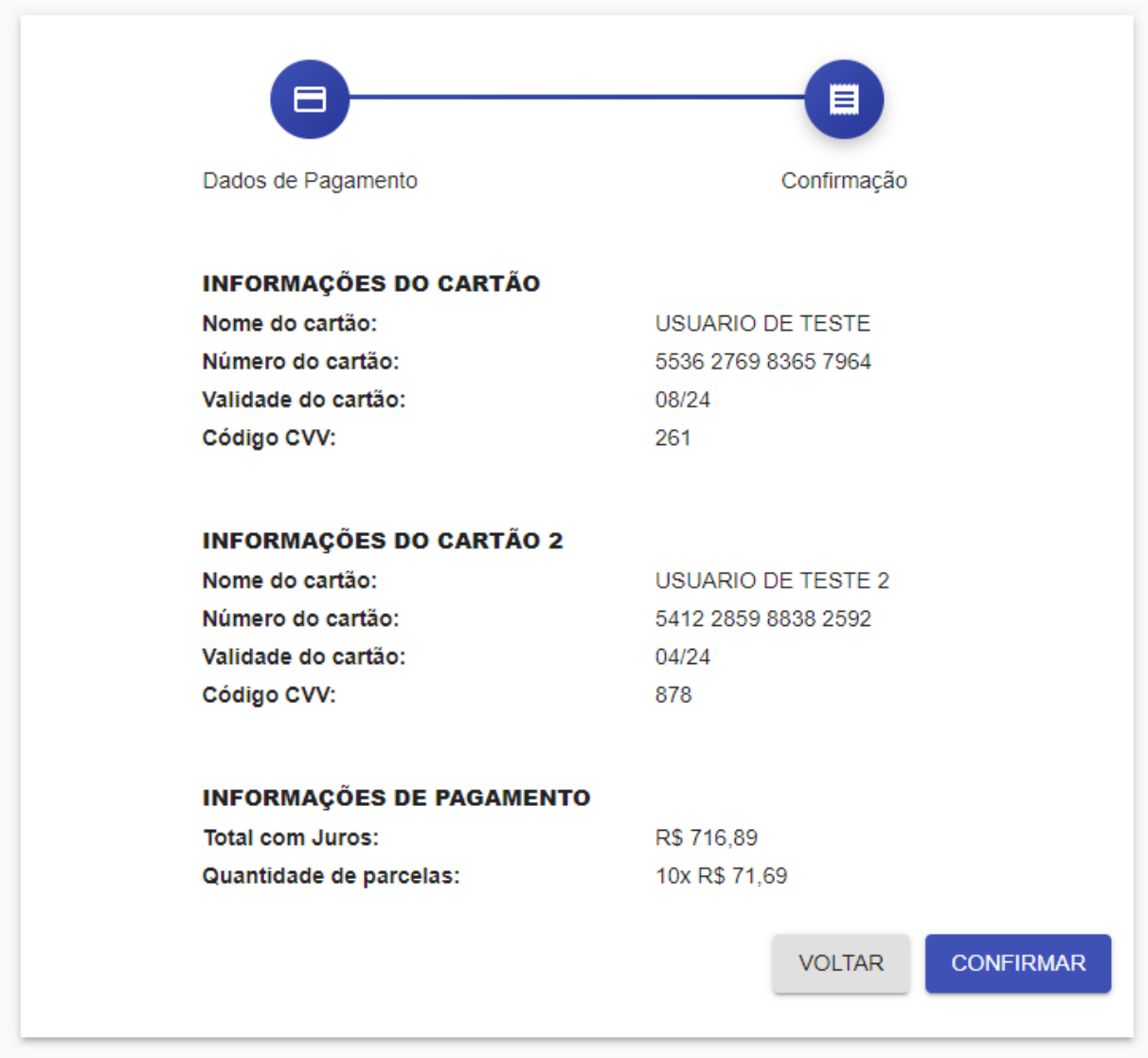



Fonte: Autor

A Figura 7 mostra que para realizar o pagamento de um boleto utilizando até dois cartões de crédito, é necessário que o usuário insira os dados referentes ao boleto em questão, bem como dados do portador de cada cartão de crédito e uma breve descrição para facilitar a identificação do boleto que está sendo pago.

A Figura 8 exibe para o usuário os dados relacionados ao pagamento para que seja feita validação pelo usuário antes de mandar os dados para o servidor.

Figura 8 - Tela de Confirmação de Pagamento - Zero Boleto




Fonte: Autor

Então, através da Figura 8, o usuário pode verificar se todos os dados estão corretos e também ter acesso ao valor que será pago, já com os juros somados e o valor que ficará para cada parcela em seus cartões de crédito.

# 5 TRABALHOS CORRELATOS

Algumas instituições financeiras já disponibilizam o pagamento de boletos com parcelamento através de cartão de crédito, como por exemplo o PicPay e o Nubank. De acordo com (PICPAY, 2021) é possível pagar boletos em até 12 vezes no cartão de crédito, com uma taxa de 3,09% calculada sobre o valor do boleto, mais 3,69% sobre cada parcela. Enquanto para o Nubank, a taxa é de 4% sobre cada parcela (NUBANK, 2021b) e permite que os clientes concentrem seus gastos na fatura do cartão de crédito do próprio Nubank e também parcelem os boletos.

Portanto, como as opções disponíveis atualmente possibilitam o pagamento somente utilizando um único cartão, que no caso, é o próprio da instituição responsável, a aplicação, que é desenvolvida neste projeto de pesquisa, disponibilizará formas de pagamento com mais de um cartão de crédito, além de cobrar taxas mais baixas do que as soluções presentes no mercado atualmente.

De acordo com Pagar.me (2021), a taxa cobrada pelo seu gateway de pagamentos é de 3,19% por transação, o que permite que a aplicação desenvolvida neste projeto, trabalhe com uma taxa de 3,60% ao mês e ainda obtenha lucros. Além de oferecer versão web do software para o pagamento de boletos e também o pagamento através de mais de um cartão de crédito.

Quadro 1 - Comparação com outras aplicações

| Aplicação | Taxa de juros | Versão web | Pagamento com mais de um cartão |
|---|---|---|---|
| PicPay | 3,09% + 3,69% por parcela | Não possui | Não possui |
| Nubank | 4,99% por parcela | Não possui | Não possui |
| ZeroBoleto | 3,60% por parcela | Possui | Possui |

Fonte: Autor

O Quadro 1 torna possível analisar de uma maneira ilustrada os benefícios, vantagens e desvantagens entre cada aplicação.

# 6 CONCLUSÃO

Este trabalho abordou o desenvolvimento de um sistema web denominado Zero Boletos, o qual tem como finalidade realizar pagamentos de boletos através de cartões de crédito. Foram utilizadas tecnologias atuais para o desenvolvimento deste projeto, de acordo com as suas necessidades e foi orientado pelo uso de uma metodologia que se mostrou mais adequada.

Dessa maneira, considera-se que o sistema Zero Boletos atendeu ao propósito esperado e que seu objetivo foi alcançado: criar uma aplicação web que permita pagar boletos através de mais de um cartão de crédito e com taxas mais baixas que as oferecidas por outras aplicações existentes no mercado.

Fica como indicação para trabalho futuro, automatizar o processo de pagamento do boleto bancário propriamente, utilizando alguma API de open banking que melhor se adeque ao projeto. Além de também permitir que o cliente possa escolher quanto do valor total deseja alocar em cada cartão de crédito.

## REFERÊNCIAS

ÁLVAREZ, R.; ALICIA, A.; ZAMORA, R. **Optimizing a Password Hashing Function with Hardware-Accelerated Symmetric Encryption.** 2018. Disponível em <https://doi.org/10.3390/sym10120705> . Acesso em: 02 nov. 2021.

AUTH0. **Articles.** 2021. Disponível em: <https://auth0.com/docs/security/tokens/json-web-tokens>. Acesso em: 02 nov. 2021a.

AUTH0. **Hashing Passwords: One-Way Road to Security.** 2021, Disponível em: <https://auth0.com/blog/hashing-passwords-one-way-road-to-security/>. Acesso em: 02 nov. 2021b.

BANKS, A.; PORCELLO, E. **Learning React: Functional Web Development with React and Redux**. 1ed. O'REILLY, 2017.

BB. **Parcelar boletos**. 2021. Disponível em: <https://www.bb.com.br/pbb/pagina-inicial/voce/produtos-e-servicos/emprestimo/organizar-meus-compromissos/parcelar-boletos#/>. Acesso em: 04 nov. 2021.

BCB. **Fintechs**. 2021. Disponível em: <https://www.bcb.gov.br/estabilidadefinanceira/fintechs>. Acesso em: 04 nov. 2021.

BERTÃO, N. **Mais da metade dos brasileiros contrata empréstimo para terminar o mês, diz pesquisa do Google.** 2019. Valor Investe. Disponível em: <https://valorinveste.globo.com/produtos/credito/noticia/2019/08/26/mais-da-metade-dos-brasileiros-contrata-emprestimo-para-terminar-o-mes-diz-pesquisa-do-google.ghtml>. Acesso em: 02 nov. 2021

BLOKDYK, G. **JSON Web Token**. 3ed. Createspace Independent Publishing Platform, 2018.

BOSCARIOLI, C.; BEZERRA, A.; BENEDICTO, M.; DELMIRO, G. **Uma reflexão sobre Banco de Dados Orientados a Objetos.** 2006. Disponível em: <https://deinfo.uepg.br/~conged/artigo4.pdf>. Acesso em: 02 nov. 2021

CARTS, David A. **A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols.** 2001. Disponível em: <http://www.madchat.fr/crypto/papers/paper751.pdf>. Acesso em: 29 nov. 2022.

CNC. **Endividamento recorde acende alerta para uso do crédito e inadimplência**. 2021. Disponível em: <https://portal-bucket.azureedge.net/wp-content/2021/08/Analise_Peic_julho_2021.pdf>. Acesso em: 02 nov. 2021.

DEJONGHE, J. **C#: Learn C# in One Day and Learn It Well**. 1 ed. Learn Coding Fast, 2015.

DEJONGHE, D. **NGINX Cookbook: Advanced Recipes for High-Performance Load Balancing**. 2 ed. O'REILLY, 2022.

DENNIS, A.; HALEY, B.; ROTH, M. R. **Análise e Projeto de Sistemas**. 5 ed. LTC, 2014.

DINIZ, B. **O Fenômeno Fintech: Tudo sobre o movimento que está transformando o mercado financeiro no Brasil e no mundo**. 1 ed. Altabooks, 2020.

FIELDING, R.; TAYLOR, R. **Principled design of the modern Web architecture.** 2000. Disponível em: <https://www.ics.uci.edu/~fielding/pubs/webarch_icse2000.pdf>. Acesso em: 02 nov. 2021

GARCIA, M.; ROJAS, H. **Hands-On Visual Studio 2022: A developer's guide to exploring new features and best practices in VS2022 for maximum productivity**. 1 ed. Packt Publishing, 2022.

GATTI, D. **VPS Toolkit.** 1 ed. Lulu.com, 2014.

JONES, M.; CAMPBELL, B.; MORTIMORE, C. **JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants.** 2015. Disponível em: <https://www.hjp.at/doc/rfc/rfc7523.html#sec_1>. Acesso em: 02 nov. 2021.

KHUAT, T. **Developing a frontend application using ReactJS and Redux.** 2018. Disponível em: <https://www.theseus.fi/bitstream/handle/10024/150837/Tung_Khuat_1301747_Thesis.pdf?sequence=1&isAllowed=y>. Acesso em: 02 nov. 2021.

LOCK, A. **ASP.NET Core in Action**. 2 ed. Manning, 2021.

LUCIDCHART. **Scrum for one: A tutorial on adapting Agile Scrum methodology for individuals.** 2021. Disponível em: <https://www.lucidchart.com/blog/scrum-for-one>. Acesso em: 02 nov. 2021.

MARQUES, F. **NUBANK: O mercado de Fintechs no Brasil.** 2018. Disponível em: <https://app.uff.br/riuff/handle/1/8807>. Acesso em: 03 nov. 2021.

MASSE, M. **REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces**. O'REILLY, 2011.

MICROSOFT. **Overview of Entity Framework Core**. 2021. Disponível em: <https://docs.microsoft.com/en-us/ef/core/>. Acesso em: 02 nov. 2021.

MIDDLETON, N.; SCHNEEMAN, R. **Heroku: Up and Running**. 1 ed. O'REILLY, 2013.

NUBANK. **Como pagar e parcelar boletos com o cartão de crédito do Nubank.** 2021a. Disponível em: <https://blog.nubank.com.br/como-pagar-boletos-com-o-cartao-de-credito-nubank/>. Acesso em: 02 nov. 2021.

NUBANK. **O que é fintech e por que esse termo ficou tão popular.** 2021. Disponível em: <https://blog.nubank.com.br/fintech-o-que-e/>. Acesso em: 03 nov. 2021b.

NUBANK. Análise. 2022. Disponível em: <https://nubank.com.br/analise/>. Acesso em: 29 nov. 2022.

PICPAY. **Pague Boletos.** 2021. Disponível em: <https://pague-boletos.picpay.com/>. Acesso em: 02 nov. 2021.

PRESSMAN, Roger S. **Engenharia de Software: Uma abordagem Profissional**. 7 ed. Bookman, 2011.

SBROCCO, J. H. T. C.; MACEDO, P. C. **Metodologias ágeis: Engenharia de software sob medida**. 1 ed. Érica, 2012.

SOMMERVILLE, I. **Engenharia de Software**. 8 ed. Pearson Universidades, 2019.

SUTHERLAND, J. **SCRUM: A arte de fazer o dobro do trabalho na metade do tempo.** LEYA, 2016.

Pagar.me. **Método de Pagamento Cartão**. 2021. Disponível em: <https://pagar.me/ofertas/>. Acesso em: 12 nov. 2021.

Pagar.me. **Antifraude para e-commerce: o que é e qual a importância**. 2022. Disponível em: <https://pagar.me/blog/antifraude-para-ecommerce/>. Acesso em: 29 nov. 2022.

PENARD, W.; WERKHOVEN, T. **On the Secure Hash Algorithm family.** 2008. Disponível em: <https://silo.tips/download/chapter-1-on-the-secure-hash-algorithm-family>. Acesso em: 16 out. 2022.

RFC 7159. **The JavaScript Object Notation (JSON) Data Interchange Format**. 2014. Disponível em: <https://www.rfc-editor.org/rfc/rfc7159>. Acesso em: 06 dez. 2022.

RIGGS, S; CIOLLI, G; BARTOLINI, G. **PostgreSQL Administration Cookbook**. 9.5/9.6 ed. Packt Publishing, 2017.

TSITOARA, M. **Beginning Git and Github: A Comprehensive Guide to Version Control, Project Management, and Teamwork for the New Developer**. 1 ed. Apress, 2019.

WETZELS, J. **Open Sesame. The Password Hashing Competition and Argon2.** 2015. Disponível em: <https://eprint.iacr.org/2016/104.pdf>. Acesso em: 02 nov. 2021.

XUNIT. **xUnit**. 2022. Disponível em: <https://xunit.net/>. Acesso em: 16 out. 2022.