UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

PLINIO LEME DA SILVA CAMPINAS

SISTEMAS DE GESTÃO DE PESSOAS APLICADO À PEQUENAS ORGANIZAÇÕES

BAURU
Novembro/2019

PLINIO LEME DA SILVA CAMPINAS

SISTEMAS DE GESTÃO DE PESSOAS APLICADO À PEQUENAS ORGANIZAÇÕES

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru, como requisito à obtenção do título de Bacharel em Sistemas de Informação.
Orientador: Prof. Dr. Eduardo Martins Morgado

BAURU
Novembro/2019

PLINIO LEME DA SILVA CAMPINAS

SISTEMAS DE GESTÃO DE PESSOAS APLICADO À PEQUENAS ORGANIZAÇÕES

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru, como requisito à obtenção do título de Bacharel em Sistemas de Informação.

Banca Examinadora

Prof. Dr. José Remo Ferreira Brega
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências - Unesp Bauru
Departamento de Ciência da Computação

Prof. Dr. Marcia Aparecida Zanoli Meira e Silva
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências - Unesp Bauru
Departamento de Ciência da Computação

Prof. Dr. Roberta Spolon
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências - Unesp Bauru
Departamento de Ciência da Computação

Bauru, _____ de _____ de _____.

Dedico esse trabalho à minha familia, principalmente meus pais, e à todos os meus professores.

Agradecimentos

Agradeço primeiramente aos meus pais por me darem todo o apoio que precisei e a UNESP pela experiência oferecida. Importante lembrar também de todos os meus professores, desde antes do ensino superior, no ensino técnico e médio que proporcionaram a minha atual formação.

Por último agradeço também a empresa Junior de Computação pela grande contribuição profissional e pessoal que o projeto de extensão me proporcionou.

Se espera-se que uma maquina seja infalivel, ela não poderá ser também inteligente.

Alan Turing

Resumo

O projeto realizado é um sistema WEB com a finalidade de facilitar atividades rotineiras de gestão de pessoas e recursos humanos em organizações pequenas. Além disso também visa demonstrar o uso de algumas das principais ferramentas utilizadas no desenvolvimento WEB atualmente. Durante a rotina diária de gerir pessoas, e informações sobre elas, algumas atividades podem custar muito tempo para o gestor responsável. Em uma organização pequena, o gestor de recursos humanos pode acabar perdendo muito tempo com atividades repetitivas, ao mesmo tempo que gerencia muito mal os dados de seus membros. Além disso, pode ser dificil visualisar os dados existentes de forma a identificar possíveis problemas ou oportunidades. Com esses problemas em vista, o projeto foi executado para auxiliar o gestor em atividades rotineiras, aumentando assim a sua produtividade. Juntamente a isso, permitir ampliar sua visão sobre as pessoas e o ambiente de trabalho.
Palavras-chave: Gestão, Pessoas, RH, Angular, MongoDB.

Abstract

The project is a WEB information system, designed to help human resources managers in their daily life routine. Managing people means to perform certain tasks constantly. Inside small organizations, the manager can spend to much of his time with those tasks, while poorly managing personal data about his members. Such problems can be misleading, hiding problems and opportunities. The solution this project presents is about building an information system, in a WEB platform, to increase HR managers productivity in their daily tasks. Preventing mistakes, while opening their minds to what's happening inside.
Keywords: RH, Managing, Manager, Angular, MongoDB.

Lista de figuras

Figura 1 – Exemplo do software da Benner 1 15
Figura 2 – Exemplo do software da Benner 2 16
Figura 3 – Usuário e Privilégios 19
Figura 4 – Cadastros básicos 19
Figura 5 – Pagamentos 20
Figura 6 – Classes de Usuários e Privilégios 21
Figura 7 – Classe de Pagamentos 21
Figura 8 – Pessoas e fichas 22
Figura 9 – Sequência para cadastro de Pessoas 22
Figura 10 – Sequência para cadastro de atividades 23
Figura 11 – Sequência para autenticação de usuário 23
Figura 12 – Visão geral do sistema 24
Figura 13 – Gravando dados no banco 28
Figura 14 – Estrutura do tipo imagem64 29
Figura 15 – Editando usuário 29
Figura 16 – Rotas 30
Figura 17 – Cadastro de atividades 31
Figura 18 – Usuário por faixa de idade 32

*

Lista de tabelas

Tabela 1 – Comparando as soluções. 15
Tabela 2 – Funcionalidades Básicas. 18
Tabela 3 – Funcionalidades Avançadas. 18

Lista de abreviaturas e siglas

SAP	Systems, Applications & Products in Data Processing
ERP	Enterprise Resource Planning
ADP	Automatic Data Processing
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
ECMA	European Computer Manufacturers Association
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
API	Application Programming Interface
BD	Banco de Dados
JS	JavaScript
SQL	Structured Query Language
MEAN	Mongo, Express, Angular, Node

Sumário

1 INTRODUÇÃO 12

2 DETALHAMENTO DO PROBLEMA 13

3 SOLUÇÕES EXISTENTES 14

4 DESCRIÇÃO DO PROJETO 17
4.1 **Especificação** 17
4.1.1 Casos de uso e requisitos 17
4.1.2 Diagramas de caso de uso 18
4.1.3 Diagramas de classe de análise 20
4.1.4 Diagramas de sequência 22
4.1.5 Visão geral 23
4.2 **Tecnologias escolhidas** 24
4.2.1 Ferramentas para construção do front-end 24
4.2.2 Ferramentas para construção do Back-end 25
4.2.3 Banco de dados 26
4.2.4 Ferramentas de apoio ao desenvolvimento 27
4.3 **Software desenvolvido** 27

5 VALIDAÇÃO 33

6 CONCLUSÃO 34

REFERÊNCIAS 35

1 INTRODUÇÃO

A gestão de pessoas impacta o desempenho de toda uma organização. A maioria dos problemas de produtividade em uma organização está relacionado ao comportamento e não a deficiências técnicas (HUSELID, 1995).

A motivação para desenvolver o sistema foi tentar facilitar a realização de um conjunto de atividades simples como contabilizar horas de trabalho e faltas fornecem uma visão geral sobre a organização. Apesar de simples essas atividades devem ser executadas constantemente e o gestor pode perder muito tempo analisando e manipulando essas informações. Tempo esse que faltará para que ele analise questões mais complexas sobre o grupo de pessoas que gerencia.

A proposta de um sistema WEB pode trazer várias vantagens e ajudar a solucionar esses problemas. Mas isso exige um software de boa qualidade, seguro e preparado para aproveitar plenamente a plataforma WEB. A usabilidade é parte importante de um sistema, para garantir que o precioso tempo dos gestores seja melhor aproveitado (FERREIRA, 2016).

O sistema foi construido em duas partes principais: um módulo básico, para cadastros e um módulo para aproveitar esses dados e construir visualizações mais trabalhadas. Apesar do nome, o módulo básico exigiu o maior cuidado, ajudando a validação das tecnologias escolhidas. Além disso as funcionalidades do módulo básico e os dados armazenados foram essenciais para implementar funções mais complexas.

O segundo módulo que este documento propôs construir foi pensado para o cálculo de métricas e visualização de dados. Calcular métricas pertinentes a gestão de pessoas e implementar formas variadas e úteis de visualizar os dados do módulo básico, como gráficos e alertas. São funcionalidades mais complexas e que dependem do bom funcionamento do módulo básico.

Tomar decisões sobre que tecnologias usar em um novo projeto Web não é simples. A indústria de software apresenta opções como nunca antes e é facil ficar perdido dentro de tantas opções de bibliotecas. As tecnologias escolhidas fazem uso do Javascript, que é a linguagem mais usada no mundo Web hoje para o desenvolvimento de aplicações. Por exemplo, uma pesquisa com 64 mil desenvolvedores realizada pelo sítio StackOverflow mostrou o JS sendo utilizado por 67% dos desenvolvedores (MENDES, 2016).

Além da implementação do sistema, o estudo das tecnologias escolhidas e da area de desenvolvimento WEB como um todo foi descrito no documento e se mostrou uma das principais vantagens da execução do projeto.

2 DETALHAMENTO DO PROBLEMA

A gestão de pessoas exige a manipulação de vários tipos de informações, sobre o pessoal a ser gerenciado e sobre a organização em si. São muitas as tarefas que cobram tempo e energia dos gestores, e até mesmo tarefas que necessitam de um cuidado especial devido a questões legais. Erros simples nestes procedimentos podem resultar em prejuízos para a organização, que podem ser desde alguns problemas financeiros, até mesmo a paralisação total de alguma atividade, devido ao não cumprimento de exigências legais (SINOPSE..., 2019).

Todos os tipos de organização necessitam de atenção para cuidar de seus recursos humanos. Mesmo empresas pequenas, que não tenham um profissional específico para isso, precisam cumprir uma série de exigências que, além de consumir tempo dos gestores, quando mal executadas podem levar a problemas de desempenho e até prejuízos diretos à saúde de funcionários e ao caixa da empresa (BITTENCOURT J. A.; MATSUMOTO, 2017).

Em qualquer tipo de organização todos podem ser impactados pelas atividades de gestão de pessoas, seja por questões na própria organização de que faz parte ou com uma outra organização relacionada.

Falhas na contabilização de horas trabalhadas, vazamento de informações pessoais sensíveis e problemas de segurança do trabalho. São exemplos que podem resultar em processos muito custosos para uma empresa. O processo de seleção de novos empregados também pode ter um impacto grande, uma série de contratações equivocadas podem gerar prejuízo por muito tempo (BASSO, 2017).

Para que o setor de recursos humanos possa desempenhar suas funções de forma adequada e segura é preciso garantir que as informações estejam facilmente acessíveis para os gestores, além de garantir a segurança das informações, impedindo que certas pessoas tenham acesso a algumas informações (UAC, 2019).

Além das questões levantadas anteriormente, o desenvolvimento do projeto também cumpre o papel de explorar algumas das tecnologias mais populares atualmente para o desenvolvimento WEB. Para construir o sistema proposto é importante contar com tecnologias escaláveis e com amplo suporte para o desenvolvimento.

3 SOLUÇÕES EXISTENTES

O sistema SAP(*Systems, Applications & Products*) e outros ERPs(*Enterprise Resource Planning*) implementam funções relativas a gestão de recursos humanos. São softwares grandes que não tem foco na gestão de pessoas. Existem softwares específicos para recursos humanos, feitos dentro e fora do Brasil, o que pode ser um fator importante. A empresa ADP (ADP, 2019), sigla para Automatic Data Processing, se destaca como uma empresa com soluções pensadas especificamente para a gestão de recursos humanos. Um exemplo é o ADP Workforce Now, que consiste em uma solução especializada em pagamentos e funcionalidades relacionadas, inclui cálculo de impostos e outras taxas. São chamados software de Payrolls ou de folha de pagamento. Segue uma referência ao *buyers guide* da software advice (ADVICE, 2019a) sobre soluções para pagamentos.

> Every company must process payroll for its employees. As a company grows, a reliable system that processes payments quickly and easily becomes critical. In order to do this, companies have two options: outsourcing to a provider, or keeping payroll in-house. Cost and time savings are the main draw when outsourcing. A company avoids the fixed costs of a full-time salaried employee and staff can focus on their core competencies. However, by outsourcing, companies lose control over the process, and have less flexibility. (ADVICE, 2019b)

Outro software investigado foi o Kronos (KRONOS, 2019) Workforce Ready, especializado no controle de horas, ausências e planejamento de tempo. Segue uma descrição das funções, retirada do mesmo *buyers guide*, sobre esse tipo de software: "Managing employee schedules Tracking the hours logged by each employee Monitoring overtime and absences due to vacation, sick leave and holidays" (ADVICE, 2019c)

A Benner (BENNER, 2019a), além de outras soluções, possui uma solução de RH(Recursos Humanos) no Brasil, segundo descrição dos vendedores, tem entre suas funcionalidades: "Folha de Pagamento, ponto eletrônico e controle de benefícios Engajamento, autoatendimento, centralização das informações de RH Recrutamento e Seleção, Competências, Feedbacks e Metas, Carreira e Sucessão e Educação corporativa e EAD" (BENNER, 2019b)

Os softwares modernos e internacionais também implementam funções de análise comportamental dos funcionários mas as funções mais comuns entre as soluções investigadas são o gerenciamento de pagamentos e contabilização de horas.

Na Tabela 1 é feita uma breve comparação entre as soluções citadas anteriormente.

Todos esses sistemas são pagos e englobam um número muito grande de funcionalidades, como no caso dos ERPs. Além disso, são desenvolvidos para organizações grandes. Desses

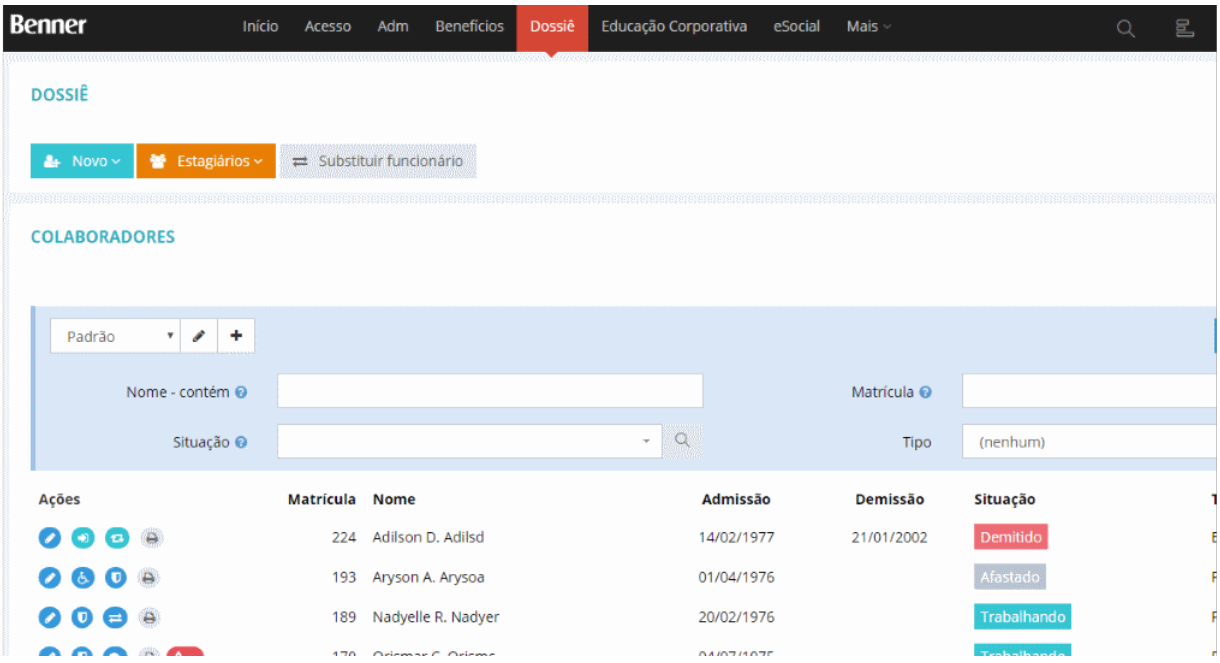
Tabela 1 – Comparando as soluções.

Solução	Vantagens	Desvantagens
SAP e ERPs	Completo e integrado	Caro e não especializado
ADP WorkForce Now	Especializado em pagamentos	Pago, muito especifico e estrangeiro
Kronos WorkForce	Especializado em controle de horas	Pago e de empresa estrangeira
Benner RH	Empresa brasileira e software completo	Pago e focado em grandes organizações

sistemas disponíveis, muitos são antigos e as soluções WEB e Mobile não estão plenamente implantadas ou não foram implementadas de forma nativa. A proposta desse documento é construir um software simples e livre, focado nas funcionalidades básicas e na visualização das informações, construído especificamente para WEB, sendo assim ideal para pequenas organizações e com pouco capital.

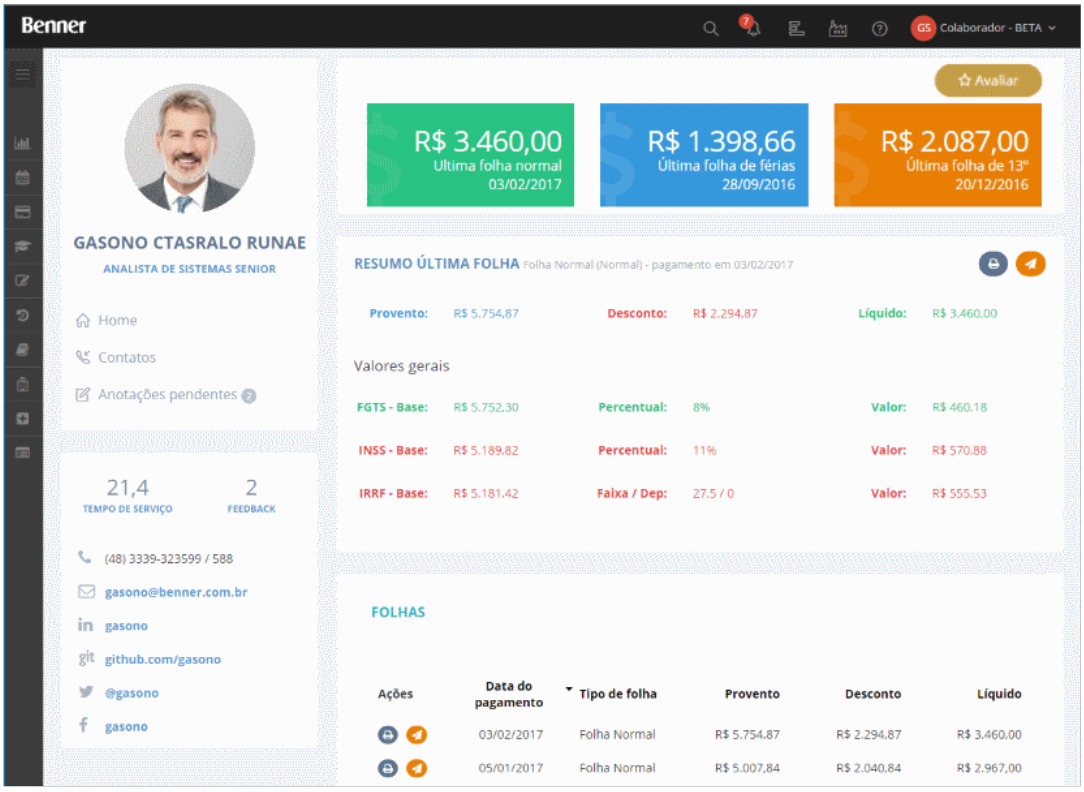
Para levantar informações para o desenvolvimento da navegação e das funcionalidades, buscou-se algumas janelas dos sistemas disponíveis. A Figura 1 exibe um cadastro de funcionários. A Figura 2 exibe o cadastro de um funcionário especifico, com dados cadastrais de contato, foto e pagamentos realizados.

Figura 1 – Exemplo do software da Benner 1



Fonte: Site da empresa.

Figura 2 – Exemplo do software da Benner 2



Fonte: Site da empresa.

4 DESCRIÇÃO DO PROJETO

Para solucionar os problemas descritos anteriormente foram levantados os requisitos e realizou-se uma modelagem inicial do sistema. Desde o início ficou claro que seria necessário implementar várias janelas diferentes, para isso pensou-se o front-end e a interface com o usuário com cuidado. Cada módulo do sistema é formado por um conjunto de janelas e classes específicas para cada funcionalidade. O primeiro módulo é responsável pelo cadastro e manutenção das informações pessoais, controle de horas simples e atividades.

A primeira parte da execução do projeto foi, portanto, escolher uma ferramenta para a construção das janelas, criar um padrão estético e de navegação, implementar um protótipo das primeiras janelas para cadastro de dados e controle de horas. Após isso, começar primeiro a testar a interface com o banco de dados, deixando tudo pronto para a conclusão do módulo básico e para o desenvolvimento dos módulos seguintes.

A conclusão do módulo básico validou as tecnologias escolhidas e abriu caminho para a construção de novas funcionalidades.

Outro módulo foi projetado para a visualização dos dados do módulo básico. Gráficos, estatísticas e métricas são o foco deste módulo, aproveitando os dados coletados por outros módulos, inicialmente, o módulo básico.

4.1 Especificação

Durante a especificação do projeto, os requisitos foram definidos inicialmente com o auxílio de casos de uso. Juntamente à isso, construiu-se diagramas de caso de uso e de classe.

4.1.1 Casos de uso e requisitos

Para selecionar os requisitos, foram descritos casos de uso. A Tabela 2 apresenta algumas funcionalidades básicas.

Tabela 2 – Funcionalidades Básicas.

Funcionalidades Básicas
Cadastrar dados pessoais presentes em um RG. Todos eles. Inclusive foto, opcionalmente.
Controlar a atividade de funcionários a partir de um cadastro de horas, com horário de entrada e saída, além das atividades realizadas, breve descrição ou atividade padronizada.
Permitir aos colaboradores visualizar as horas contabilizadas no sistema e comparar com o que é exigido.
Permitir o controle de advertências e infrações dos códigos da empresa, atraves de notificações.
Produzir um sistema com um padrão estético e de navegação adequados para a função do sistema e que garantam produtividade aos usuários do sistema.
Organizar através de um login e de níveis de privilégio, o acesso às informações do sistema. Com informações mais sensíveis disponíveis apenas para alguns gestores. Além disso, cada colaborador poderá visualizar suas próprias informações individualmente.
Todas as funcionalidades devem ser acessíveis pela WEB.

A implementação das funcionalidades avançadas e complementares sofreram mudanças durante o desenvolvimento do projeto, os caso de uso na Tabela 3 abaixo foram utilizados para a implementação do módulo avançado.

Tabela 3 – Funcionalidades Avançadas.

Funcionalidades Avançadas
Permitir visualizar, em uma ficha, informações sumarizadas de diversas fontes, como salário, férias, horas trabalhadas ausências e outras informações relevantes para a gestão de pessoas avançado.
Exportar os dados do sistema para PDF ou Planilha
Permitir ao gestor uma visão dos dados com gráficos, relatórios ou outras visualizações.
Cadastrar pagamentos, somente para consultas.
As principais funções devem estar adequadas para visualização em dispositivos móveis.

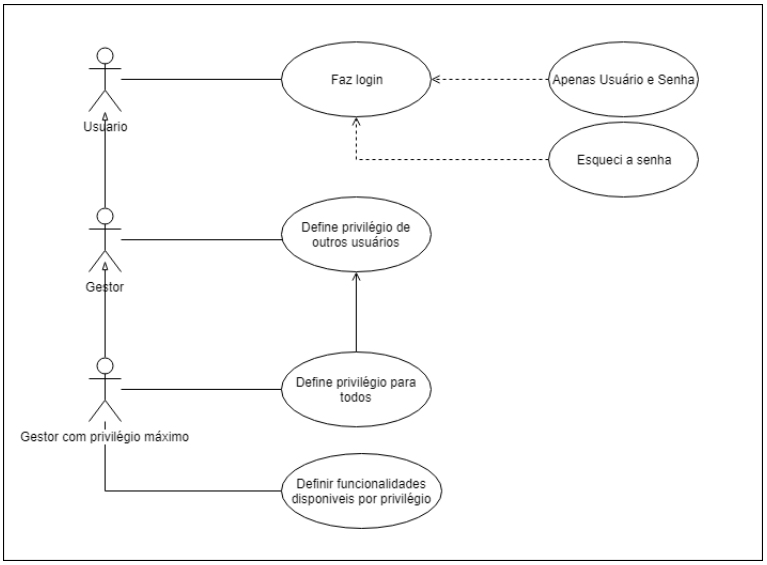
4.1.2 Diagramas de caso de uso

Após a descrição textual dos casos de uso, seguem os diagramas, produzidos a partir do texto anterior.

Na Figura 3 se ve um diagrama que modela como os cadastros de usuários devem funcionar no sistema, de acordo com o seu privilégio.

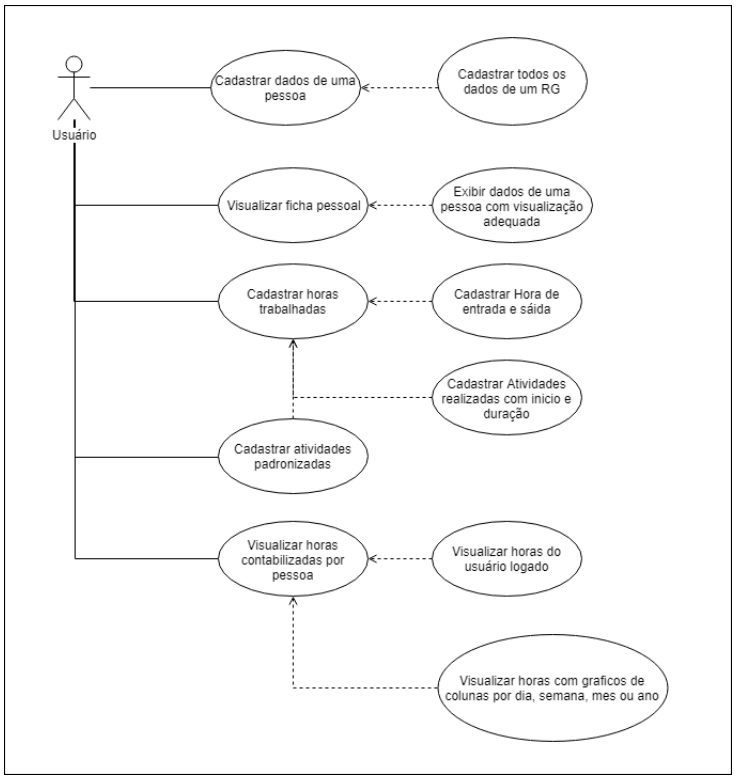
A Figura 4 mostra os principais cadastros de dados so sistema, como o de pessoa e o de atividades. Enquanto isso na Figura 5 é exibido a modelagem do cadastro de pagamentos, parte do módulo avançado.

Figura 3 – Usuário e Privilégios



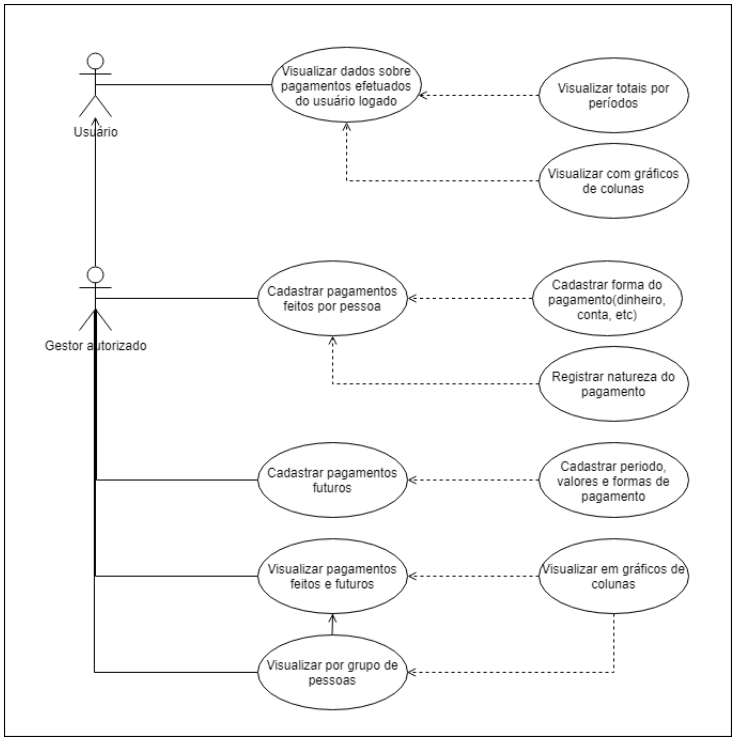
Fonte: Elaborada pelo autor.

Figura 4 – Cadastros básicos



Fonte: Elaborada pelo autor.

Figura 5 – Pagamentos



Fonte: Elaborada pelo autor.

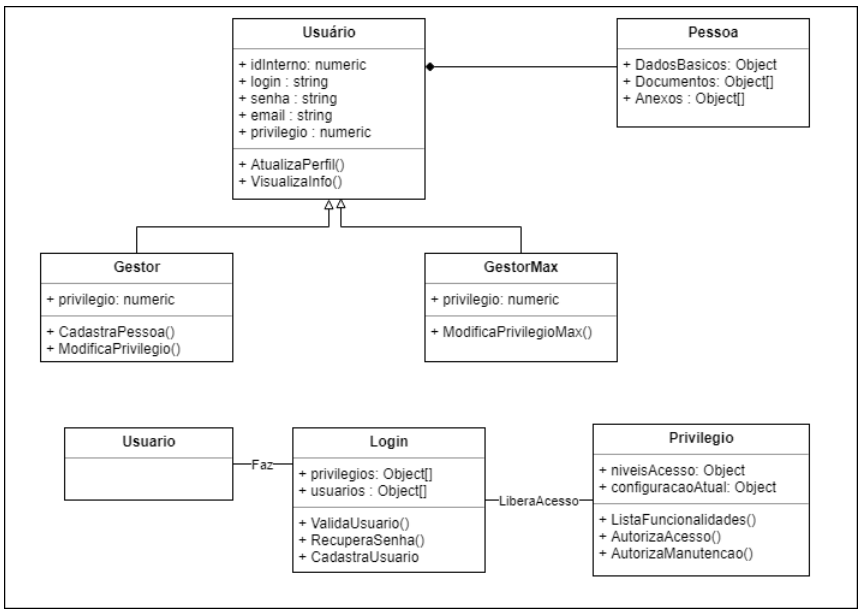
4.1.3 Diagramas de classe de análise

Dando sequência à modelagem, foram desenhados diagramas de classe de análise.

Na Figura 6 é mostrada a modelagem inicial das classes do sistema responsáveis por tratar do login e do acesso às funcionalidades do sistema. Na Figura 7 é exibido o vínculo da classe de pagamentos com os usuários, gestores e a classe de mensagens.

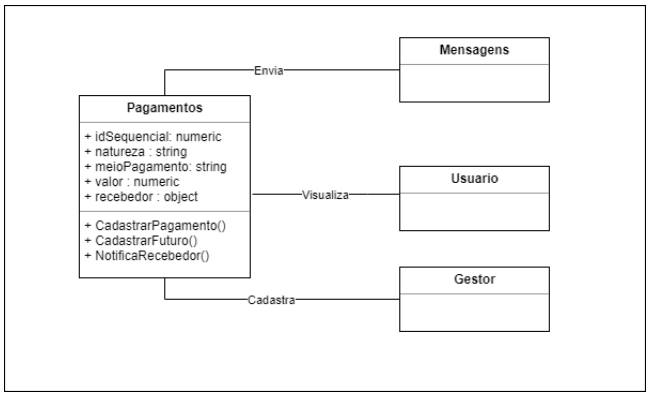
Na Figura 8 o cadastro de pessoas é mais detalhado e relacionado com a funcionalidade das fichas, pertencente ao módulo avançado.

Figura 6 – Classes de Usuários e Privilégios



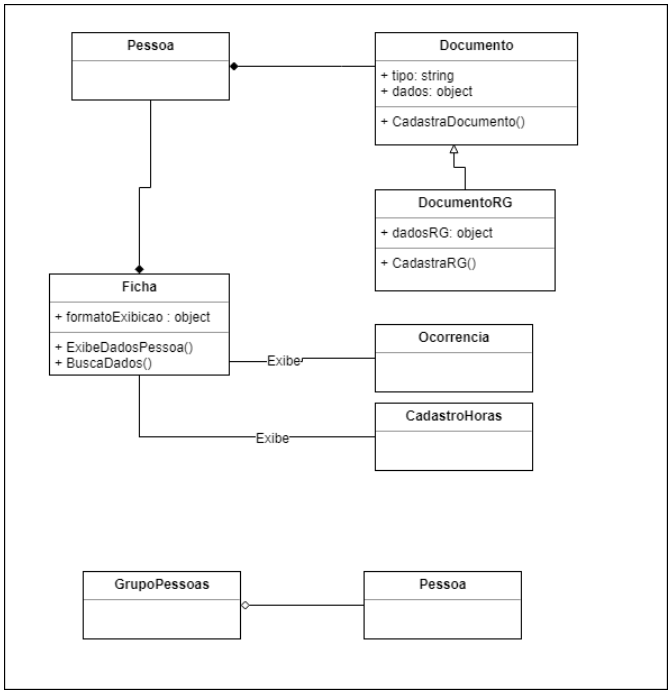
Fonte: Elaborada pelo autor.

Figura 7 – Classe de Pagamentos



Fonte: Elaborada pelo autor.

Figura 8 – Pessoas e fichas



Fonte: Elaborada pelo autor.

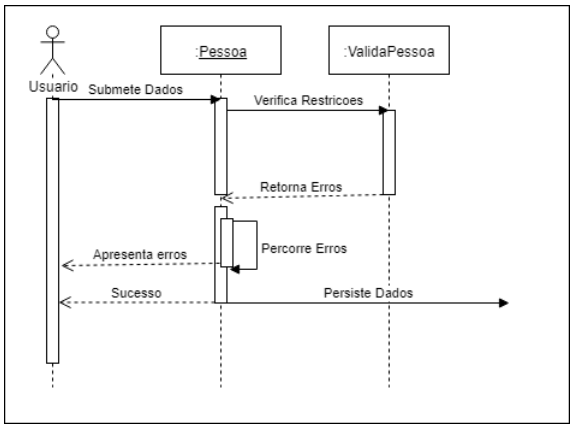
4.1.4 Diagramas de sequência

Os diagramas de sequência foram desenvolvidos apenas para algumas funcionalidades, principalmente as funções de cadastro, como o de pessoas, atividades.

A Figura 9 mostra o diagrama para o cadastro de pessoas, enquanto na Figura 10 se ve o diagrama para o cadastro de atividades.

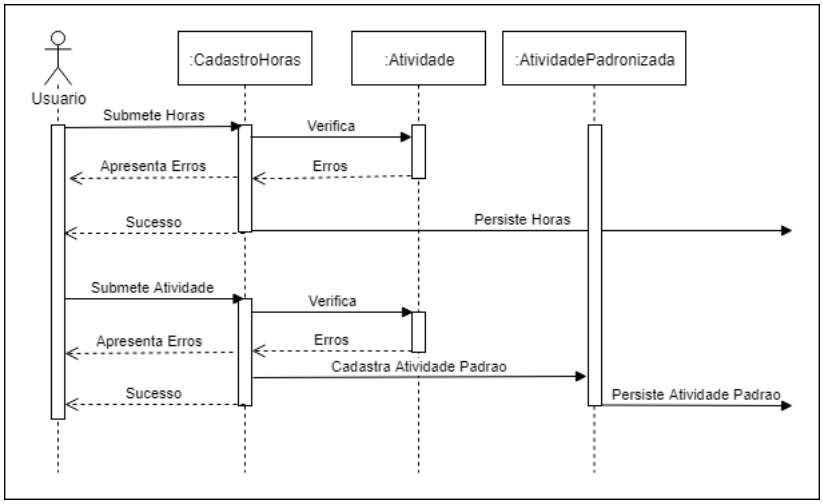
Além desses cadastros, o processo de login e autentificação também foi modelado, conforme pode ser visto na Figura 11.

Figura 9 – Sequência para cadastro de Pessoas



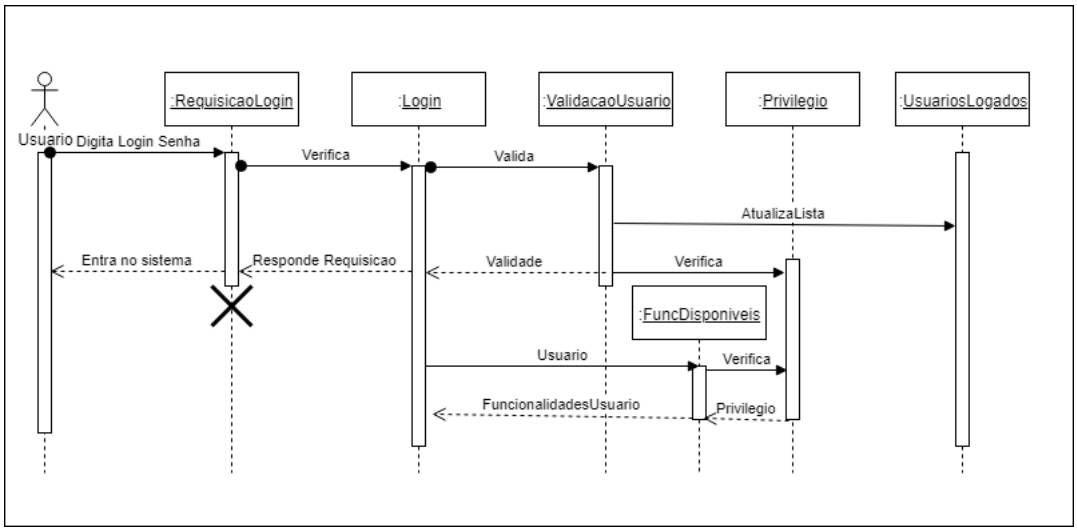
Fonte: Elaborada pelo autor.

Figura 10 – Sequência para cadastro de atividades



Fonte: Elaborada pelo autor.

Figura 11 – Sequência para autenticação de usuário

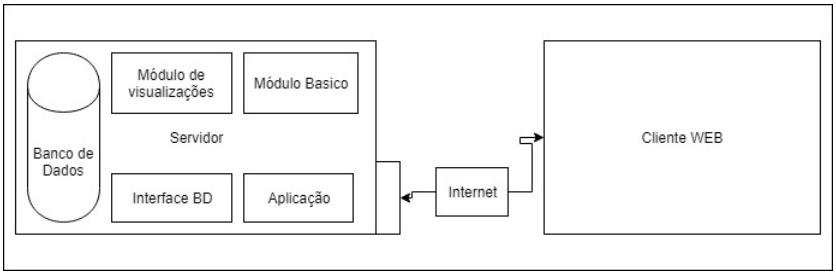


Fonte: Elaborada pelo autor.

4.1.5 Visão geral

A Figura 12 foi criada para exibir uma visão geral simplificada do sistema como um todo. Nele pode-se ver as principais partes desenvolvidas, o cliente WEB, o banco de dados e a interface entre ele e a aplicação como um todo. O módulo básico e o módulo avançado também estão representados.

Figura 12 – Visão geral do sistema



Fonte: Elaborada pelo autor.

4.2 Tecnologias escolhidas

Parte fundamental do desenvolvimento do sistema foi a escolha das tecnologias. O principal critério utilizado foi a popularidade da tecnologia, e consequente disponibilidade de material de apoio para o desenvolvimento.

4.2.1 Ferramentas para construção do front-end

Para a construção do front-end e das janelas, realizou-se uma pesquisa pelos principais frameworks utilizados atualmente. Entre eles estão: AngularJs, versão mais antiga do Angular (ANGULARJS, 2019), Angular 2+ (ANGULAR2+, 2019), Vue (VUE, 2019) e React (REACT, 2019). O Angular, seja na versão JS ou na versão 2 ou mais, é o mais utilizado, algumas vezes criticado pela complexidade do framework e pela longa curva de aprendizado. O Vue e o React seriam soluções mais simples, mas o conteúdo de apoio disponível e a base de usuários são significativamente menores.

Apesar de semelhantes, tanto no nome quanto na sintaxe de suas diretivas, o AngularJS e as versões 2+ são substancialmente diferentes. O AngularJS foi o primeiro a ser construído e utiliza o JavaScript como linguagem. A partir do Angular 2, no entanto, a linguagem utilizada é o TypeScript (TYPESCRIPT, 2019). Além disso ele conta com outras funções como o Angular Cli, que auxilia o manejo dos projetos à partir de comandos próprios do Angular, para criação de projetos, componentes e serviços.

O TypeScript é uma variação do JavaScript que permite declarar tipos para as variáveis, o que não é possível no JavaScript normal, pelo menos não explicitamente. A primeira vista, não precisar atribuir tipos para as variáveis pode parecer uma facilidade no desenvolvimento. Mas conforme a complexidade das tarefas aumenta, o uso de tipos se faz mais necessário. Isso ocorre principalmente para facilitar uso da orientação a objeto e de padrões de desenvolvimento de software.

Enquanto o JavaScript é uma linguagem interpretada, o TypeScript é uma linguagem compilada e o resultado dessa compilação é um código JavaScript. Por isso, é possível utilizar

ferramentas e código feito em JavaScript nas aplicações que utilizam TypeScript.

O conteúdo de apoio disponível na internet foi o fator determinante na escolha do Angular 2+ como ferramenta para a construção do front-end do sistema. A versão escolhida foi o Angular 7.

O Angular é uma ferramenta muito útil no desenvolvimento das aplicações WEB, facilitando a separação do software em componentes. Cada componente é constituído de um arquivo HTML(*Hyper Text Markup Language*), outro arquivo CSS(*Cascading Style Sheets*) e um outro para o TypeScript. Enquanto o CSS é responsável pelo posicionamento e estilização dos elementos HTML contidos no componente; o arquivo TypeScript é responsável por tratar a parte lógica dos componentes, como eventos e propriedades.

Além do Angular, foram estudados frameworks para auxiliar o desenho dos componentes visuais, utilizando padrões estéticos previamente pensados, e facilitando a implementação de componentes responsivos ao tamanho da tela.

Nesse sentido, o framework Bootstrap (BOOTSTRAP, 2019) é muito útil para estilizar os componentes e torná-los responsivos. Também foi estudado o uso do Angular Material (ANGULARMATERIAL, 2019), uma biblioteca de componentes prontos para serem utilizados. O Angular Material tem esse nome devido ao uso do Material Design, um padrão estético muito utilizado pelo Google e, portanto, muito conhecido pela maioria das pessoas.

Para a construção de gráficos, a biblioteca de javascript Chart.js (CHATJS, 2019) facilita o desenho de gráficos, gerados no cliente com dados recebidos de requisições. O Chart.js pode ser facilmente integrado com o Angular e foi utilizado na construção dos gráficos no sistema.

4.2.2 Ferramentas para construção do Back-end

Para o back-end, as opções levantadas foram construir um servidor com PHP (PHP, 2019) e Laravel (LARAVEL, 2019) ou construir o servidor com Node.js (NODEJS, 2019), utilizando javascript server side. Ambas são soluções muito utilizadas atualmente para o desenvolvimento WEB. O Node.js costuma ser utilizado juntamente com o Angular, contando com muito conteúdo de apoio para a utilização das duas tecnologias juntas.

A segunda opção foi escolhida, mais uma vez, levando em conta o conteúdo disponível na internet mas também a facilidade de desenvolvimento, uma vez que tanto o Angular quanto o Node trabalham com versões da linguagem javascript, embora o Angular use TypeScript.

O Node.js é uma ferramenta que utiliza o V8 JavaScript engine do Chrome, para executar código javascript no lado do servidor, como se houvesse um navegador executando os códigos javascript no servidor.

Como o Node é construído com base em uma versão específica da engine do Chrome,

essa versão específica não é atualizada com frequência, ao contrário do javascript, que é uma linguagem viva e que esta sendo atualizada com alguma frequência. O que pode levar a problemas de compatibilidade.

Uma outra versão do javascript é chamada de ES6 (ECMASCRIPT6, 2019) ou javascript 6. Esses nomes são relativos ao ECMAScript 6, especificação da linguagem javascript controlada pela ECMA (ECMA, 2019), abreviatura para *European Computer Manufacturers Association*.

Para utilizar as funcionalidades do ES6 sem problemas de compatibilidade, é utilizado o Babel (BABELJS, 2019), como o nome já indica, ele traduz o código JS para o Node. Em outras palavras, ele compila o código javascript nas versões mais recentes e gera código javascript na versão aceita pelo Node. Isso permite até mesmo misturar no mesmo código fonte funcionalidades de versões diferentes, uma vez que no fim, tudo será traduzido.

Durante a pesquisa das tecnologias, uma vez escolhidas as tecnologias citadas acima, outra ferramenta surgiu como uma escolha óbvia, o Express.js. Óbvia porque existe uma stack de desenvolvimento WEB famosa pela sua sigla, chamada de MEAN stack (MEANSTACK, 2019). A sigla MEAN vem de Mongo, Express, Angular e Node. Às vezes chamadas de MEAN2 ou MEAN+ quando utilizada com a versão 2+ do Angular.

O Express (EXPRESSJS, 2019) é responsável por facilitar o uso de requisições HTTP(*Hypertext Transfer Protocol*), possibilitando a criação de uma API no servidor com um esforço muito baixo, restando ao programador se preocupar em definir as rotas e nomenclaturas à serem utilizadas para as requisições. Assim fica mais fácil implementar as operações no banco de dados.

4.2.3 Banco de dados

Para a infraestrutura de banco de dados, os bancos MySQL (MYSQL, 2019), MongoDB (MONGODB, 2019), PostgreSQL (POSTGRESQL, 2019) e FireBase (FIREBASE, 2019) foram considerados inicialmente, porque são muito utilizados. Com exceção do primeiro, os outros apresentam paradigmas e conceitos diferentes dos bancos de dados utilizados normalmente. Podem possuir uma curva de aprendizado maior, para quem já está acostumado com bancos SQL(*Structured Query Language*) normais.

A escolha mais natural era o MySQL, devido a facilidade e conceitos amplamente difundidos, bem como uma grande base de usuários. Porém a opção escolhida foi o MongoDB, após os estudos técnicos mostrarem as vantagens de utilizar a MEAN stack. O MongoDB é muito utilizado e conta com ferramentas para facilitar a integração com o Node.js e a implementação das operações básicas no banco de dados a partir das requisições HTTP vindas do cliente.

O Mongo, ao contrário das outras opções, é um banco de dados não-relacional e NoSQL. Ele não trabalha com tabelas mas sim com coleções de documentos. Embora coleções

e documentos sejam análogos às tabelas e registros em um banco relacional, existem diferenças importantes. Uma coleção não exige um formato específico para um documento. É possível inserir documentos com formato de dados completamente diferentes entre si. Quando não é definida uma chave unica para uma coleção, o MongoDB irá gerar um ID único automaticamente, no momento da inserção. Esse ID gerado é o que permite trabalhar com documentos diferentes dentro de uma coleção, sem ter grandes problemas. Porém, definir uma estrutura para a coleção pode ser útil.

Para facilitar o uso do Mongo foi utilizado o Mongoose (MONGOOSE, 2019), uma ferramenta que auxilia na criação de modelos para os documentos que se deseja inserir. Com ele é possível definir uma estrutura para uma coleção, nomear os campos desejados, atribuir um tipo para eles e criar regras de validação. Citando exemplos, ele permite dizer que um campo é obrigatório, ou definir um valor padrão, ou default, caso o documento seja iniciado sem valor neste campo. Também é possível declarar valores máximos e mínimos, entre outras validações. Além das validações ja existentes no Mongoose, é possível criar validações personalizadas.

4.2.4 Ferramentas de apoio ao desenvolvimento

Durante o desenvolvimento foi utilizada a IDE(*Integrated Development Environment*) Visual Studio Code (VSCODE, 2019), que possui extensões que facilitam o desenvolvimento com Angular e javascript.

Para o gerenciamento do banco de dados, o Robo Mongo (ROBOMONGO, 2019) foi utilizado, ele permite criar coleções de documentos, realizar manutenção diretamente nos dados entre outras funcionalidades.

Para testar as requisições HTTP utilizadas pela API(*Application Programming Interface*) com o back-end, escolheu-se o Postman (POSTMAN, 2019), que permite o envio de requisições para o servidor, servindo para testar as rotas e operações.

4.3 Software desenvolvido

A escolha do desenvolvimento do software para a plataforma Web abriu muitas possibilidades de uso de ferramentas. Lembrando que a escolha das tecnologias teve como um dos critérios o uso de tecnologias recentes e muito utilizadas na atualidade.

Uma das dificuldades foi trabalhar com operações assíncronas, uma vez que a conclusão de algumas operações precisam ser executadas sincronizadamente, enquanto outras não.

O uso da linguagem javascript, além da sua variação, o typescript, não foi um grande entrave no desenvolvimento. Muito pelo contrário, oferecem facilidades que ajudaram a reduzir o tempo de desenvolvimento.

Um exemplo de operação que deve ser sincronizada é mostrado no código da Figura 13, em typescript, ilustra o uso dos métodos subscribe, presentes em objetos do tipo observables. O método recebe em um de seus parâmetros uma função que será executada após a operação inicial ser realizada. No caso da Figura 13, envia uma imagem para o banco e espera receber o id, gerado automaticamente pelo Mongo, para então gravá-lo nos dados do usuário cadastrado.

Figura 13 – Gravando dados no banco

```
this.imagem64Service.addImagem64(this.imagemPerfil.nome, this.imagemPerfil.tipo,
                                 this.imagemPerfil.descricao, this.imagemPerfil.base64)
.subscribe( (res: any) => { this.imagemPerfil.id = res._id; },
// onError, é o método executado em caso de erro
() => {
  this.enviaUsuario(pNome, pUsuario, pEmail, pDataNasc, pSenha, pPrivilegio);
  console.log('Erro no envio da imagemPerfil');
},
// onComplete enviando imagemPerfil, envia usuario
() => {
  this.enviaUsuario(pNome, pUsuario, pEmail, pDataNasc, pSenha, pPrivilegio);
});
```

Fonte: Elaborada pelo autor.

Na Figura 13 é possível ver o uso da variavel base64. Esse nome é utilizado porque a imagem é guardada no banco no formato base64, que nada mais é do que uma cadeia de caracteres.

A Figura 14 mostra o objeto mongoose utilizado para guardar as imagens. Como pode-se ver, a imagem é armazenada como uma string.

Figura 14 – Estrutura do tipo imagem64

```
const Schema = mongoose.Schema;

let Imagem64 = new Schema({
    nome: {
        type: String
    },

    tipo: {
        type: String
    },

    base64: {
        type: String
    },

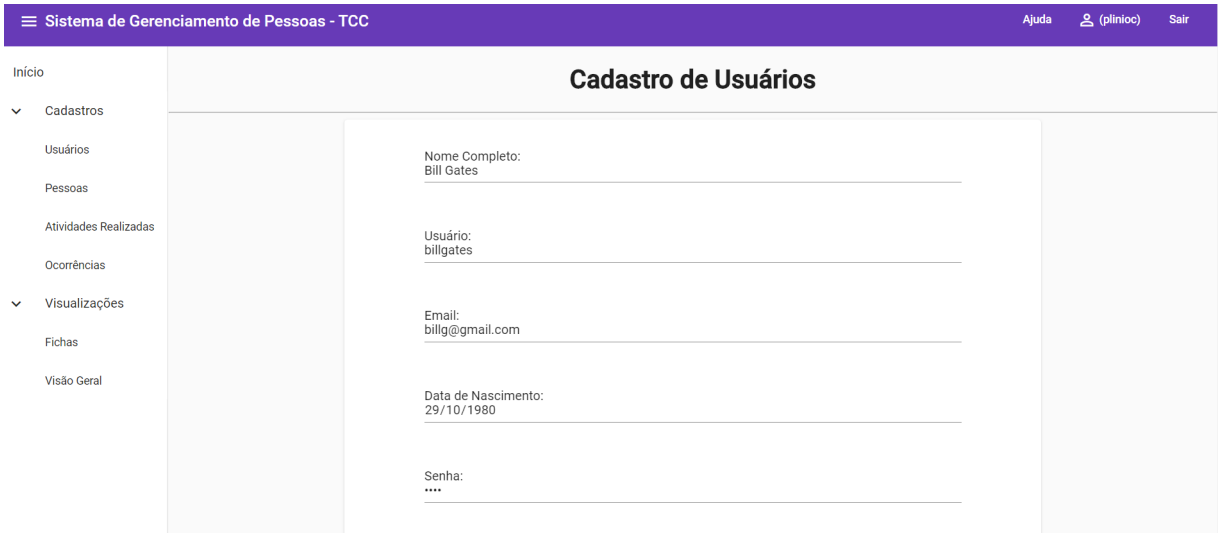
    descricao: {
        type: String
    }

});
```

Fonte: Elaborada pelo autor.

O cadastro de usuário citado anteriormente é feito atraves de um formulário, A Figura 15 mostra o cadastro de usuários aberto no modo de edição.

Figura 15 – Editando usuário



Fonte: Elaborada pelo autor.

Quando um formulário é submetido, é emitida uma requisição http, tratada no servidor node com auxilio do ExpressJs. São definidas rotas, que padronizam como serão as requisições

para cada tipo de operação. A Figura 16 mostra como são realizadas requisições por dados do cadastro de atividades.

Figura 16 – Rotas

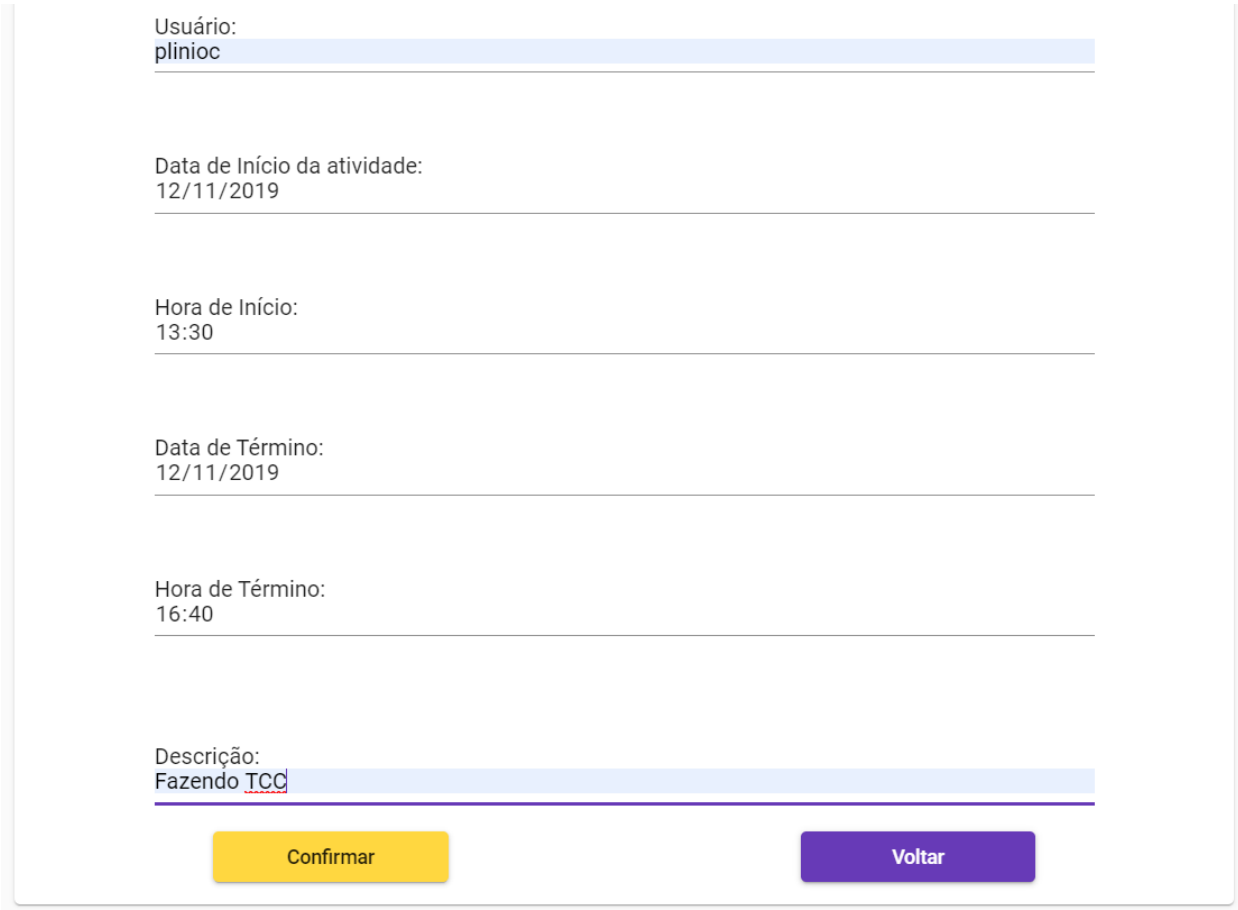
```
AtividadesRoutes.route('/:id').get((req, res) => {
    Atividade.findById(req.params.id, (err, atividade) => {
        if(err)
            console.log(err);
        else
            res.json(atividade);
    });
});

// Procura por usuário
AtividadesRoutes.route('/usuario/:usuario').get((req, res) => {
    Atividade.find({ usuario: req.params.usuario }, (err, atividades) => {
        if(err)
            console.log(err);
        else
            res.json(atividades);
    });
```

Fonte: Elaborada pelo autor.

O cadastro de atividades é outro cadastro importante no sistema. O cadastro permite informar atividades com inicio e fim em dias diferentes. Porém por padrão a data final é preenchida com a data inicial, indicando que a atividade foi realizada continuamente durante o dia. A leitura para o registro de horas dos usuários é feita apartir do cadastro dessas atividades, como pode ser observado na Figura 17.

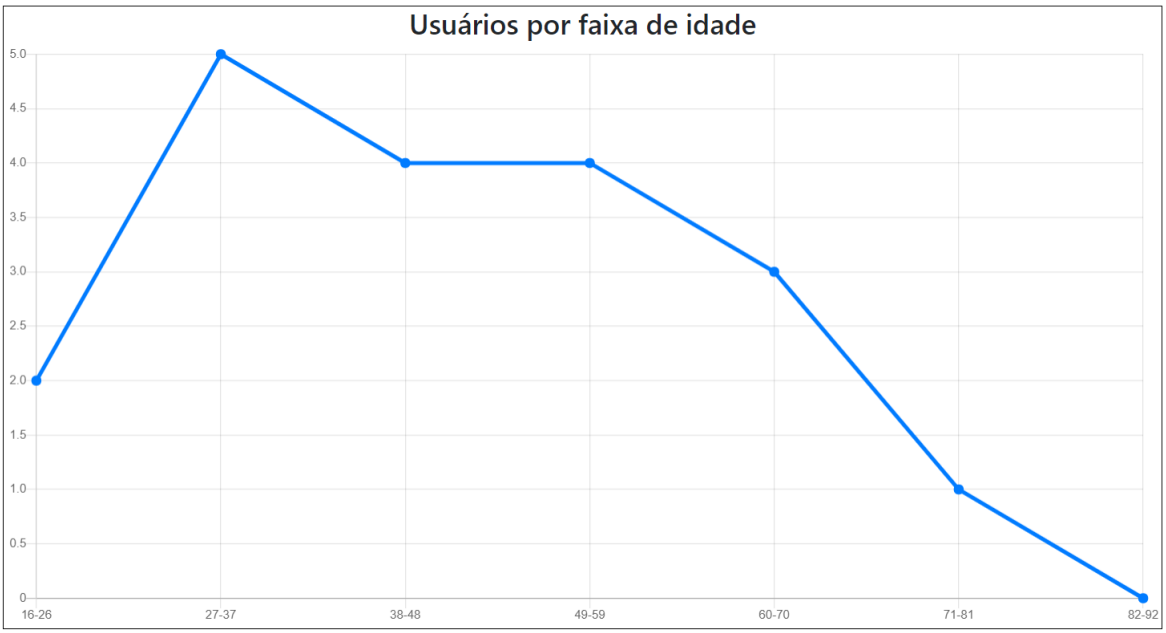
Figura 17 – Cadastro de atividades



Fonte: Elaborada pelo autor.

Um dos gráficos implementados é o de usuários por faixa de idade. Para o desenvolvimento dos gráficos utilizou-se a ferramenta Chat.Js, citada anteriormente entre as tecnologias escolhidas. Oferece uma interface simples para implementação de vários tipos de gráfico, como de barras, de pizza ou de linha. O gráfico mostrado na Figura 18 é do tipo de linha.

Figura 18 – Usuário por faixa de idade



Fonte: Elaborada pelo autor.

5 VALIDAÇÃO

Os estudos técnicos e a busca bibliográfica serviram para avaliar as possíveis tecnologias e o projeto como um todo. Também realizou-se uma prototipação do cliente WEB para dar uma noção da aparência e navegação do sistema, mas que também validou as escolhas do projeto.

Para testar e validar os formulários de cadastros e a infraestrutura de banco de dados foi necessário cadastrar um volume razoável de informação pelas janelas criadas, executando as formas de manutenção disponíveis, como Consultar, Cadastrar, Editar e Apagar.

O uso e a navegação também foram testados para avaliar, além da consistência das operações com os dados, a navegabilidade e eficiência da janela na função que ela se propõe executar.

Além de ser o primeiro a ser construído, o módulo básico foi o primeiro a ser validado, buscando assim validar a infraestrutura em conjunto. Para a validação do módulo de visualização de dados, os dados cadastrados devem ter uma "aparência" real para que os gráficos e estatísticas possam ser visualizados.

Para alimentar o cadastro de horas, utilizou-se uma variação aleatória dentro de um periôdo de alguns meses, para conseguir gerar gráficos mais realistas.

6 CONCLUSÃO

As funcionalidades dos dois módulos construídos para o sistema devem satisfazer as necessidades básicas dos usuários de desempenhar tarefas de gerenciamento de recursos humanos. Os cadastros desenvolvidos englobam as principais funcionalidades disponíveis em softwares comerciais. Além disso, a disponibilidade WEB das funcionalidades permitirá aos gestores um acompanhamento das informações em qualquer momento e lugar.

Os desafios de projeção e implementação foram muitos. O aprendizado das ferramentas escolhidas se mostrou mais longo do que o esperado inicialmente. Apesar disso as escolhas se mostraram vantajosas no período final do desenvolvimento. A correção de problemas e a extensão de funcionalidades já implementadas criou menos problemas do que a implementação inicial das mesmas.

A modelagem do software durante o período de especificação foi muito útil para no início do desenvolvimento, servindo como um norte. Porém foram necessárias mudanças durante a implementação, unindo classes e funções em uma só, por exemplo.

Uma falha encontrada na especificação inicial foi a ausência de alguma tentativa formal de estimar o tempo para o desenvolvimento do sistema. O planejamento foi feito com base em um cronograma geral, sem detalhamento das horas.

O desenvolvimento Web se revelou uma área muito ampla. As tecnologias investigadas oferecem uma grande variedade de opções e funcionalidades que ainda podem ser exploradas. Para citar exemplos, o Angular e o MongoDB possuem documentações muito extensas e detalhadas, oferecendo um bom suporte tanto no aprendizado inicial quanto no aprofundamento nas funções mais avançadas. Enfim, o desenvolvimento do sistema foi bem sucedido na exploração das tecnologias escolhidas.

Como trabalhos futuros seria importante melhorar a performance de navegação no sistema. A capacidade efetiva do sistema em cumprir as tarefas de forma rápida e segura, como proposta inicialmente, é muito afetada por problemas na navegação. Um desenho mais profissional também pode tornar o cliente WEB mais agradável visualmente. As funcionalidades do módulo avançado também permitem muita expansão, com opções mais relevantes e personalizadas de visualização dos dados, mais gráficos e métricas.

Referências

ADP. *Site institucional*. 2019. Disponível em: <https://www.adp.com/about-adp.aspx>. Acesso em 07 de junho de 2019.

ADVICE, S. *Site institucional, empresa avaliadora de softwares*. 2019a. Disponível em: <https://www.softwareadvice.com>. Acesso em 14 de junho de 2019.

ADVICE, S. *Buyers guide de softwares Payroll*. 2019b. Disponível em: <https://www.softwareadvice.com/hr/payroll-software-comparison/#buyers-guide>. Acesso em 13 de junho de 2019.

ADVICE, S. *Buyers guide de softwares de Time and Attendance*. 2019c. Disponível em: <https://www.softwareadvice.com/hr/time-attendance-comparison/#buyers-guide>. Acesso em 13 de maio de 2019.

ANGULAR2+. 2019. Disponível em: <https://angularjs.org/>. Acesso em 23 de julho de 2019.

ANGULARJS. 2019. Disponível em: <https://angularjs.org/>. Acesso em 23 de julho de 2019.

ANGULARMATERIAL. 2019. Disponível em: <https://material.angular.io/>. Acesso em 15 de novembro de 2019.

BABELJS. 2019. Disponível em: <https://babeljs.io/>. Acesso em 15 de novembro de 2019.

BASSO, D. A. *Gestão De Pessoas Na Micro E Pequena Empresa*. 2017. Disponível em: <https://www.mastereditora.com.br/periodico/20140721_162611.pdf>. Acesso em 12 de junho de 2019.

BENNER. *Site institucional*. 2019a. Disponível em: <https://www4.benner.com.br>. Acesso em 20 de junho de 2019.

BENNER. *Descrição da empresa Benner sobre sua solução de RH*. 2019b. Disponível em: <https://www4.benner.com.br/solucoes-para-reducao-de-custos/terceirizacao-de-servicos/gestao-de-recursos-humanos>. Acesso em 20 de junho de 2019.

BITTENCOURT J. A.; MATSUMOTO, A. S. *A influência da legislação trabalhista na gestão das pequenas empresas no Brasil*. 2017. Disponível em: <https://www.aedb.br/seget/arquivos/artigos16/31824361.pdf>. Acesso em 12 de junho de 2019.

BOOTSTRAP. 2019. Disponível em: <https://getbootstrap.com/>. Acesso em 24 de julho de 2019.

CHATJS. 2019. Disponível em: <https://www.chartjs.org>. Acesso em 24 de julho de 2019.

ECMA. 2019. Disponível em: <https://www.ecma-international.org/>. Acesso em 15 de novembro de 2019.

ECMASCRIPT6. 2019. Disponível em: <http://es6-features.org/>. Acesso em 15 de novembro de 2019.

EXPRESSJS. 2019. Disponível em: <https://expressjs.com/pt-br/>. Acesso em 15 de novembro de 2019.

FERREIRA, . e. L. *Avaliação da usabilidade em sistemas de informação: o caso do Sistema Submarino*. 2016. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1415-65552003000200007>. Acesso em 06 de junho de 2019.

FIREBASE. 2019. Disponível em: <https://firebase.google.com/?hl=pt-BR>. Acesso em 24 de julho de 2019.

HUSELID, M. A. *The Impact Of Human Resource Management Practices On Turnover, Productivity, And Corporate Financial Performance*. 1995. Disponível em: <https://journals.aom.org/doi/abs/10.5465/256741>. Acesso em 11 de maio de 2019.

KRONOS. *Site institucional*. 2019. Disponível em: <https://www.kronos.com/about-us>. Acesso em 20 de junho de 2019.

LARAVEL. 2019. Disponível em: <https://php.net>. Acesso em 24 de julho de 2019.

MEANSTACK. 2019. Disponível em: <http://meanjs.org/>. Acesso em 15 de novembro de 2019.

MENDES, M. *Tecnologias Web em 2018: O que escolher ao começar um novo projeto*. 2016. Disponível em: <https://medium.com/@marco_s_mendes/tecnologias-web-em-2018-o-que-escolher-ao-come%C3%A7ar-um-novo-projeto-f396a1d76859>. Acesso em 01 de novembro de 2019.

MONGODB. 2019. Disponível em: <https://www.mongodb.com>. Acesso em 24 de julho de 2019.

MONGOOSE. 2019. Disponível em: <https://mongoosejs.com>. Acesso em 15 de novembro de 2019.

MYSQL. 2019. Disponível em: <https://www.mysql.com>. Acesso em 24 de julho de 2019.

NODEJS. 2019. Disponível em: <https://nodejs.org/en>. Acesso em 24 de julho de 2019.

PHP. 2019. Disponível em: <https://php.net>. Acesso em 23 de julho de 2019.

POSTGRESQL. 2019. Disponível em: <https://www.postgresql.org>. Acesso em 24 de julho de 2019.

POSTMAN. 2019. Disponível em: <https://www.getpostman.com/>. Acesso em 15 de novembro de 2019.

REACT. 2019. Disponível em: <https://pt-br.reactjs.org>. Acesso em 23 de julho de 2019.

ROBOMONGO. 2019. Disponível em: <https://robomongo.org/>. Acesso em 15 de novembro de 2019.

SINOPSE das Rotinas Trabalhistas. 2019. Disponível em: <http://www.guiatrabalhista.com.br/tematicas/rotinastrabalhistas.htm>. Acesso em 11 de maio de 2019.

TYPESCRIPT. 2019. Disponível em: <https://www.typescriptlang.org/>. Acesso em 11 de novembro de 2019.

UAC. *Importance Of Information Security In Organizations Information Technology Essay*. 2019. Disponível em: <https://www.uniassignment.com/essay-samples/information-technology/importance-of-information-security-in-organizations-information-technology-essay.php>. Acesso em 07 de junho de 2019.

VSCODE. 2019. Disponível em: <https://code.visualstudio.com/>. Acesso em 15 de novembro de 2019.

VUE. 2019. Disponível em: <https://vuejs.org>. Acesso em 23 de julho de 2019.