INSTITUTO FEDERAL DO SUDESTE DE MINAS GERAIS

CAMPUS AVANÇADO BOM SUCESSO

CURSO DE ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ANDRÉ EUGENIO NETO

DESENVOLVIMENTO DE UM SISTEMA FINANCEIRO PARA CONTROLE DE ORÇAMENTO PESSOAL

BOM SUCESSO

2020

ANDRÉ EUGENIO NETO

DESENVOLVIMENTO DE UM SISTEMA FINANCEIRO PARA CONTROLE DE ORÇAMENTO PESSOAL

Trabalho de Conclusão de Curso submetido ao Instituto Federal do Sudeste de Minas Gerais, Campus Avançado de Bom Sucesso como parte dos requisitos necessários para a obtenção do Grau de Curso Superior em Análise e Desenvolvimento de Sistemas.

Orientador: Graziany Thiago Fonseca

BOM SUCESSO

2020

Dados internacionais de catalogação na publicação (CIP)
Bibliotecária responsável Maria de Lourdes Cardoso CRB-6/3242

E87d Eugenio Neto, André, 1992 -

Desenvolvimento de um sistema financeiro para controle do orçamento pessoal / André Eugenio Neto. -- 2020.

45 f. : il. ; 30 cm.

Orientador: Graziany Thiago Fonseca

Monografia (Graduação) - Instituto Federal do Sudeste de Minas Gerais, Campus Avançado Bom Sucesso, Coordenadoria de Curso de Tecnologia em Análise de Desenvolvimento de Sistemas, 2020.

1. Software – Desenvolvimento. 2. Desenvolvimento ágil de software. 3. Controle financeiro. 4. Finanças pessoais. I. Fonseca, Graziany Thiago. II. Instituto Federal do Sudeste de Minas Gerais, Campus Avançado Bom Sucesso. III. Título.

CDD: 005.1

MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO SUDESTE DE MINAS GERAIS

COMPROVANTE DE APROVAÇÃO DO PROJETO POR BANCA EXAMINADORA Nº 7 / 2021 - BSCNA(11.01.10.01.01.02)

Nº do Protocolo: NÃO PROTOCOLADO

Juiz de Fora-MG, 27 de Janeiro de 2021

TERMO DE APROVAÇÃO

André Eugenio Neto

DESENVOLVIMENTO DE UM SISTEMA FINANCEIRO PARA CONTROLE DE ORÇAMENTO PESSOAL

Este Trabalho de Conclusão de Curso foi julgado e aprovado como requisito parcial para a obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais - *Campus* Avançado Bom Sucesso.

Assinado digitalmente em 25/02/2021 08:06)
GRAZIANY THIAGO FONSECA
PROFESSOR ENS BASICO TECN TECNOLOGICO
Matrícula: 1966904

(Assinado digitalmente em 02/02/2021 21:20)
VICTOR SCHMIDT COMITTI
PROFESSOR ENS BASICO TECN TECNOLOGICO
Matrícula: 3082930

(Assinado digitalmente em 02/02/2021 21:18)
PEDRO HENRIQUE DE OLIVEIRA E SILVA
PROFESSOR ENS BASICO TECN TECNOLOGICO
Matrícula: 1758559

Para verificar a autenticidade deste documento entre em https://sig.ifsudestemg.edu.br/documentos/ informando seu número:
7, ano: **2021**, tipo: **COMPROVANTE DE APROVAÇÃO DO PROJETO POR BANCA EXAMINADORA**, data de
emissão: **27/01/2021** e o código de verificação: **351c19997b**

Dedico este trabalho a todos que contribuíram direta ou indiretamente em minha formação acadêmica.

AGRADECIMENTOS

Primeiramente agradeço a Deus que, de alguma forma, me conduziu até aqui.

Aos professores orientadores, que durante todo este processo me acompanharam assiduamente, dando todo o auxílio necessário para o desenvolvimento deste trabalho.

Aos meus pais que sempre me incentivaram a ser determinado.

A minha namorada que me ajudou e, também, compreendeu as minhas ausências temporárias.

RESUMO

O presente trabalho apresenta todo o processo de desenvolvimento de um sistema para o controle de finanças pessoais. Com base na metodologia de desenvolvimento de software ágil SCRUM, foram criados todos os artefatos necessários para realizar a análise e a implementação do sistema. O uso do padrão de modelagem UML – Unified Modeling Language, tornou possível a melhor compreensão do sistema, apresentando todos os requisitos necessários, e facilitando a construção e o desenvolvimento do software. O sistema de Controle de finanças pessoais é um sistema web que permite que o usuário controle suas finanças, controlando suas movimentações, saldos e suas contas. Para seu desenvolvimento foi utilizado o framework Codeigniter, baseado na linguagem de programação PHP, juntamente com o banco de dados MySQL por ser fácil desenvolver aplicações web e criar telas com menos tempo.

Palavras-chave: Controle financeiro. Sistema web. Orçamento. Finanças.

ABSTRACT

The present report is about the development of a system for personal finances controlling. Based on the SCRUM agile software development methodology, all necessary artifacts were created to carry out the analysis and implementation of the system. The use of the UML - Unified Modeling Language modeling standard, made possible a better understanding of the system, presenting all the necessary requirements, and facilitating the construction and development of the software. The Personal Finance Control system is a web-based system that allows the user to control their finances, controlling their movements, balances and accounts. For its development, the Codeigniter framework was used, based on the PHP programming language, together with the MySQL database as it is easy to develop web applications and create screens with less time.

Keywords: Financial control. Web system. Budget. Finances

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

APP – Aplicativo

CSS – Cascading Style Sheets

HTML – Hyper Text Markup Language

JS – Java Script

MVC – Model-View-Controller

PDF – Adobe Acrobat Reader

PHP – Hypertext Preprocessor

SQL – Structured Query Language

WEB – World Wide Web

SUMÁRIO

1. INTRODUÇÃO

Organizar-se financeiramente é condição essencial para o desenvolvimento pessoal e profissional. A correria do dia-a-dia muitas vezes impede que as pessoas dediquem seu tempo para organizar e acompanhar sua vida financeira.

Para Cerbasi (2015), é necessário detalhar todos os gastos mensais e agir de acordo com essa informação, procurando alternativas que proporcionem uma vida financeira saudável. O autor ainda enfatiza que o planejamento financeiro familiar não será eficiente se não existir equilíbrio no orçamento, o que significa gastar menos do que ganha e investir a diferença com o pensamento no futuro.

O planejamento financeiro sempre foi necessário para a vida das pessoas, mas nem todos os indivíduos dão a devida importância para a administração de seu orçamento.

Segundo uma pesquisa realizada em 2020, com 813 consumidores das 27 capitais brasileiras, pelo Serviço de Proteção ao Crédito (SPC Brasil, 2020), foi demonstrado que aproximadamente 48% dos entrevistados não realizam o controle sob o seu orçamento, 25% confiam apenas na memória para anotar as despesas, 20% não fazem nenhum registro dos ganhos e gastos e 2% delegam a função para terceiros.

O controle do orçamento pessoal, se realizado de maneira bem ordenada e com o hábito de realizar as atualizações, é o segredo para um autocontrole financeiro, evitando despesas inesperadas ao final do mês.

Sendo assim, para que se obtenha uma melhor compreensão dos dados financeiros é necessário consolidá-los para que eles possam ser visualizados da melhor forma possível. A maneira capaz de facilitar e resolver esta situação é o desenvolvimento de um sistema que realiza o controle de forma sucinta e inteligente.

Desse modo, torna-se necessário criar ferramentas que facilitem o armazenamento de dados financeiros e, com base nessas informações, possibilitar a geração de relatórios, além de também ajudar as pessoas a terem um controle orçamentário para que obtenham uma vida financeira saudável.

A ferramenta proposta irá permitir o cadastro de receitas e despesas do usuário. Além do cadastro, será possível gerar relatórios que relacionam ganhos e gastos mensais, o que permite entender o que ocorre com as finanças sem a necessidade de verificar vários extratos diferentes, o que pode ser complicado e difícil de compreender.

Para o desenvolvimento deste trabalho foi utilizado o framework Codeigniter, baseado na linguagem de programação PHP, juntamente com o banco de dados MySQL. Além das ferramentas citadas, foi necessário a utilização de um servidor web que é responsável por processar as requisições do usuário.

Após o desenvolvimento do sistema foi realizada a etapa de validação da ferramenta, onde foi utilizado uma base de dados fictícia de um controle financeiro de orçamento pessoal de uma pessoa física, onde foram inseridos dados de receitas e despesas mensais.

Para esse projeto foi necessário aprofundar os conhecimentos sobre as ferramentas e etapas existentes para o desenvolvimento de um sistema web, bem como a realização de uma pesquisa de mercado sobre plataformas e aplicativos que controlam o orçamento pessoal.

Após esse processo, iniciou-se a análise das particularidades e características do sistema, tendo como foco verificar quais informações seriam necessárias para obter um controle financeiro básico e funcional para o usuário. Todas as etapas realizadas durante o processo de desenvolvimento dos sistemas estão descritas no capítulo 3.

Este trabalho está dividido em quatro capítulos. No primeiro capítulo apresenta-se o projeto, expondo uma breve contextualização e apresentando a problemática vislumbrada, assim como o objetivo geral. No segundo capítulo, é realizada uma revisão da literatura relacionada à área de desenvolvimento de sistemas, engenharia de software e banco de dados. No terceiro capítulo é apresentado todo o processo de desenvolvimento do sistema, bem como os resultados alcançados, onde são descritos detalhadamente os requisitos do sistema e os diagramas, alinhados a cada funcionalidade que a ferramenta oferecerá. Por fim, o quarto capítulo contém as considerações finais e possibilidades para trabalhos futuros.

1.2. Objetivo

1.2.1. Objetivo Geral

Apresentar todo o processo de desenvolvimento de um sistema para o controle de finanças pessoais.

1.2.2. Objetivo Específico

Visando atingir o objetivo principal, alguns objetivos específicos são requeridos, entre eles:

- Análise da literatura de temas relacionados;
- Levantamento de requisitos da ferramenta a ser criada;
- Modelagem do sistema;
- Criação da Base de Dados;
- Desenvolvimento do Sistema;
- Validação da ferramenta;

2. REFERENCIAL TEÓRICO

Este capítulo apresenta os assuntos que fundamentam este trabalho, como o valor do controle financeiro e o poder da informação, tecnologias envolvidas e conceitos computacionais utilizados na construção da ferramenta.

2.1. A importância das informações

Para Tuomi (1999), os dados são simples fatos que quando tratados e estruturados são transformados em informação. A informação se torna conhecimento quando é analisada e descrita em um contexto ou até mesmo quando se dá um significado a ela. Sendo assim, a informação é o principal requisito para a geração do conhecimento.

O aumento da quantidade de informação gerada está em evidência uma vez que diversos estudiosos analisam este fenômeno e meios de extrair conhecimento de toda esta informação.

> A velocidade e a amplitude com que o conhecimento gerado passou a ser compartilhado provocaram o surgimento de uma dinâmica de reaproveitamento e produção de novos conhecimentos, bem como o aparecimento de novas necessidades de tratar a informação. (RAMOS; BRASCHER, 2009, p. 57)

Segundo Mattos (2007), não basta fornecer ou obter informações, é necessário fazê-las compreensíveis e compreendê-las. Porém, o que é uma informação importante para uma pessoa, pode não ser para outra, pois há tipos diferentes de interesse na mesma informação, que representada de outra maneira, ou ainda juntamente com outros dados, toma um formato diferente de representação.

Levando em consideração a subjetividade da relevância de informações, que varia de pessoa para pessoa, é importante dar ao usuário do sistema a liberdade de visualizar e organizar suas informações da forma desejada, seja através de relatórios, gráficos, planilhas e outros formatos.

2.2. Finanças pessoais

Para Massaro (2015), o hábito de consumir é um tema universal, pois todas as pessoas consomem, e o ato de consumo engloba o uso de recursos financeiros. O autor também evidencia que as propagandas são um fator determinante para o hábito de consumir, e que tamanha influência emocional pode acarretar em uma doença. Ele reforça que é necessário

distinguir o que é necessidade e o que é desejo, pois a maior parte do desequilíbrio financeiro consiste na falta de conhecimento das pessoas não conseguirem enxergar o que é cada coisa.

Conforme exposto por Massaro (2015) existem quatro termos para o desenvolvimento de um controle financeiro: patrimônio, receitas, despesas e fluxo de caixa. O patrimônio é tudo aquilo que o indivíduo possui; as receitas significam as entradas de dinheiro do indivíduo, seja por atividade profissional ou outros tipos de renda; as despesas são toda a saída de dinheiro, as quais podem ser fixas e variáveis. E, por último, o fluxo de caixa, que é a ordem de lançamento das entradas e saídas de dinheiro, que pode ser representado pela soma das receitas subtraídas pelas despesas.

Para Silva (2007), o fluxo de caixa é uma demonstração onde são apresentados os recebimentos e pagamentos efetuados em um determinado período. A autora ainda especifica, que o fluxo de caixa é o resultado do fluxo financeiro, contemplando um controle de todas as entradas e saídas de dinheiro do patrimônio de uma pessoa. Vale ressaltar que um fluxo de caixa diário auxilia no entendimento de quais são os dias que mais entraram ou saíram dinheiro da conta.

2.3. Ferramentas para controle financeiro

Devido à necessidade de controle e organização da vida financeira, muitas ferramentas têm surgido no cenário global.

De acordo com o site Organizze (2020), o Organizze é um aplicativo de gerenciamento financeiro online, que permite lançar despesas manualmente e agrupar por categorias. Também é possível anexar comprovantes e realizar transferências entre contas bancárias cadastradas dentro do aplicativo. A solução é disponibilizada para Web, iOS e smartphones Android.

Conforme verificado em Mobills (2020), a ferramenta Mobills é um app de controle financeiro que funciona como um controle de contas, saldos e cartão de crédito. Além dos lançamentos das despesas, é possível definir objetivos financeiros que auxiliam nos planejamentos futuros. Com a missão de ajudar os usuários em educação financeira, é possível ver postagens de dicas financeiras no blog oficial do aplicativo. O app está disponível para Web, iOS e smartphones Android.

Existe também uma plataforma chamada Guiabolso, consultado em Oficial da net (2018), que unifica contas e cartões em um só lugar. Há também a possibilidade de pesquisar os melhores empréstimos com base nos dados do usuário e contratar o serviço pelo próprio

aplicativo. Com a opção radar de CPF, é permitido rastrear a situação que o CPF se encontra. O app está disponível para Web, iOS e smartphones Android.

2.4 Engenharia de Software

Para Padua (2000), o conceito de engenharia tem correlação com diferentes variáveis, como a arte, atendimento das necessidades humanas, conhecimentos científicos, conhecimentos empíricos, habilitações específicas, recursos naturais, formas adequadas e processos. O autor ainda menciona que a engenharia de software utiliza resultados da ciência e fornece problemas para seus estudos e, por isso, não deve ser confundida com a Ciência da Computação.

De acordo com Pressman (2006), a engenharia de software está relacionada na elaboração e utilização de princípios de engenharia com a finalidade de conseguir softwares a preços acessíveis, que sejam confiáveis e que tenham eficiência para trabalhar em máquinas reais.

Para Sommerville (2011), a engenharia de software é importante, pois os indivíduos em geral dependem dos sistemas avançados, sendo assim, o profissional deve se preocupar em desenvolver um sistema de forma rápida, que seja econômico e confiável.

É essencial usar técnicas da engenharia de software para o desenvolvimento de sistemas, pois nesse processo de evolução é verificado os requisitos essenciais para o cliente e a validação para evitar possíveis falhas.

Conforme explicado por Sommerville (2011), é mais barato adotar técnicas da engenharia de software, em vez de simplesmente escrever os programas como se fossem algum projeto pessoal, a maior despesa ocorre nas implementações solicitadas após o sistema começar a ser usado.

Para a realização de um projeto é necessário obter um gerenciamento ágil e interativo, de modo que se obtenha melhor gestão de tempo e distribuição de recursos para a equipe. Para o desenvolvimento do sistema proposto neste trabalho foram utilizadas algumas técnicas da metodologia de desenvolvimento ágil Scrum.

Como parte deste método, uma das partes fundamentais para este projeto são os requisitos. Estes são definidos durante os estágios iniciais do desenvolvimento de um sistema, como uma especificação do que deverá ser implementado. Eles descrevem um comportamento, uma propriedade ou um atributo do sistema. Podem também definir restrições ao processo de desenvolvimento do sistema, conforme cita Sommerville (2011).

2.5 Scrum

O Scrum é definido de uma maneira simples, da seguinte forma:

> Scrum" é um framework estrutural que está sendo usado para gerenciar o desenvolvimento de produtos complexos desde o início de 1990. Scrum não é um processo ou uma técnica para construir produtos; em vez disso, é um framework dentro do qual você pode empregar vários processos ou técnicas. O Scrum deixa claro a eficácia relativa das práticas de gerenciamento e desenvolvimento de produtos, de modo que você possa melhorá-las. (SCHWABER E SUTHERLAND, 2013, p. 3)

O Scrum está inteiramente ligado ao processo de desenvolvimento de software. Em concordância com Sommerville (2011), uma característica marcante do Scrum são os ciclos de *sprint*. Um sprint é uma forma de planejamento onde o trabalho a ser realizado é avaliado, seleciona-se os recursos para o desenvolvimento do projeto e o software é implementado.

Na abordagem em questão, existem três fases, conforme exposto por Sommerville (2011). A primeira é o planejamento do projeto, onde são estabelecidos os objetivos do projeto e da arquitetura do software. Após isso, ocorre uma quantidade de ciclos de *sprin*t onde cada um desses gera um incremento do sistema. A última e terceira fase é o encerramento do projeto e a elaboração da documentação exigida como manuais de usuário e um *feedback* do que foi absorvido com o desenvolvimento do projeto.

Como exposto por Sommerville (2011), a ideia principal do Scrum é que todas as pessoas envolvidas no projeto possuam poderes para tomar decisões e o *Scrum Master,* substituto da denominação "gerente de projeto", é um facilitador que organiza reuniões diárias, registra decisões, comunica com cliente e gerência externamente a equipe.

Esse método pode ser considerado como uma abordagem que proporciona uma melhor forma de visualizar os problemas, e serve como um roteiro para o processo de desenvolvimento e alcance de resultados em equipe.

2.6 Banco de Dados

De acordo com Thomson e Welling (2005), um banco de dados tem a capacidade de armazenar, pesquisar, classificar e recuperar dados de forma bastante eficiente.

Segundo o Manual de Referências do Mysql 4.1 (2020), um banco de dados é uma coleção de dados estruturados, que pode ser uma simples lista de compras ou uma grande quantidade de informação de sua rede corporativa. O funcionamento consiste em adicionar, acessar e processar dados armazenados em um banco de dados de um computador, e para

realizar essas tarefas é necessário obter um sistema de gerenciamento de banco de dados como o Servidor MySQL.

2.6.1 MySQL

Para Oliveira (2002), SQL (Structured Query Language) é um conjunto de comandos de manipulação de banco de dados, que permite criar uma estrutura, além de incluir, excluir, modificar e pesquisar informações nas tabelas inseridas na base.

O MySQL é um servidor robusto de bancos de dados *SQL* rápido, que pode ser usado em sistemas de produção com uma grande quantidade de dados e também pode ser embutido em software de uso em massa, conforme verificado no Manual de Referências do Mysql 4.1 (2010).

Conforme descrito por Thomson e Welling (2005), o servidor de MySQL, controla o acesso aos dados armazenados para ter certeza que muitos usuários possam logar ao mesmo tempo, fornecer consultas rápidas aos dados e assegurar que somente os usuários que tenham permissão obtenham acesso.

De acordo com o Manual de Referências do Mysql 4.1 (2020), o sistema de gerenciamento de banco de dados MySQL, é um banco de dados relacional que armazena seus dados em várias tabelas diferentes, uma solução melhor do que armazenar os dados apenas num só local, pois proporciona maior velocidade. A linguagem padrão mais usada para o acesso aos bancos de dados é a Linguagem *SQL*, definida pelo Padrão ANSI/ISO SQL.

2.7 Framework - Codeigniter

Codeigniter é um framework voltado para pessoas que constroem sites usando PHP. Seu objetivo é permitir desenvolver projetos com base num conjunto de bibliotecas para tarefas comumente necessárias, bem como uma interface simples e com uma estrutura lógica para acessar essas bibliotecas. Disponível em Codeigniter (2020).

O Codeigniter usa o padrão Model, View, Controller (MVC) para organizar os arquivos de forma que os dados, a apresentação e o fluxo do aplicativo fiquem em partes separadas. Disponível em Codeigniter (2020).

Um aplicativo Codeigniter pode ser executado hospedado em um servidor web usando virtualização. Disponível em Codeigniter (2020).

2.8 Padrão de Projeto MVC (Model, View e Controller)

Um padrão de projeto é uma solução encontrada para um determinado problema dentro de um contexto de desenvolvimento de um software que pode ser reutilizada na resolução de problemas similares em outros contextos. Para a realização deste trabalho procurou se utilizar o Padrão de Projeto MVC (Model, View e Controller). O MVC é um padrão de projeto arquitetural, que possibilita a divisão do projeto em camadas muito bem definidas.

Conforme explicado por Macoratti (2002), a organização em camadas é a forma de proporcionar independência entre os componentes que vai atingir a eficiência e facilidade de manutenção de software.

A arquitetura MVC (Modelo Visualização Controle) divide a funcionalidade envolvida na manutenção e apresentação dos dados de uma aplicação, e está relacionado com a arquitetura da aplicação e em como os componentes se comunicam, de acordo com Macoratti (2002).

O modelo envolve os dados e as regras que comandam o acesso e a alteração dos dados e também fornece ao controlador a capacidade de se comunicar com as funcionalidades das aplicações incluídas pelo próprio modelo.

Um elemento de visualização renderiza todo o conteúdo do modelo e encaminha para o controlador todas as ações do usuário.

Um controlador é quem define como a aplicação deve-se comportar, pois é responsável por interpretar as ações do usuário e mapear para chamadas do modelo.

2.9 Linguagem de Programação PHP

O Hypertext Preprocessor (PHP), é uma linguagem de programação muito utilizada e é o principal atrativo para o desenvolvimento web, que tem como objetivo o desenvolvimento de páginas que serão geradas de forma dinâmica e rápida.

Para Thomson e Welling (2005), o PHP é uma linguagem que consiste na criação de scripts especificamente para a WEB. Dentro do script HTML, é possível inserir código de PHP que será executado toda vez que a página for aberta. O código de PHP é interpretado no servidor Web e gera HTML ou outra saída que o visitante verá.

Referente às capacidades que o PHP possui, Thomson e Welling (2005) citam o alto desempenho da ferramenta, interfaces para sistemas diferentes de banco de dados, bibliotecas integradas, baixo custo do serviço, suporte orientado a objetos e disponibilidade de código-fonte.

2.10 Html

O HTML descreve o conteúdo e a estrutura de um documento por meio de TAGS, que tem como objetivo formatar uma página. Cada tag contém um comando ou atributo e por meio dessas exibições é possível inserir músicas, frases, imagens entre outros recursos.

Os arquivos em HTML são interpretados por navegadores. Para Gallois (2008), uma página web oferece vários elementos que são utilizados para interagir com o usuário, sendo para mostrar visualmente ou não. As páginas web utilizam Hypertext Markup Language (HTML) para realizar as formatações dos dados de modo que os comandos mudem a maneira como os dados são dispostos na tela.

Na figura 1 e figura 2, demonstrou-se exemplos de codificação e o resultado do comando em HTML.

Figura 1: Exemplo de um código em HTML

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title> TCC - IF SUDESTE MG </title>
  </head>
  <body>

    <h1>Olá</h1>
    <h5>If Sudeste MG - Bom Sucesso MG</h5>

  </body>
</html>
```

Fonte: Elaborado pelo autor

Figura 2: Resultado apresentado do código em HTML

Fonte: Elaborado pelo autor

3. DESENVOLVIMENTO DO SISTEMA

3.1. Definição do ambiente

3.1.1. Coleta, análise e modelagem dos requisitos

O sucesso no desenvolvimento de sistemas está diretamente ligado a uma análise de requisitos bem elaborada. Pressman (2006) evidencia que a engenharia de requisitos ajuda a entender melhor o sistema proposto e posteriormente facilita na resolução do problema.

Sommerville (2011) classifica os requisitos de software em requisitos funcionais, não funcionais e organizacionais.

Os requisitos funcionais são as declarações das funções que o sistema deve oferecer, como o sistema se comporta com entradas particulares e como o sistema deve se comportar em situações específicas. O termo função é usado no sentido genérico da operação que pode ser realizada pelo sistema, seja por meio de comandos dos usuários, seja pela ocorrência de eventos internos ou externos ao sistema. Em alguns casos, os requisitos funcionais podem também explicitamente definir o que o sistema não deve fazer.

Os requisitos não-funcionais são as restrições nas funções oferecidas pelo sistema. Incluem restrições de tempo, restrições no processo de desenvolvimento, padrões, e qualidades globais de um sistema, como manutenibilidade, usabilidade, desempenho, custos e várias outras;

O desenvolvimento do sistema deu-se início analisando quais componentes da área financeira e área de desenvolvimento de software, seria essencial para a composição da ferramenta proposta.

Nesta análise, verificou-se que para realizar o controle financeiro, a plataforma teria que possuir telas de cadastro de receitas e despesas, onde deveria possuir valores e datas de lançamentos para cada situação. Também verificou a necessidade de permitir o cadastro de categorias por ser essencial definir o tipo de receitas e despesas cadastradas. Para os relatórios, observou que é fundamental desenvolver filtros de categorias e datas para que o usuário tenha diversas opções de analisar as informações.

Caso o usuário cadastre alguma informação errada ou que possua informação desnecessária, o sistema terá que permitir a alteração ou exclusão do dado.

3.1.2. Requisitos do sistema

a) Requisitos funcionais

Quadro 1: Requisitos funcionais

Código	Título	Descrição	Condições
RF01	Cadastro de usuário	O sistema deverá permitir inserir um usuário no banco de dados para que uma pessoa possa acessar o sistema.	-
RF02	Alteração usuário	O sistema deverá dar permissão para atualizar os dados do usuário	Ter efetuado login
RF03	Excluir usuário	O sistema deverá dar a possibilidade de excluir a conta de usuário.	Ter efetuado login
RF04	Cadastrar Categoria	O sistema deverá permitir que o usuário cadastre categorias de receita ou despesa	Ter efetuado login
RF05	Editar Categoria	O sistema deverá permitir a atualização de dados da categoria	Ter efetuado login, existir cadastro de categoria
RF06	Excluir Categoria	O sistema deverá permitir a exclusão da categoria cadastrada	Ter efetuado login, existir cadastro de categoria
RF07	Cadastrar Receitas	O sistema deverá permitir o cadastro de receitas	Ter efetuado login
RF08	Editar Receitas	O sistema deverá permitir a atualização dos lançamentos de receitas realizadas	Ter efetuado login, existir cadastro de receitas

RF09	Excluir Receitas	O sistema deverá permitir a exclusão dos lançamentos de receitas realizados	Ter efetuado login, existir cadastro de receitas
RF10	Gerar Relatório por categoria de receita	O sistema deverá gerar relatório de despesa por categoria e permitir a exportação para pdf	Ter efetuado login
RF11	Gerar Relatório de receitas	O sistema deverá gerar relatório por data de lançamento e permitir a exportação para pdf	Ter efetuado login
RF12	Cadastrar Despesas	O sistema deverá permitir o cadastro de despesas	Ter efetuado login
RF13	Editar Despesas	O sistema deverá permitir a atualização dos lançamentos de despesas realizadas	Ter efetuado login, existir cadastro de despesas
RF14	Excluir Despesas	O sistema deverá permitir a exclusão dos lançamentos de despesas realizados	Ter efetuado login, existir cadastro de despesas
RF15	Gerar Relatório por categoria de despesa	O sistema deverá gerar relatório de despesa por categoria e permitir a exportação para pdf	Ter efetuado login
RF16	Gerar Relatório de despesas	O sistema deverá gerar relatório por data de lançamento e permitir a exportação para pdf	Ter efetuado login
RF17	Disponibilizar Dashboard	O sistema deverá conter um dashboard com as informações de receita e despesa	Ter efetuado login

RF18	Login	O sistema deve permitir o acesso apenas para usuários que tenham realizado cadastro	Ter realizado cadastro

Fonte: Elaborado pelo autor

b) Requisitos não funcionais

Quadro 2: Requisitos não funcionais

Código	Descrição
RF01	A interface com o usuário deve ser de fácil entendimento para que não exista necessidade de treinamento
RNF02	Toda geração de relatórios não deve ultrapassar 5 segundos para obter resposta

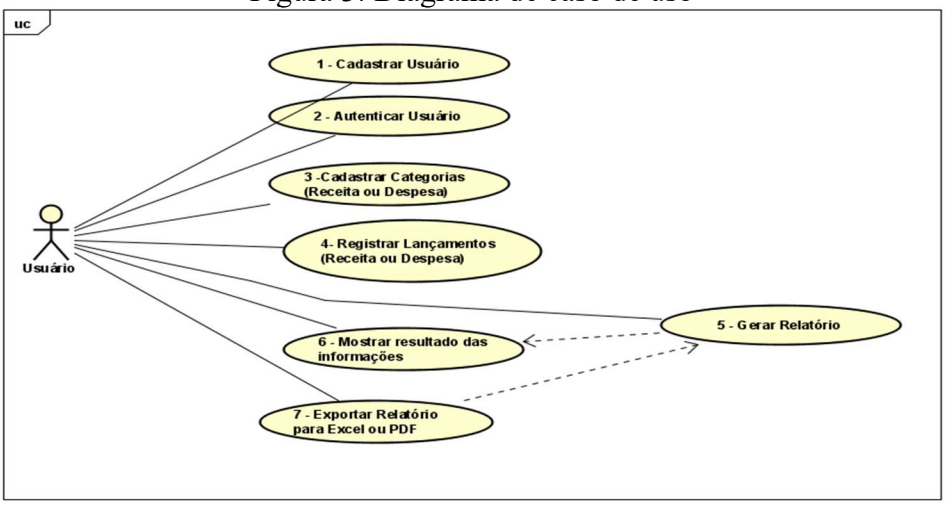
Fonte: Elaborado pelo autor

3.1.3. Diagrama de Caso de Uso

Os casos de uso permitem identificar as interações entre o usuário e o sistema, permitindo verificar quais são os atores envolvidos no processo. Ele representa as metas dos usuários assim como os procedimentos necessários para satisfazer essas metas. É demonstrado através de um diagrama e representa possíveis utilizações do sistema que pode ser uma pessoa ou subsistema.

Na figura 3 é demonstrado o funcionamento básico do sistema através de um caso de uso, onde o usuário é o ator que se relaciona com os sete casos de uso necessários para o funcionamento da ferramenta.

Figura 3: Diagrama de caso de uso



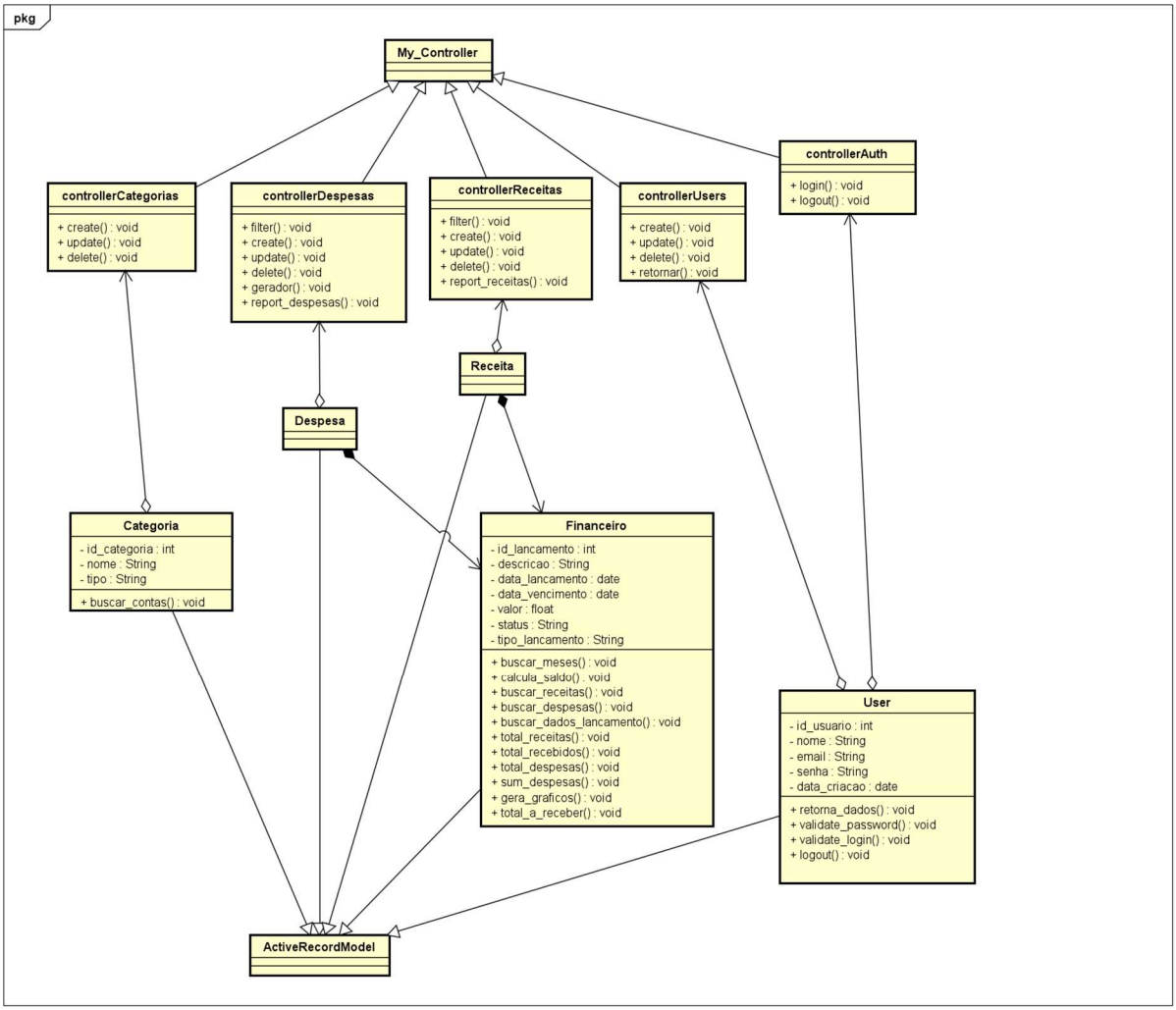
Fonte: Elaborado pelo autor

3.1.4. Diagrama de Classes

Conforme exposto por NUNES e O'NEILL (2004), o diagrama de classes descreve a estrutura dos objetos em um sistema, onde o objeto é caracterizado por um conjunto de propriedades, um comportamento e identidade. As propriedades dizem respeito as características que definem o objeto, transportado para vários atributos, onde os valores estabelecem a situação do objeto. O comportamento é caracterizado pelas operações que o objeto pode realizar. A identidade identifica um objeto em particular como um único semelhante. A classe em si, representa uma generalização sobre um conjunto de objetos que dividem o mesmo comportamento.

A figura 4 demonstra o diagrama de classe em que se baseou o desenvolvimento do sistema deste presente trabalho.

Figura 4: Diagrama de classes



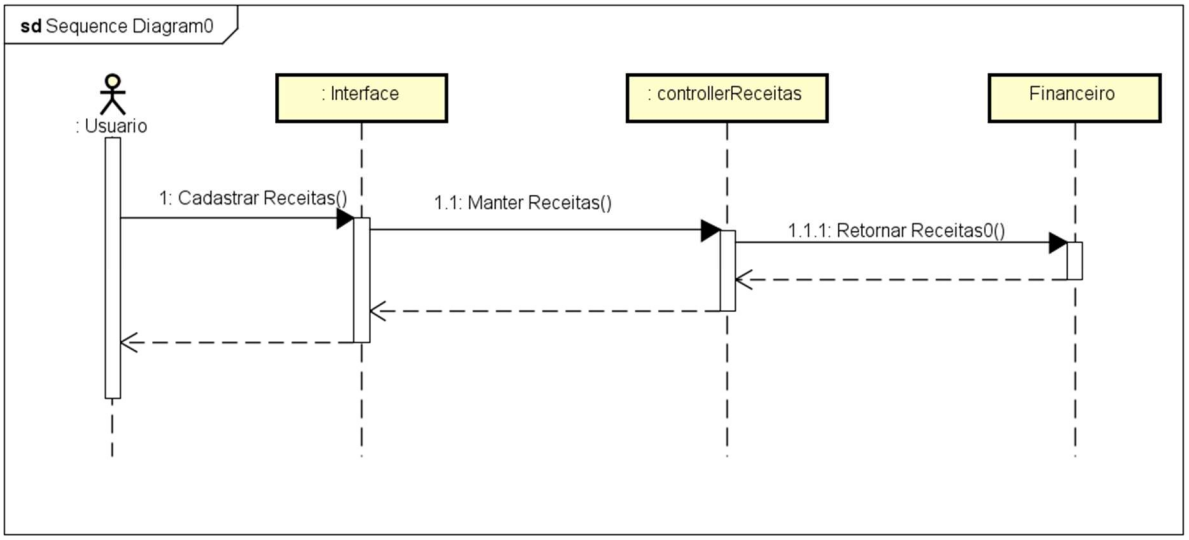
Fonte: Elaborado pelo autor

3.1.5. Diagrama de Sequência

Segundo NUNES e O'NEILL (2004) um diagrama de sequência demonstra as interações entre objetos a partir do encadeamento temporal das mensagens. O diagrama mostra uma interação (uma sequência de mensagens trocadas entre vários objetos do sistema). Nele você deve descrever uma sequência particular de funcionamento.

A figura 5, mostra um exemplo do processo de interação do usuário com o cadastramento e armazenamento de receita, em que é semelhante ao cadastro de despesas e categorias.

Figura 5: Diagrama de sequência



Fonte: Elaborado pelo autor

3.1.6. Modelo Lógico de Banco de Dados

Através da ferramenta Mysql WorkBench, foi realizada a modelagem do banco de dados onde criou-se as tabelas necessárias para armazenamento dos dados. O banco de dados do controle financeiro contemplará as tabelas categorias, financeiros, users e core.

A tabela *users* é utilizada para o armazenamento dos usuários cadastrados no sistema. Nesta tabela contém as informações que identificam o usuário, como também dados para a realização do login.

Na entidade *categorias,* são armazenadas as categorias de receita e despesa cadastradas pelo usuário. Nessa tabela existe um atributo de nome *codusu* que é gravado a identificação do usuário que cadastrou a informação, e esta é a chave estrangeira que relaciona com a tabela *users*

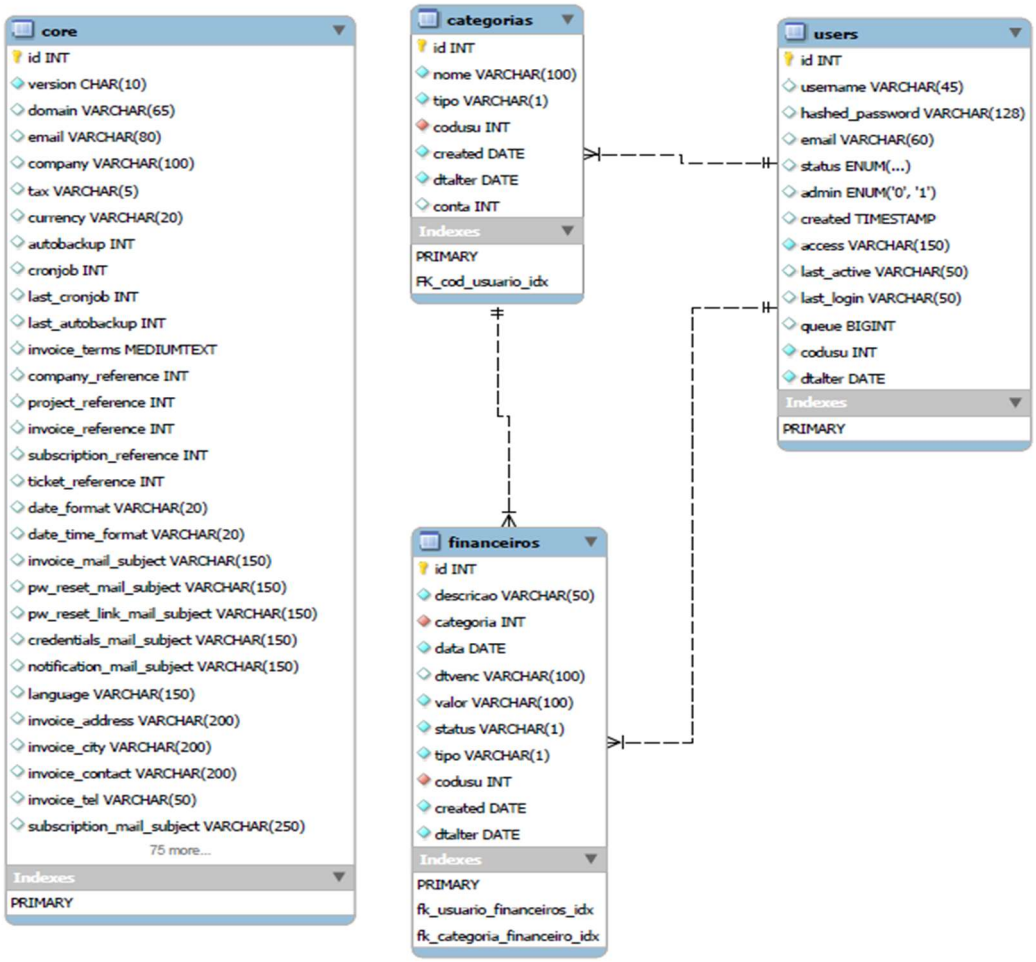
A tabela *financeiros* armazena todas as movimentações financeiras do usuário. Essa entidade contém chaves estrangeiras referenciando a tabela usuário, para que seja possível verificar o responsável pelo cadastro da informação e também referenciando a tabela categoria, para caracterizar qual categoria pertence ao lançamento.

As informações da tabela *core* são as configurações do sistema core, o núcleo do Codeigniter.

Para controlar as modificações realizadas em cada tabela, criou-se duas colunas do tipo date para armazenar as datas de alterações das informações e também da criação da tabela.

A figura 6 ilustra um modelo lógico do desenvolvimento do banco de dados.

Figura 6: Modelo lógico do banco de dados

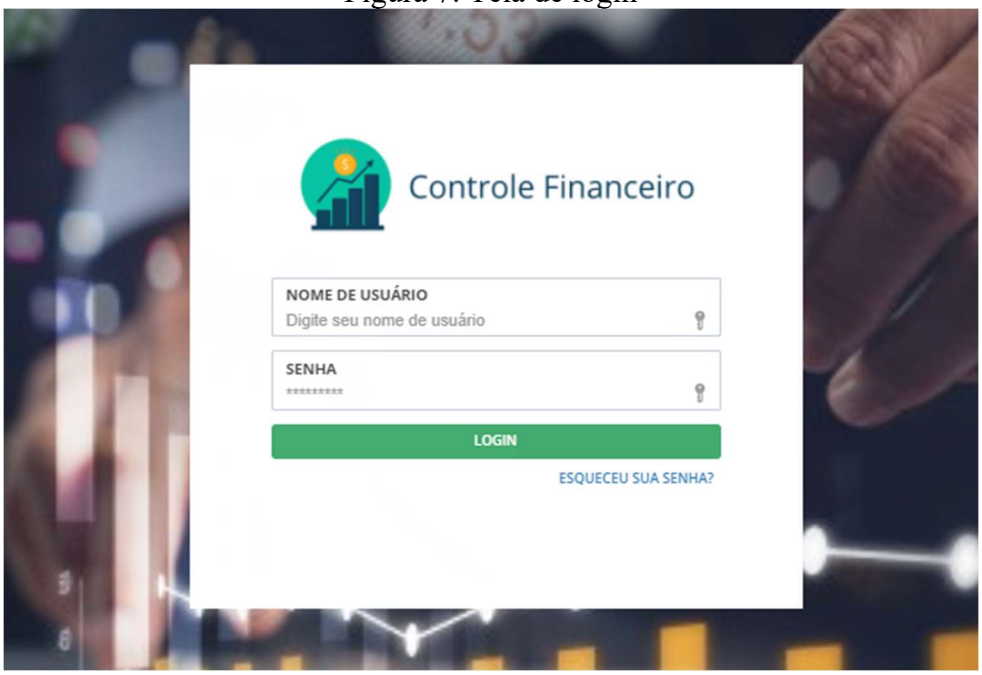


Fonte: Elaborado pelo autor

3.2. Desenvolvimento da interface

A interface é o meio de interação entre o usuário e o sistema. A figura 7 demonstra a tela de login que tem a funcionalidade de liberar acesso ao usuário do sistema, com verificação de login e controle por senha.

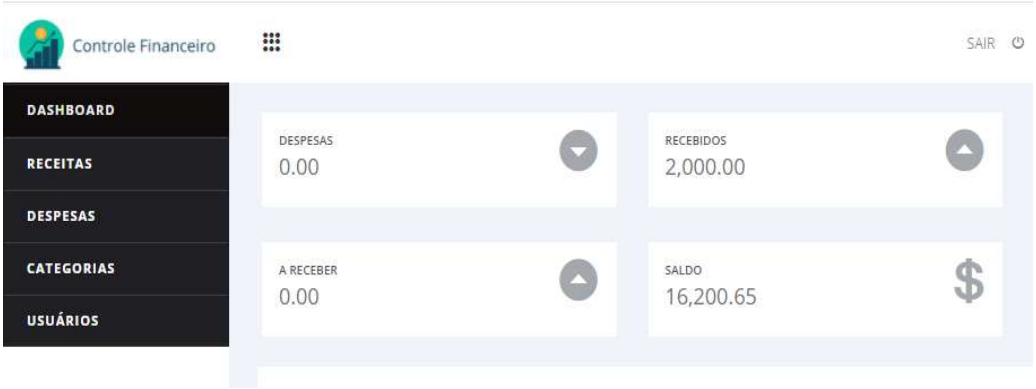
Figura 7: Tela de login



Fonte: Sistema de controle financeiro

Após a tela de login, foi desenvolvido a tela principal, conforme figura 8, no que o usuário irá interagir com o menu lateral conforme sua necessidade, pois é nessa interface que ele irá acessar as demais funcionalidades do sistema. O usuário poderá acessar o dashboard, cadastro de receitas, despesas e categorias, visualização de relatórios como as receitas e despesas mensais.

Figura 8: Tela principal



Fonte: Sistema de controle financeiro

A proposta de um Dashboard é melhorar a visualização dos dados financeiros do usuário de modo a permitir uma melhor análise e planejamento do ano vigente, e por isso foi

desenvolvido a tela de Dashboard, de acordo com a figura 9. Nessa tela foi utilizado gráficos de linhas para demonstrar a evolução dos lançamentos, seja receita ou despesa.

No gráfico “Receitas/Despesas”, é realizado um comparativo no qual pode-se observar se a despesa está maior ou menor com relação a receita de determinado mês.

Figura 9: Tela de dashboard

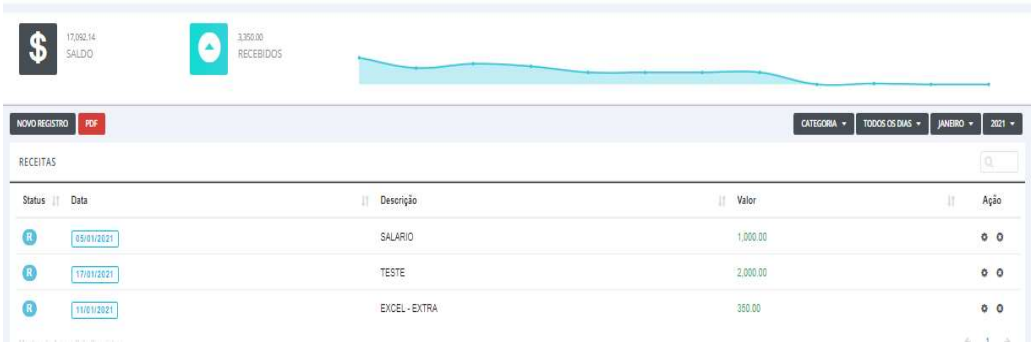
Fonte: Sistema de controle financeiro

Na figura 10, é ilustrado a interface de receitas, onde é possível visualizar os lançamentos e realizar filtros de datas conforme o período desejado. As modificações e exclusões dos lançamentos realizados são feitas na coluna “Ação” e a situação da receita pode ser visualizada na coluna “Status”, sendo “R” para recebido e “P” para pendente.

O usuário poderá cadastrar novas receitas clicando em “Novo Registro”, e conforme demonstra na figura 11, deverá inserir a data de lançamento, valor, status, descrição, e categoria para poder identificar do que se trata a informação.

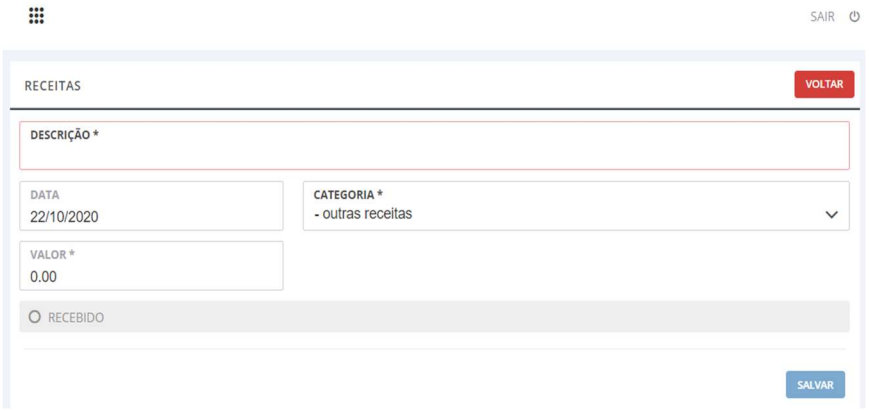
A importação das informações em formato PDF é realizada pelo botão “PDF” e essa função foi desenvolvida por meio da ferramenta JasperReport.

Figura 10: Tela de informações de receitas



Fonte: Sistema de controle financeiro

Figura 11: Tela de lançamento de receitas



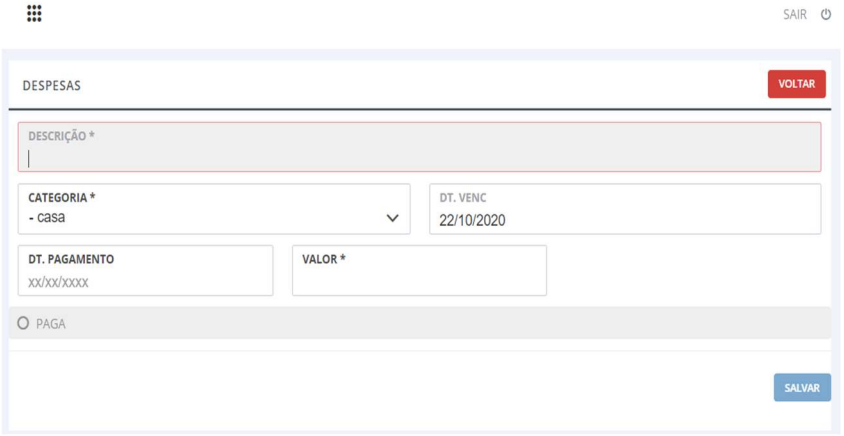
Fonte: Sistema de controle financeiro

A figura 12 mostra a tela de despesas, onde pode-se visualizar as informações cadastradas pelo usuário, sendo possível realizar filtros por período e categoria de despesas. Através do botão “Novo registro”, é permitido cadastrar novas despesas, inserindo a descrição, valor, categoria, status, data de lançamento e data de vencimento, conforme figura 13. As alterações das informações também são realizadas na coluna “Ação” e a situação do lançamento pode ser analisada na coluna “status”, sendo “P” para “Pago” e “A” para “Aguardando Pagamento”. Nesta interface também é possível verificar as informações cadastradas e importar para arquivo em formato PDF.

Figura 12: Tela de informações de despesas

Fonte: Sistema de controle financeiro

Figura 13: Tela de lançamento de despesas



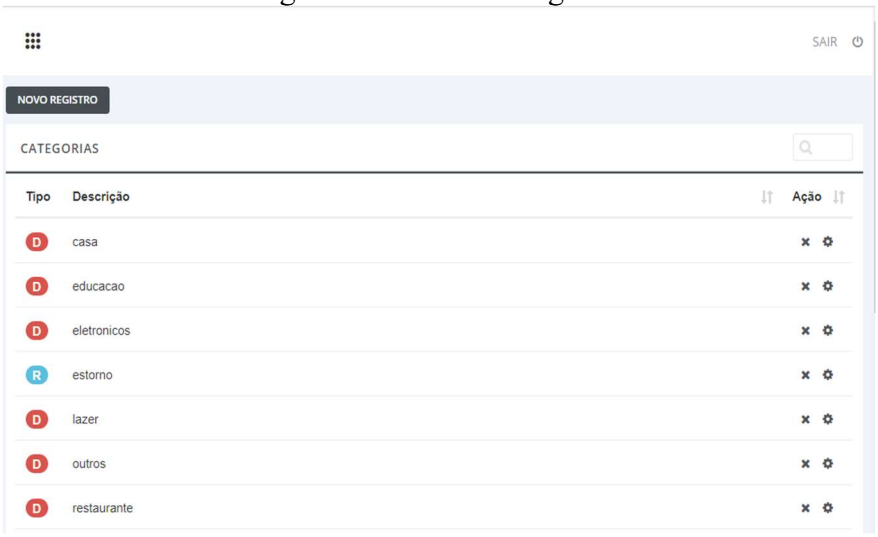
Fonte: Sistema de controle financeiro

A tela de categorias, de acordo com figura 14, foi idealizada para cadastrar as informações que dividem os tipos de lançamentos, sendo estes separados em classes conforme a natureza do lançamento: mensalidade, lazer, imposto e etc.

Nesta tela também é possível visualizar as categorias cadastradas, como também cadastrar, editar e excluir as informações. O cadastro é realizado pelo botão “Novo registro” e as modificações podem ser realizadas na coluna ação, através dos ícones editar e excluir.

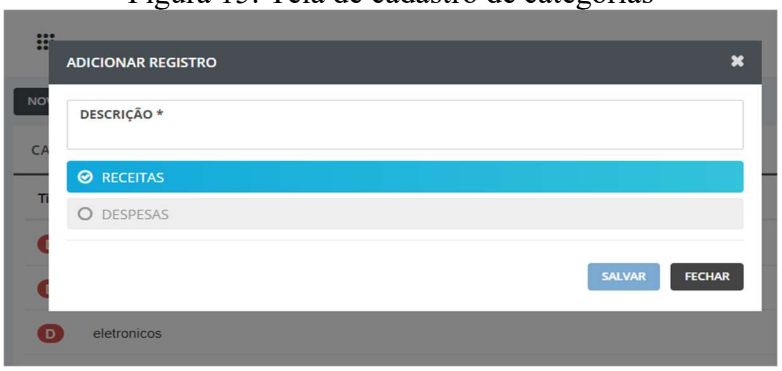
A figura 15 ilustra a tela de cadastro de categoria, onde o usuário precisa inserir a descrição da categoria e identificar se a categoria se refere a receita ou despesa.

Figura 14: Tela de categorias



Fonte: Sistema de controle financeiro

Figura 15: Tela de cadastro de categorias

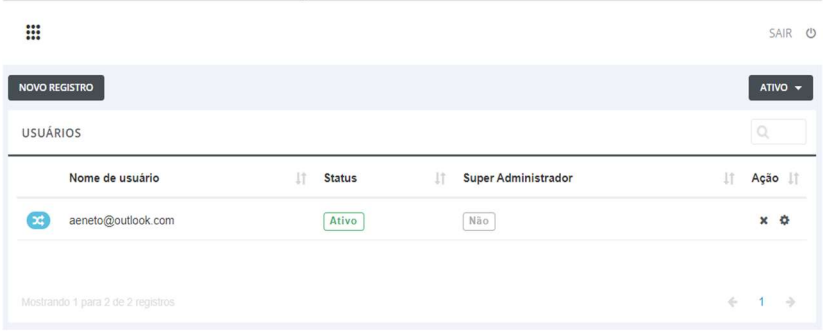


Fonte: Sistema de controle financeiro

A tela de usuário contém as informações dos usuários cadastrados no sistema, conforme demonstrado na figura 16. Nesta tela o usuário poderá visualizar, editar, excluir e cadastrar novos usuários.

De acordo com a figura 17, para o cadastro de usuário deverá ser inserido o nome de login e a senha, que deverá ser informada duas vezes para critério de validação. Caso o usuário necessite modificar a senha ou excluir algum outro usuário, poderá ser realizado na coluna “Ação”, através dos ícones alterar e excluir.

Figura 16: Tela de usuário



Fonte: Sistema de controle financeiro

Figura 17: Tela de cadastro de usuário

Fonte: Sistema de controle financeiro

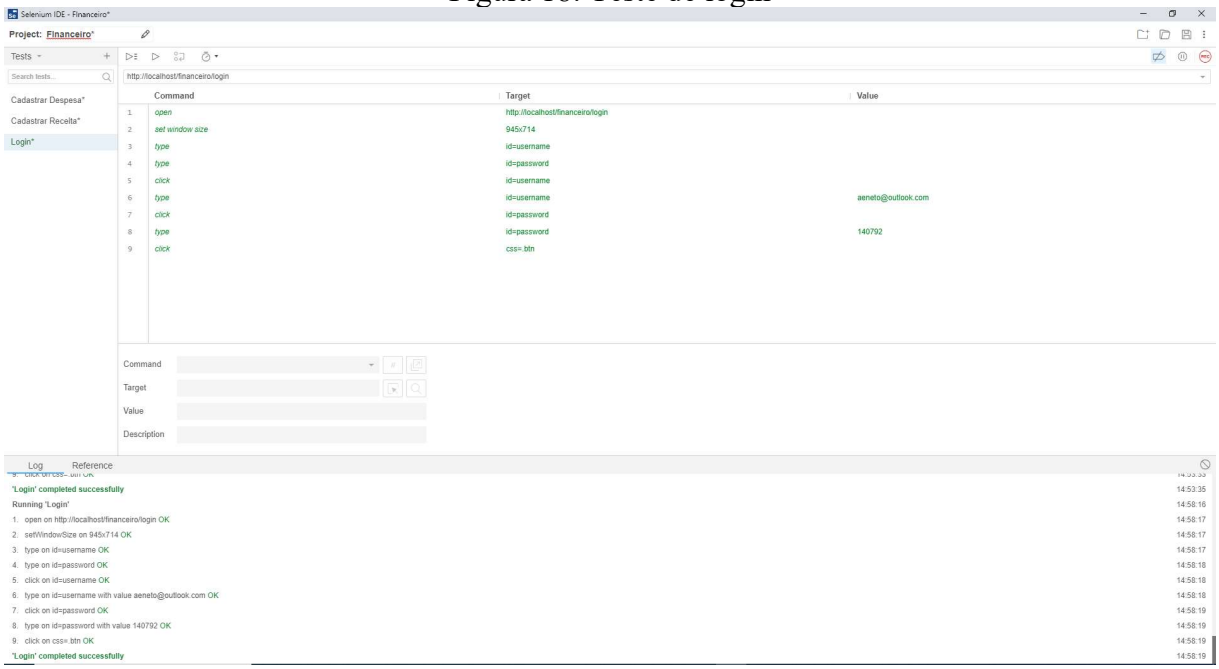
3.3. Testes e demonstrações

A criação de modelos constitui-se em etapas detalhadas que mostram todos os fatos verificados no problema, apresentando informações importantes que facilitam a comunicação entre os desenvolvedores, o que consequentemente ajuda na construção do sistema. Também se faz necessário demonstrar o resultado proposto neste presente trabalho.

O teste foi realizado através da extensão Selenium IDE, que é uma ferramenta utilizada para testar aplicações web pelo browser de forma automatizada. A aplicação grava as informações digitadas pelo usuário e repete a situação retornando o resultado do teste realizado.

O primeiro teste feito foi a tentativa de logar na plataforma, de modo que o Selenium grava as credenciais e o processo para ser realizado o login. O teste retornou com sucesso conforme demonstrado na figura 18:

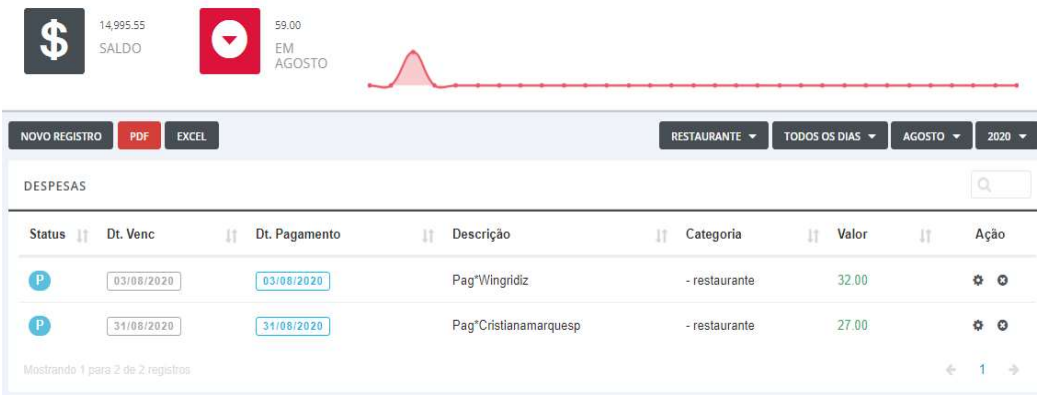
Figura 18: Teste de login



Fonte: Sistema de teste Selenium IDE

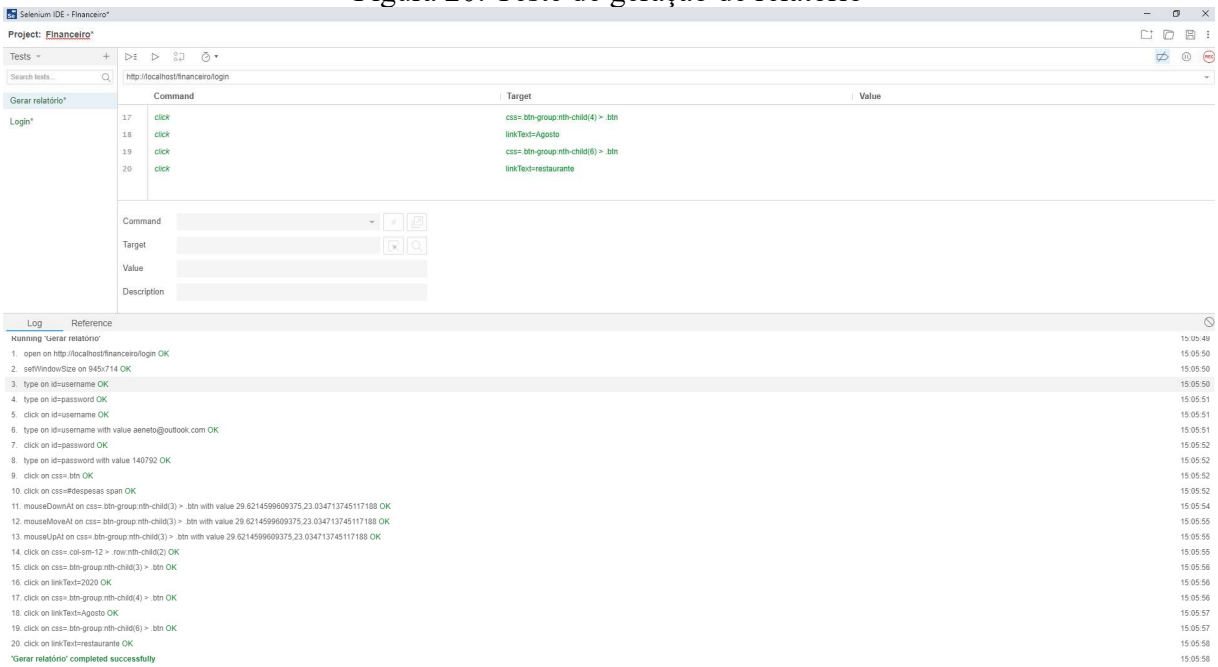
No exemplo abaixo, podemos considerar que foi necessário gerar um relatório de despesas na categoria restaurante, em um determinado mês de 2020, o que resultou em uma despesa de R$59,00 no total. A figura 19, mostra o relatório gerado no sistema. A figura 20 demonstra o teste que identifica todo o processo para gerar o relatório e retorna à comprovação que o sistema consegue gerar as informações sem erros.

Figura 19: Demonstração de relatório de despesas



Fonte: Sistema de controle financeiro

Figura 20: Teste de geração de relatório



Fonte: Sistema de teste Selenium IDE

3. CONCLUSÃO E TRABALHOS FUTUROS

O objetivo do presente trabalho foi desenvolver um sistema para o controle de finanças pessoais, apresentando de forma simples que é possível gerenciar as despesas e receitas de uma pessoa, através de uma plataforma Web. Este objetivo foi atingido considerando que o software é capaz de armazenar as informações financeiras em um banco de dados, sendo possível gerar relatório e resumos da vida financeira do usuário, o que facilita na gestão financeira pessoal.

As ferramentas bem como a metodologia utilizada no desenvolvimento deste trabalho atenderam às expectativas, visto que auxiliaram na redução no tempo de execução das atividades de forma prática e competente.

Esse trabalho foi desenvolvido para trazer praticidade ao usuário e mesmo assim existem melhorias que podem ser aplicadas para aperfeiçoar o uso do sistema. Uma das melhorias, que poderia ser pensada para trabalhos futuros, seria a divisão de parcelas de forma automática e outra função seria a integração dos lançamentos de cartão de crédito com a ferramenta deste trabalho.

REFERÊNCIAS

CARVALHO, B. V.; MELLO, C. H. Aplicação do método ágil scrum no desenvolvimento de produtos de software em uma pequena empresa de base tecnológica. **Gest. Prod**., São Carlos, v. 19, n. 3, p. 557-573, 2012.

CERBASI, G. **Casais inteligentes enriquecem juntos: finanças para casais**. [*S.l.*]: Sextante, 2014.

CERBASI, G. **Como Organizar Sua Vida Financeira**. [*S.l.*]: Sextante / Gmt, 2015.

CODEIGNITER 4. Codeigniter 4 user guide. Disponível em: https://codeigniter4.github.io/userguide/intro/index.html. Acesso em: 17 nov. 2020.

DEBONI, J. E. **Modelagem orientada a objetos com a UML:** técnicas de análise, documentação e projeto de sistemas. São Paulo: Futura, 2003.

GALLOIS, F. **Curso Básico de HTML**. Joinville: Creative Commons, 2008.

MACORATTI, José Carlos. **Padrões de projeto**: design patterns. 2002. Disponível em: http://www.macoratti.net/vb_pd1.htm Acesso em: 20 de outubro de 2020.

MASSARO, A. **Como cuidar de suas finanças pessoais**. [*S.l.*]: Conselho Federal de Administração, 2015.

MATTOS, E. **Programação de Softwares em Java**. [*S.l.*]: Universo dos Livros Editora LTDA, 2007.

MAZZOLA, V. B. **Engenharia de software:** conceitos básicos. Santa Catarina: UFSCAR. 2010

MEDEIROS, L. F. de. **Banco de Dados: princípios e prática**. [*S.l*].: [*s.n*.] 2007.

MOBILLS. **Mobills Labs**. 2020. Disponível em: https://www.mobills.com.br/. Acesso em: 01 out. 2020.

MMySQL Team (2010), **Manual de Referência do MySQL 4.1**. Disponível em: https://downloads.mysql.com/docs/refman-4.1-pt.a4.pdf. Acesso em: 26 de maio de 2020.

NUNES, M; O'NEILL, H. **Fundamental de UML.** Lisboa: FCA Editora de Informática, 2004.

OLIVEIRA, C.H.P. **SQL:** curso prático**.** São Paulo: Novatec Editoria Ltda, 2002

ORGANIZZE. **Organizze 2020**. Disponível em: www.organizze.com.br. Acesso em: 01 out. 2020.

PADRÕES DE PROJETO: O modelo MVC - Model View Controller. Disponível em: http://www.macoratti.net/vbn_mvc.htm. Acesso em: 17 nov. 2020.

PADUA, W. F. **Engenharia de Software**: introdução e uma visão do processo de software. Rio de Janeiro: LTC, 2020.

PAULA, F.; PÁDUA, W. **Engenharia de software**: fundamentos, métodos e padrões. Rio de Janeiro: LTC, 2008.

MANUAL DO PHP. PHP Documentation Group. PHP 2020. Disponível em: https://www.php.net/manual/pt_BR/preface.php. Acesso em: 01 jun. 2020.

PRESSMAN, R. S. **Engenharia de software**. 6. ed. São Paulo: Mc Graw Hill, 2006.

RAMOS, H.; BRASCHER, M. Aplicação da descoberta de conhecimento em textos para apoio à construção de indicadores informáticos para a área de C&T. **Ci. Inf.**, Brasília, v. 38, n. 2, ago. 2009.

SCHWABER, K.; SUTHERLAND, J. **Um guia definitivo para o Scrum:** as regras do jogo. Los Angeles, Creative Commons Corporation, 2013.

SILVA, Maria de Lourdes. **Contabilidade pessoal:** uma proposta para a contabilização do patrimônio das pessoas. 2007. Trabalho de Conclusão de Curso (Graduação em Ciências Contábeis) - Universidade Federal de Santa Catarina, Centro Sócio-Econômico. Curso de Ciências Contábeis, Florianópolis, 2007.

SOMMERVILLE, I. **Engenharia de software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

THOMSON, L.; WELLING, L. **PHP e MySQL Desenvolvimento Web**. 3. ed. Rio de Janeiro: Elsevier Editora LTDA, 2005.

TOP 5 aplicativos para organizar suas finanças. Oficial da net 2018. Disponível em: https://www.oficinadanet.com.br/apps/21642-top-5-aplicativos-para-organizar-suas-financas. Acesso em: 01 out. 2020.

TUOMI, I. Data is more than knowledge: implications of the reversed knowledge hierarchy for knowledge management and organization memory. **Journal of Management Information Systems**, California, v. 16, n. 3, p. 103- 117, Winter, 1999.

SPC Brasil. **48% dos brasileiros não controlam o próprio orçamento, revela pesquisa CNDL/SPC Brasil.** Disponível em: https://www.spcbrasil.org.br/imprensa/noticia/7176. Acesso em: 01 out. 2020.

APÊNDICE A

Modelo Físico de Banco de Dados - SQL

```
CREATE DATABASE `financeiro`
USE `financeiro`;

DROP TABLE IF EXISTS `categorias`;

CREATE TABLE `categorias` (
  `id` int NOT NULL AUTO_INCREMENT,
  `nome` varchar(100) NOT NULL,
  `tipo` varchar(1) NOT NULL,
  `codusu` int NOT NULL,
  `created` date NOT NULL,
  `dtalter` date NOT NULL,
  `conta` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `FK_cod_usuario_idx` (`codusu`),
  CONSTRAINT `FK_cod_usuario` FOREIGN KEY (`codusu`) REFERENCES
`users` (`id`)
)
CREATE TABLE `financeiros` (
  `id` int NOT NULL AUTO_INCREMENT,
  `descricao` varchar(50) NOT NULL,
  `categoria` int NOT NULL,
  `data` date NOT NULL,
  `dtvenc` varchar(100) DEFAULT NULL,
  `valor` varchar(100) NOT NULL,
  `status` varchar(1) NOT NULL,
  `tipo` varchar(1) NOT NULL,
  `codusu` int NOT NULL,
  `created` date NOT NULL,
  `dtalter` date NOT NULL,
  PRIMARY KEY (`id`),
```

```
  KEY `fk_usuario_financeiros_idx` (`codusu`),
  KEY `fk_categoria_financeiro_idx` (`categoria`),
  CONSTRAINT `fk_categoria_financeiro` FOREIGN KEY (`categoria`) REFERENCES `categorias` (`id`),
  CONSTRAINT `fk_usuario_financeiros` FOREIGN KEY (`codusu`) REFERENCES `users` (`id`)
)

CREATE TABLE `users` (
  `id` int NOT NULL AUTO_INCREMENT,
  `username` varchar(45) DEFAULT NULL,
  `hashed_password` varchar(128) DEFAULT NULL,
  `email` varchar(60) DEFAULT NULL,
  `status` enum('active','inactive','deleted') DEFAULT NULL,
  `admin` enum('0','1') DEFAULT NULL,
  `created` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `access` varchar(150) NOT NULL DEFAULT '1,2',
  `last_active` varchar(50) DEFAULT NULL,
  `last_login` varchar(50) DEFAULT NULL,
  `queue` bigint DEFAULT '0',
  `codusu` int NOT NULL,
  `dtalter` date NOT NULL,
  PRIMARY KEY (`id`)
)
CREATE TABLE `core` (
  `id` int NOT NULL,
  `version` char(10) NOT NULL DEFAULT '0',
  `domain` varchar(65) DEFAULT NULL,
  `email` varchar(80) DEFAULT NULL,
  `company` varchar(100) DEFAULT NULL,
  `tax` varchar(5) DEFAULT NULL,
  `currency` varchar(20) DEFAULT NULL,
  `autobackup` int DEFAULT NULL,
```

```
`cronjob` int DEFAULT NULL,
`last_cronjob` int DEFAULT NULL,
`last_autobackup` int DEFAULT NULL,
`invoice_terms` mediumtext,
`company_reference` int DEFAULT NULL,
`project_reference` int DEFAULT NULL,
`invoice_reference` int DEFAULT NULL,
`subscription_reference` int DEFAULT NULL,
`ticket_reference` int DEFAULT NULL,
`date_format` varchar(20) DEFAULT NULL,
`date_time_format` varchar(20) DEFAULT NULL,
`invoice_mail_subject` varchar(150) DEFAULT NULL,
`pw_reset_mail_subject` varchar(150) DEFAULT NULL,
`pw_reset_link_mail_subject` varchar(150) DEFAULT NULL,
`credentials_mail_subject` varchar(150) DEFAULT NULL,
`notification_mail_subject` varchar(150) DEFAULT NULL,
`language` varchar(150) DEFAULT NULL,
`invoice_address` varchar(200) DEFAULT NULL,
`invoice_city` varchar(200) DEFAULT NULL,
`invoice_contact` varchar(200) DEFAULT NULL,
`invoice_tel` varchar(50) DEFAULT NULL,
`subscription_mail_subject` varchar(250) DEFAULT NULL,
`logo` varchar(150) DEFAULT NULL,
`template` varchar(200) DEFAULT 'blueline',
`paypal` varchar(5) DEFAULT '1',
`paypal_currency` varchar(200) DEFAULT 'EUR',
`paypal_account` varchar(200) DEFAULT '',
`invoice_logo` varchar(150) DEFAULT 'assets/blueline/img/invoice_logo.png',
`pc` varchar(150) DEFAULT NULL,
`vat` varchar(150) DEFAULT NULL,
`ticket_email` varchar(250) DEFAULT NULL,
`ticket_default_owner` int DEFAULT '1',
`ticket_default_queue` int DEFAULT '1',
```

```
`ticket_default_type` int DEFAULT '1',
`ticket_default_status` varchar(200) DEFAULT 'new',
`ticket_config_host` varchar(250) DEFAULT NULL,
`ticket_config_login` varchar(250) DEFAULT NULL,
`ticket_config_pass` varchar(250) DEFAULT NULL,
`ticket_config_port` varchar(250) DEFAULT NULL,
`ticket_config_ssl` varchar(250) DEFAULT NULL,
`ticket_config_email` varchar(250) DEFAULT NULL,
`ticket_config_flags` varchar(250) DEFAULT '/notls',
`ticket_config_search` varchar(250) DEFAULT 'UNSEEN',
`ticket_config_timestamp` int DEFAULT '0',
`ticket_config_mailbox` varchar(250) DEFAULT NULL,
`ticket_config_delete` int DEFAULT '0',
`ticket_config_active` int DEFAULT '0',
`ticket_config_imap` int DEFAULT '1',
`stripe` int DEFAULT '0',
`stripe_key` varchar(250) DEFAULT NULL,
`stripe_p_key` varchar(255) DEFAULT '',
`stripe_currency` varchar(255) DEFAULT 'USD',
`bank_transfer` int DEFAULT '0',
`bank_transfer_text` longtext,
`estimate_terms` longtext,
`estimate_prefix` varchar(250) DEFAULT 'EST',
`estimate_pdf_template` varchar(250) DEFAULT 'templates/estimate/blueline',
`invoice_pdf_template` varchar(250) DEFAULT 'templates/invoice/blueline',
`second_tax` varchar(5) DEFAULT '',
`estimate_mail_subject` varchar(255) DEFAULT 'New Estimate #{estimate_id}',
`money_format` int unsigned NOT NULL DEFAULT '1',
`money_currency_position` int unsigned NOT NULL DEFAULT '1',
`pdf_font` varchar(255) DEFAULT 'NotoSans',
`pdf_path` int unsigned NOT NULL DEFAULT '1',
`registration` int unsigned NOT NULL DEFAULT '0',
`authorize_api_login_id` varchar(255) DEFAULT NULL,
```

```
  `authorize_api_transaction_key` varchar(255) DEFAULT NULL,
  `authorize_net` int DEFAULT NULL,
  `authorize_currency` varchar(30) DEFAULT NULL,
  `invoice_prefix` varchar(255) NOT NULL DEFAULT '',
  `company_prefix` varchar(255) NOT NULL DEFAULT '',
  `quotation_prefix` varchar(255) NOT NULL DEFAULT '',
  `project_prefix` varchar(255) NOT NULL DEFAULT '',
  `subscription_prefix` varchar(255) NOT NULL DEFAULT '',
  `calendar_google_api_key` varchar(255) DEFAULT NULL,
  `calendar_google_event_address` varchar(255) DEFAULT NULL,
  `default_client_modules` varchar(255) DEFAULT NULL,
  `estimate_reference` int NOT NULL,
  `login_background` varchar(255) DEFAULT 'blur.jpg',
  `login_logo` varchar(255) DEFAULT '',
  `login_style` varchar(255) DEFAULT 'left',
  `custom_colors` int DEFAULT '1',
  `top_bar_background` varchar(60) DEFAULT '#FFFFFF',
  `top_bar_color` varchar(60) DEFAULT '#333333',
  `body_background` varchar(60) DEFAULT '#D8DCE3',
  `menu_background` varchar(60) DEFAULT '#2c3e4d',
  `menu_color` varchar(60) DEFAULT '#FFFFFF',
  `cor_texto_menu` varchar(20) NOT NULL,
  `cor_hover_menu` varchar(20) NOT NULL,
  `cor_sub_menu` varchar(20) NOT NULL,
  `cor_hover_submenu` varchar(20) NOT NULL,
  `primary_color` varchar(60) DEFAULT '#28a9f1',
  `twocheckout_publishable_key` varchar(250) DEFAULT '',
  `twocheckout_private_key` varchar(250) DEFAULT '',
  `twocheckout_seller_id` varchar(250) DEFAULT '',
  `twocheckout` int DEFAULT '0',
  `twocheckout_currency` varchar(250) DEFAULT 'USD',
  PRIMARY KEY (`id`)
)
```