UNIVERSIDADE FEDERAL DO CEARÁ

INSTITUTO UNIVERSIDADE VIRTUAL

CURSO DE GRADUAÇÃO EM SISTEMAS E MÍDIAS DIGITAIS

JEMIMA FONSECA LUZ

DIPLOMATA: UMA PLATAFORMA PARA FACILITAÇÃO DO ACESSO AO CONHECIMENTO CIENTÍFICO

FORTALEZA

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

L994d Luz, Jemima Fonseca.
Diplomata: uma plataforma para facilitação do acesso ao conhecimento científico / Jemima Fonseca Luz. – 2022.
83 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Instituto UFC Virtual, Curso de Sistemas e Mídias Digitais, Fortaleza, 2022.
Orientação: Prof. Dr. Leonardo Oliveira Moreira.

1. Acesso ao conhecimento. 2. Plataforma web. 3. Linguagem acadêmica. 4. Ilustração. 5. Vue. I. Título.
CDD 302.23

JEMIMA FONSECA LUZ

DIPLOMATA: UMA PLATAFORMA PARA FACILITAÇÃO DO ACESSO AO CONHECIMENTO CIENTÍFICO

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas e Mídias Digitais do Instituto Universidade Virtual da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas e Mídias Digitais.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Leonardo Oliveira Moreira (Orientador)
Universidade Federal do Ceará (UFC)

Profa. Me. Mara Franklin Bonates
Universidade Federal do Ceará (UFC)

Prof. Dr. Emanuel Ferreira Coutinho
Universidade Federal do Ceará (UFC)

Aos meus pais, por me incentivarem a seguir essa jornada desafiadora e repleta de momentos apaixonantes. Nunca poderia imaginar o quão rica seria toda essa caminhada.

AGRADECIMENTOS

Começo o fim desse ciclo, agradecendo a quem tornou o seu início possível. Aos meus pais: “seu Jamilton, e dona Jerusa” que sempre me transmitiram confiança e incentivo no início dessa fase onde eu pouco ou quase nada sabia sobre o que esperar da vida. O senso de responsabilidade talvez tenha vindo disso, vendo que eu mesmo sem a certeza de nada tinha o apoio dos dois. Com eles aprendi que confiança não é transmitida onde há certezas, é o que você transmite quando há incertezas. É criar a certeza onde não há.

Aos meus irmãos: José, que foi fundamental nos meus primeiros passos na programação. Lembro das noites em que ele me ajudava tirando dúvidas e me ensinando a pensar. À Verena, que sempre me inspirou pela sua maturidade. À Monique, que sempre me cativou com suas conversas, e me fez perceber que tudo é sobre a importância de aprender.

Ao Alek, por ter me acolhido na minha pior fase, e ter sido um porto seguro, mesmo com todas as minhas instabilidades. Agradeço por me ensinar não a ter, mas a ser esperança. Por me segurar em todas as vezes que cheguei ao meu limite por trabalhar demais e não descansar. Não tenho como mensurar o tamanho da benção que foi te conhecer.

Agradeço ao meu amigo Bruno e seus pais, por serem minha família em uma outra cidade, em uma nova realidade, sempre presentes para me ajudar e compartilhar bons momentos. Vocês me adotaram e eu adotei vocês.

Ao meu time maravilhoso chamado 5ights! Carmen, Emili, Rosi, e Bruno. Que equipe! Definitivamente a melhor parte do curso foi ter conhecido e trabalhado com vocês. Mesmo tendo nos afastado com o passar do tempo, guardo vocês com muito carinho!

Ao meu primo Lucas, por dividir comigo sua companhia em uma parte valiosa dessa jornada que foi a faculdade. Apesar de perdermos o contato, mantenho a lembrança divertida que foi essa fase.

Ao Prof. Leonardo, por me inspirar com seu trabalho, conhecimento, organização e por todo o suporte. Para mim foi uma honra tê-lo como orientador.

Se hoje me encontrei, é um reflexo das pessoas que me encontraram antes, e permitiram que eu aprendesse com elas. Carrego comigo a certeza que ao fim desse ciclo seguirei aprendendo e que, mais importante do que o quanto você sabe é o que você faz com o conhecimento que lhe foi entregue. A vida que vivemos é um momento no tempo em que nos submetemos à ignorância e dela não podemos sair sem a humildade de aprender com algo ou alguém. A todo momento aprendemos. E se pararmos, de alguma forma, morremos.

“E no meio do inverno, eu finalmente aprendi
que dentro de mim havia um verão invencível.”
(Albert Camus)

RESUMO

A produção científica é uma poderosa ferramenta, graças aos seus processos metodológicos sólidos e a linguagem precisa. Ao mesmo tempo, a linguagem acadêmica é também uma barreira para que o público geral acesse e possa usufruir desses conhecimentos tão valiosos. Existe um grande vão entre a população e o conhecimento científico, os desincentivos governamentais para com as universidades públicas e a crescente cultura de desinformação favorecida pela praticidade só aumentam esse vão. Entendendo o potencial da internet e suas tecnologias em aspectos como alcance, distribuição e acessibilidade, surge o questionamento: como podemos usar as tecnologias web para popularizar o acesso à ciência? Este trabalho se dispôs a desenvolver uma solução web que facilite o acesso aos conteúdos acadêmicos para o público em geral, sem comprometer suas fontes originais, fomentando a cultura de busca pelo conhecimento científico e visando reduzir as barreiras entre a sociedade e o meio acadêmico. Os resultados alcançados com o teste de usabilidade sobre o protótipo com voluntários, o questionário SUS (System Usability Scale) revelaram uma aderência positiva pelos usuários com solução proposta, bem como a realização da publicação da primeira versão em um domínio online atendendo todos os requisitos estabelecidos durante o processo de desenvolvimento.

Palavras-chave: Acesso ao conhecimento. Plataforma web. Linguagem acadêmica. Ilustração. Vue.

ABSTRACT

Scientific production is a powerful tool, thanks to its solid methodological processes and precise language. At the same time, academic language is also a barrier for the general public to access and enjoy this valuable knowledge. There is a large gap between population and scientific knowledge, government disincentives towards public universities and the growing culture of disinformation favored by practicality only widen this gap. Understanding the potential of the internet and its technologies in terms of reach, distribution and accessibility, the question arises: how can we use web technologies to popularize access to science? This work set out to develop a web solution that facilitates access to academic content for the general public, without compromising its original sources, fostering a culture of searching for scientific knowledge and aiming to reduce barriers between society and academia. The results achieved with the usability test on the prototype with volunteers, the SUS questionnaire (System Usability Scale) revealed a positive adherence by users with the proposed solution, as well as the publication of the first version in an online domain meeting all the established requirements. during the development process.

Keywords: Access to knowledge. Web platform. Academic language. Illustration. Vue.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

CSS	*Cascading Style Sheets*
DOM	*Document Object Model*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
IDE	*Integrated Development Environment*
MVC	*Model-View-Controller*
NPM	*Node Package Managment*
PNG	*Portable Network Graphics*
SMD	Sistemas e Mídias Digitais
SPAs	*Single-Page Applications*
SUS	*System Usability Scale*
TCLE	Termo de Consentimento Livre e Esclarecido
UFC	Universidade Federal do Ceará
URL	*Uniform Resource Locator*

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização e Motivação

O projeto Diplomata surgiu a partir de um trabalho na disciplina de Tópicos Avançados em Design de Interfaces Gráficas, do Bacharelado em Sistemas e Mídias Digitais (SMD) da Universidade Federal do Ceará (UFC). O desafio era desenvolver uma solução digital relacionada com a área da saúde. Dentro da metodologia da disciplina foram realizadas as etapas de exploração, análise da demanda em relação a proposta de solução e a prototipação do produto final. O requisito principal para condução do trabalho neste período era a criação de uma solução digital voltada ao contexto de saúde, e mais especificamente, relacionada com o problema da diabetes.

Na etapa de exploração foram realizadas: Uma pesquisa de artigos científicos para analisar o que a ciência tinha sobre diabetes até então; Uma pesquisa para descobrir quais soluções digitais relacionadas ao diabetes já estavam disponíveis ao público; Definição do problema.

Já na etapa de análise da demanda, foi realizada uma pesquisa com o público, da qual 112 voluntários participaram. Essa pesquisa abordou: dados demográficos como gênero, idade, nível de escolaridade e localização geográfica; Perguntas relacionadas com o acesso a literacia digital; Perguntas relacionadas ao uso de ferramentas de pesquisa científica; A relação dos voluntários com a produção científica. Os resultados da pesquisa de análise da demanda podem ser acompanhados através do link: Pesquisa sobre acesso à produção científica no Brasil.

Ao final da disciplina o Projeto Diplomata tem o seu primeiro protótipo de interface disponibilizado no *Figma*. E quanto aos resultados dos dados levantados pela pesquisa de demanda, 80% dos voluntários declaravam ser favoráveis ao projeto. Desse modo foi entendido que o protótipo da interface desenvolvido na disciplina poderia ganhar vida através de um sistema web, onde sua premissa foi adaptada para o Projeto de TCC de modo a ser uma solução digital que promova o letramento científico em seus usuários a partir de fontes acadêmicas, e não mais apenas se restringido ao contexto da saúde e diabetes.

De acordo com Tilly (2006 apud ANNAN, 2003), noventa e cinco por cento da ciência é criada nos países que abrigam apenas um quinto da população mundial, e na área da saúde por exemplo, grande parte da produção científica produzida negligencia os problemas que afligem a maioria da população mundial.

O acesso desigual ao conhecimento científico não é apenas um problema por si só, pois ainda nas ideias de Tilly (2006): “O conhecimento confere vantagens políticas, financeiras e existenciais aos que o detém.”. Os “donos” do conhecimento conseguem perpetuar as relações e instituições que sustentam seus privilégios, em detrimento aos privilégios de terceiros.

A produção científica é uma poderosa ferramenta, graças aos processos metodológicos sólidos e sua linguagem precisa. Ao mesmo tempo, o padrão elevado que se tornou uma base sólida para evoluirmos em diversas áreas (humanas, exatas e biológicas) se torna também uma barreira que impede as camadas mais vulneráveis da sociedade a acessarem esse conhecimento, devido à sua linguagem robusta e altamente técnica.

Diante dessa perspectiva, foi considerada a importância de se reforçar a facilitação do acesso à produção acadêmica. E por isso, o eixo do projeto foi ampliado para promover não apenas a literacia em saúde, mas em outras áreas científicas. Para lidar com o desafio de simplificar o acesso ao conhecimento de um artigo científico, dado o seu nível de complexidade, podemos considerar as mídias digitais como ferramentas aliadas. Uma vez que a própria publicação eletrônica, afirma Castro (2006), trouxe perspectivas infinitas para promover mudanças na cultura da comunicação científica. O acesso livre pela Internet contribui para a democratização e o acesso equitativo à informação científica.

Em paralelo, a ciência também vem sofrendo contínuos desincentivos, através da redução dos orçamentos das universidades públicas por parte do governo, crescimento da desinformação, e fomento da cultura negacionista através de redes sociais. Mas como a ciência pode ser defendida e se tornar uma prioridade novamente, se a maior parcela da população desconhece o seu valor? Como podemos facilitar o acesso ao conhecimento científico para pessoas leigas? Mais especificamente: como podemos aproveitar as tecnologias web para popularizar o acesso ao conhecimento científico? Assim surge o Projeto Diplomata como um recurso para ajudar na difusão e promover acessibilidade aos conteúdos acadêmicos.

1.2 Objetivos

O objetivo geral deste trabalho é desenvolver um sistema que facilite o acesso aos conteúdos acadêmicos para o público geral, sem comprometer suas fontes originais, a fim de reduzir as barreiras entre a sociedade e o meio acadêmico.

Segundo Marconi e Lakatos (2003) os objetivos específicos visam, de um lado, atingir o objetivo geral e, de outro, aplicá-los a situações particulares. Assim, para alcançar o

objetivo geral, os seguintes objetivos específicos foram elencados:

a) elucidar o problema da acessibilidade acadêmica;
b) definir a logística de simplificação do conhecimento de um artigo científico (como é feito o processo de facilitação e inserção de imagens ilustrativas para um artigo);
c) desenvolver uma plataforma web como solução proposta; e
d) avaliar o sistema por meio dos aspectos de usabilidade.

1.3 Estrutura do Documento

Esta relatório técnico está organizado em seis capítulos. O Capítulo 1 descreve a introdução, destacando a contextualização, motivação, questões de pesquisa e os objetivos. Já o Capítulo 2 aborda todos os conceitos teóricos necessários para uma melhor compreensão do trabalho. O Capítulo 3 apresenta todas os métodos científicos utilizados e também todas as etapas adotadas para concretização do trabalho. Já o Capítulo 4 analisa e discute os resultados experimentais da pesquisa para fins de validação do Diplomata. Por fim, o Capítulo 5 conclui o trabalho e apresenta os trabalhos futuros que podem dar continuidade ao presente trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os conceitos técnicos sobre aplicações web, design responsivo, e dos principais *frameworks* utilizados para o desenvolvimento do produto multimídia. Além disso também serão levantados os conceitos necessários para contextualizar o problema central que motivou o desenvolvimento do diplomata como desigualdade, desigualdade categórica e o impacto da internet no fluxo da comunicação científica.

2.1 Aplicações Web

As informações disponibilizadas na web são entregues, geralmente, por meio de documentos (TANENBAUM; STEEN, 2007). Esses documentos disponibilizados na web são chamados de páginas e podem ser desenvolvidos por meio de materiais multimidiáticos como textos, áudios, vídeos, animações, imagens, arquivos binários etc (NASCIMENTO *et al.*, 2021). Para Coulouris *et al.* (2013), a web tem evoluído sem mudar sua arquitetura básica que é baseada em três componentes tecnológicos padrões principais: i) *HyperText Markup Language* (HTML) que é utilizada como uma linguagem de marcação para estruturar e preencher a página web com o seu devido conteúdo, possibilitando a interpretação pelos navegadores web; ii) *Uniform Resource Locator* (URL) que é uma forma de identificar os documentos e outros recursos dispostos na web; e iii) arquitetura cliente-servidor (KUROSE; ROSS, 2010) com regras padrão para interação entre navegadores e outros clientes por meio do protocolo de rede *HyperText Transfer Protocol* (HTTP), facilitando a obtenção de documentos e outros recursos disponíveis em servidores web (NASCIMENTO *et al.*, 2021).

A base da web é a transferência de páginas do servidor para o cliente por meio do protocolo HTTP (TANENBAUM; STEEN, 2007). As páginas estáticas são arquivos mais simples mantidos no servidor com conteúdo fixo, ou seja, textos e mídias que não se alteram com requisições (SOUZA, 2018). Da mesma forma que as páginas estáticas, os servidores também possuem outros objetos ou recursos estáticos, tais como imagens, áudios, documentos de representação de dados etc. Estes objetos e recursos estáticos são caracterizados por não se modificarem, em tempo de execução, diante de uma requisição específica ou passagem de parâmetros (NASCIMENTO *et al.*, 2021).

Já as páginas, objetos ou recursos dinâmicos são caracterizados por terem seus conteúdos gerados, em tempo de execução, por meio das linguagens de programação (SOUZA,

2018). Além disso, eles possuem capacidades de receberem parâmetros e também se personalizarem conforme uma determinada requisição de um cliente. Assim, aplicações baseadas na web têm, em boa parte da sua implementação, a adoção de páginas, objetos e recursos dinâmicos. Em resumo, aplicações web possuem programas, escritos em alguma linguagem de programação, que são invocados por meio de URLs, podendo passar parâmetros e escolher o método HTTP usado para interagir com o programa (NASCIMENTO *et al.*, 2021).

2.2 Design Responsivo

O design responsivo resolve os problemas de compatibilidade de páginas da web exibidas em diferentes resoluções, diferentes plataformas e diferentes tamanhos de tela, além de trazer uma experiência de alta qualidade aos usuários (LI; ZHANG, 2019). Neste sentido, o conteúdo apresentado em páginas web com design responsivo se torna de amplo acesso, pois mitiga limitações de dispositivos, resoluções e plataformas. Além disso, não prejudica os aspectos de experiência do usuário ao utilizar o conteúdo mesmo em uma resolução menor.

Segundo Li e Zhang (2019) a ideia central de design web responsivo é "prioridade móvel e aprimoramento progressivo". A "prioridade móvel" significa que, ao projetar uma página, consideramos totalmente a diversidade de tamanho e resolução do dispositivo móvel e, primeiro, projetamos o efeito da página do dispositivo móvel, de modo a realizar o *layout* do dispositivo móvel adaptável. Em seguida, o efeito da página no lado do computador é levado em consideração, o que é benéfico para aumentar a eficiência do desenvolvimento da web (LI; ZHANG, 2019).

Já o "Aprimoramento progressivo" refere-se à conversão gradual da exibição da página de tamanho pequeno para tamanho grande sob a premissa de dar prioridade a dispositivos móveis. No espaço de exibição limitado de tamanho pequeno, o conteúdo da página deve ser destacado e o desempenho do conteúdo deve ser simplificado. Com o tamanho aumentando e o conteúdo inalterado, o desempenho do conteúdo deve ser aprimorado e a exibição da página melhor (LI; ZHANG, 2019).

2.3 Framework Vue.js

O *framework* Vue.js surgiu inspirado pelo *framework* AngularJS, ele foi criado pelo Evan You na época em que trabalhava no Google e lançado oficialmente em 2014 (FELIZARDO,

2018). Sua ideia foi criar uma versão mais leve e diferente do AngularJS, sem muitos recursos extra. Vue (pronuncia-se /vju/, como “view”, em inglês) é um *framework* progressivo para a construção de interfaces de usuário (VUEJS, 2022).

Ao contrário de outros *frameworks* monolíticos, Vue.js foi projetado desde sua concepção para ser adotável incrementalmente. A biblioteca principal é focada exclusivamente na camada visual (*view layer*), sendo fácil adotar e integrar com outras bibliotecas ou projetos existentes. Por outro lado, Vue.js também é perfeitamente capaz de dar poder a sofisticadas *Single-Page Applications* (SPAs) quando usado em conjunto com ferramentas modernas e bibliotecas de apoio (VUEJS, 2022).

Para entender o diferencial do Vue.js, é preciso colocar em perspectiva outros *frameworks front-end* semelhantes: AngularJS e React.js por exemplo. O AngularJS (criado e mantido pela Google) é um *framework* que separa bem as responsabilidades por arquivos, onde um componente possui em sua estrutura um arquivo HTML (*view*), um arquivo .ts (*controller*), e um *Cascading Style Sheets* (CSS) pelo menos. Essa organização o torna mais verboso, exige mais atenção do programador quanto às suas boas práticas e oferece um ecossistema fechado e auto-suficiente, com menor necessidade de recorrer a bibliotecas externas.

O React.js (criado e mantido pelo Facebook) por sua vez, abriga o *controller* e a *view* juntos em um arquivo .tsx, de modo que o componente pode se tratar de uma classe ou uma função que retorne HTML. Assim o React.js oferece maior flexibilidade, com um ecossistema aberto e maior necessidade de recorrer a bibliotecas externas, como a Redux (para controle de estado) e a React Router (para rotas). Desse modo podemos encarar o React.js como um *framework* modular, com menos exigências e padrões de como o código deve ser feito.

Já o Vue.js se enquadra no meio termo entre os extremos delimitados pelo AngularJS e pelo React.js. Ele mantém a divisão de responsabilidades proposta pelo Angular, mas aplica essa proposta de modo análogo à praticidade do React.js: dentro de um único arquivo. Sua estrutura é dividida em três partes - *template* HTML, *script* (*controller*) e *style* (CSS) - mas ao mesmo tempo ela não é obrigatória (NEGRÃO, 2022). Dessa forma o Vue.js transita entre os dois potenciais: a robustez proposta pelo Angular e a flexibilidade proposta pelo React.js. Além disso, é também um projeto *open source*, que diferente dos concorrentes é mantido abertamente pela comunidade. As principais características do Vue.js são:

- reutilização de código através do uso de componentes;
- gerenciamento de estado;

- facilidade para trabalhar com dados externos (*models*, como no padrão *Model-View-Controller* (MVC));
- maior nível de programação lógica permitida na *Document Object Model* (DOM);
- ganho de performance; e
- possui uma comunidade *open source* ativa.

2.4 Framework NodeJS

O NodeJS foi criado pelo pesquisador Ryan Dahl, em 2009. O principal diferencial do NodeJS gira em torno dele possibilitar o uso de JavaScript no lado do servidor (ONLINE, 2022). Antes da sua criação, a linguagem JavaScript era interpretada somente pelos navegadores. Com o surgimento do NodeJS, a possibilidade de sistemas terem JavaScript no *front-end* e no *back-end* trouxe com ele um grande ganho de performanceqwwwws.

Além disso o NodeJS também é *single-thread*, sendo que na maioria das linguagens usadas em servidores são *multi-threads*. Em um servidor *multi-thread* as diferentes requisições recebidas criam novas *threads*, essa *threads* por sua vez consomem mais recursos da máquina onde o servidor se encontra, deixando-a mais lenta. Já na *single-thread* do NodeJS isso não acontece pois todas as requisições recebidas são listadas nesse única *thread* e respondidas uma de cada vez (ONLINE, 2022).

O NodeJS também possui o *Node Package Managment* (NPM), um gerenciador de pacotes muito popular na comunidade de desenvolvimento web. Com ele é possível estabelecer um padrão de construção, instalação, atualização e desinstalação em bibliotecas usadas como dependências de um projeto web. As principais características do NodeJS são:

- construído em JavaScript, que é uma linguagem muito popular;
- possibilita o uso de JavaScript no lado do servidor;
- baixo consumo de memória e processamento de CPU; e
- possui uma comunidade *open source* ativa.

2.5 O problema da desigualdade

Segundo Tilly (2006), a desigualdade é uma relação entre pessoas ou conjuntos de pessoas na qual a interação gera mais vantagens para um dos lados. Sendo a desigualdade categórica a ocorrência dessa relação em cima de categorias que se definem por meio das

diferenças entre os seus grupos, como empregadores e empregados, professores e alunos, médicos e pacientes, e assim em diante.

Essa circunstância revela como a liberdade de uma parte produz a falta de liberdade da outra. Tilly (2006) pontua que apesar da retórica acadêmica em contrário, os produtores e distribuidores têm poucos incentivos para disseminar seus conhecimentos onde quer que eles possam gerar boas consequências, e muitos incentivos para impedir que isso ocorra.

Desse modo é possível imaginar que o potencial de impacto do meio acadêmico é comprometido pelo problema da desigualdade. De modo que os indivíduos mais próximos do meio acadêmico possuem acesso a conhecimentos que podem lhes assegurar vantagens tangentes ao esclarecimento cientifico, enquanto a parcela da população que não está inserida nesse meio perde pelo simples fato de desconhecê-lo.

2.6 O impacto da internet no fluxo da comunicação científica

Diante deste problema causado pela desigualdade, podemos questionar quais seriam alternativas possíveis para contorná-lo. Segundo Castro (2006), a internet democratizou o acesso à informação, permitindo que os países adotassem metodologias e tecnologias similares, independentemente de seu estágio de desenvolvimento. Mas não apenas isso como alterou o fluxo da comunicação científica que anteriormente era linear com o modelo de publicação eletrônica.

A internet rompeu barreiras geográficas e temporais nesse fluxo de comunicação, de modo que muitos processos sofreram reduções significativas na sua duração. Castro (2006) ressalta que: "O fluxo da comunicação científica tradicional, baseado em etapas sucessivas e dependentes entre si, com longos períodos de tempo entre cada instância, passa a ser realizado, no espaço virtual, sem imposições temporais e de espaço físico. A dinâmica de transmissão de informação e de publicação na Internet permite que as ações se sucedam concomitantemente, e não mais em intervalos regulares."

Como por exemplo: dois cientistas que moram em diferentes cidades e desejam escrever um artigo colaborativamente. No modelo de fluxo linear seria limitada à realização de encontros presenciais e comunicações baseadas em cartas e telefonia juntamente a todo o processo burocrático exigido para publicação em uma revista analógica. Já no ambiente virtual, é possível todo uma orquestra da produção através de recursos virtuais como conferências online para realização de reuniões, escrita colaborativa através de ferramentas editoriais que permitem

edição simultânea, e a própria publicação através de uma revista eletrônica.

Diante dos ganhos desse modelo digital em agilidade, dinamicidade e interatividade, faz sentido a hipótese de que a criação de uma ferramenta digital voltada para a facilitação do conhecimento científico pode apresentar um grande potencial na tarefa de distribuição da ciência ao publico geral.

3 METODOLOGIA

Este capítulo apresenta a metodologia utilizada para a construção do projeto, dividida cinco etapas importantes: considerações sobre a facilitação de conteúdos, tecnologias, desenvolvimento, resultados e próximos passos.

3.1 Considerações importantes sobre o processo de facilitação dos conteúdos acadêmicos

Partindo do questionamento de "como podemos facilitar o acesso ao conhecimento científico para o público geral?" precisamos ter em mente que o vocabulário presente nos artigos acadêmicos não é de fácil compreensão para pessoas com baixos índices de escolarização e até mesmo para pessoas que foram escolarizadas mas que perderam ou nunca tiveram contato com as universidades. Pois muitas vezes ele possui termos específicos ao contexto acadêmico daquele artigo. Na área de saúde temos muitos termos associados aos marcadores fisiológicos, nomes estruturas e compostos orgânicas. Esses termos tendem a ser melhor compreendidos por quem já está inserido no ecossistema em que aquele artigo foi produzido, como um aluno de um dos cursos da área de saúde, por exemplo. E ao mesmo tempo, também não seria oportuno simplificar o conteúdo textual de um artigo, pois pode comprometer a precisão do método científico.

Diante dessa compreensão, os artigos científicos presentes na plataforma deverão passar por um processo de facilitação do conhecimento. Esse processo consiste no emprego de ilustrações para acompanhar/representar os trechos textuais do artigo em questão, e será aplicado apenas nas seções de resumo (objetivos, métodos, resultados, conclusões).

Outra intenção da plataforma, além de trazer o conhecimento acadêmico para o público em geral, é despertar o interesse do usuário pelo conhecimento científico. Para isso, as ilustrações foram limitadas apenas aos trechos introdutórios do artigo. A fim de desmistificar o primeiro contato com o conteúdo ali presente, mas manter o desafio que a obra científica carrega consigo e preservar a cultura de desenvolvimento do pensar, instigando mais pessoas a se aventurarem nela.

A plataforma Diplomata será exclusivamente o veículo que irá disponibilizar esses artigos com versões ilustradas em um ambiente virtual.

3.2 Escolha de tecnologias e ferramentas

1. *Front-end*:
    - *Figma* para prototipação da interface.
    - *Framework Vue.js* para desenvolvimento da interface.
    - Plataforma *Vercel* para realização do *deploy*.
2. *Back-end*:
    - *MongoDB* como banco de dados.
    - *Framework Express.js* para desenvolvimento do servidor.
    - Plataforma *Heroku* para realização do *deploy*.

3.3 Processo de desenvolvimento

3.3.1 Conteúdo

O desenvolvimento da plataforma não envolve o processo de ilustração de um artigo, isto é: como um artigo é analisado e preparado para receber ilustrações com o objetivo de facilitar a compreensão do que está escrito nele. Esse deve ser um processo à parte. Neste trabalho partimos do ponto de partida onde já existe um artigo ilustrado (que foi ilustrado no período da disciplina de Tópicos Avançados em Design de Interfaces Gráficas), e que servirá como uma amostra de dados a fim de realizar testes e provar o funcionamento do sistema. A seguir foram documentadas as etapas necessárias para criação de um artigo ilustrado:

- Processo de ilustração do artigo escolhido como amostra:
    1. Criação de um protótipo no *Figma* com 2 versões: uma para telas grandes (*desktop* e *tablets*), e a outra para telas pequenas (celulares).
    2. Recorte de todo conteúdo que compõe o resumo do artigo.
    3. Fragmentação de cada parte do recorte (objetivo, método, resultados ..) em períodos.
    4. Disposição dos períodos em sequência, respeitando a ordem presente no artigo.
    5. Seleção de imagens / ícones de acordo com cada período.
    6. Organização da sequência respeitando os limites de cada versão (telas grandes e pequenas).
    7. Caso seja necessário: dispor de linhas para conectar cada etapa e/ou evidenciar a

ordem lógica / relações entre os períodos.

8. Geração (exportação) de duas imagens *.png*: uma para cada versão.

3.3.1.1 Critérios de seleção

Ainda no período da disciplina de Tópicos Avançados em Design de Interfaces Gráficas, durante a etapa de exploração houve um processo de pesquisa em equipe por artigos acadêmicos relacionados com a área da saúde (ser relacionado com a área da saúde foi um requisito definido na disciplina para orientação da pesquisa), esse processo foi responsável por definir o artigo que seria ilustrado e utilizado como exemplo de dados no sistema. A pesquisa foi conduzida em cima dos seguintes critérios:

- Índice: Título;
- Coleções/Países: Brasil;
- Periódicos: Scielo;
- Idioma: Português;
- Ano da publicação: 5 últimos anos (2021 até 2016);
- Tipo de literatura: Revisão sistemática ou mapeamento sistemático de artigos, artigos de revisão, relatos de caso.
- Áreas temáticas
  - Nutrição; Alimentos ultra-processados; Suplementação; Conscientização do diabetes; Diabetes no Brasil; Rotina de exercícios físicos e sua relação com o diabetes;
- Palavras chaves:
  - Diabetes; Síndrome metabólica; Obesidade; Alimentos ultra-processados; Alimentos saudáveis; Hipertrigliceridemia;
- Critérios de inclusão
  - Ser um estudo que trata sobre o contexto alimentar;
  - Ser um estudo que trata sobre alimentos ultra-processados;
  - Ser um estudo que trata sobre alimentação saudável;
  - Ser um estudo que trata sobre a conscientização do diabetes no brasil;
  - Ser um estudo que trata sobre exercícios físicos para pacientes com diabetes;
- Critérios de exclusão
  - Estudos secundário ou terciário;
  - Não ser publicado em periódico ou conferência revisada por pares;

- Ser escrito que outra língua que não língua portuguesa;
- Não se referir ao contexto de diabetes e/ou obesidade;
- Artigo com 3 ou menos páginas;
- Texto completo não é acessível;
- Ser um livro ou os anais de uma conferência;
- Ser um artigo publicado há mais de 5 anos;
- Ser um artigo com coleta de dados realizada há mais de 5 anos;

A partir da filtragem, foram selecionados 69 artigos e cada membro da equipe (3 no total) se encarregou de ler 23 artigos aplicando os critérios de inclusão e exclusão. Desses, 28 foram aprovados, e dos 28, 3 foram escolhidos para guiar um processo de formação de personas e levantamento de soluções para possíveis problemas envoltos à temática do artigo.

3.3.1.2 Artigo selecionado

Das 3 soluções levantadas a partir de cada um dos artigos, uma era a plataforma Diplomata, cujo a proposta envolvia promover a literacia em saúde baseada no artigo “Avaliação dos efeitos de um programa educativo em diabetes: ensaio clínico randomizado” que foi ilustrado para servir como amostra de dado no protótipo da interface construída no *Figma*, como mostram as Figuras 1 e 2.

Figura 1 – Interface (versão de telas grandes)

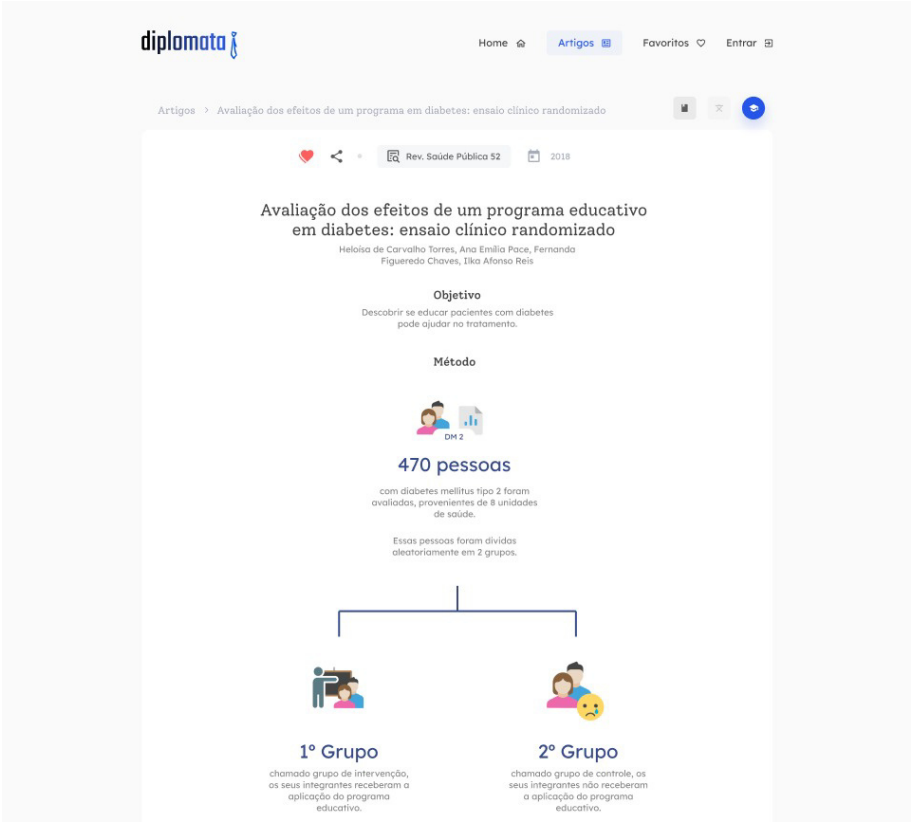


Figura 2 – Interface (versão de telas pequenas)

O artigo em questão pode ser encontrado no *link*: Revista de Saúde Pública v.52. Os links de imagens abaixo mostram o resultado final do processo de ilustração (não foram incluídas diretamente neste documento por possuírem um tamanho vertical muito extenso, o que compromete sua visualização):

- Link: Ilustração para telas pequenas
- Link: Ilustração para telas grandes

3.3.2 Levantamento de requisitos: Front-end

RF01 Menu de navegação responsivo para telas pequenas e grandes, contendo as opções: “Home”, “Artigos”, “Favoritos”, “Entrar”.

RF02 As páginas “Home”, “Favoritos” e “Entrar” deverão retornar um conteúdo do tipo “Em breve”. Dado o tempo de desenvolvimento, o escopo foi reduzido para entregar apenas a página “Artigos” apresentando conteúdo (que é onde os artigos com ilustrações serão apresentados).

RF03 I - Página “Artigos” exibindo uma listagem de artigos disponíveis, e uma seção lateral com filtros de categorias organizados por nomes.

RF03 II - Os itens da listagem de artigo devem conter: título, lista de autores, ano de publicação, revista/periódico onde foi publicado, botão para favoritar, botão para compartilhar e botão “Ler artigo”.

RF04 I - Página “Artigo” responsivo para telas pequenas e grandes, contendo os detalhes referentes ao artigo selecionado previamente na lista da página “Artigos”. Apresentando duas abas de exibições distintas: uma para informações gerais e outra para visualização das ilustrações.

RF04 II - A aba de exibição de informações gerais devem conter trechos sobre autores, palavras chaves e como realizar a referenciação do artigo.

RF04 III - Na aba de exibição do artigo ilustrado devem conter os parágrafos iniciais do artigos após passar pelo processo de facilitação do conteúdo científico, apresentando imagens e divisões claras entre os períodos dos parágrafos.

3.3.3 Levantamento de requisitos: Back-end

RF01 Realizar a persistência e coleta de dados de um banco externo não relacional (*MongoDB*).

RF02 Definição de um modelo de dados para os artigos.

RF03 Disponibilização das rotas para artigos.

3.3.4 Desenvolvimento: Back-end

3.3.4.1 Visão geral de bibliotecas

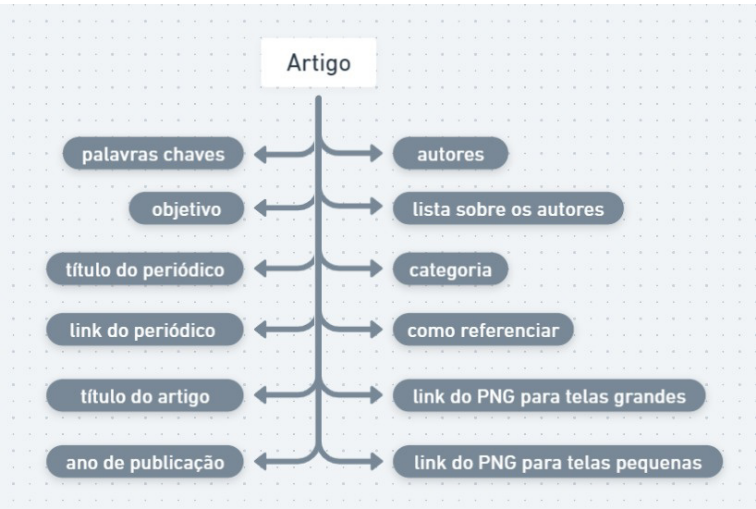
O repositório *back-end* é construído em cima do *framework Express.js*, além da própria biblioteca do *express*, ele conta com outras bibliotecas auxiliares como:

- *Cors* para permitir requisições originadas do *front-end*.
- *Dotenv* para gerenciamento de variáveis de ambiente.
- *Mongoose* para conexão com o banco *MongoDB* e preparação dos modelos de dados.
- *Nodemon* para ganho de performance no processo de desenvolvimento.

3.3.4.2 Modelo de dados

Foi necessário idealizar um modelo de dados para padronização dos artigos e suas informações dentro da aplicação. Para isso, usamos o seguinte modelo representado na Figura 3:

Figura 3 – Diagrama do modelo de dados de um artigo



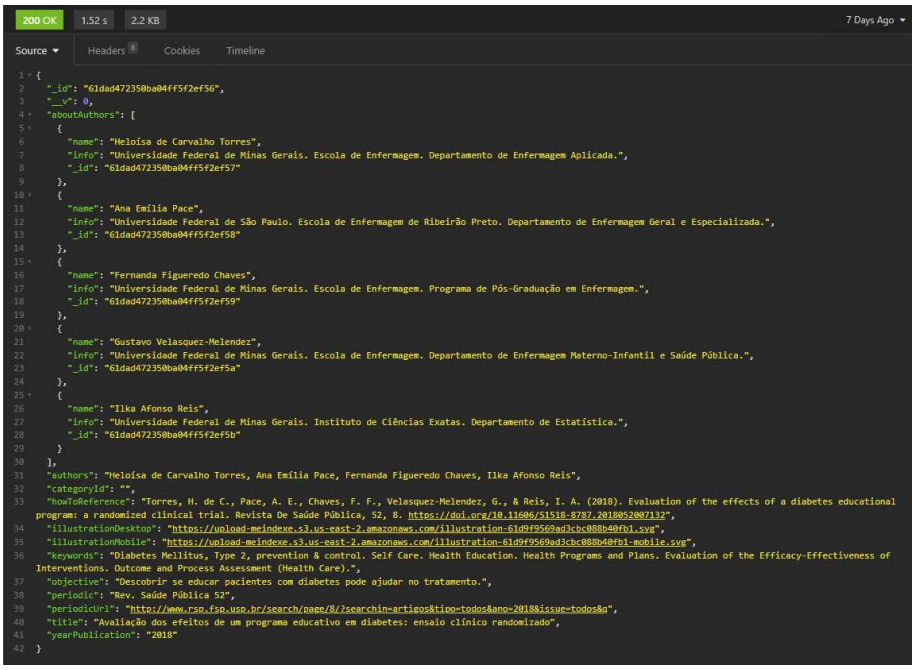
1. Atributo "**title**" para armazenar o título (tipo: texto).
2. Atributo "**authors**" para armazenar os nomes dos autores (tipo: texto).
3. Atributo "**aboutAuthors**" para armazenar uma lista com os detalhes sobre a formação dos autores (tipo: lista de objetos).
4. Atributo "**objective**" para armazenar o objetivo (tipo: texto).
5. Atributo "**keywords**" para armazenar palavras chaves (tipo: texto).
6. Atributo "**howToReference**" para armazenar um exemplo de referência do artigo (tipo: texto).
7. Atributo "**yearPublication**" para armazenar o ano de publicação (número).
8. Atributo "**periodic**" para armazenar o nome da revista ou periódico (tipo: texto).
9. Atributo "**periodicUrl**" para armazenar o *link* da revista ou periódico (tipo: texto).
10. Atributo "**categoryId**" para armazenar o *id* da categoria ao qual o artigo pertence (tipo: numero).
11. Atributo "**ilustrationDesktop**" para armazenar o *link* da ilustração *Portable Network Graphics* (PNG) na versão para telas grandes (tipo: texto).
12. Atributo "**ilustrationMobile**" para armazenar o *link* da ilustração PNG na versão para telas pequenas (tipo: texto).

Os atributos, assim como todo o código são escritos em inglês a fim de manter uma padronização para favorecer a comunidade de desenvolvedores e os próximos membros do projeto (Figura 4).

Figura 4 – Código: Definição do modelo de dados de artigo dentro do servidor com a utilização da biblioteca *mongoose* (linhas 1 até 22)

Durante a comunicação entre o servidor e o cliente, esse modelo é convertido da sua respectiva linguagem para o formato *JSON* e assim inversamente também ocorre. Nesse formato os dados trafegam entre *front-end* e *backend* de acordo com as requisições que vão sendo solicitadas entre os mesmos (Figura 5).

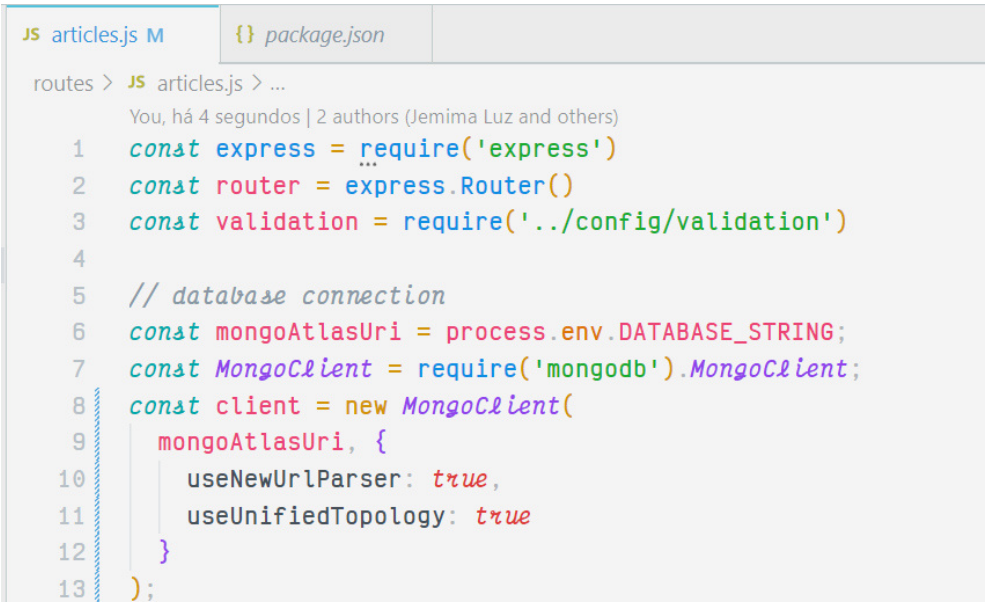
Figura 5 – Exemplo do modelo de um artigo em formato *JSON* (formato que é fornecido do sistema backend para o frontend realizar a exibição)



3.3.4.3 Conexão com o banco de dados

A conexão do servidor com o banco é feita no arquivo de rotas de artigos da pasta *routes/articles.js*. Para criação do roteador utilizamos a função *Router()* da biblioteca *express* (linha 2). Em seguida declaramos a *URI* de conexão chamada *mongoAtlasUri* que é uma variável de ambiente (linha 6), e utilizamos a classe *MongoClient* fornecida pela biblioteca *mongodb* para configurar a conexão com o banco (linhas 7 até 13) como é mostrado na Figura 6.

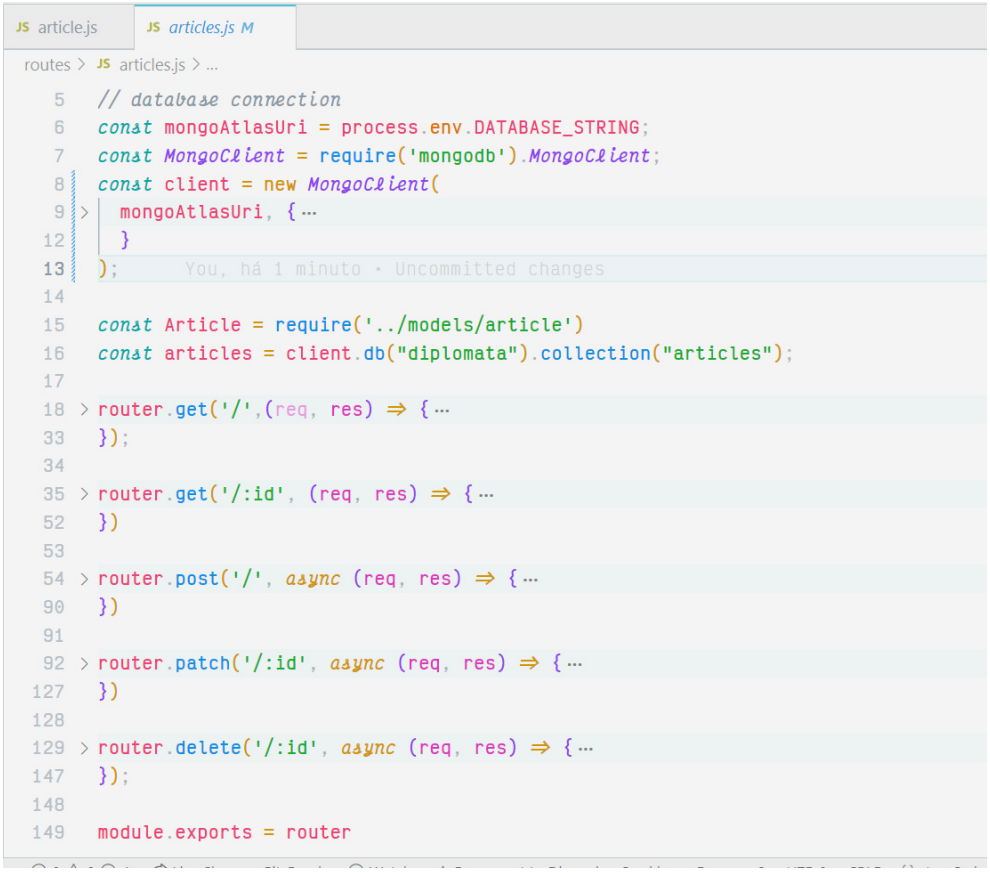
Figura 6 – Código: Configuração de conexão com o banco de dados (linhas 6 a 13)



3.3.4.4 Definição das rotas

Após a configuração da conexão foram criadas as rotas para os artigos. Nessa etapa foi necessária a importação do modelo de dados (linha 15) presente no aquivo da pasta */models/article.js* apresentado pela Figura 7 e também o registro de uma variável chamada *articles* (linha 16) que remete a coleção de artigos presente no banco de dados. A partir desse ponto já é possível a criação das rotas (linhas 18 até 147) e por fim ocorre a exportação do roteador como um módulo (linha 149), para que ele possa ser importado por outros artigos como o *server.js*.

Figura 7 – Código: criação do roteador e definição das 5 rotas (GET all, GET by id, POST, PATCH e DELETE)



Toda vez que uma rota é executada, ela irá abrir uma conexão com o banco de dados. Para que isso ocorra, utiliza-se a função *connect()* (linha 19 da Figura 8) daquela variável *client* para realizar essa tarefa. Dentro dela é construído todo o procedimento de consulta, manipulação ou remoção sobre modelo de dados correspondente ao tipo de método REST daquela rota (linhas 20 até 31).

Figura 8 – Código: Exemplo da conexão sendo aberta em dentro de uma rota GET

Para utilização das rotas pelo servidor, é realizada a importação do roteador dentro do arquivo *server.js* (linha 10 da Figura 9) para vinculação do mesmo a um caminho. Essa vinculação acontece por meio da função *use()* (linha 20) acessada através da instância do servidor *express* (variável *app* declarada na linha 4).

Figura 9 – Código: Importação e utilização do módulo *articlesRouter* dentro da aplicação

3.3.5 Desenvolvimento: Frontend

3.3.5.1 Visão geral de bibliotecas

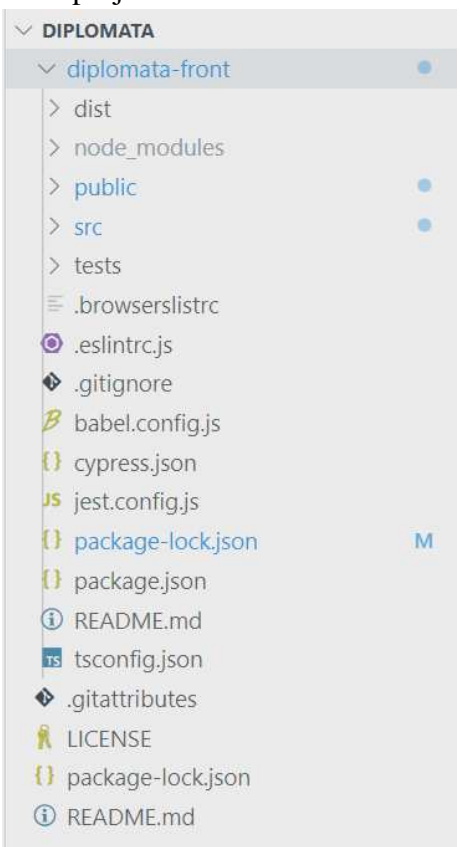
O repositório *front-end* é construído em cima do *framework vue.js*, além da própria biblioteca do *vue*, ele conta com outras bibliotecas auxiliares como:

- *Vuex* para gerenciamento de estado.
- *Vue* router para gerenciamento de rotas.
- *Dotenv* para carregamento de variáveis sensíveis.
- *Axios* para comunicação via requisições com o *back-end*.
- *Sass* como pre-processador CSS.

3.3.5.2 Apresentando a estrutura de pastas do projeto

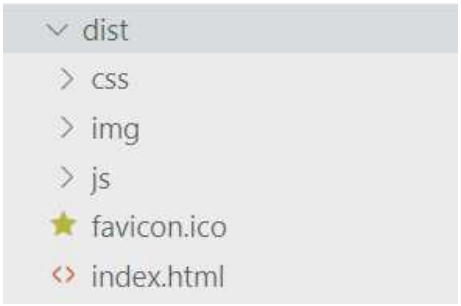
A Figura 10 mostra o diretório raiz, que foi criado e organizado pelo *vue-cli*, onde a partir do comando “vue create my-project”, ele já prepara toda estrutura minimamente necessária para um projeto *vue*.

Figura 10 – Estrutura de pastas do projeto



dist: é onde ficam localizados os arquivos finais do projeto, conforme a Figura 11 mostra, os quais são gerados através do comando “npm run build”.

Figura 11 – Pasta dist



public: é onde fica localizado o arquivo *.html* de entrada da versão de desenvolvimento (Figura 12). Esse arquivo contém alguns vínculos importantes como *link* de fontes, bibliotecas CSS e principalmente, a *tag* raiz ** ** que será correspondente a instância *vue*, é através dessa *tag* que o *vue* será vinculado ao HTML.

Figura 12 – Código: arquivo public/dist/index.html

index.html M

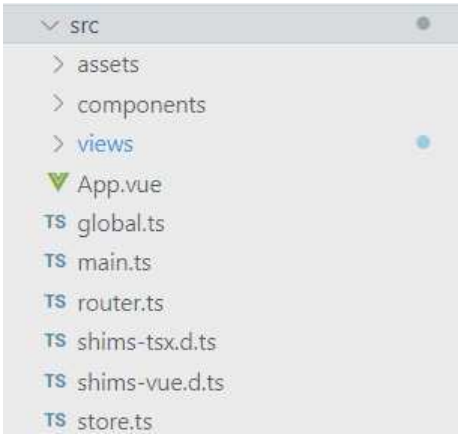
diplomata-front > public > index.html > ...

You, há 15 segundos | 2 authors (Jemima Luz and others)

```
1  <!DOCTYPE html>
2  <html lang="">
3    <head>
4      <meta charset="utf-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width,initial-scale=1.0">
7      <link rel="icon" href="<%= BASE_URL %>favicon.ico">
8      <title><%= htmlWebpackPlugin.options.title %></title>
9
10     <!-- Biblioteca tailwind -->   "Biblioteca": Unknown word.
11     <link href="https://unpkg.com/tailwindcss@^2/dist/tailwind.min.css" rel="stylesheet">
12
13     <!-- Google Fonts -->
14     <link rel="preconnect" href="https://fonts.googleapis.com">
15     <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
16
17   </head>
18   <body style="background-color: #f6f6f6;">
19     <noscript>
20       <strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't work properly
         without JavaScript enabled. Please enable it to continue.</strong>
21     </noscript>
22     <div id="app"></div>
23     <!-- built files will be auto injected -->
24   </body>
25 </html>
26
```

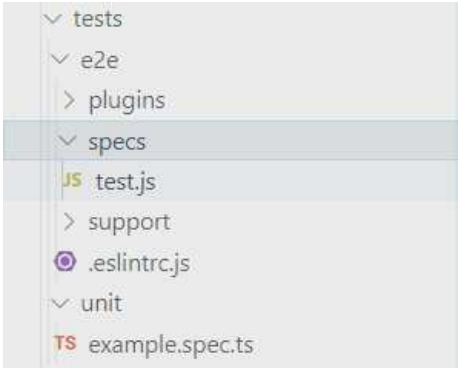
src: é a pasta onde se encontram os arquivos de desenvolvimento do projeto, todo o trabalho de desenvolvimento se concentra nessa pasta (Figura 13). Essa pasta será aprofundada no tópico 3.3.5.4.

Figura 13 – Pasta src



tests: essa pasta é onde ficam os testes em código da aplicação, ela é gerada automaticamente durante a criação do projeto pelo *vue-cli* (Figura 14). Ela pode englobar os teste unitários, *end-to-end* e os testes de integração. Neste caso não houve trabalho de desenvolvimento de testes devido ao curto prazo para desenvolvimento do projeto.

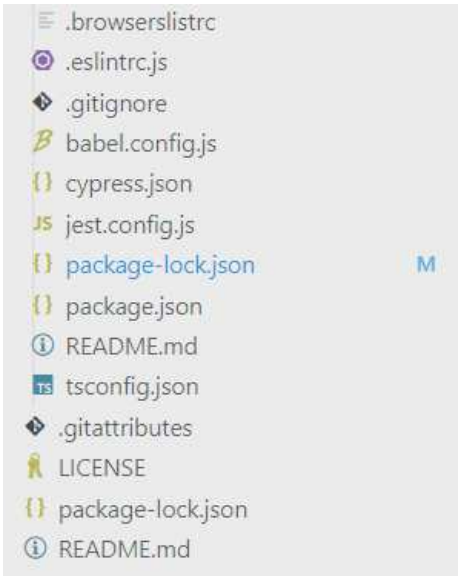
Figura 14 – Pasta tests



3.3.5.3 Arquivos de configurações

Os demais arquivos presentes na raiz do projeto são arquivos de configurações diversas (Figura 15). Desde configurações de quais arquivos serão rastreados pelo *git* como ".gitignore", até configurações da biblioteca de testes *end-2-end* "cypress.json" e gerenciamento das bibliotecas utilizadas no projeto como "package.json" e "package-lock.json.

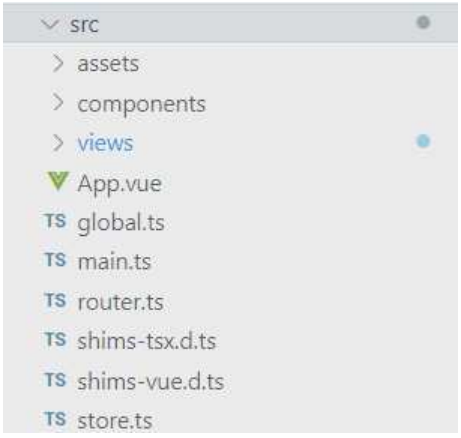
Figura 15 – Arquivos de configurações



3.3.5.4 A estrutura da pasta src

A pasta src quando expandida se apresenta da seguinte forma (Figura 16):

Figura 16 – A estrutura de pastas do src

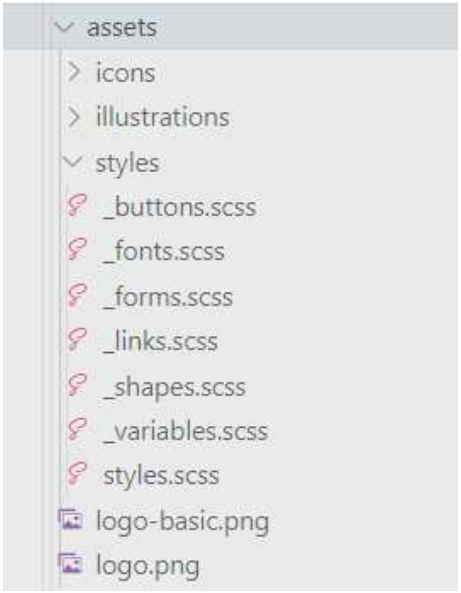


assets: é a pasta onde ficam imagens e os *partials* do *sass*.

Os *partials* são arquivos de código *sass* que não são renderizados na *build* do projeto, dessa forma eles fornecem códigos de estilos que podem ser utilizados sob demanda por outras partes do projeto, sem sobrecarregar o tamanho da *build* final. Um *partial* é caracterizado por um sublinhado na frente do nome do arquivo, dessa forma ele é ignorado pelo processo que converte código *sass* em CSS durante a *build* do projeto.

Na pasta *assets/styles* (Figura 17) foram centralizados estilos de fontes, botões, formulários, *links*, formas e variáveis.

Figura 17 – Diretório assets



Os estilos presentes nos *partials* estão principalmente na forma de *mixins*, um recurso do *sass* que funciona como um *template* para reutilização de códigos CSS (Figura 18). Ao criar um *mixin*, dentro do seu escopo é inserido algum código CSS, e toda vez que ele for referenciado com um *"@include"*, esse código é replicado para o seletor que o recebeu.

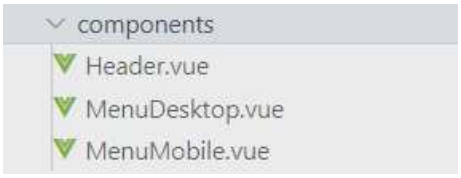
Figura 18 – Código: Exemplo de um mixin no css

Na Figura 19, a linha 64 atribui todo o estilo do *mixin* “articleItemReadButtonAppearance” para o seletor “.go-to-article”.

Figura 19 – Código: Exemplo de um mixin sendo usado na estilização de um componente

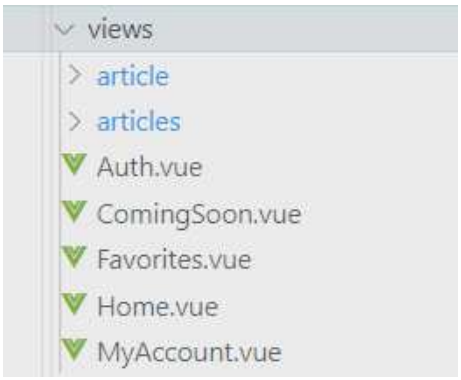
components: aqui ficam arquivos de componentes *.vue* gerais da aplicação (Figura 20).

Figura 20 – Diretório components



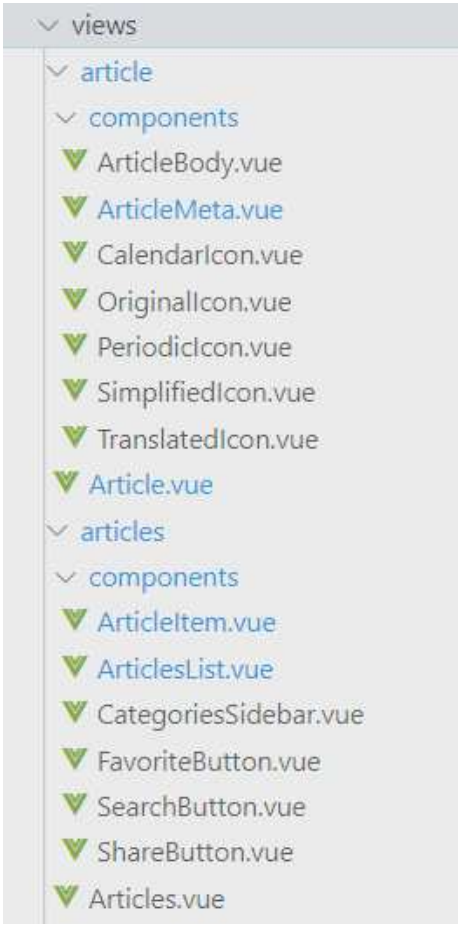
views: nessa pasta ficam as telas *.vue* que são análogas as páginas do site (Figura 21).

Figura 21 – Diretório views



No caso das pastas *views/article* e *views/articles* (Figura 22), elas foram separadas em pastas para organização dos seus componentes específicos (essas foram as únicas *views* desenvolvidas na versão piloto, justamente por isso elas já tem seus componentes definidos).

Figura 22 – Diretório views: exibindo detalhes dos diretórios article e articles



3.3.5.5 Os arquivos principais

App.vue: Enquanto o *main.ts* cria a instancia *vue*, o arquivo *app* abriga a *tag* (de mesmo atributo *id* do arquivo *public/index.html*) que irá se conectar com o HTML de entrada da versão de desenvolvimento.

Desse modo o arquivo “App” representa o conteúdo que irá preencher o HTML de entrada, e funciona como um ponto de partida para todo o resto do código que será desenvolvido em *vue*, podendo ser considerado como o “coração” da árvore de componentes *vue*.

No atributo *class* da *div #app* (linha 2 da Figura 23) podemos observar a utilização do *tailwind* para fornecer alguns estilos de responsividade.

Figura 23 – Código: arquivo App.vue

global.ts: Arquivo para armazenamento de variáveis globais (não é um arquivo principal, mas auxiliar), representado pela Figura 24.

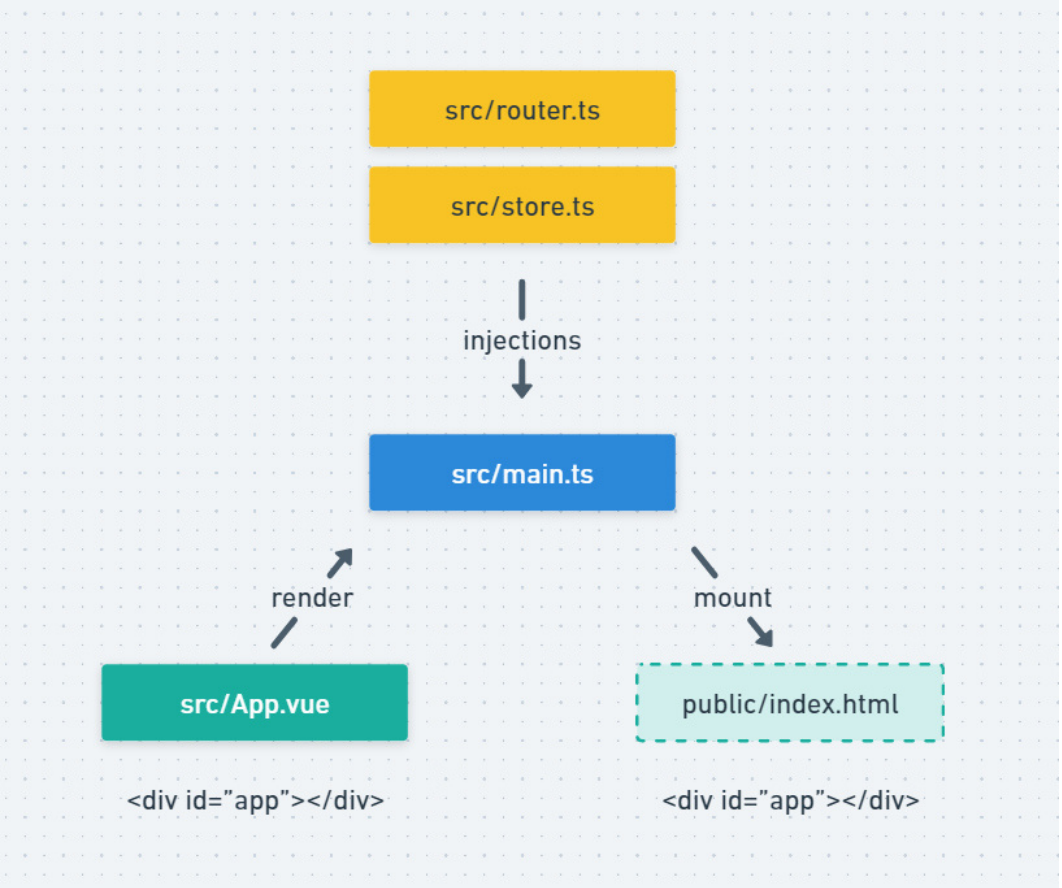
Figura 24 – Código: arquivo global.ts

main.ts: É onde fica a instância *vue*, nele ela é criada, preenchida (com *router* e *store*), e montada dentro do HTML de entrada da pasta */public*, utilizando o conteúdo do *App.vue* para preenchê-lo (Figura 25).

Figura 25 – Código: arquivo main.ts

O diagrama ilustrado pela Figure 26 mostra este processo:

Figura 26 – Diagrama: como funciona o arquivo main.ts



router.ts: é onde vinculamos cada *view* a uma rota, como por exemplo */articles* que está vinculada à *view Articles* (linha 19 da Figura 27). Essas rotas ficam armazenadas na variável "routes" (linha 14) que por sua vez é utilizada dentro da instância do roteador (linha 28).

Figura 27 – Código: arquivo router.ts

TS router.ts M

diplomata-front > src > TS router.ts > ...

```
1  import Vue from 'vue'
2  import VueRouter, { RouteConfig } from 'vue-router'
3
4  // import Home from '../views'
5  import Articles from './views/articles/Articles.vue'
6  import Article from './views/article/Article.vue'
7  // import Favorites from './views/Favorites.vue'
8  // import MyAccount from './views/MyAccount.vue'
9  // import Auth from './views/Auth.vue'
10 import ComingSoon from './views/ComingSoon.vue'
11
12 Vue.use(VueRouter)
13
14 const routes: Array<RouteConfig> = [
15   { path: '/', name: 'Home', component: ComingSoon },
16   { path: '/auth', name: 'Auth', component: ComingSoon },
17   { path: '/favorites', name: 'Favorites', component: ComingSoon },
18   { path: '/my-account', name: 'MyAccount', component: ComingSoon },
19   { path: '/articles', name: 'Articles', component: Articles },
20   {
21     path: '/articles/:id',
22     name: 'Article',
23     component: Article,
24     props: true
25   },
26 ]
27
28 const router = new VueRouter({
29   mode: 'history',
30   base: process.env.BASE_URL,
31   routes
32 })
33
34 export default router
```

shims-tsx.d.ts e shims-vue.d.ts: não são arquivos principais, mas ajudam a *Integrated Development Environment* (IDE) a reconhecer e operar com o formato ".vue", eles facilitam as importações e exportações entre arquivos *.vue*.

store.ts: nesse arquivo é onde fica o gerenciamento de estado da aplicação (Figura 28), útil para um cenário onde por exemplo exista um usuário e o mesmo favoritou alguns artigos. Com o *store* é possível centralizar essas informações para o caso de serem requisitadas por

diferentes componentes. No caso desse projeto, não houve desenvolvimento em cima do *store* devido ao escopo reduzido.

Figura 28 – Código: arquivo store.ts

3.3.5.6 Anatomia dos componentes

O componente principal *App.vue* abriga o componente *Header* (que dispõe do menu de navegação e seus respectivos *links*) e uma *tag* <*router-view*> para exibir dinamicamente as *views* de acordo com as rotas que são ativadas pelo menu do *Header.vue*.

As *views* “Auth.vue”, “Favorites.vue”, “Home.vue” e “MyAccount.vue” não foram desenvolvidas na versão piloto, mas já tiveram seus arquivos definidos para favorecer a organização do projeto e a delimitação das rotas. A *view* “ComingSoon.vue” é utilizada como conteúdo provisório a ser exibido nessas rotas que não foram desenvolvidas (Figuras 29, 30, 31).

Figura 29 – Diagrama: como funciona o arquivo App.vue

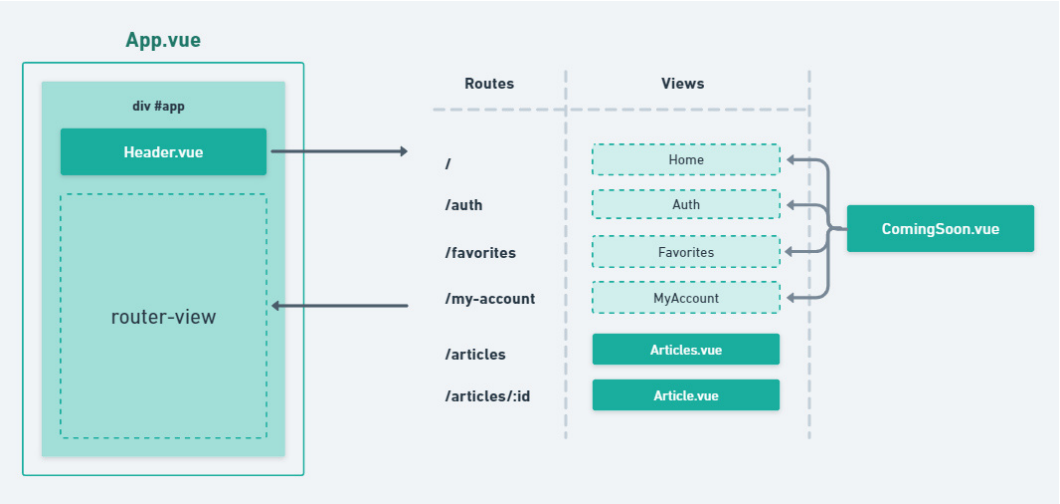


Figura 30 – Código: trecho template do arquivo App.vue

App.vue M

diplomata-front > src > App.vue > Vetur > {} "App.vue"

You, há 1 segundo | 2 authors (Jemima Luz and others)

```
1  <template>
2    <div id="app" class="mx-auto lg:w-5/6 xl:w-4/5 2xl:mx-auto">
3      <Header />
4      <router-view />
5    </div>
6  </template>
```

Figura 31 – Código: arquivo Header.vue

App.vue M | Header.vue M

diplomata-front > src > components > Header.vue > Vetur > {} "Header.vue"

You, há 6 segundos | 2 authors (Jemima Luz and others)

```
1  <template>
2
3    <header>
4      <MenuDesktop />
5      <MenuMobile />
6    </header>
7
8  </template>
```

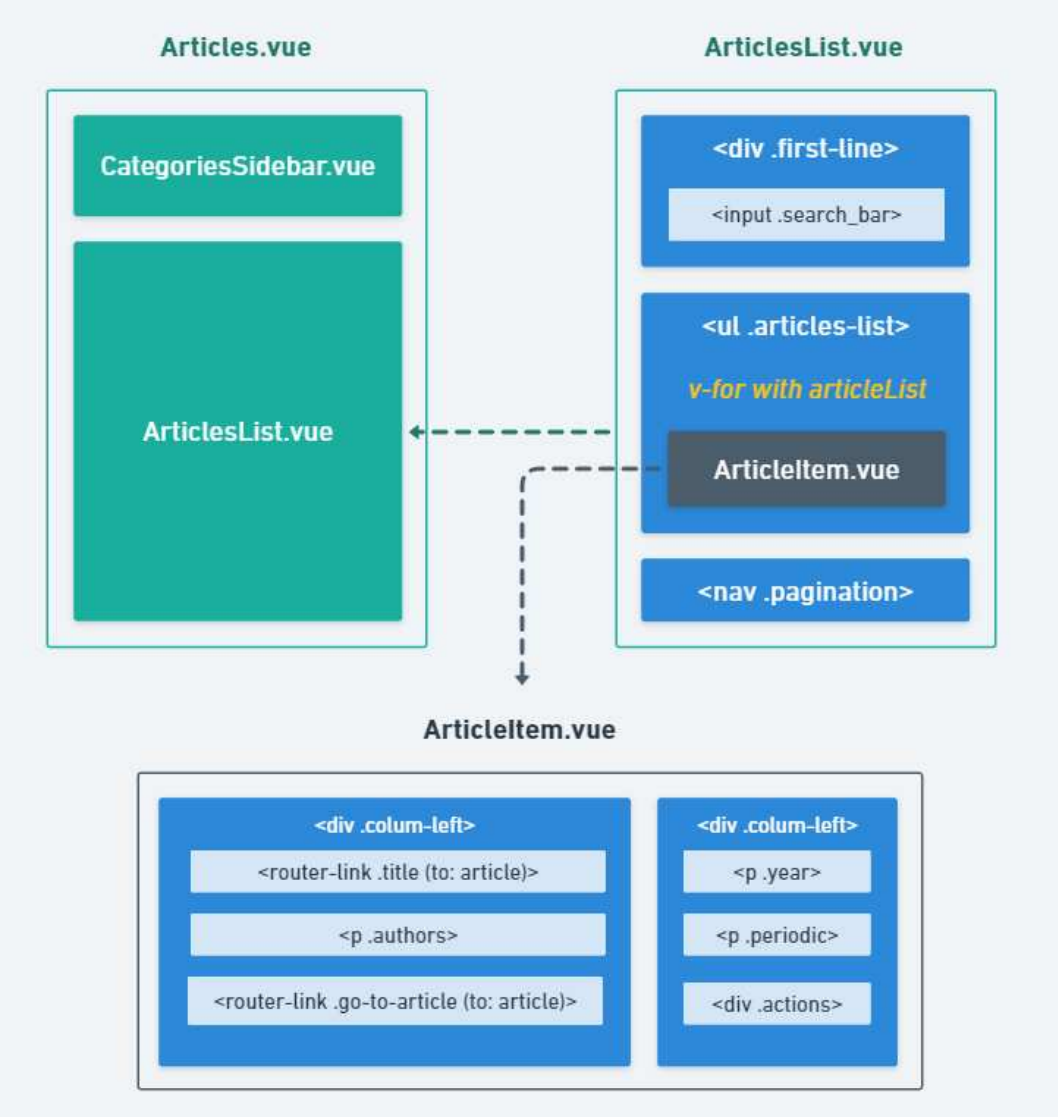
Ao acessar a rota */articles*, a *view Articles.vue* é exibida (Figura 32). Ela é composta por 2 componentes principais:

- O componente *CategoriesSidebar.vue* que exibe o painel de categorias (visível apenas em telas grandes).
- O componente *ArticlesList.vue* que exibe a listagem dos artigos disponíveis na plataforma.

No componente *ArticlesList.vue*, temos 3 partes principais:

- Uma *div* nomeada “first-line” que abriga a barra de pesquisa.
- Uma *div* nomeada “articles-list” que exibe a listagem dos artigos através do componente *ArticleItem.vue*.
- Uma *tag nav* nomeada “pagination” onde ficam os botões de navegação entre páginas.

Figura 32 – Diagrama: estrutura de componentes do arquivo Articles.vue



O componente filho *ArticlesList.vue* (Figura 33) recebe a lista de artigos como *props* (*:articlesList*) (linha 4).

Figura 33 – Código: trecho template do arquivo Articles.vue

Articles.vue M

diplomata-front > src > views > articles > Articles.vue > Vetur > {} "Articles.vue"

You, há 1 segundo | 2 authors (You and others)

```
1  <template>
2    <div id="articles">
3      <CategoriesSidebar class="md:ml-5 lg:ml-0" />
4      <ArticlesList :articlesList="articles" class="md:mr-5 lg:mr-0" />
5    </div>
6  </template>
```

Durante a inicialização (linha 40 a 42 da Figura 34) do ciclo de vida do componente *Articles.vue*, a função *getArticles()* (linhas 30 a 39) é chamada para carregar as informações fornecidas pelo *back-end* e preencher a lista “articles”.

Figura 34 – Código: trecho script do arquivo Articles.vue

Articles.vue M

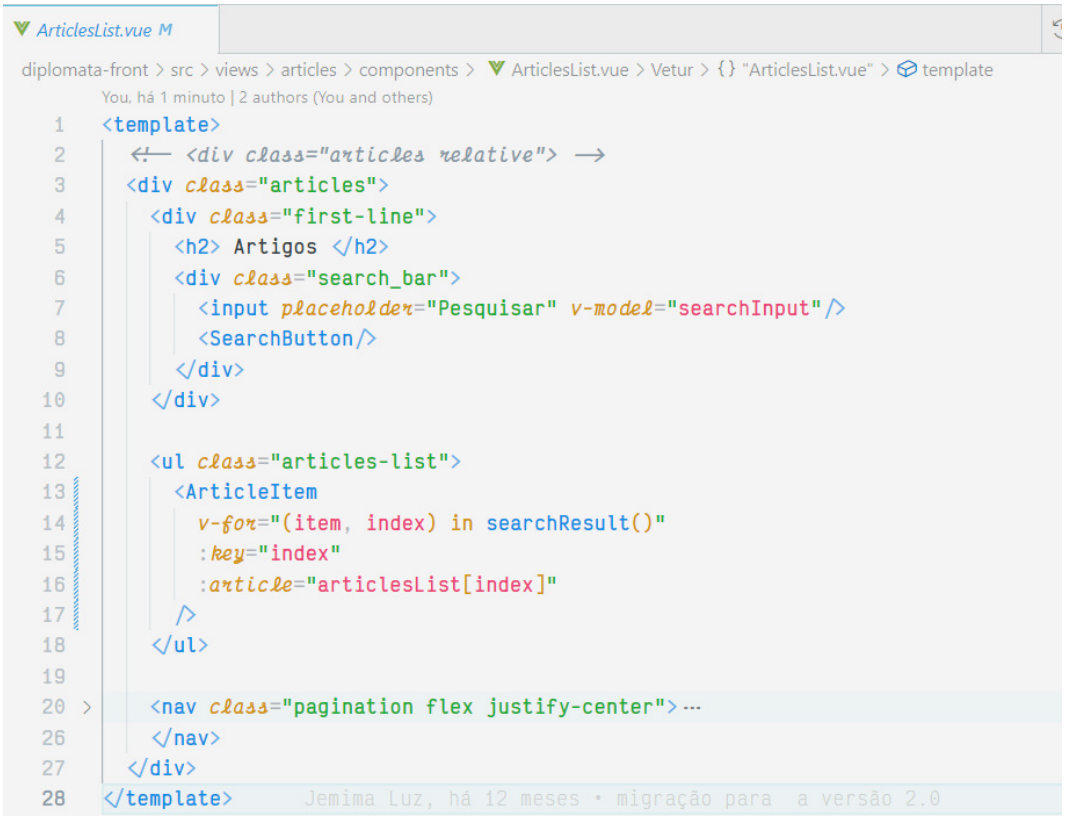
diplomata-front > src > views > articles > Articles.vue > Vetur > {} "Articles.vue"

```
8  <script>   Virtual script not found, may missing <script lang="ts"> / "allowJs"
9  import Vue from 'vue';
10 import Component from 'vue-class-component';
11 import axios from 'axios'    "axios": Unknown word.
12
13 import { apiUrl } from '../../global'
14
15 import CategoriesSidebar from './components/CategoriesSidebar.vue'
16 import ArticlesList from './components/ArticlesList.vue'
17
18 @Component({
19   name: 'Articles',
20   components: {
21     CategoriesSidebar,
22     ArticlesList,
23   },
24   data() {
25     return {
26       articles: []
27     }
28   },
29   methods: {
30     getArticles() {
31       axios.get(`${apiUrl}/articles`).then(    "axios": Unknown word.
32         res => {
33           this.articles = res.data
34         }
35       ).catch(
36         err => { console.log(err) }
37       )
38     }
39   },
40   mounted() {
41     this.getArticles();
```

O mecanismo de listagem de artigos ocorre através da diretiva *v-for* integrada à uma função de filtragem *searchInput()* (linha 14). A diretiva *v-for* já opera em cima da lista que foi retornada após o mecanismo de filtragem para realizar a exibição dinâmica dos itens inclusos nela. Cada item da lista é passado como valor à *props* (*:article*) (linha 16 da Figura 36). Representados pelas Figuras 35, 36.

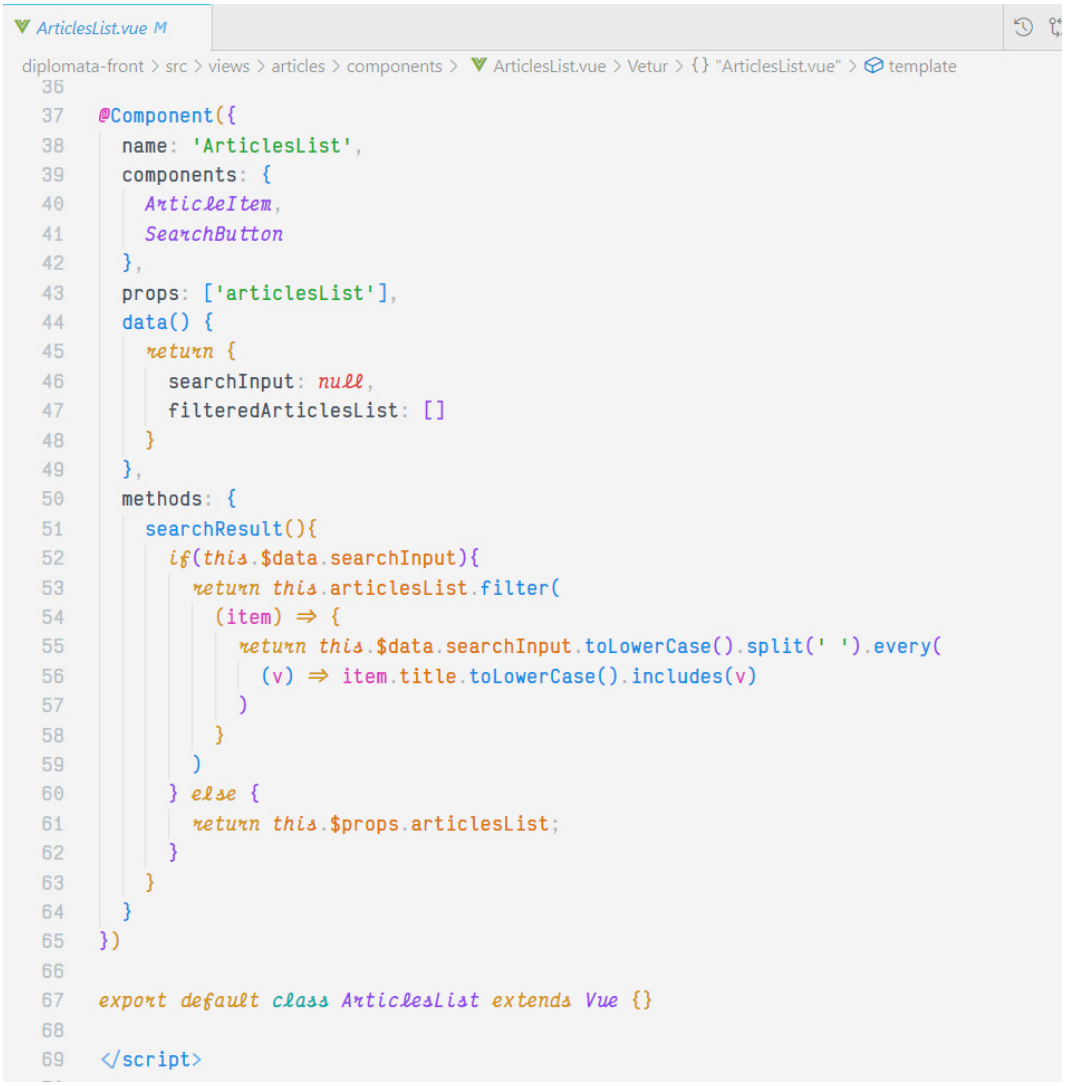
Figura 35 – Diagrama: como funciona o componente ArticlesList.vue

Figura 36 – Código: trecho template do arquivo ArticlesList.vue



A função de filtro utiliza o valor do *v-model searchInput* para filtrar resultados na lista de artigos (*articlesList*), retornando uma lista com os resultados correspondentes como mostra a Figura 37.

Figura 37 – Código: trecho do script do arquivo ArticlesList.vue



Nas Figuras 38, 39 e 40, o componente *ArticleItem* recebe como *props* um objeto *article* e utiliza-o para renderizar as informações do artigo no HTML através da interpolação de dados entre o *script* e o *template*.

Figura 38 – Diagrama: como funciona o arquivo ArticleItem.vue

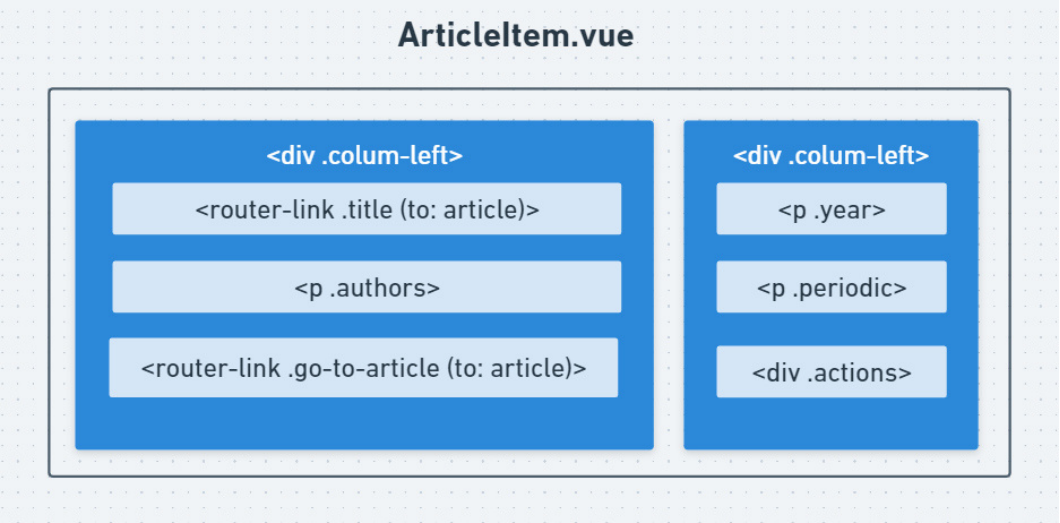


Figura 39 – Código: trecho do template do arquivo ArticleItem.vue

ArticleItem.vue M

diplomata-front > src > views > articles > components > ArticleItem.vue > Vetur > { } "ArticleItem.vue"

You, há 19 segundos | 2 authors (Jemima Luz and others)

```
1  <template>
2
3  <li class="article-item">
4    <div class="column-left sm:w-3/5 md:3/4">
5      <router-link :to="{ path: '/articles/' + article._id }" class="title">
6        {{ article.title }}
7      </router-link>
8
9      <p class="authors hidden sm:flex"> {{ article.authors }} </p>
10
11     <p class="authors sm:hidden"> {{ article.authors.slice(0,20) + '... ' }} </p>
12
13     <router-link to="" class="go-to-article"> Ler artigo </router-link>
14   </div>
15
16   <div class="column-right sm:w-2/5 md:1/4">
17     <p class="year"> {{ article.yearPublication }} </p>
18     <p class="periodic"> {{ article.periodic }} </p>
19     <div class="actions">
20       <ShareButton/>
21       <FavoriteButton/>
22     </div>
23   </div>
24 </li>
25
26 </template>
27
28 > <script>   Virtual script not found, may missing <script lang="ts"> / "allowJs": true /
42
```

Figura 40 – Código: trecho do script do arquivo ArticleItem.vue

```
ArticleItem.vue M
diplomata-front > src > views > articles > components > ArticleItem.vue > Vetur > {} "ArticleItem.vue"
You, há 1 minuto | 2 authors (Jemima Luz and others)
1  > <template> ···
27
28   <script>    Virtual script not found, may missing <script lang="ts"> / "a
29   import { Component, Vue } from 'vue-property-decorator';
30
31   import ShareButton from './ShareButton.vue'
32   import FavoriteButton from './FavoriteButton.vue'
33
34   @Component({
35     name: 'ArticleItem',
36     components: { ShareButton, FavoriteButton },
37     props: ['article']
38   })
39
40   export default class ArticleItem extends Vue {}
41   </script>
42
```

Uma vez acessado a rota */articles/:id* de um item, a *view Article.vue* é exibida. Ela pode ser dividida em 3 partes relevantes (representadas pelas Figuras 41 e 42):

- Uma *div* de classe “article_header” onde ficam os botões de ação (compartilhar e favoritar), o ano e a revista da publicação.
- O componente *ArticleMeta.vue* (que só aparece quando o valor da variável *currentView* é igual ao valor “original”). Ele apresenta as informações sobre os autores, palavras chaves e como referenciar o artigo.
- O componente *ArticleBody.vue* (que só aparece quando o valor da variável *currentView* é igual ao valor “simplified”). Ele exibe o conteúdo que foi simplificado com ilustrações.

Figura 41 – Diagrama: como funcionam os tipos de exibições de um artigo

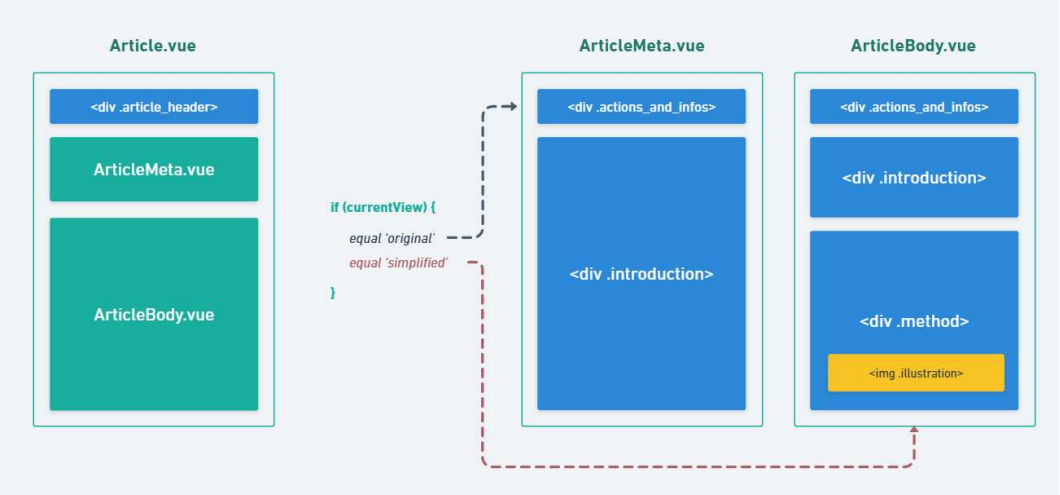
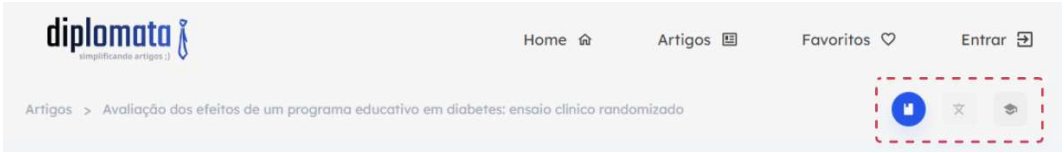


Figura 42 – Código: trecho do template do arquivo Article.vue

O mecanismo de cambiamento da variável *currentView* acontece dentro da *div* “article_header”, que dispõe dos botões que definem o tipo de exibição do artigo em questão (Figura 43). De acordo com qual botão está ativo, a variável *currentView* tem um valor diferente:

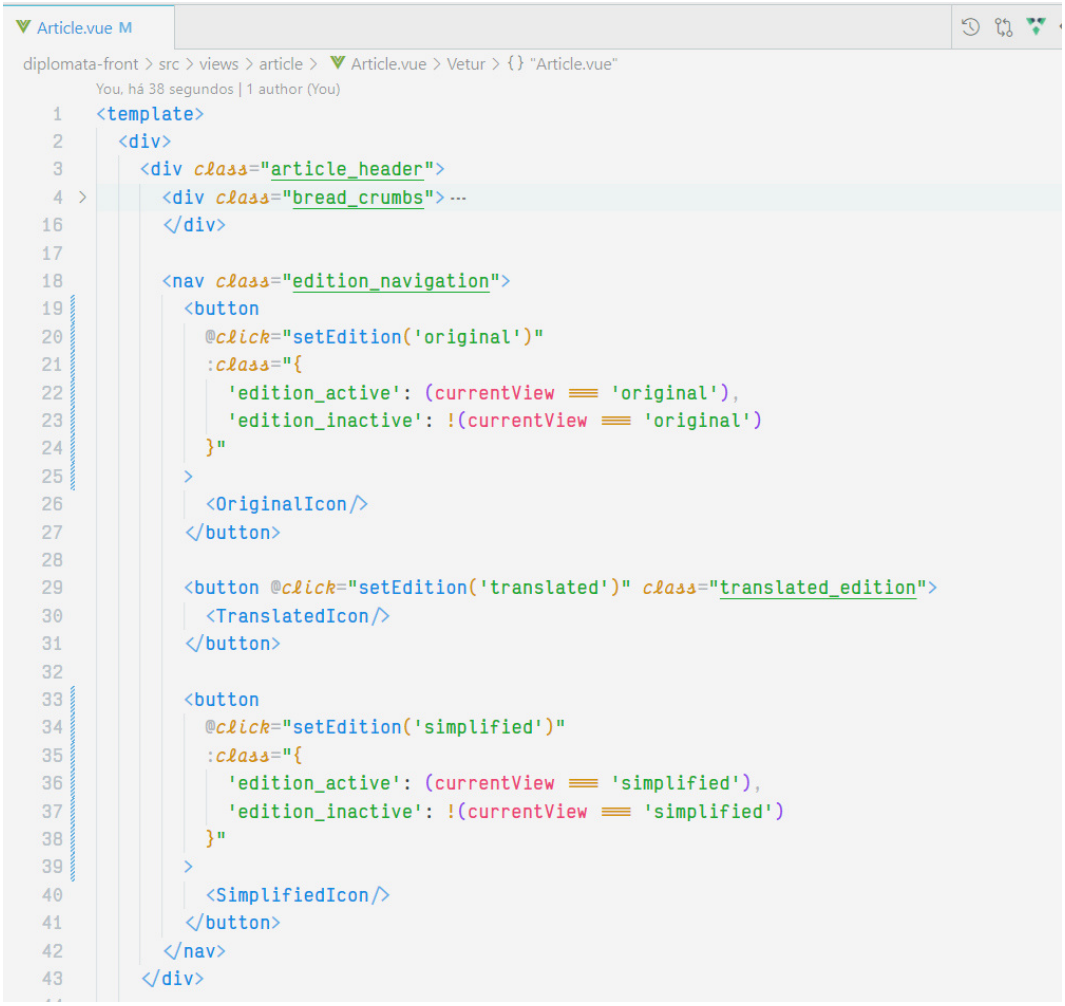
- Botão com ícone de livro, atribui o valor “original” (linhas 18 a 27).
- Botão com ícone de idioma (desabilitado), atribui o valor “translated” (linhas 29 a 31).
- Botão com ícone de capelo, atribui o valor “simplified” (linhas 33 a 39).

Figura 43 – Código: article interface views.png



Os botões possuem classes com estilos ativos e inativos (.edition_active e .edition_inactive) que são interpoladas de acordo com o valor assumido pela variável *currentView* (linhas 21 a 24 e 35 a 38 da Figura 44), dessa forma quando um botão está ativo, os outros estarão inativos.

Figura 44 – Código: trecho do template do arquivo Article.vue



A função *setEdition()* é responsável por atualizar o valor da variável *currentView* (linha 88 a 90 da Figura 45) baseada nos eventos de cliques nos botões que definem o tipo de exibição (linha 19 a 39).

Figura 45 – Código: trecho do script do arquivo Article.vue

As Figuras 46 e 47 mostram que a exibição das informações do artigo nas *views* *ArticleMeta* e *ArticleBody view* também acontecem por meio da interpolação de dados entre *script* e *template*. Ambos componentes recebem um objeto artigo como *props*.

Figura 46 – Código: trecho do template do arquivo ArticleMeta.vue

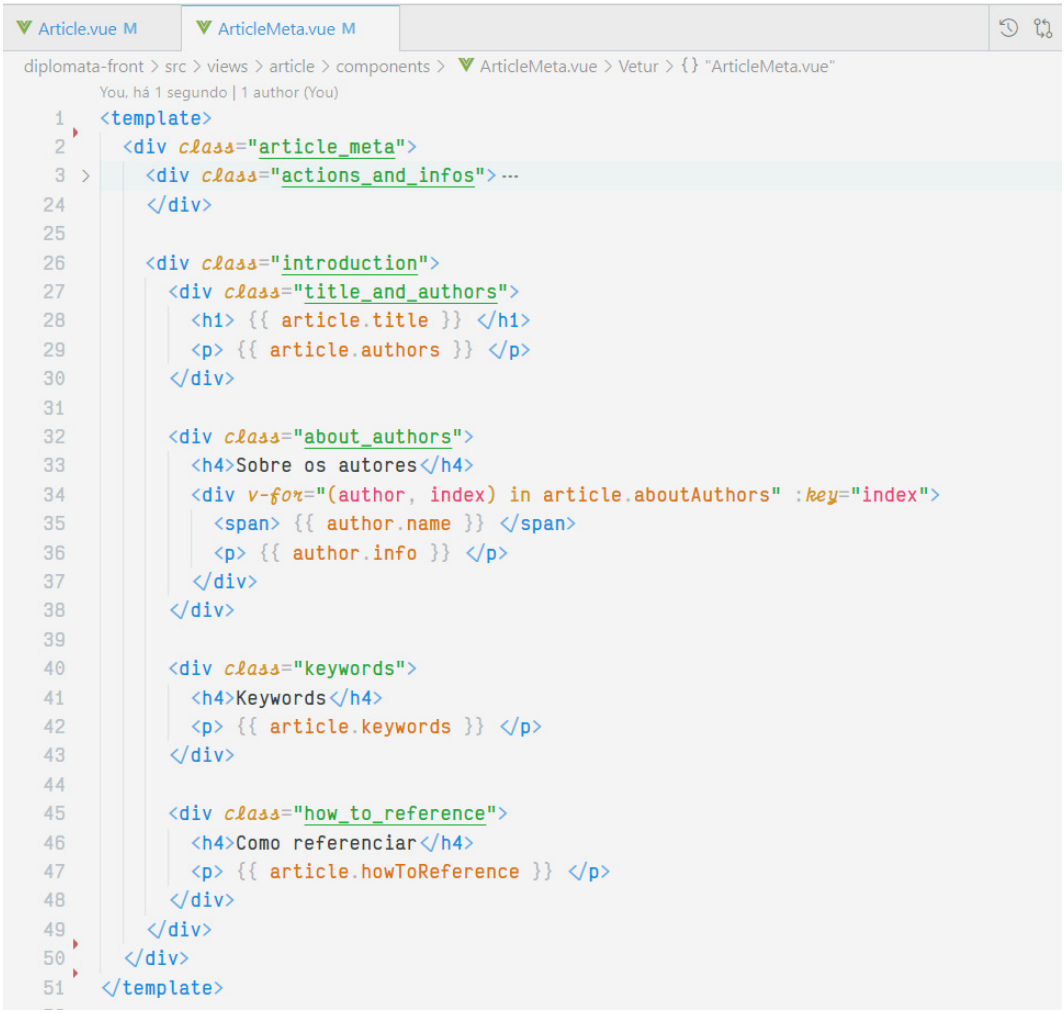
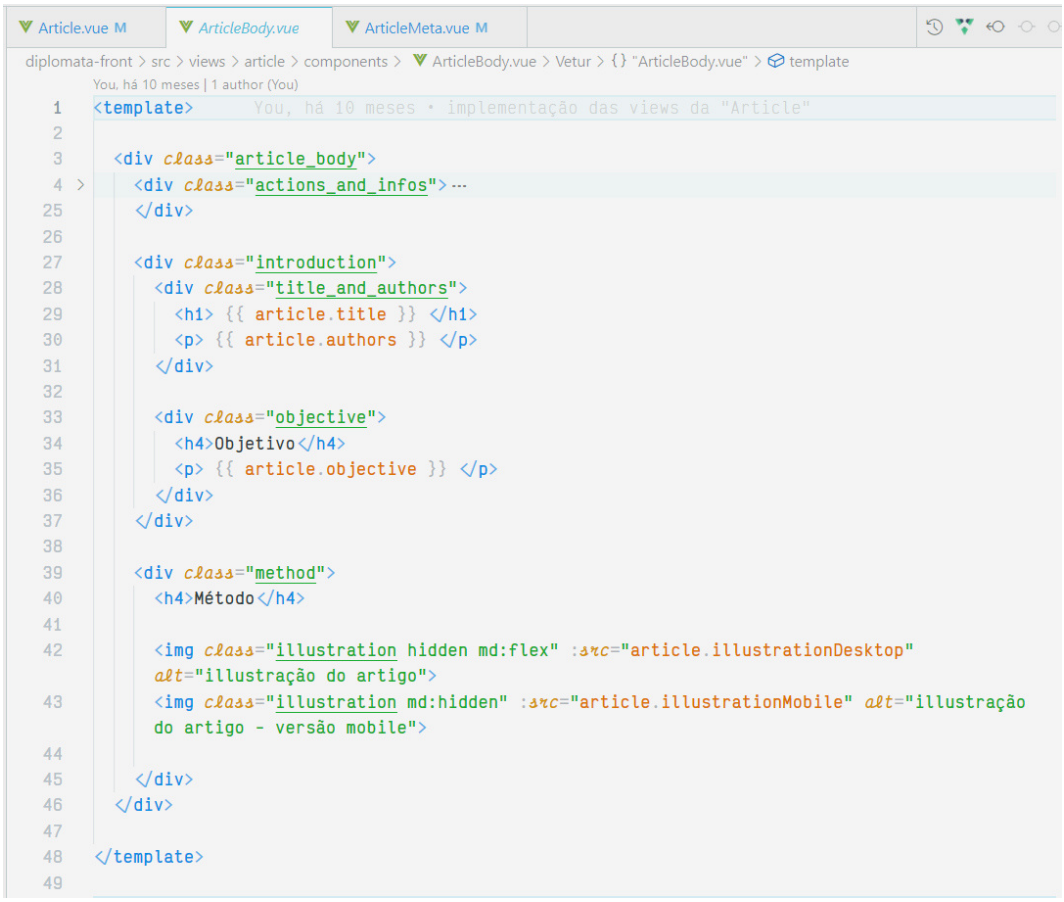


Figura 47 – Código: trecho do template do arquivo ArticleBody.vue



3.4 Análise dos resultados

O processo de análise consiste na aplicação de um teste quantitativo e um questionário de usabilidade do tipo *System Usability Scale* (SUS) (BOUCINHA; TAROUCO, 2013). O teste envolve a realização de 4 atividades com um protótipo da interface, seguida do questionário de SUS com 10 perguntas. As tarefas elencadas para o teste foram:

1. Navegar da página “Home” para a página “Artigos”.
2. Abrir a página de detalhes de um artigo em específico.
3. Navegar com o *scroll* do mouse pela sequência de ilustração de um artigo na exibição simplificada.
4. Acessar a exibição de informações do artigo como autores, palavras chaves e como referenciar.

O questionário por sua vez, irá medir a usabilidade do produto a partir de 10 premissas que serão avaliados com a escala Lickert (JúNIOR; COSTA, 2014):

1. Eu acho que gostaria de usar esse sistema com frequência.
2. Eu acho o sistema desnecessariamente complexo.
3. Eu achei o sistema fácil de usar.
4. Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.
5. Eu acho que as várias funções do sistema estão muito bem integradas.
6. Eu acho que o sistema apresenta muita inconsistência.
7. Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente.
8. Eu achei o sistema atrapalhado de usar.
9. Eu me senti confiante ao usar o sistema.
10. Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.

O teste de usabilidade e o questionário foram realizado com 12 integrantes, sendo aplicado através de uma plataforma voltada para testes de usabilidade: o <https://maze.co/>. Com essa plataforma todo o processo de coleta de dados pôde ocorrer de forma não linear, onde os voluntários podem realizar os testes de modo independente e online, em paralelo a coleta de dados de outros usuários simultâneos. Não apenas a coleta como o processamento de dados também é favorecido, visto que a própria plataforma organiza os dados e gera gráficos com métricas a partir disso.

É importante ressaltar a necessidade de um Termo de Consentimento Livre e Es-

clarecido (TCLE) a ser lido e aceito pelos voluntários logo no início da pesquisa. Esse termo resguarda a universidade em relação ao direito de utilização dos dados que são coletados na pesquisa. O exemplo a seguir foi utilizado nos testes executados pelo *maze*. O conteúdo de apresentação TCLE pode ser visualizado no Apêndice A.

4 RESULTADOS

Este capítulo trata dos resultados obtidos com o teste e questionário de usabilidade, bem como o resultado final do desenvolvimento da plataforma, considerando os requisitos funcionais levantados no capítulo de metodologia.

4.1 Teste e questionário de usabilidade

O teste de usabilidade apresenta os resultados com porcentagens que são dividas em 3 grupos possíveis: 1° Direct Success (Sucesso direto): Usuários que completaram a tarefa seguindo o fluxo esperado. 2° Indirect Success (Sucesso indireto): Usuários que completaram a tarefa sem seguir o fluxo esperado. 3° Give-up / Bounce (Desistir / Pular): Usuários que desistiram ou pularam para próxima tarefa

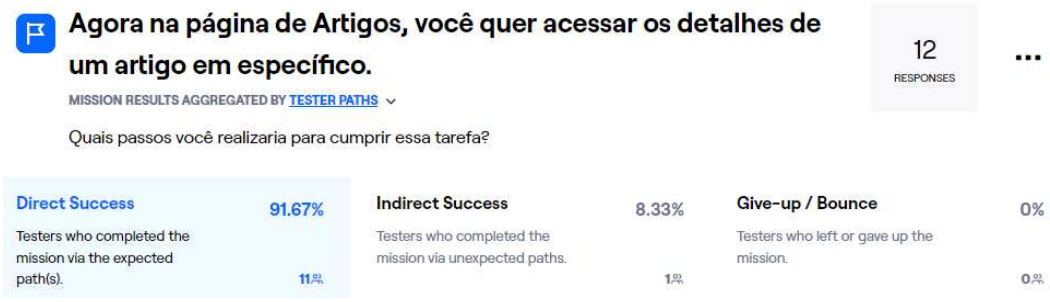
A seguir serão listadas as tarefas do teste de usabilidade juntamente com suas respectivas taxas.

Tarefa 1: 100% de sucesso direto, 0% de sucesso indireto e 0% de desistência/pulos. [48]

Figura 48 – Tarefa 1: Navegar da página "Home"para a página "Artigos"

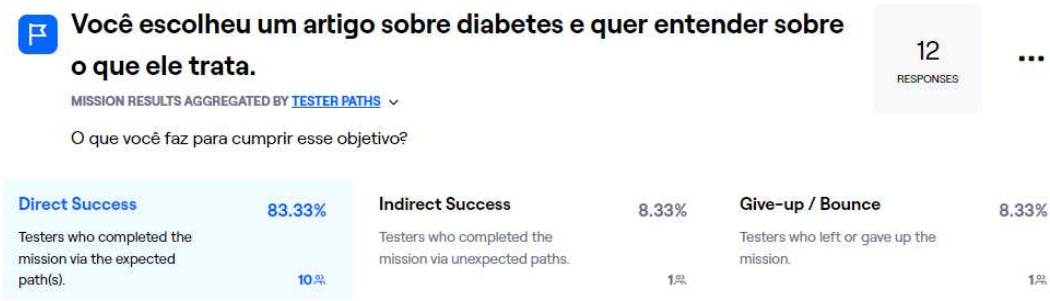
Tarefa 2: 91.67% de sucesso direto, 8.33% de sucesso indireto e 0% de desistência/-pulos. [49]

Figura 49 – Tarefa 2: Abrir a página de detalhes de um artigo em específico.



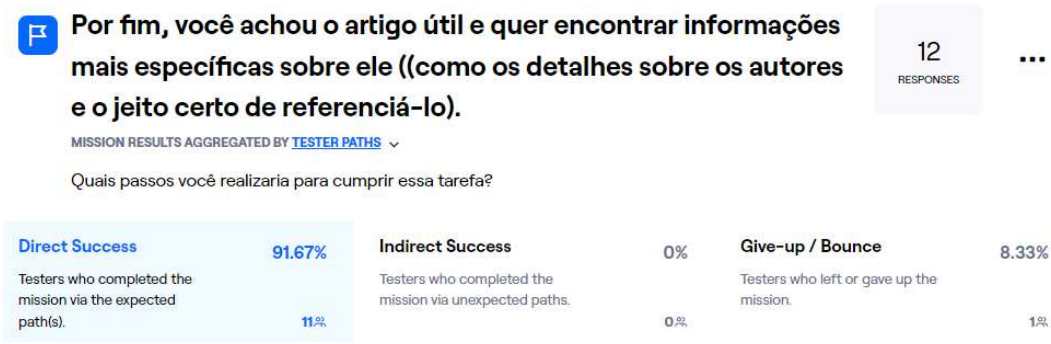
Tarefa 3: 83.33% de sucesso direto, 8.33% de sucesso indireto e 8.33% de desistência/pulos. [50]

Figura 50 – Tarefa 3: Navegar com o scroll do mouse pela sequência de ilustração de um artigo na exibição simplificada.



Tarefa 4: 91.67% de sucesso direto, 0% de sucesso indireto e 8.33% de desistência/-pulos. [51]

Figura 51 – Tarefa 4: Acessar a exibição de informações do artigo como autores, palavras chaves e como referenciar.



O questionário de usabilidade, por sua vez, apresenta os resultados com porcentagens distribuídas pela escala Lickert, classificados com a numeração de 1 até 5, sendo 1 correspondente ao "Discordo"e 5 correspondente ao "Concordo", com os intermediários 2, 3 e 4 assumindo valores relativos aos extremos 1 e 5.

A seguir serão listadas as tarefas do teste de usabilidade juntamente com suas respectivas taxas. [52, 53, 54, 55, 56, 57, 58, 59, 60, 61]

Figura 52 – Questão 1: 0% em 1, 8% em 2, 8% em 3, 25% em 4 e 58% em 5.

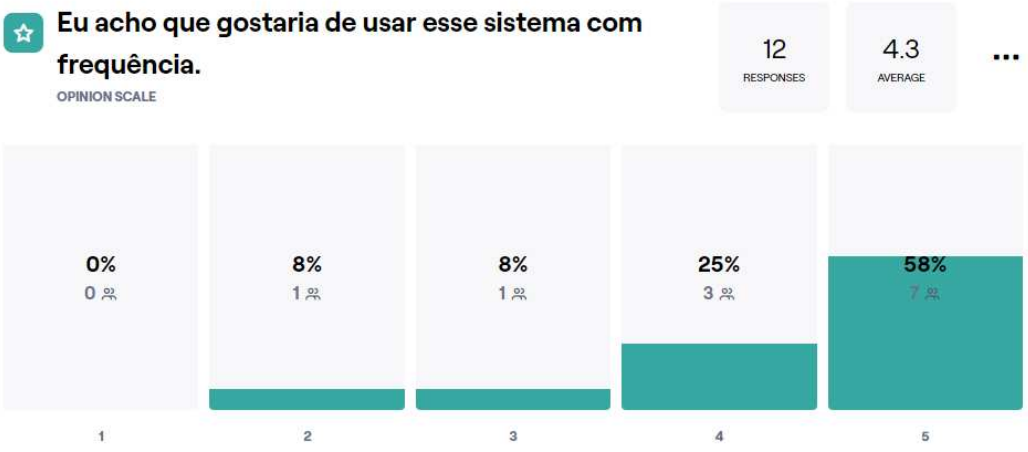


Figura 53 – Questão 2: 75% em 1, 17% em 2, 0% em 3, 8% em 4 e 0% em 5.

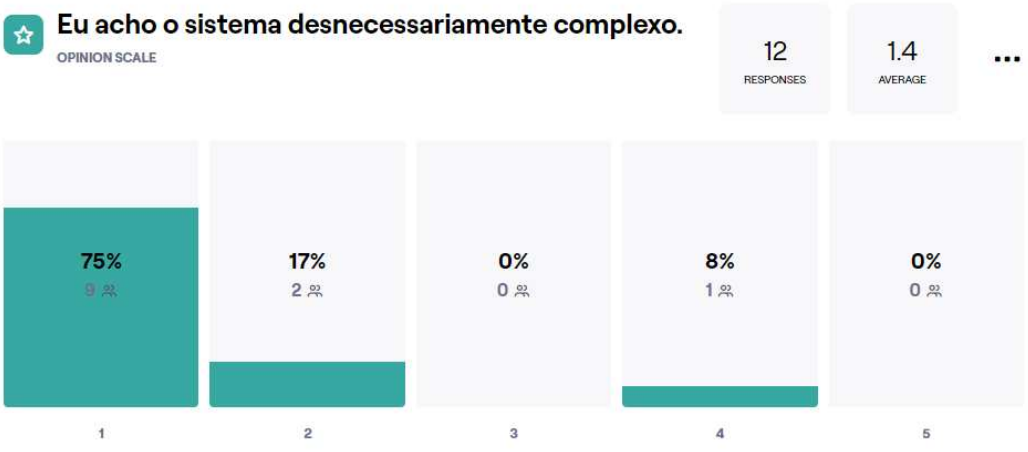


Figura 54 – Questão 3: 8% em 1, 0% em 2, 8% em 3, 17% em 4 e 67% em 5.

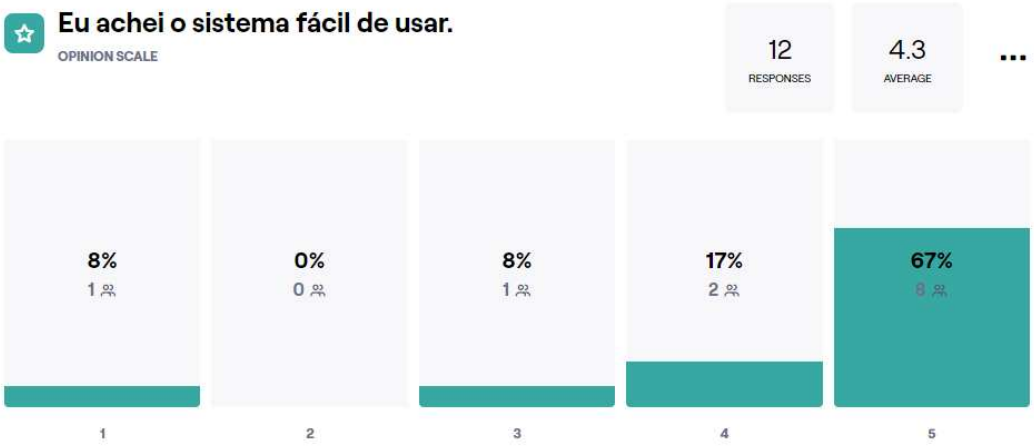


Figura 55 – Questão 4: 92% em 1, 8% em 2, 0% em 3, 0% em 4 e 0% em 5.

Figura 56 – Questão 5: 0% em 1, 8% em 2, 0% em 3, 17% em 4 e 75% em 5.

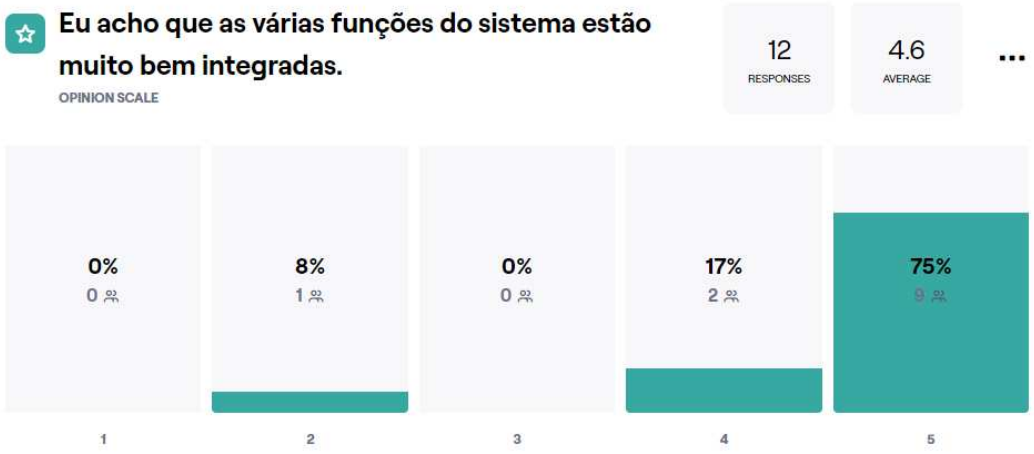


Figura 57 – Questão 6: 83% em 1, 8% em 2, 0% em 3, 0% em 4 e 8% em 5.

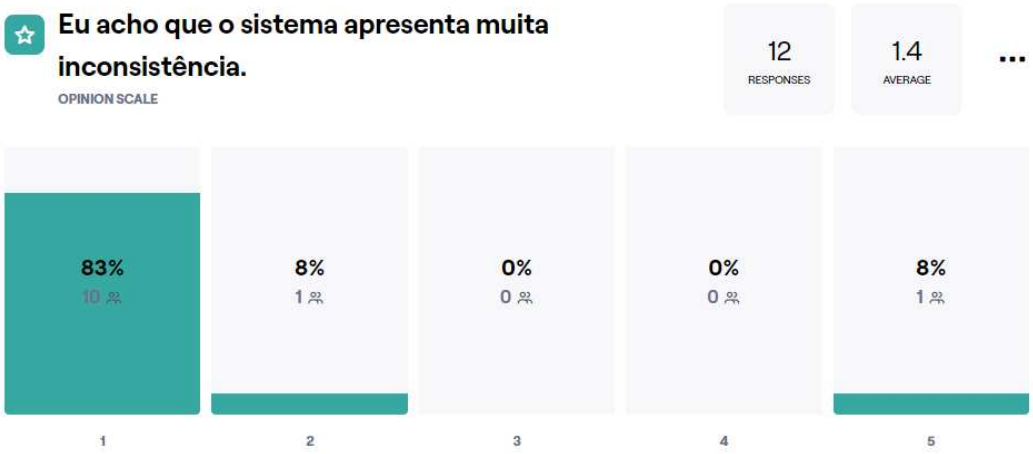


Figura 58 – Questão 7: 0% em 1, 0% em 2, 8% em 3, 17% em 4 e 75% em 5.

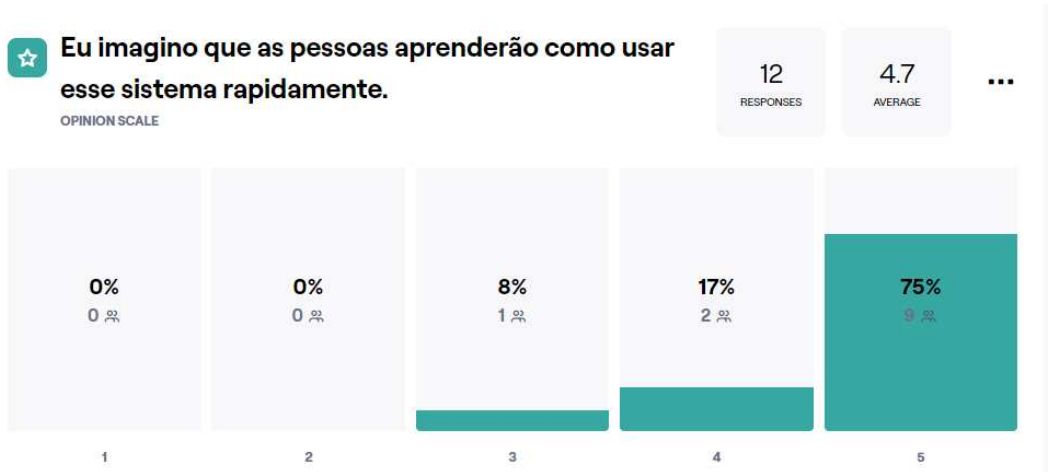


Figura 59 – Questão 8: 83% em 1, 8% em 2, 0% em 3, 0% em 4 e 8% em 5.

Figura 60 – Questão 9: 83% em 1, 18% em 2, 0% em 3, 0% em 4 e 8% em 5.

Figura 61 – Questão 10: 8% em 1, 0% em 2, 0% em 3, 25% em 4 e 67% em 5.

4.2 Requisitos funcionais: Frontend

RF01 Menu [62, 63, 64]

Figura 62 – Captura de tela: Menu *desktop*

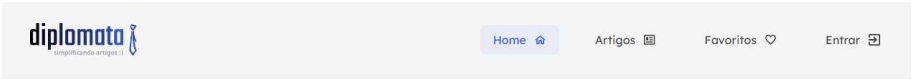
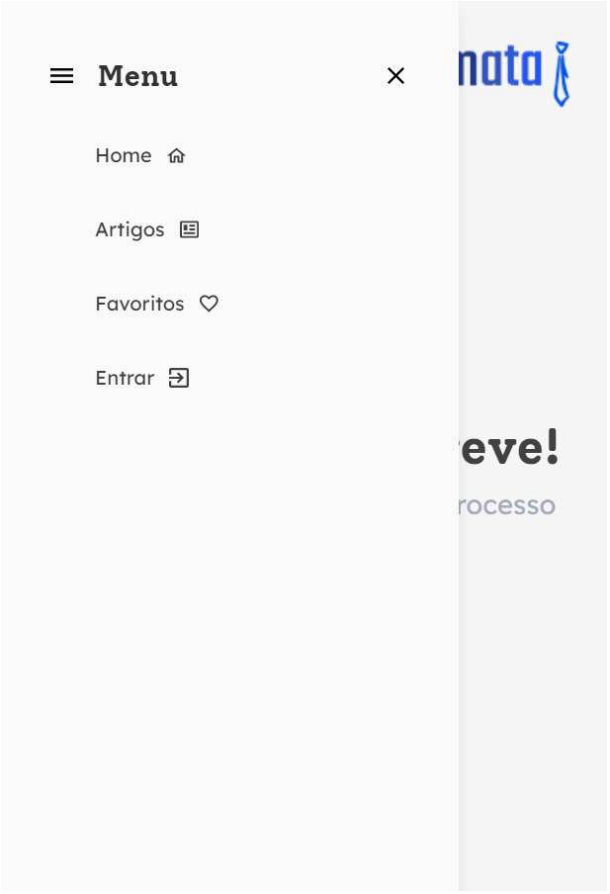


Figura 63 – Captura de tela: Menu *mobile* fechado

Figura 64 – Captura de tela: Menu mobile aberto



RF02 Conteúdo “Em breve” para páginas fora do escopo [65]

Figura 65 – Captura de tela: Página Home

RF03 I - Página artigos [66 e 67]

RF03 II - Conteúdo da lista de artigos [66]

Figura 66 – Captura de tela: Página Artigos *desktop*

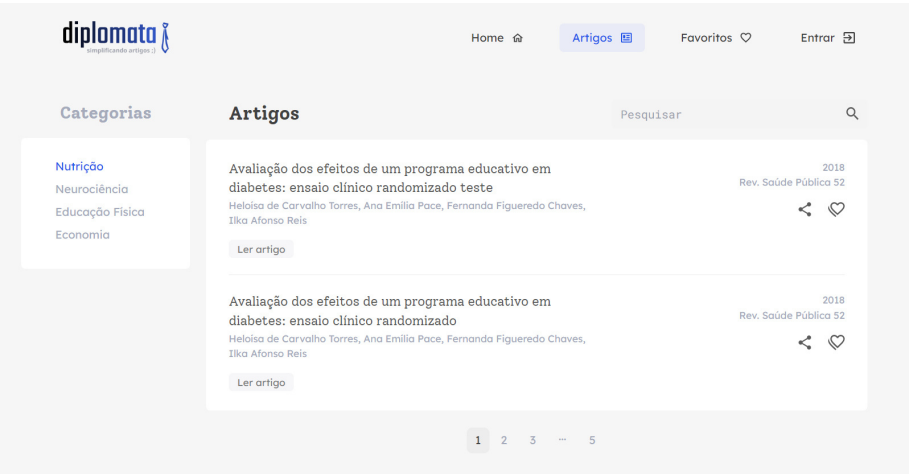
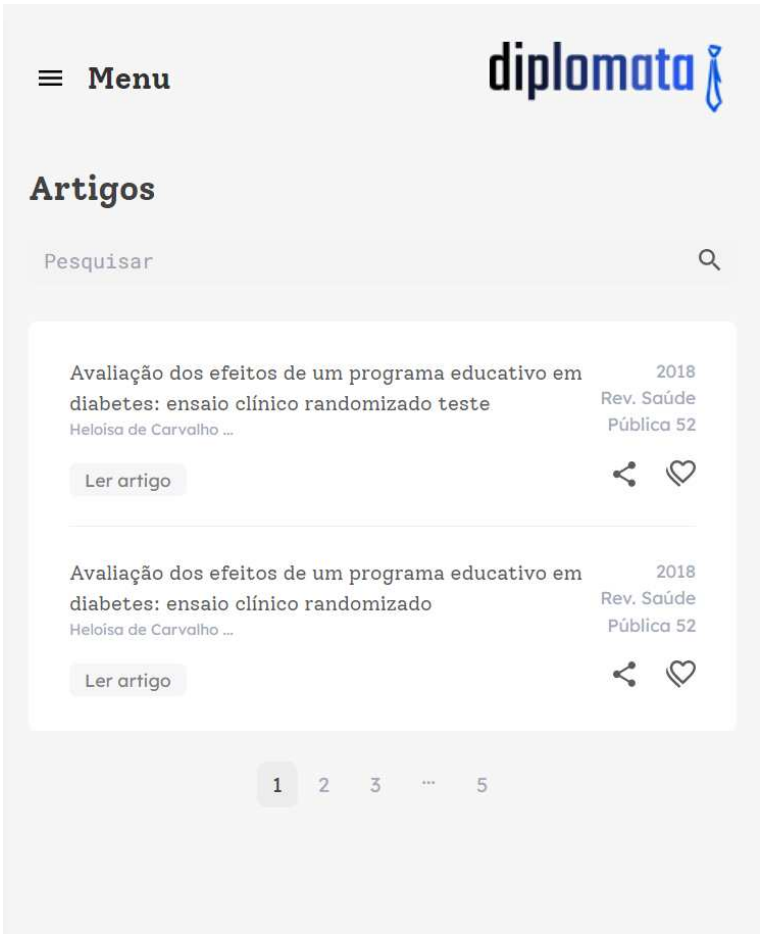


Figura 67 – Captura de tela: Página Artigos *mobile*



RF04 I - Página de um artigo específico [68, 69, 70, 71]

Figura 68 – Captura de tela: Página de um artigo em especifico *desktop*

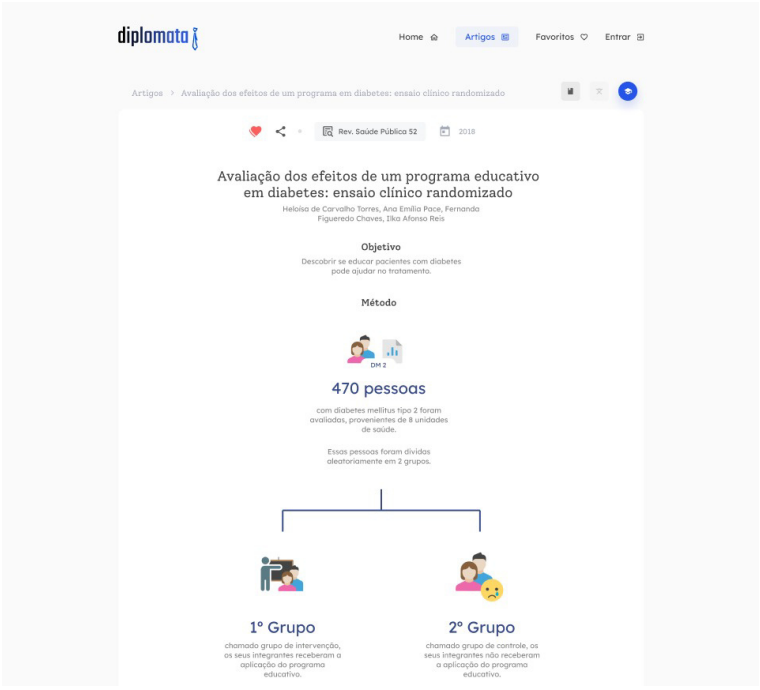


Figura 69 – Captura de tela: Página de um artigo em especifico (exibição de informações gerais) *desktop*

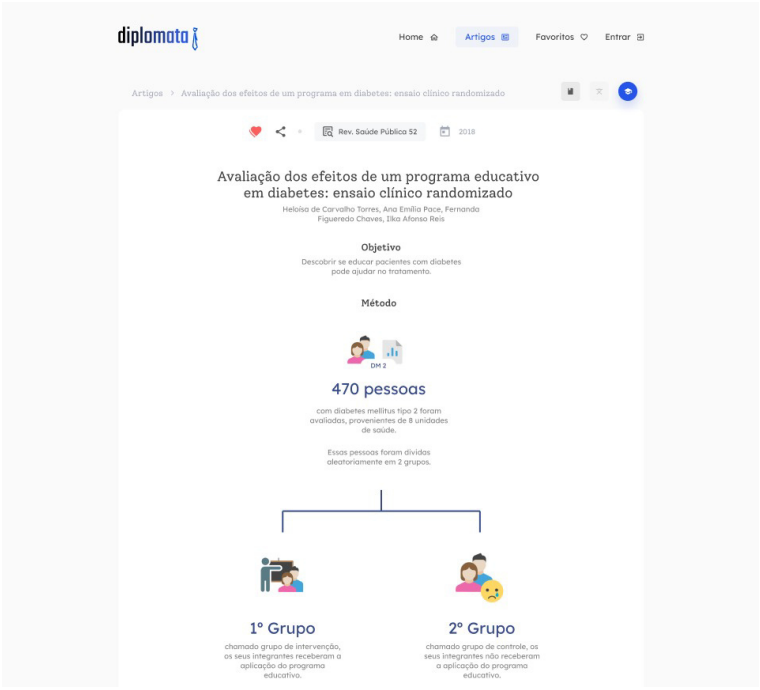


Figura 70 – Captura de tela: Página de um artigo em especifico *mobile*

Figura 71 – Captura de tela: Página de um artigo em especifico (exibição de informações gerais) *mobile*

4.3 Requisitos funcionais: Backend

RF01 Servidor deve persistir e coletar os dados de um banco externo e não relacional (MongoDB). [Observação] Nessa versão do projeto, o escopo foi pensado de modo reduzido a fim o curto prazo de tempo para desenvolvimento do software, portanto a aplicação deve ser estruturada para armazenar e manipular apenas um modelo de dados: o modelo de artigo. (Concluído)

RF02 Definição de um modelo de dados para os artigos. (Concluído)

RF03 Disponibilização das rotas para artigos:

- *GET* (listar todos) e *POST* (criar um) para o caminho */articles*. (Concluído)
- *GET* (listar um), *PATCH* (alterar um) e DELETE (excluir um) para o caminho */articles/:id*. (Concluído)

4.4 Requisitos não funcionais

- Os serviços frontend e backend devem estar disponibilizados de forma online 24h / dia, 7 dias / semana. (Concluído)
- A plataforma deverá conter um repositório backend e um frontend, que devem ser disponibilizados em um domínio público virtual para viabilizar os testes e utilização do sistema. (Concluído)

5 CONCLUSÕES E TRABALHOS FUTUROS

Este último capítulo encerra o trabalho realizado com as considerações finais e os próximos passos que podem ser desenvolvidos como uma evolução natural da plataforma Diplomata.

5.1 Considerações finais

Este trabalho levantou a discussão sobre a facilitação do acesso ao conhecimento científico, em conjunto com a exploração da frente tecnológica como recurso capaz de auxiliar na distribuição da produção científica considerando essas questões de facilitação dos conteúdos acadêmicos. Além disso e designado como objetivo central a ser alcançado, este trabalho documentou todo o processo de desenvolvimento da primeira versão da plataforma Diplomata, explorando o potencial das principais tecnologias apresentadas (*Vue* e *Express.js*). Também foi revelado toda estruturação do processo de elaboração de um artigo ilustrado, que também é um ponto central em relação a solução sobre acessibilidade ao conhecimento científico.

O capítulo de metodologia organiza e dispõe de toda trajetória percorrida desde a ideação até a consolidação do produto, evidenciando através do capítulo de resultados que os objetivos gerais e específicos foram alcançados e tornando a hipótese deste trabalho assegurada.

A versão publicada da plataforma se encontra no domínio <https://diplomata.vercel.app/>, e ela atende todos os requisitos funcionais levantados no capítulo de metodologia, sendo possível para o usuário final a navegação entre páginas através do menu, o acesso à listagem de artigos, o acesso a um artigo específico e suas diferentes opções de exibição, incluído conteúdo ilustrado e informações gerais. Também é possível conferir a plataforma executando no vídeo de amostra do site publicado em <https://www.youtube.com/watch?v=4gb-5wah2IA>.

5.2 Trabalhos Futuros

É importante ressaltar que o escopo atendido foi limitado às necessidades do prazo de construção desse trabalho, e considerando o tempo de produção médio de um TCC, não seria viável incorporar muitos requisitos no projeto. Uma vez que ele precisa ser planejado, desenvolvido e documentado em relatório. Dessa forma podemos considerá-lo um escopo pequeno comparado à plataformas disponibilizadas na internet de um modo geral. Este é um ponto central para a continuidade do projeto nos seus próximos passos, podendo ter exploradas

funcionalidades que já foram pensadas mas não atendidas nessa primeira versão como por exemplo: permanência de usuários com acesso a entrada e saída mediante credenciais no sistema, possibilitando o acesso à informações individualizadas como favoritos e histórico de leitura.

De modo final pode-se elencar os seguintes alvos futuros:

- Inclusão de usuários.
  - Implementação de formulário para criação de contas de usuários.
  - Implementação de formulários para login e *logoff* da plataforma.
  - Possibilidade do usuário favoritar artigos quando logado.
  - Consulta dos artigos favoritos pelo usuário.
  - Consulta do histórico de leitura pelo usuário.
- Implementação de um painel de compartilhamento do link de um artigo na plataforma em redes sociais.
- Implementação dos modelos de categorias.
- Inclusão de filtros junto a listagem de artigos.
- Disponibilização de um modo de exibição voltado à tradução de artigos.
- Criação de uma lista de e-mails vinculada ao site para comunicações de novidades em relação a plataforma.
- Análise de melhorias possíveis em cima do fluxo de elaboração de ilustrações para artigos.
- Definição de uma meta em número de artigos ilustrados.

REFERÊNCIAS

ANNAN, K. A challenge to the world's scientists. **Science**, v. 299, n. 5612, p. 1485–1485, 2003. Disponível em: <https://www.science.org/doi/abs/10.1126/science.299.5612.1485>.

BOUCINHA, R. M.; TAROUCO, L. M. R. **Avaliação de ambiente virtual de aprendizagem com o uso do sus-system usability scale.** 2013. Disponível em: <https://www.seer.ufrgs.br/renote/article/view/44479>.

CASTRO, R. C. F. Impacto da internet no fluxo da comunicação científica em saúde. **Revista de Saúde Pública [online]**, v. 40, n. spe, p. 57–63, 2006. ISSN 1518-8787.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. **Sistemas Distribuídos: Conceitos e Projeto**. 5a.. ed. Bookman Editora, 2013. ISBN 9788582600542. Disponível em: <https://books.google.com.br/books?id=6WU3AgAAQBAJ>.

FELIZARDO, A. **O que é Vue.js**. 2018. Disponível em: <http://www.andrefelizardo.com.br/blog/o-que-e-vue-js>. Acessado em: 14 de novembro de 2022.

JúNIOR, S. D. da S.; COSTA, F. J. **Mensuração e escalas de verificação: uma análise comparativa das escalas de Likert e Phrase Completion.** 2014. 61 p. Disponível em: <http://sistema.semead.com.br/17semead/resultado/trabalhospdf/1012.pdf>.

KUROSE, J. F.; ROSS, K. W. **Redes de computadores e a Internet: uma abordagem top-down**. 5a. ed. [S.l.]: Addison Wesley, 2010. ISBN 9788588639973.

LI, N.; ZHANG, B. The design and implementation of responsive web page based on html5 and css3. In: **2019 International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)**. [S.l.: s.n.], 2019. p. 373–376.

MARCONI, M. d. A.; LAKATOS, E. M. **Fundamentos de Metodologia Científica**. 5a. ed. São Paulo: Atlas, 2003. ISBN 85-224-3397-6.

NASCIMENTO, J. V. B. do; NETO, M. M.; COUTINHO, E. F.; MOREIRA, L. O. Um levantamento sobre os aspectos técnicos dos principais riscos de segurança e ataques em aplicações web. **Revista Sistemas e Mídias Digitais (RSMD)**, v. 6, n. 1, julho 2021. ISSN 2525-9555. Disponível em: <http://revistasmd.virtual.ufc.br/arquivos/volume-6/numero-1/rsmd-v6-n1-6.pdf>.

NEGRÃO, L. **Angular vs React vs Vue.js**. 2022. Disponível em: <https://www.alura.com.br/artigos/angular-vs-react-vs-vue-js>. Acessado em: 14 de novembro de 2022.

ONLINE, R. **Node.js: descubra tudo sobre a linguagem e suas aplicações**. 2022. Disponível em: <https://www.remessaonline.com.br/blog/node-js-descubra-tudo-sobre-a-linguagem-e-suas-aplicacoes/>. Acessado em: 14 de novembro de 2022.

SOUZA, W. C. **Construtor de Sistemas Web**. 2018. Monografia de Graduação. Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Instituto Federal de Santa Catarina, Gaspar, Brasil.

TANENBAUM, A. S.; STEEN, M. V. **Distributed Systems: Principles and Paradigms**. 2nd. ed. [S.l.]: Pearson Prentice Hall, 2007. ISBN 9780132392273.

TILLY, C. O acesso desigual ao conhecimento científico. **Tempo Social [online]**, v. 18, n. 2, p. 47–63, 11 2006. ISSN 1809-4554.

VUEJS. **Introdução a Vue.js**. 2022. Disponível em: <https://br.vuejs.org/v2/guide/>. Acessado em: 14 de novembro de 2022.

APÊNDICE A – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO (TCLE)

Você está sendo convidado por Leonardo Moreira Oliveira, professor da Universidade Federal do Ceará - Campus Pici, e por Jemima Fonseca Luz, aluno do curso de Sistemas e Mídias Digitais da Universidade Federal do Ceará - Campus Pici, como participante da pesquisa intitulada "Diplomata: Uma plataforma para facilitação do acesso ao conhecimento científico".

Você não deve participar contra a sua vontade. Leia atentamente as informações a seguir e faça qualquer pergunta que desejar, para que todos os procedimentos desta pesquisa sejam esclarecidos.

O objetivo desta pesquisa é utilizar uma ferramenta para a facilitação do acesso ao conhecimento científico, e avaliar sua utilização sob diversos aspectos.

Basicamente você será solicitado a realizar 4 tarefas em um protótipo da plataforma, e em seguida deverá preencher um questionário com 10 perguntas.

As respostas deverão ser baseadas em sua experiência na utilização da ferramenta.

(i) Eu declaro que é de livre e espontânea vontade que estou participando desta pesquisa.

(ii) Eu declaro que li cuidadosamente este Termo de Consentimento Livre e Esclarecido e que, após sua leitura, tive a oportunidade de fazer perguntas sobre o seu conteúdo, como também sobre a pesquisa, e recebi explicações que responderam por completo minhas dúvidas.

(iii) Eu declaro que já tive contato com artigos científicos ou que precisei pesquisar por artigos científicos. (Este é um requisito necessário para a avaliação da ferramenta).