**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**PUC Minas Virtual**

**Pós-graduação *Lato Sensu* em Engenharia de *Software***

Projeto Integrado

Relatório Técnico

Sistema de controle de clientes para empresa desenvolvedora de software

Rodrigo Casadei Alanis

Belo Horizonte

Abril de 2022.

# Projeto Integrado

**Sumário**

Projeto Integrado 4

1. Cronograma de Trabalho 5

2. Introdução 5

3. Definição Conceitual da Solução 7

3.1 Diagrama de Casos de Uso 7

3.2 Requisitos Funcionais 7

3.3 Requisitos Não-funcionais 8

4. Protótipo Navegável do Sistema 9

5. Diagrama de Classes de Domínio 9

6. Arquitetura da Solução 10

6.1 Padrão Arquitetural 10

6.2 C4 model - Diagrama de Contexto 10

7. Frameworks de Trabalho 10

8. Estrutura Base do Front End 10

9. Modelo Relacional ou Projeto de Banco de Dados NoSQL 10

10. Plano de Testes 10

11. Relatório de Execução de Testes de Software 10

12. Apropriação de Horas no Projeto 10

13. Código da Aplicação 11

14. Avaliação Retrospectiva 11

14.1 Objetivos Estimados 11

14.2 Objetivos Alcançados 11

14.3 Lições aprendidas 11

15. Referências 12

## Cronograma de Trabalho

|  |  |  |  |
| --- | --- | --- | --- |
| **Datas** | | **Atividade / Tarefa** | **Produto / Resultado** |
| **De** | **Até** |
| 01 / 04 / 22 | 01 / 04 / 22 | 1. Pesquisa e elaboração do problema. | Início da Introdução do projeto. |
| 02 / 04 / 22 | 03 / 04 / 22 | 2. Pesquisa por bibliografias e escrita da Introdução. | Elaboração da Introdução. |
| 04 / 04 / 22 | 05 / 04 / 22 | 3. Análise e escrita dos requisitos. | Elaboração dos Requisitos funcionais e Requisitos não-funcionais. |
| 06 / 04 / 2 | 06 / 04 / 22 | 4. Análise dos requisitos e criação do Diagrama de casos de uso. | Elaboração do diagrama de casos de uso. |
| 07 / 04 / 22 | 07 / 04 / 22 | 5. Confecção dos wireframes das telas principais. | Elaboração dos wireframes das telas principais. |
| 08 / 04 / 22 | 10 / 04 / 22 | 6. Análise e evolução dos wireframes para um protótipo interativo. | Elaboração de um protótipo navegável e interativo. |
| 11 / 04 / 22 | 11 / 04 / 22 | 7. Análise dos requisitos e criação do Diagrama de classes de domínio. | Elaboração do diagrama de classes de domínio. |
| 12 / 04 / 22 | 13 / 04 / 22 | 8. Revisão de todo o material produzido e criação do repositório com os arquivos solicitados. | Criação do repositório e conclusão da primeira etapa do projeto. |
| 15 / 05 / 22 | 15 / 05 / 22 | 9. Correção dos erros e alterações frente as sugestões propostar pelo orientador | Correções e alterações referentes à etapa 1 para o início da etapa 2. |
| 08 / 07 / 22 | 11 / 07 / 22 | 10. Escolha das tecnologias e estudo e escolha dos padrões arquiteturais a serem utilizados. | Após a escolha das tecnologias, a pesquisa e escolha dos padrões foram realizadas. |
| 11 / 07 / 22 | 11 / 07 / 22 | 11. Confecção do modelo C4. | Confecção do modelo C4 baseado no material já entregue. |
| 14 / 07 / 22 | 23 / 07 / 22 | 12. Escolha dos frameworks disponíveis nas tecnologias e testes com os mesmos. | Após as escolhas, foram realizados testes com o uso dos frameworks e poucos quase todos foram mantidos. |
| 23 / 07 / 22 | 30 / 07 / 22 | 13. Escolha da funcionalidade e desenvolvimento do back end. | Após a escolha da funcionalidade, um escopo de funções na API do back end foi definida para ser desenvolvida. |
| 01 / 08 / 22 | 24 / 08 / 22 | 14. Desenvolvimento do front end e adaptações e correções no back end. | No decorrer do desenvolvimento do front end, algumas alterações e correções tiveram de ser realizadas no back end. |
| 22 / 08 / 22 | 24 / 08 / 22 | 15. Deploy do banco de dados, back end e front end. | Estudo e deploy das camadas em plataformas disponíveis na nuvem. |
| 25 / 08 / 22 | 27 / 08 / 22 | 16. Confecção de representação do Modelo relacional implementado no sistema. |  |
| 28 / 08 / 22 | 05 / 09 / 22 | 17. Confecção do plano de testes e a elaboração dos mesmos. |  |
| 06 / 09 / 22 | 10 / 09 / 22 | 18. Execução e registro dos testes. |  |
| 11 / 09 / 22 | 23 / 09 / 22 | 19. Desenvolvimento das outras funcionalidades do sistema e refinamento da estrutura. |  |
| 23 / 09 / 22 | 25 / 09 / 22 | 20. Realização da avaliação retrospectiva. |  |

## Introdução

A cada ano, a tecnologia está mais presente não apenas no cotidiano das pessoas, mas também no funcionamento das empresas, que precisam ser mais ágeis e produtivas para atender às expectativas de clientes mais exigentes.

Na visão de McGEE e PRUSAK (1994):

Nas próximas décadas, a informação, mais do que a terra ou o capital, será a força motriz na criação de riquezas e prosperidade. [...] numa economia de informação, a concorrência entre as organizações baseia-se em sua capacidade de adquirir, tratar, interpretar e utilizar a informação de forma eficaz. (p. 3).

Acredita-se que esta é a realidade sobretudo de empresas de desenvolvimento de software, que buscam otimização em seus processos e relações com seus clientes em um mundo mais interconectado do que nunca. Segundo REZENDE e ABREU (2000, p.63), entre os benefícios que empresas buscam obter a partir de sistemas de informação, destaca-se para o uso de empresas deste setor neste desafio:

Suporte à tomada de decisão com maior qualidade e confiabilidade; melhor prestação de serviços e vantagens competitivas; elaboração de produtos de melhor qualidade; geração de oportunidades de negócios e aumento da rentabilidade; maior segurança nas informações e menos erros; aperfeiçoamento dos processos, trazendo maior eficiência; redução na carga de trabalho e desperdícios e maior controle das operações.

Diante de um crescente número de clientes e aumento de requisições de alterações e implementações de novas funcionalidades, o tempo dos integrantes tornou-se escasso e valioso na Global Soft, empresa de desenvolvimento de software de Araçatuba, uma cidade do interior paulista. Combinado a isso, a gestão dos dados dos clientes, suas mensalidades e faturas, persiste de forma manual no dia a dia dela, subtraindo mais tempo com estas atividades. Seja em papéis avulsos ou em documentos e planilhas armazenados em computadores diferentes, as informações estão espalhadas, muitas vezes consumindo ainda mais tempo em seu acesso e tornando a integridade e segurança dos dados questionável, como conta Paulo Roberto, CEO da empresa.

Em busca de otimização dos processos de gestão e criação de um ambiente unificado e protegido para o armazenamento dos dados dos clientes e suas pendências financeiras, o presente trabalho sugere que a empresa se beneficiaria desenvolvendo um sistema próprio para o controle de clientes. Isto traria, de acordo com o autor deste projeto integrado, maior agilidade e economia de tempo para os integrantes, também possibilitando análises e melhor controle sobre as finanças, seus clientes e tomada de decisões. Além de cobrir algumas lacunas, este sistema proporcionaria comodidades e novos serviços aos clientes, como consultas online de seu saldo devedor ou faturas pagas, e abertura de tickets de suporte. Este sistema central de controle também teria potencial de abrir portas para futuras integrações com os softwares fornecidos e mantidos pela empresa aos seus clientes.

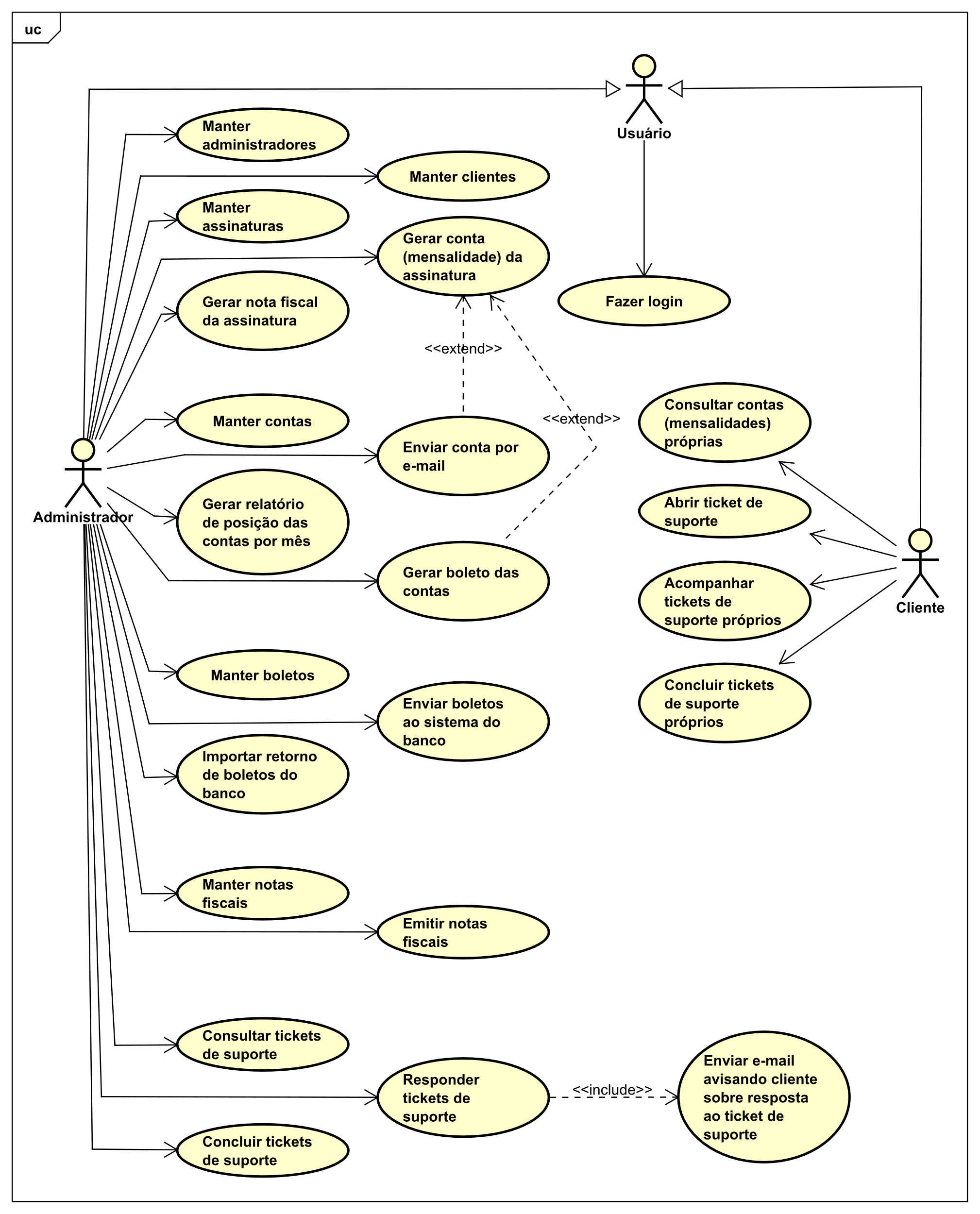
O objetivo geral deste trabalho é elaborar a descrição do projeto de um sistema de controle de clientes para uma empresa de desenvolvimento de software.

Os objetivos específicos são:

* Definir a solução conceitualmente, descrevendo os requisitos do sistema e suas interfaces;
* Definir o melhor padrão arquitetural, linguagem, frameworks e sistema de banco de dados para seu desenvolvimento;
* Desenvolver, testar e hospedar as primeiras etapas deste sistema;

## Definição Conceitual da Solução

## Diagrama de Casos de Uso



## Requisitos Funcionais

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Descrição Resumida** | **Dificuldade (B/M/A)\*** | **Prioridade**  **(B/M/A)\*** |
| RF01 | O usuário deve fazer login no sistema. | B | A |
| RF02 | O administrador deve manter administradores. | M | A |
| RF03 | O administrador deve manter clientes. | M | A |
| RF04 | O administrador deve manter assinaturas. | M | A |
| RF05 | O administrador deve gerar conta (mensalidade) da assinatura. | B | A |
| RF06 | O administrador deve gerar nota fiscal da assinatura. | A | M |
| RF07 | O administrador deve manter contas. | M | A |
| RF08 | O administrador deve enviar conta por e-mail. | B | A |
| RF09 | O administrador deve gerar relatório de posição das contas por mês. | M | M |
| RF10 | O administrador deve gerar boleto das contas. | M | M |
| RF11 | O administrador deve manter boletos. | M | M |
| RF12 | O administrador deve enviar boletos ao sistema do banco. | A | M |
| RF13 | O administrador deve importar retorno de boletos do banco. | A | B |
| RF14 | O administrador deve manter notas fiscais. | A | M |
| RF15 | O administrador deve emitir notas fiscais. | A | M |
| RF16 | O administrador deve consultar tickets de suporte. | M | B |
| RF17 | O administrador deve concluir tickets de suporte | B | B |
| RF18 | O administrador deve responder tickets de suporte. | B | B |
| RF19 | O sistema deve enviar um e-mail ao cliente avisando sobre a resposta ao ticket de suporte | B | B |
| RF20 | O cliente deve consultar contas (mensalidades) próprias. | M | M |
| RF21 | O cliente deve abrir ticket de suporte. | B | B |
| RF22 | O cliente deve acompanhar tickets de suporte próprios. | M | B |
| RF23 | O cliente deve concluir tickets de suporte próprios. | B | B |

\* B = Baixa, M = Média, A = Alta.

## Requisitos Não-funcionais

|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Prioridade**  **B/M/A** |
| RNF01 | Compatibilidade com os navegadores mais utilizados (Chrome, Edge e Firefox). | A |
| RNF02 | O sistema deve ficar disponível durante 90% do tempo de expediente da empresa, que funciona das 08h às 11h30 e das 13h às 17h30. | A |
| RNF03 | O sistema deve possuir tempo de resposta abaixo de 500 ms no processamento de operações síncronas. | A |
| RNF04 | O sistema deve fazer uso de design responsivo nas interfaces gráficas. | M |
| RNF05 | O sistema deve possuir autenticação de usuário para consumo de webservices do sistema pelos sistemas oferecidos pela empresa aos seus clientes. | M |
| RNF06 | Integração com sistemas do Banco Bradesco para envio das remessas de boletos. | B |

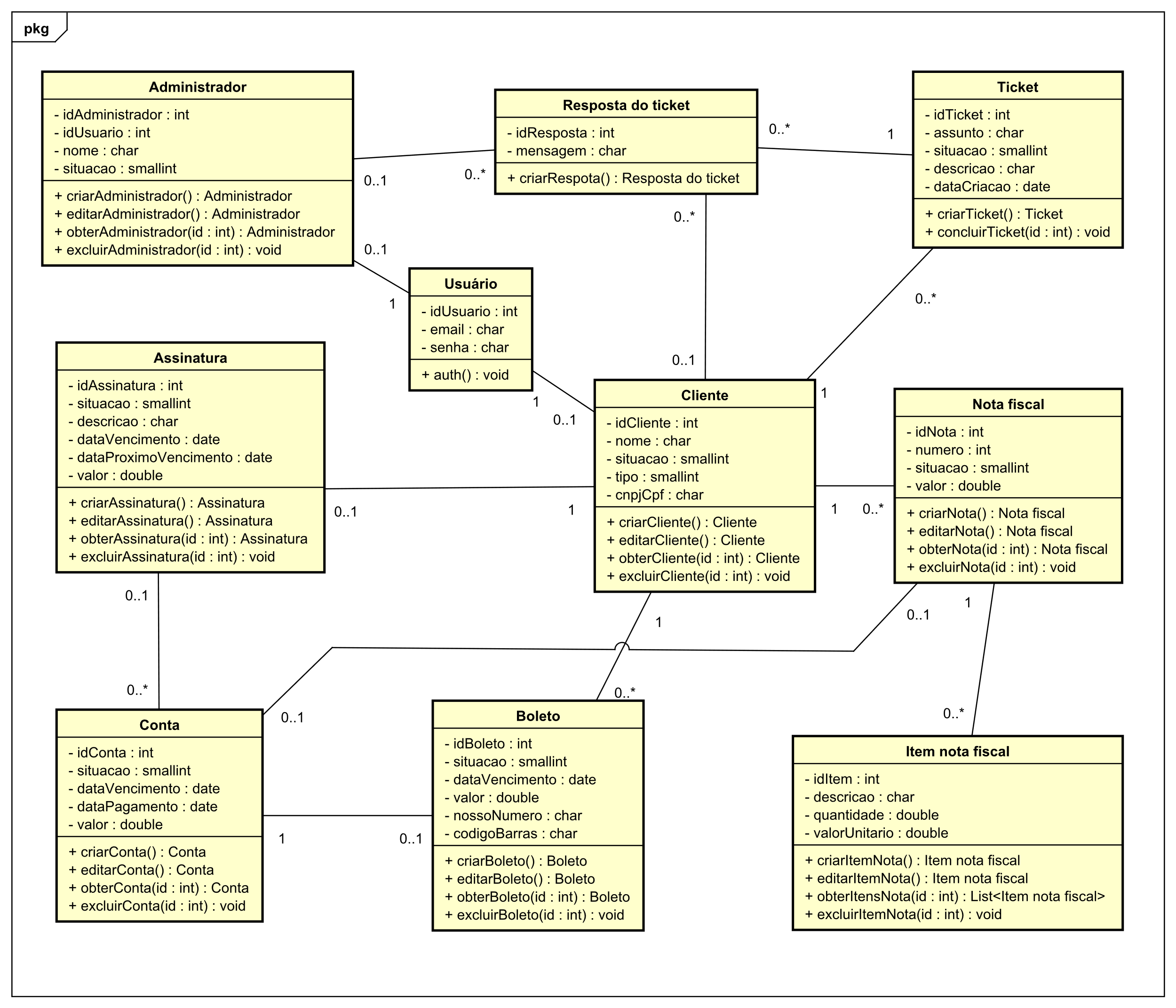
## Protótipo Navegável do Sistema

Wireframes e vídeo demonstrando o protótipo navegável e interativo encontram-se no repositório do projeto, dentro da pasta Documentos.

Link para o repositório do projeto: <https://github.com/rodrigovk/projeto-integrado-engsoftware-pucminas.git>.

Link de acesso ao protótipo: <https://www.figma.com/proto/TXFZ2sEaT8AtQeZRm9AIaG/Projeto-Integrado---Prot%C3%B3tipo---PUC-Minas?scaling=min-zoom&page-id=0%3A1&starting-point-node-id=5%3A377&node-id=5%3A377>.

## Diagrama de Classes de Domínio

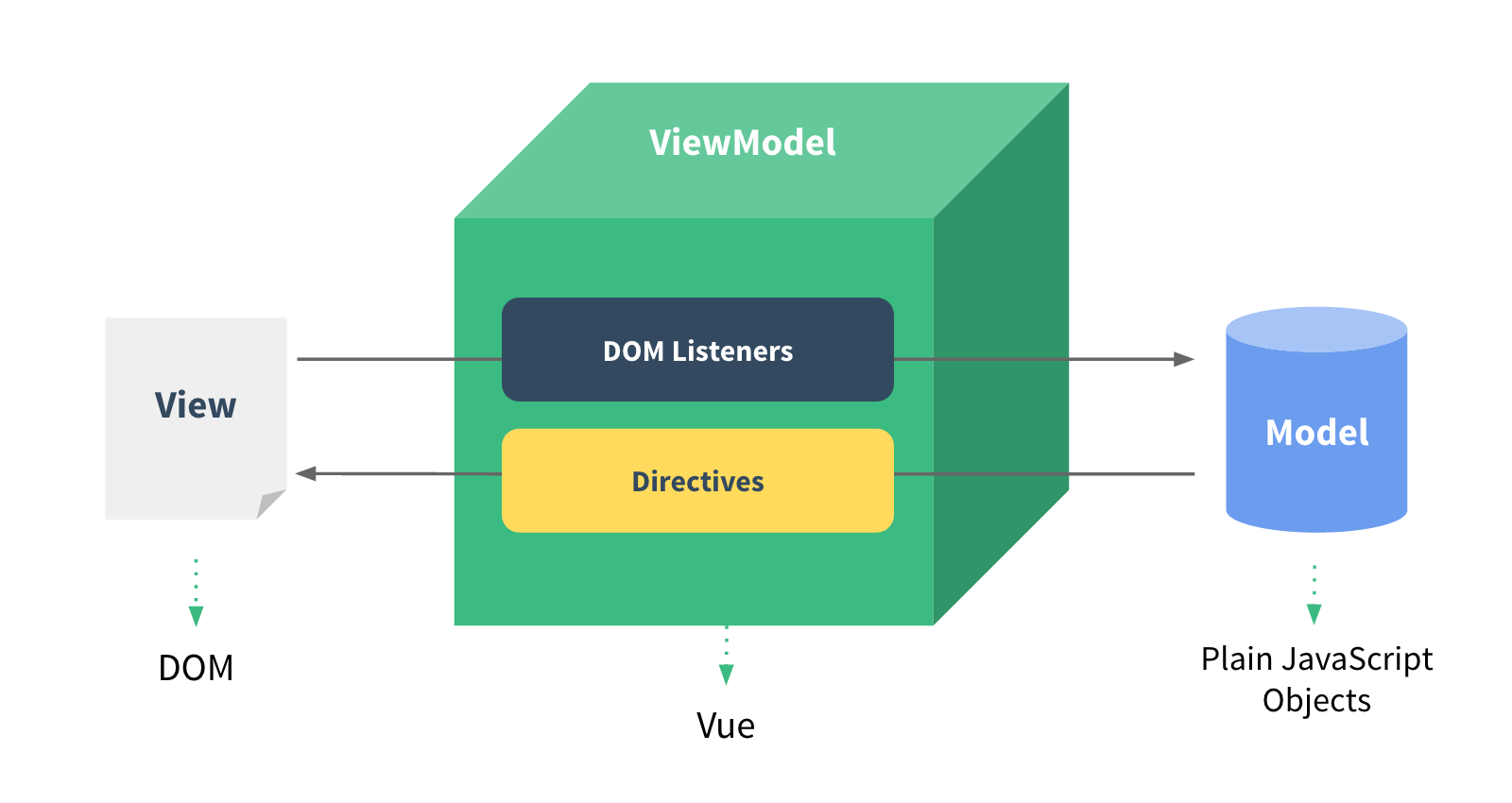


## Arquitetura da Solução

## Padrão Arquitetural

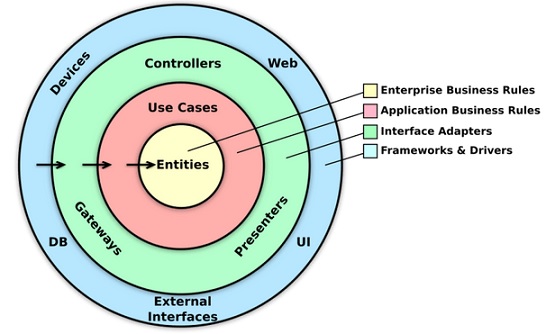
Para o desenvolvimento do front end do sistema, foi escolhido o padrão arquitetural MVVM (Model-View-ViewModel), muito comum entre os frameworks web atuais (MENDES, 2020, p. 108). Consiste em um padrão utilizado para a criação de interfaces de usuário e é otimizado para desenvolvimento multiplataforma, além de simplificar testes de unidade e ser flexível (MENDES, 2020, p. 51). Sua escolha está ligada diretamente à escolha do framework web, o Vue 3, que tem seu funcionamento atrelado a este padrão.

Neste padrão, a View é a interface do usuário, podendo ser uma tela, parte de uma tela ou um simples elemento que será contido em uma tela, agindo apenas como uma interface para entrada e saída de dados. A View é ligada à ViewModel, onde estão contidas todas as operações que serão utilizadas para gerenciar os elementos e coordenar as entradas e saídas da View. O Model, por sua vez, é o responsável pelos dados da aplicação, e fará a comunicação diretamente com a ViewModel, provendo e recebendo informações.



Referente ao desenvolvimento do Back end, a arquitetura limpa foi escolhida para proporcionar uma base de código de fácil manutenção e baixo acoplamento dos frameworks e módulos utilizados. Proposta por Robert Martin, também conhecido como Uncle Bob, esta arquitetura permite a construção de um software flexível, ao mesmo tempo que continua robusto e escalável.

Como demonstrado na imagem a seguir, esta arquitetura estabelece uma comunicação entre as camadas partindo da camada mais externa, e adentrando uma camada por vez, de uma forma que apenas a camada externa adjacente tenha conhecimento da próxima camada interna.



## C4 model - Diagrama de Contexto



No diagrama de contexto apresentado estão ilustrados os dois tipos de pessoas que acessarão o sistema, no caso, “Administradores” e ”Clientes”, possuindo permissões de acesso a telas e funções distintas. O sistema está representado pelo bloco “Sistema de controle de clientes”, ele é usado pelas pessoas citadas, e comunica-se com o sistema externo de e-mails, representado pelo bloco “Sistema de e-mail”. Este, após ser acionado pelo sistema desenvolvido, envia e-mails para a pessoa “Cliente”.

## Frameworks de Trabalho

O framework Express, minimalista e flexível, um grande auxiliador na criação de APIs, foi utilizado para a criação de uma API RESTful que opera como o Back end do sistema. Aliado ao Prisma, um novo ORM baseado em TypeScript para Node.js, estes foram os principais frameworks utilizados no desenvolvimento do Back end. Também foi utilizado o módulo nodemailer para o envio de e-mails.

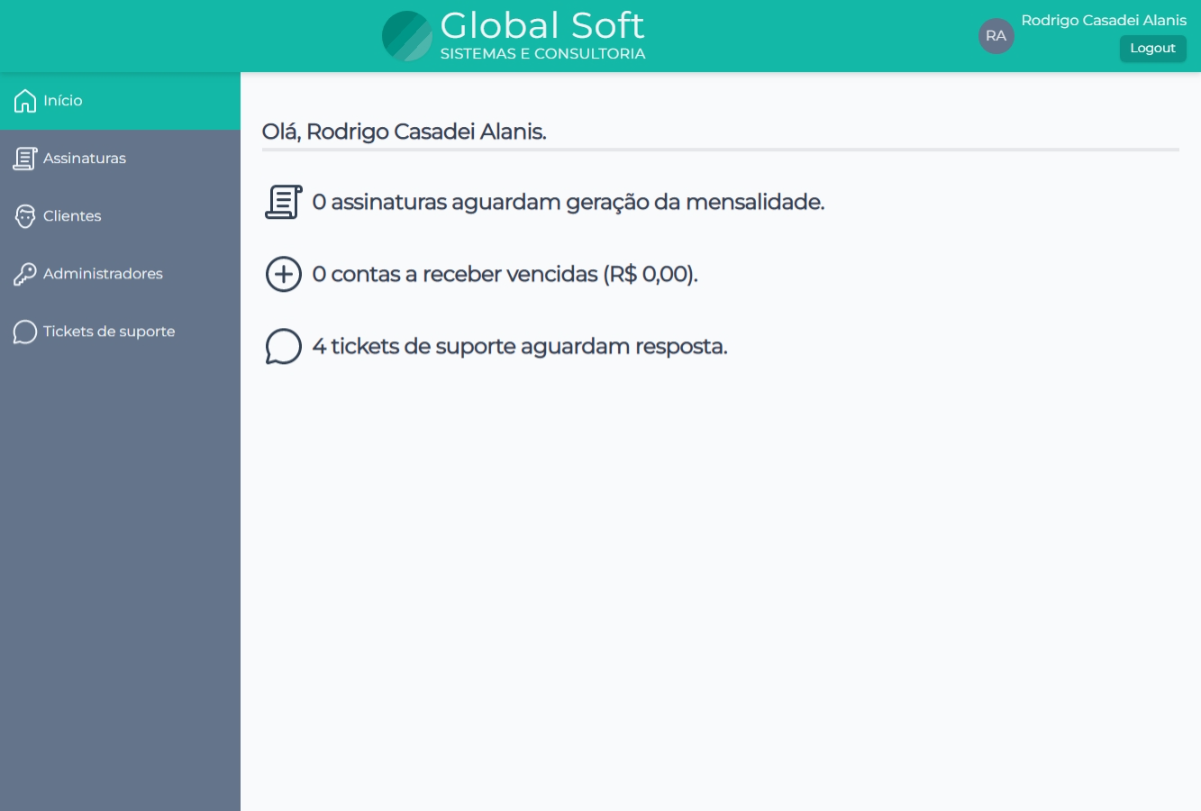
No lado Front end utilizou-se o framework web Vue 3, um dos mais utilizados ao lado de React e Angular, aliado à linguagem JavaScript. Para a facilitação na implementação do CSS, o Tailwind foi escolhido, consistindo em uma ferramenta que se integra e lê o código e arquivos HTML escritos, e gera um arquivo CSS estático. A ferramenta Vite foi utilizada para a renderização da interface durante o desenvolvimento e para a construção dos arquivos finais de produção. Também foram utilizados os módulos Axios, para a realização das requisições de comunicação com a API RESTful do Back end, e Pinia, responsável pelo armazenamento dos dados em memória utilizados no Front end.

Para a Persistência, o framework Prisma foi empregado para o acesso e escrita dos dados do sistema, agindo em paralelo com uma base de dados PostgreSQL.

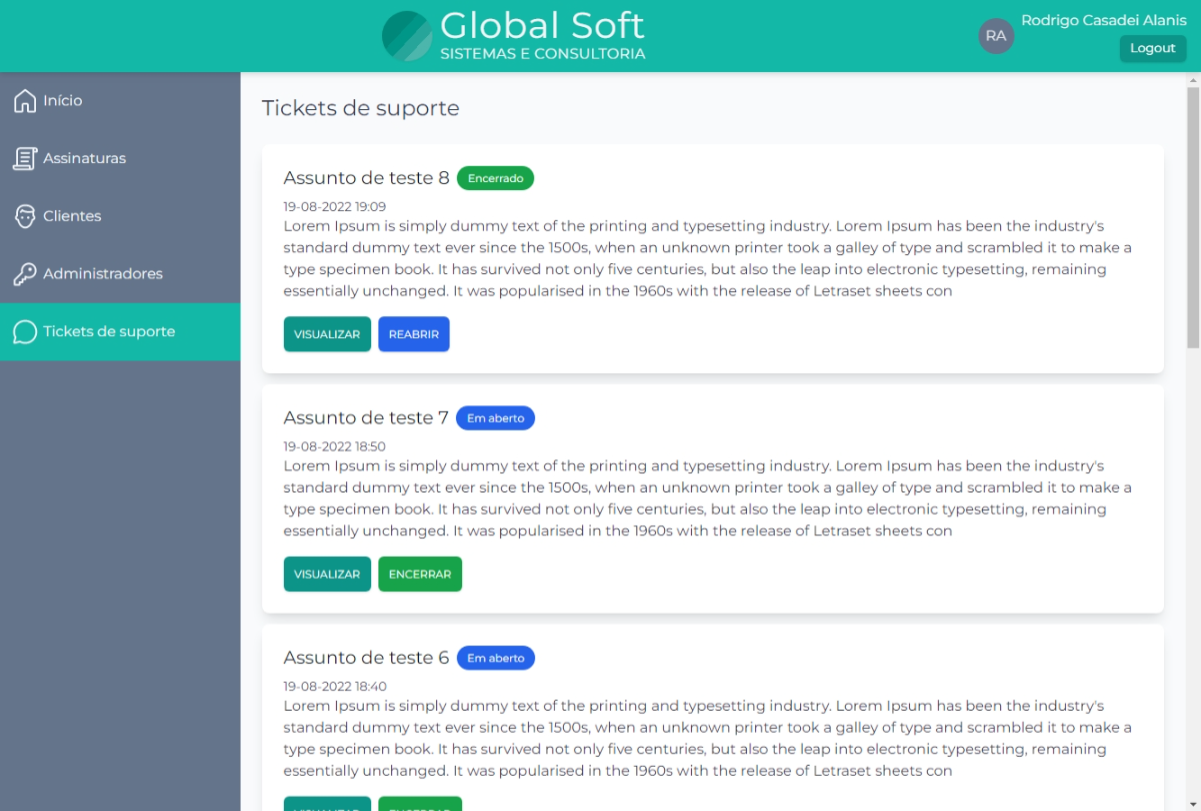
## Estrutura Base do Front End



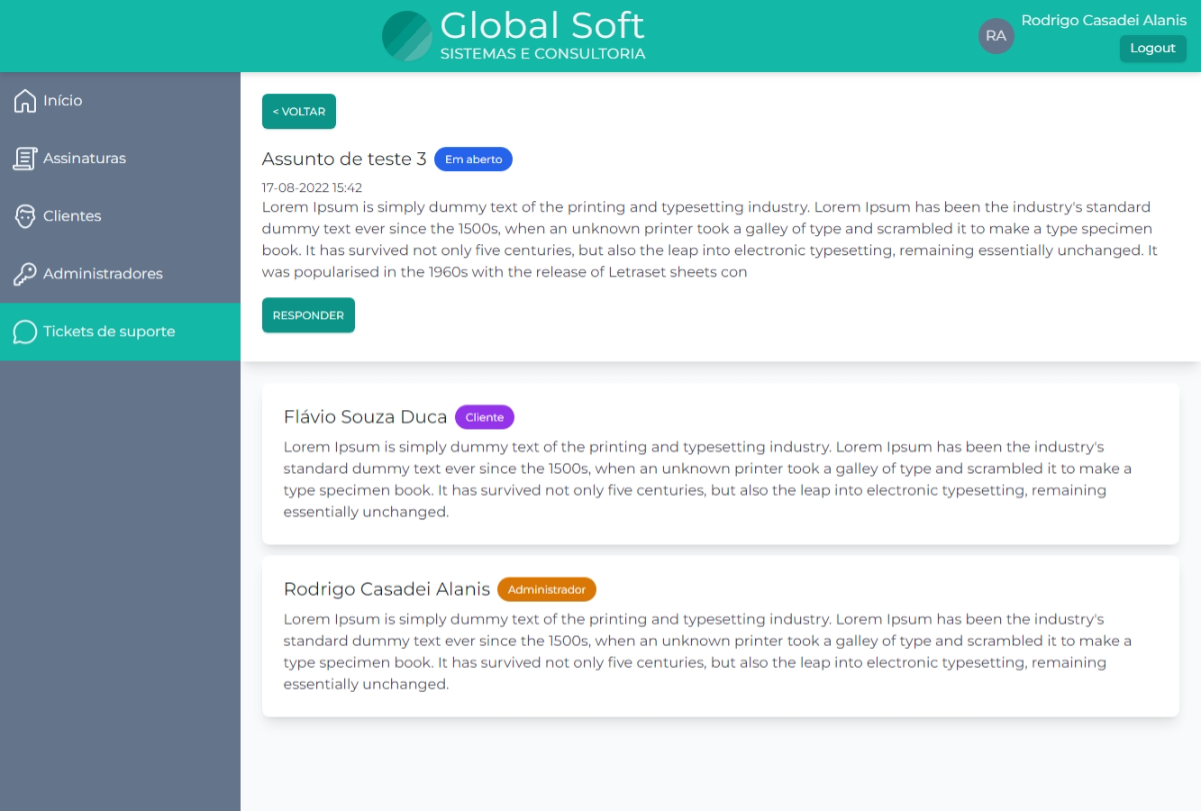
Página de login onde o usuário é autenticado.



Página inicial com acesso aos menus das outras páginas.



Página de tickets de suporte abertos por clientes e acompanhados pelos mesmos e administradores.



Página de detalhes de ticket, onde o cliente que o criou e os administradores acompanham o ticket e enviam novas respostas.

## Modelo Relacional ou Projeto de Banco de Dados NoSQL

## Plano de Testes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Caso de uso** | **Objetivo do caso de teste** | **Entradas** | **Resultados esperados** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Relatório de Execução de Testes de Software

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Caso de teste** | **Saída esperada** | **Resultados encontrados** | **Aprovado?** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Apropriação de Horas no Projeto

|  |  |  |
| --- | --- | --- |
| **Histórico de apropriação de horas** | | |
| **Data do registro** | **Atividade** | **Quantidade de horas** |
| 01/04/22 | Pesquisa e elaboração do problema. | 03 |
| 02/04/22 | Pesquisa das bibliografias e escrita da introdução do projeto. | 06 |
| 03/04/22 | Análise e escrita dos requisitos. | 04 |
| 04/04/22 | Análise e escrita dos requisitos. | 03 |
| 05/04/22 | Criação do diagrama de casos de uso. | 02 |
| 07/04/22 | Alterações nos requisitos e criação do diagrama de casos de uso. | 03 |
| 07/04/22 | Confecção dos wireframes das telas principais. | 04 |
| 08/04/22 | Término dos wireframes das telas principais. | 03 |
| 08/04/22 | Início da criação do protótipo. | 02 |
| 10/04/22 | Manutenção no protótipo. | 02 |
| 11/04/22 | Manutenção no protótipo. | 03 |
| 12/04/22 | Término do protótipo. | 01 |
| 13/04/22 | Análise dos requisitos e criação do diagrama de classes de domínio | 02 |
| 14/04/22 | Gravação do vídeo do protótipo. | 01 |
| 14/04/22 | Revisão do material produzido. | 02 |
| 14/04/22 | Criação do repositório com os arquivos solicitados. | 01 |
| 15/05/22 | Correções e alterações sugeridas pelo coordenador. | 03 |
| 08/07/22 | Escolha e estudo sobre as tecnologias utilizadas e os padrões arquiteturais. | 06 |
| 11/07/22 | Confecção do modelo C4. | 02 |
| 14/07/22 | Escolha dos frameworks e testes sobre os mesmos. | 10 |
| 23/07/22 | Escolha da funcionalidade e desenvolvimento do back end. | 25 |
| 01/08/22 | Desenvolvimento do front end e adaptações e correções no back end. | 40 |
| 22/08/22 | Deploy do banco de dados, back end e front end. | 05 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Código da Aplicação

Link para o repositório do projeto: <https://github.com/rodrigovk/projeto-integrado-engsoftware-pucminas.git>.

Site da aplicação: <https://rodrigovk.github.io/projeto-integrado-engsoftware-pucminas-frontend-dist>.

Credenciais para acessos de teste:

|  |  |  |
| --- | --- | --- |
| **E-mail** | **Senha** | **Função** |
| rodrigocasadeialanis@hotmail.com | Rodrigo@senha1234 | Administrador |
| flaviosduca@gmail.com | Flavio@senha5678 | Cliente |
| tadeurf7@gmail.com | Tadeu@senha1234 | Cliente |

A primeira funcionalidade implementada é o controle de tickets de suporte, consistindo em uma página onde todos os tickets de suporte são exibidos para o administrador logado ou todos os tickets de suporte próprios são exibidos para o cliente logado. Administradores podem encerrar e reabrir os tickets. E uma página acessada a partir de um desses tickets listados, onde são exibidos os detalhes do ticket escolhido e a lista de suas respostas, assim como novas respostas podem ser enviadas por administradores ou pelo cliente criador do ticket de suporte. Quando um administrador responde um ticket, o sistema envia automaticamente um e-mail para o cliente criador do ticket avisando-o da nova resposta.

Vídeo apresentando o sistema com sua primeira funcionalidade implementada: <https://github.com/rodrigovk/projeto-integrado-engsoftware-pucminas/blob/main/Documentos/Primeira%20funcionalidade.mp4>.

## Avaliação Retrospectiva

## Objetivos Estimados

## Objetivos Alcançados

## Lições aprendidas

|  |  |  |
| --- | --- | --- |
|  | **Retrospectiva (Lições Aprendidas)** | |
|  | **Descrição da Lição** | **Classificação** |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |
| 12 |  |  |

## Referências

MCGEE, James. PRUSAK, Laurence. **Gerenciamento estratégico da informação**: aumente a competividade e a eficiência de sua empresa utilizando a informação como uma ferramenta estratégica. Rio de Janeiro: Campus, 1994.

REZENDE, Denis A. ABREU, Aline F. **Tecnologia da informação**: aplicada a sistemas de informações empresariais. São Paulo: Atlas, 2000.

MENDES, Marco Aurélio de Souza. Arquitetura de Sistemas Web: Princípios, Práticas e Tecnologias. 2020. ed. Creative Commons, 2020.