UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

LÍGIA FALCÃO NOGUEIRA MIRANDA

AMIGOS DO REI: UMA APLICAÇÃO WEB PARA ESTUDOS BÍBLICOS INFANTIS

GUARAPUAVA

2025

LÍGIA FALCÃO NOGUEIRA MIRANDA

AMIGOS DO REI: UMA APLICAÇÃO WEB PARA ESTUDOS BÍBLICOS INFANTIS

King's Friends: A web aplication for children's bible studies

Trabalho de Conclusão de Curso de Graduação apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná, Campus Guarapuava, como requisito parcial para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet.

Orientador: Prof. Dr. Roni Fabio Banaszewski

GUARAPUAVA

2025

4.0 Internacional

Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

LÍGIA FALCÃO NOGUEIRA MIRANDA

AMIGOS DO REI: UMA APLICAÇÃO WEB PARA ESTUDOS BÍBLICOS INFANTIS

Trabalho de Conclusão de Curso de Graduação apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná, Campus Guarapuava, como requisito parcial para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet.

Data de aprovação: 19/fevereiro/2025

Roni Fabio Banaszewski
Doutor
Universidade Tecnológica Federal do Paraná

Sediane Carmem Lunardi Hernandes
Doutora
Universidade Tecnológica Federal do Paraná

Diego Marczal
Doutor
Universidade Tecnológica Federal do Paraná

GUARAPUAVA
2025

Dedico este trabalho à Deus e à minha família, que me apoia em todas as decisões da minha vida, sendo meu suporte todos os dias e me ajudando com ideias.

AGRADECIMENTOS

Gostaria de agradecer em primeiro lugar a Deus, à quem sem Ele eu não sou nada e foi para trabalho de sua obra que esse trabalho foi projetado.

Agradecer à minha família que sempre esteve ao meu lado, me ajudou a ter ideias para o projeto e respeitou todo meu tempo me incentivando, apoiando e motivando para seguir até o fim.

Gostaria de agradecer a Igreja Presbiteriana da Bonsucesso por me permitir utilizar o Amigos do Rei como base para meu projeto.

Agradeço ao meu orientador Prof. Dr. Roni Fabio Banaszewski, por todo seu incentivo, auxílio e ideias sempre disposto a me ajudar e motivar ao longo de todo esse trabalho.

Aos meus amigos e colegas de trabalho que me incentivaram a continuar fazendo, me ajudaram nos momentos de dúvidas e torceram por mim a cada momento desse trabalho.

Agradeço ao Prof. Dr. Diego Marczal que me ajudou em tantas áreas desse projeto e em dúvidas, onde mesmo não sendo meu orientador dedicou parte do seu tempo para me auxiliar.

Enfim, a todos os que por algum motivo contribuíram para a realização deste projeto de alguma forma.

RESUMO

O presente trabalho aborda a temática da organização e gestão das aulas da Escola Bíblica Dominical (EBD) no âmbito do Ministério Infantil da Igreja Presbiteriana Bonsucesso (IPB) de Guarapuava. A EBD é um espaço de ensino cristão voltado para crianças de diferentes faixas etárias, e sua eficácia é essencial para a transmissão de valores e princípios religiosos. No entanto, o atual sistema de organização das aulas apresenta desafios significativos. As informações sobre escalas, materiais, e outros aspectos relevantes são compartilhadas por meio de ferramentas como WhatsApp e Google Drive. Isso tem levado a problemas de comunicação, confusão e dificuldades na preparação das aulas por parte dos professores. Diante desse cenário, este trabalho apresenta o desenvolvimento de uma aplicação web que centraliza e simplifica a organização das informações da EBD. O sistema desenvolvido oferece uma interface simples para professores, alunos e pais por ter todas as informações direto na sua tela inicial, promovendo a melhoria na qualidade das aulas e incentivando a participação ativa dos pais na educação religiosa de seus filhos. A plataforma permite que os professores acessem suas escalas e os materiais necessários para as aulas, além de disponibilizar recursos multimídia, sugestões de atividades e um espaço para o compartilhamento de informações sobre as aulas realizadas. Os pais também, eles poderão acompanhar o progresso de seus filhos na EBD e participar ativamente do processo educacional. A aplicação web desenvolvida ainda não está em uso pelos interessados, mas espera-se que contribua significativamente para a eficiência na organização das aulas da EBD, tornando o ensino mais estruturado, acessível e dinâmico para as crianças. A adoção desse ambiente digital voltado à educação religiosa representa um avanço na gestão do Ministério Infantil, permitindo maior organização, engajamento dos professores e participação ativa dos pais. Dessa forma, a Igreja Presbiteriana Bonsucesso reforça sua missão de transmitir valores e princípios cristãos às novas gerações, auxiliando para que os professores forneçam um ensino mais eficaz e alinhado às necessidades contemporâneas.

Palavras-chave: estudo bíblico; angular; laravel; infantil; educação.

ABSTRACT

This paper addresses the topic of the organization and management of EBD classes within the scope of the Children's Ministry of the IPB of Guarapuava. The EBD is a Christian educational space aimed at children of various age groups, and its effectiveness is essential for transmitting religious values and principles. However, the current system for organizing classes presents significant challenges. Information regarding schedules, materials, and other relevant aspects is shared through tools such as WhatsApp and Google Drive. This has led to communication problems, confusion, and difficulties for teachers in preparing the classes. In light of this scenario, this work presents the development of a web application that centralizes and simplifies the organization of EBD information. The developed system provides a simple interface for teachers, students, and parents by having all the information directly on their home screen, promoting an improvement in the quality of classes and encouraging active parental participation in their children's religious education. The platform allows teachers to access their schedules and the necessary materials for the classes, as well as providing multimedia resources, activity suggestions, and a space for sharing information about the classes held. Parents, too, will be able to track their children's progress in the EBD and actively participate in the educational process. The developed web application is not yet in use by the interested parties, but it is expected to contribute significantly to the efficiency in organizing EBD classes, making education more structured, accessible, and dynamic for the children. The adoption of this digital environment focused on religious education represents an advance in the management of the Children's Ministry, allowing for greater organization, teacher engagement, and active parental participation. In this way, the Bonsucesso Presbyterian Church reinforces its mission to transmit Christian values and principles to new generations, helping teachers to provide more effective education that is aligned with contemporary needs.

Keywords: bible study; angular; laravel; children; education.

LISTA DE FIGURAS

Figura 1 – Exemplo de histórias no app Bíblia para Crianças **16**
Figura 2 – Dinâmica do Scrum **18**
Figura 3 – Quadro Kanban **19**
Figura 4 – Ciclo do TDD **20**
Figura 5 – Fluxo de Trabalho **23**
Figura 6 – Tela de Login **33**
Figura 7 – Tela Inicial **34**
Figura 8 – Tela de Perfil **35**
Figura 9 – Modelagem do Banco de Dados **37**
Figura 10 – Modelagem do Banco de Dados Final **39**
Figura 11 – Tela de Login Final **45**
Figura 12 – Tela Inicial Final **45**
Figura 13 – Menu **47**
Figura 14 – Tela Perfil Final **47**
Figura 15 – Calendário **48**
Figura 16 – Adicionar Eventos **49**
Figura 17 – Turmas **50**
Figura 18 – Classes **51**
Figura 19 – Cadastro de Usuários **52**

LISTA DE TABELAS

Tabela 1 – Histórias de Usuário (Ordenado por Peso) **31**
Tabela 2 – Histórias Removidas **32**
Tabela 3 – Sugestões de Histórias Futuras **55**

LISTA DE ABREVIATURAS E SIGLAS

Siglas

ACID	*Atomicity, Consistency, Isolation and Durability* (Atomicidade, Consistência, Isolamento e Durabilidade)
CLI	*Command Line Interface* (Interface de Linha de Comando)
EBD	Escola Bíblica Dominical
IPB	Igreja Presbiteriana Bonsucesso
MVC	*Model-View-Controller* (Modelo-Visão-Controlador)
SQL	*Structured Query Language* (Linguagem de Consulta Estruturada)
TDD	*Test Driven Development* (Desenvolvimento Orientado a Testes)
VCS	*Version Control System* (Sistema de Controle de Versão)

SUMÁRIO

1 INTRODUÇÃO 11

1.1 Justificativa 12

1.2 Objetivos 13

1.2.1 Objetivo geral 13

1.2.2 Objetivos específicos 14

2 TRABALHOS RELACIONADOS 15

2.1 Moodle 15

2.2 Rosângela Trajano 15

2.3 Bíblia para Crianças 16

3 REFERENCIAL TEÓRICO 17

3.1 Processo de Desenvolvimento 17

3.1.1 Método *Scrum* 18

3.1.2 Método Kanban 19

3.1.3 Desenvolvimento Orientado a Testes 20

3.1.4 Técnica de Priorização MoSCoW 21

3.2 Ferramentas de desenvolvimento 22

3.2.1 GitHub 22

3.2.2 Git e GitFlow 22

3.3 Tecnologias do lado do cliente 24

3.3.1 Angular 24

3.4 Tecnologias do lado do servidor 24

3.4.1 Laravel 25

3.4.2 Docker 25

3.4.3 MySQL 26

3.4.4 Digital Ocean 26

3.5 Considerações Finais 26

4 MATERIAIS E MÉTODOS 28

4.1 Materiais 28

4.2 Métodos 29

5 ANÁLISE E PROJETO DO SISTEMA 31

5.1 Análise do sistema 31
5.2 Prototipação das Telas 32
5.3 Modelagem do Banco de Dados 35
6 DESENVOLVIMENTO 40
6.1 Fluxo de Desenvolvimento 40
6.2 Detalhamento das Sprints 40
6.2.1 Sprints 1 a 4 - Configuração do Ambiente de Desenvolvimento 40
6.2.2 Sprint 5 - Estruturação do Banco de Dados 42
6.2.3 Sprint 6 - Autentificação e Autorização para acesso de dados 43
6.2.4 Sprint 7 - Desenvolvimento das Telas de Login e Página Inicial 44
6.2.5 Sprint 8 e 9 - Criação da Tela de Perfil e Menu 46
6.2.6 Sprint 10 - Implementação da Tela de Calendário 47
6.2.7 Sprint 11 - Desenvolvimento das Telas de Turmas e Eventos 49
6.2.8 Sprint 12 - Cadastro de Usuários 51
6.3 Considerações sobre o Desenvolvimento 52
7 CONCLUSÃO 54
7.1 Trabalhos Futuros 54
REFERÊNCIAS 56

1 INTRODUÇÃO

O primeiro encontro da criança com o ambiente escolar desempenha um papel crucial no seu desenvolvimento integral, abrangendo aspectos psicológicos, físicos, intelectuais e sociais. Portanto, esse período na escola é essencial para o crescimento da criança, ajudando-a a aprimorar suas habilidades cognitivas e motoras. A Educação Infantil desempenha um papel fundamental ao permitir que a criança interaja socialmente além do círculo familiar, proporcionando-lhe a oportunidade de aprender a conviver e se relacionar em sociedade. Isso, por sua vez, contribui para o desenvolvimento de habilidades cruciais para sua formação como ser humano (LÜCKE, 2019).

No entanto, a escola não se limita a ser o único ambiente onde as crianças adquirem conhecimentos acadêmicos, como língua portuguesa e matemática. Antes mesmo de frequentar a escola, as crianças muitas vezes têm a oportunidade de aprender em outros ambientes, como creches e igrejas.

Ensinar crianças com idades entre zero e doze anos é um desafio significativo, mas instruí-las nos princípios cristãos de acordo com a Bíblia representa um desafio ainda maior. Os educadores das escolas dominicais nas igrejas evangélicas em todo o Brasil e no mundo se concentram nos conceitos fundamentais que moldarão a personalidade e o caráter dessas crianças. Eles buscam mantê-las envolvidas e concentradas, garantindo que a mensagem seja transmitida de maneira clara e simples, adaptada a cada faixa etária. O objetivo principal é tocar o coração das crianças, de modo que levem esses ensinamentos consigo ao longo de toda a vida (BERTINATTI; NASCIMENTO, 2011).

Conforme mencionado no livro de Provérbios, capítulo 22, versículo 6, é de extrema importância iniciar a educação religiosa desde tenra idade, para que esses princípios se tornem uma parte integral de suas vidas ao longo do tempo. Com o objetivo de atender às necessidades das diversas faixas etárias das crianças, as igrejas evangélicas introduziram o conceito de Escola Bíblica Dominical (EBD) especializadas, que se concentram em oferecer um ensino com linguagem adequada para cada grupo etário.

> Ensine a criança no caminho em que deve andar, e ainda quando for velho não se desviará dele. (BÍBLIA. . . , 2017).

Na Igreja Presbiteriana Bonsucesso (IPB) de Guarapuava, por meio do seu Ministério Infantil conhecido como "Amigos do Rei", o ensino voltado para crianças é estruturado em quatro faixas etárias distintas. O primeiro grupo, chamado de "babies", abrange crianças de zero a três anos. O segundo, denominado de "primário", engloba crianças de quatro a seis anos. O terceiro grupo, intitulado "juniores", atende crianças de sete a nove anos. Por fim, o grupo de "pré-adolescentes" inclui crianças de dez a doze anos. Além disso, o ministério conta com uma equipe de teatro e diversas professoras que se revezam no atendimento às diferentes turmas. No entanto, devido à complexidade das atividades e ao envolvimento de muitas pessoas

no ministério, a forma atual de disponibilização das informações tem gerado confusões para alguns professores. Eles têm dificuldades em saber qual será a aula no próximo domingo, quais materiais estão disponíveis para conduzir a aula, quem estará responsável pelo ensino e para qual turma, além de como manter a continuidade e a conexão com o conteúdo ensinado pelo professor da semana anterior.

Com o intuito de contribuir para a organização eficaz dos cronogramas e programas de aula, bem como para aprimorar a preparação das professoras, este Trabalho de Conclusão de Curso apresenta o desenvolvimento de um sistema voltado para a organização da EBD. O sistema foi desenvolvido com acesso restrito, exigindo autenticação para garantir a segurança das informações. Ele conta com uma interface simplificada e intuitiva, abrangendo todo o conteúdo anual, além de áreas multimídia que incluem atividades, vídeos e músicas direcionadas a diferentes faixas etárias.

Além disso, os pais dos alunos da EBD, onde podem acessar, por meio do calendário, as aulas ministradas em cada dia. Nessa área, os responsáveis poderão visualizar o resumo e o conteúdo completo das aulas dadas, bem como identificar quais professoras estiveram responsáveis por cada classe. Essa funcionalidade tem como objetivo incentivar o acompanhamento do aprendizado em casa e reforçar os conteúdos abordados em sala.

Embora o sistema já tenha sido desenvolvido, sua implementação para uso efetivo pelos interessados ainda não ocorreu. No entanto, espera-se que, futuramente, sua adoção contribua significativamente para a melhoria da organização das aulas da EBD, proporcionando um ensino mais estruturado e acessível, uma vez por causa do sistema os professores conseguem se programar melhor para as aulas tendo o conteúdo disponível com mais tempo e já indicando quando será sua aula, além de fortalecer o envolvimento dos pais no processo educacional. Dessa forma, a Igreja Presbiteriana Bonsucesso reafirma seu compromisso em transmitir valores e princípios cristãos às novas gerações por meio de uma abordagem digital inovadora.

1.1 Justificativa

O Ministério atualmente utiliza um arquivo de texto no Word para registrar as datas dos domingos em que haverá aulas e compartilha-o por meio do Google Drive no grupo de WhatsApp dos professores do ministério infantil. Este arquivo contém a lista de domingos, geralmente abrangendo três meses, listados um abaixo do outro com o nome do professor(a) designado para cada turma. No entanto, esta não é a maneira mais prática de localizar suas escalas, uma vez que por diversas vezes os professores acabam sendo alterados e os professores acabam perdendo suas agendas de aula.

Quando ocorrem mudanças na escala, os professores as comunicam pelo WhatsApp. Contudo, com frequência, essas informações se perdem no meio do conteúdo das aulas e dos compartilhamentos de materiais, o que resulta em problemas como a dificuldade de substituir um professor ou, em último caso, a necessidade de juntar classes às pressas. Isso pode levar a

situações em que os professores não estão devidamente preparados para assumir uma turma que precisa ser coberta.

Quanto a conteúdos e materiais, estes estão disponíveis em forma de imagens escaneadas da apostila usada nas aulas, armazenadas em um Google Drive[1]. Além disso, algumas atividades também são escaneadas para serem usadas durante as aulas. No entanto, existem algumas limitações notáveis, como:

- **Qualidade das Imagens**: A qualidade das imagens não é muito alta, o que pode prejudicar a legibilidade e utilidade do material.
- **Cores Limitadas**: As imagens são em preto e branco, com poucas opções de atividades ou diversificação adequada para atender a diferentes faixas etárias.
- **Fotos das Aulas**: As fotos das aulas são compartilhadas no grupo, o que aumenta a quantidade de conversas no grupo, consome espaço nos celulares devido às mídias e torna difícil a tarefa de categorizar as fotos em relação às classes a que se referem, ao dia em que foram tiradas e à compilação desse material completo para métricas de final de ano.

Devido às dificuldades em localizar informações no grupo, muitas vezes é necessário repetir detalhes, como a próxima lição a ser ministrada no domingo seguinte ou a lista de professores escalados para a semana em questão. Isso frequentemente ocorre em um momento muito próximo ao evento, tornando difícil realizar substituições, e já houve casos em que os professores não tinham conhecimento de sua inclusão na escala. Quando é necessário realizar pesquisas, agendar reuniões ou comunicar informações cruciais, muitas vezes não se consegue atrair a devida atenção, uma vez que não há uma maneira eficaz de destacar essas informações.

Visando corrigir estes problema, foi idealizado um sistema para centralizar todas as informações para aulas de maneira simplificada e organizada, onde ao logar os professores irão já ter acesso a sua próxima aula, o dia e todas as informações sobre a aula de maneira resumida permitindo ver mais para ver tudo completo, sendo assim de maneira rápida os professores vão ter acesso ao dia de suas aulas além de seu conteúdo.

1.2 Objetivos

1.2.1 Objetivo geral

Desenvolver uma aplicação web para centralizar os dados da EBD voltada para crianças de zero a doze anos, visando facilitar a preparação antecipada de aulas pelos professores e aproximar os pais do conteúdo ministrado.

[1] Plataforma de compartilhamento de arquivos com opção de armazenamento em nuvem (GOOGLE, 2012).

1.2.2 Objetivos específicos

- Facilitar a identificação da próxima data de escala e da respectiva turma na página inicial.
- Possibilitar a visualização de todas as datas relevantes para o usuário por meio de um calendário.
- Disponibilizar materiais aos professores para a próxima aula, incluindo dados da apostila, sugestões de atividades, vídeos e músicas.
- Permitir o compartilhamento de informações da aula realizada no domingo, como fotos e um breve resumo dos materiais utilizados.
- Apresentar uma tela com as próximas aulas que deverão ser dadas de maneira clara e fácil.
- Promover um ambiente de fácil, com informações claras e diretas na página inicial, direcionando o grupo do WhatsApp exclusivamente para informações específicas, priorizando o sistema para todas as demais comunicações.

2 TRABALHOS RELACIONADOS

Atualmente, há uma variedade de sites e aplicativos que permitem o armazenamento de dados com o objetivo de centralizar conteúdos didáticos, organizados por disciplinas e professores. Além disso, existem algumas opções voltadas especificamente para conteúdo bíblico. Nos próximos tópicos, serão apresentados sistemas disponíveis no mercado que compartilham partes desta proposta, embora não sejam idênticos em sua abordagem.

2.1 Moodle

Utilizado no mundo inteiro por ser um código aberto multi-idioma, o Moodle é uma plataforma de ensino projetada para oferecer a educadores, administradores e alunos um sistema único, robusto e integrado para criar ambientes de aprendizagem personalizados. De facil entendimento, uma vez que usa uma interface simples, recursos de arrastar e soltar para montagem com documentação abrangente e atualizada, juntamente com melhorias contínuas na usabilidade. Além disso, essa plataforma oferece a mais ampla variedade de ferramentas para apoiar tanto a aprendizagem presencial quanto cursos totalmente online, permitindo personalizar ativando ou desativando recursos básicos e integrando facilmente todos os elementos necessários para um curso, aproveitando sua ampla gama de recursos incorporados, que incluem ferramentas colaborativas externas, como fóruns, wikis, chats e blogs (MOODLEDOCS, 2021).

Como referência de compartilhamento de dados de aula e calendário organizado para os dados e compartilhamentos de materiais, o Moodle será utilizado como exemplo de inspiração nesse projeto para desenvolver a tela inicial, além de conteúdos e calendário, com conteúdos voltados mais para o público entre 6 à 12 anos.

2.2 Rosângela Trajano

Dedicada ao ensino infantil, a página de Rosângela Trajano abriga uma ampla gama de materiais de estudo, abrangendo desde o primeiro até o nono ano e englobando disciplinas como filosofia, sociologia, artes, ensino religioso, antropologia, educação financeira, entre outras. Organizado em subcategorias, o site oferece sugestões de atividades, incluindo material para colorir, além de histórias escritas de forma acessível para garantir que as crianças compreendam o conteúdo em uma linguagem adaptada a elas (TRAJANO, 2023).

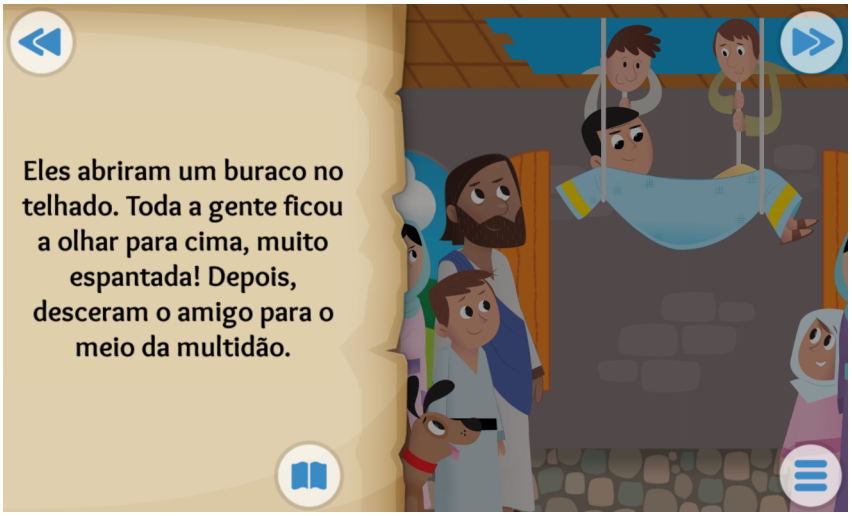
A plataforma seria utilizada de referência como compartilhamento de materiais para estudos bíblicos cristão, focado para de divulgação de materiais e conteúdos ao disponibilizar as aulas.

2.3 Bíblia para Crianças

Do grupo YouVersion, o aplicativo Bíblia para Crianças está disponível para smartphones e tablets Android e IOS. É uma aplicação totalmente gratuita, disponível em várias línguas em todo o mundo. O objetivo desse aplicativo é apresentar as histórias da Bíblia de forma ilustrada e com linguagem simplificada para crianças, oferecendo aventuras interativas, animações, perguntas sobre a história e jogos que buscam proporcionar uma experiência agradável e cuidadosamente projetada. O objetivo é motivar as crianças a retornarem sempre, contribuindo para o desenvolvimento de um amor pela Palavra de Deus desde tenra idade (YOUVERSION, 2013).

A plataforma seria utilizada de referência como referência para como distribuir essas histórias voltada ao público infantil entre os 0 aos 6 anos, com resumos mais simples e uma linguagem mais adequada para essa idade como ideias de materiais como pode ser visto na Figura 1.

Figura 1 – Exemplo de histórias no app Bíblia para Crianças



Fonte: Adaptado de YouVersion (2013).

3 REFERENCIAL TEÓRICO

Para o desenvolvimento de uma aplicação web simples, organizada e destinada ao uso das professoras, com o objetivo de facilitar a condução de suas aulas e otimizar as operações gerais do Ministério Infantil, faz-se necessário realizar um levantamento do que será necessário e entender qual o foco final da aplicação. Neste caso, busca-se com a ferramenta não apenas beneficiar os líderes do Ministério, permitindo que os líderes forneçam aos professores de maneira antecipada quais os conteúdos e dias de suas aulas permitindo que os mesmos consigam se preparar com antecedência para suas aulas e assim prestar aulas mais bem elaboradas, mas também apoiar as professoras na manutenção de uma sequência lógica de ensino. Isso é especialmente relevante devido à prática de rodízios de professoras a cada domingo, visando evitar a fadiga, permitindo assim uma continuidade coerente no desenvolvimento das aulas, além de oferecer ideias que facilitam o processo de ensino, centralizando o material necessário para a preparação das aulas das professoras. Mas acima de tudo, entregar as crianças que frequentam o Amigos do Reino uma experiência que permita com que elas absorvam da melhor maneira possível os ensinamentos de Deus para a vida delas.

Para viabilizar esta aplicação, é imprescindível contemplar as diversas ferramentas disponíveis no mercado, abrangendo linguagens de programação, frameworks e bibliotecas. A seleção criteriosa dessas tecnologias influencia diretamente na qualidade e no prazo de implementação do sistema.

Nesse contexto, é crucial seguir boas práticas de programação e empregar técnicas que facilitem a manutenção do sistema, garantindo seu funcionamento contínuo ao longo do tempo. A busca por uma abordagem cuidadosa na escolha e implementação das tecnologias visa não apenas atender às necessidades imediatas, mas também estabelecer uma base sólida para futuras expansões e aprimoramentos, consolidando assim o impacto positivo desejado na experiência educacional das crianças.

Neste capítulo, serão exploradas tecnologias essenciais para impulsionar o desenvolvimento do projeto, abrangendo ferramentas e metodologias que serão empregadas ao longo da aplicação.

3.1 Processo de Desenvolvimento

O processo de desenvolvimento abrange todo o período de criação do sistema. Por mais que no desenvolvimento deste sistema tenha apenas um desenvolvedor, é crucial utilizar práticas e metologias que auxiliem a organizar todas as etapas do projeto, para garantir que todas as partes importantes do sistemas sejam desenvolvidas e entregues ao final da aplicação no tempo estipulado. Neste contexto, para que isso possa ser possível, foram utilizadas técnicas de gerenciamento e organização de um sistema.

3.1.1 Método *Scrum*

Apesar de método no nome, *Scrum* não se propõe a ser uma metodologia com processos padronizados onde segue-se de maneira metódica uma série de etapas sequenciais para que obtenha os resultados desejados. O *Scrum* é um *framework* ágil amplamente utilizado no desenvolvimento de software para gerenciar e organizar projetos complexos, proporcionando uma abordagem flexível e iterativa para o desenvolvimento de produtos, enfatizando a colaboração, a adaptabilidade e a entrega contínua de valor ao cliente. Ele divide projetos em períodos denominados como *Sprints*, que duram algumas semanas e a definição de papeis como o *Product Owner*, *Scrum Master* e *Developers*. O *Product Owner* representa os interesses do cliente, logo, será o responsável em fornecer os requisitos do sistema juntamente com a ordem em que devem ser executados, além disso, ele quem irá avaliar a qualidades das entregas. O *Scrum Master* é o facilitador do processo, sendo assim, será o responsável de garantir com que o *Scrum* seja compreendido e aplicado de forma correta. Os *Developers* serão responsáveis em criar o produto seguindo os pedidos do cliente segundo a organização do *Scrum Master*. Para gestão de atividades, existem reuniões regulares, como o *Sprint Planning* para planejamento, o *Daily Scrum* para sincronização diária, o *Sprint Review* para mostrar o que foi feito e o *Sprint Retrospective* para avaliar e melhorar.

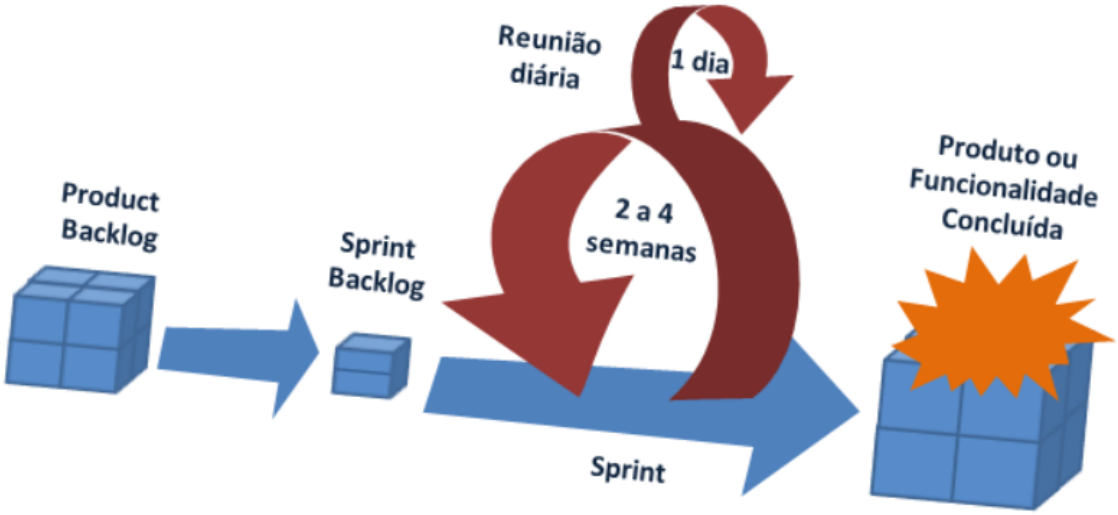


Figura 2 – Dinâmica do Scrum
Fonte: Adaptado de MindMaster (2023).

O *Scrum* utiliza listas de tarefas chamadas *Product Backlog* e o *Sprint Backlog*, que ajudam a priorizar o que precisa ser feito. O *Product Backlog* é uma lista em constante evolução, onde os itens podem ser adicionado, excluídos ou revisto pelo *Product Owner*, nela existem todas as funcionalidades, requisitos, melhorias e correções que precisam ser feitas ao produto, representando as demandas do cliente e também requisitos identificados pela equipe de desenvolvimento. O *Sprint Backlog* é uma lista dos itens do *Product Backlog* selecionados durante

a parte de planejamento para a *Sprint* que tem duração média de duas a quatro semanas, sendo o plano da entrega do produto ao longo do período, sendo exemplificada na Figura 2 (MINDMASTER, 2023).

3.1.2 Método Kanban

O método Kanban geralmente é usado junto com o *Scrum*, e consiste em uma abordagem visual para gerenciar o trabalho, originalmente desenvolvida no Japão para otimizar a produção. Hoje em dia, é amplamente utilizado em diversos contextos, incluindo desenvolvimento de software, gerenciamento de projetos, e até mesmo em tarefas do dia a dia. Para criar um quadro Kanban, é utilizado um painel subdividido em colunas que representam os estágios do processo, tais como "A fazer", "Em andamento"e "Concluído", conforme ilustrado na Figura 3. Em cada coluna, as tarefas são representadas por cartões que incluem informações cruciais, como descrição e responsável. Esses cartões são movidos entre as colunas de acordo com o progresso do desenvolvimento, proporcionando uma visualização clara e dinâmica do fluxo de trabalho. Para evitar sobrecarga, cada coluna tem um limite de cartões que podem estar em andamento simultaneamente.

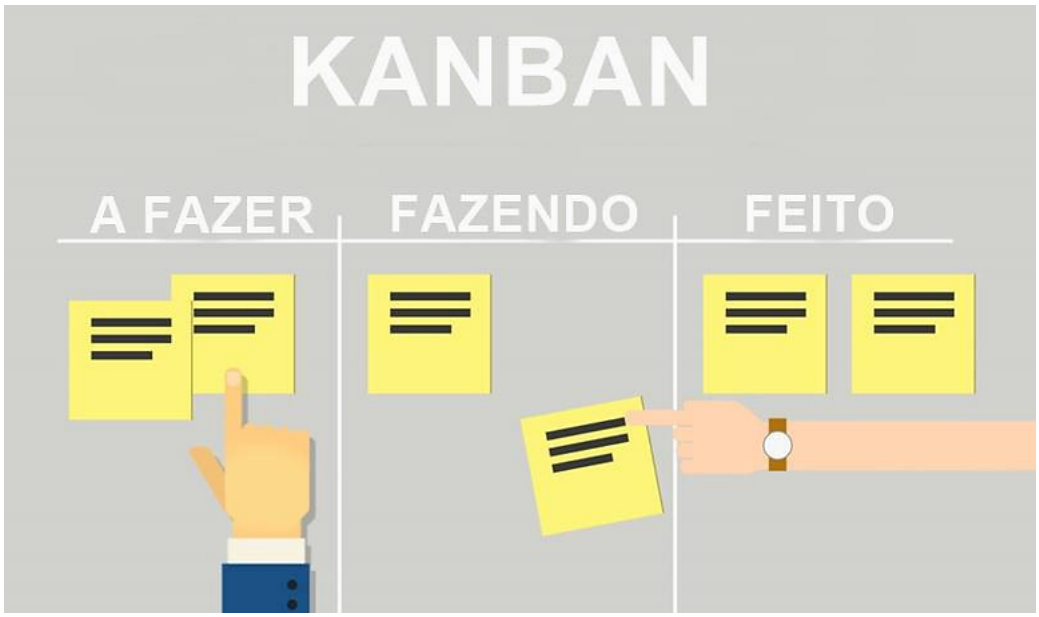


Figura 3 – Quadro Kanban
Fonte: Adaptado de Freitas (2022).

Com este processo, todos os membros da equipe podem visualizar facilmente o fluxo de trabalho, identificar gargalos e entender o que está sendo feito, além disso, é adaptável a mudanças e permite ajustes contínuos. Se algo não está funcionando, a equipe pode modificá-lo facilmente. Ao usar o Método Kanban, as equipes podem manter um fluxo de trabalho mais suave, priorizar tarefas de forma eficaz e responder rapidamente às mudanças nas demandas

do projeto. É uma ferramenta valiosa para promover a colaboração e aumentar a eficiência em diversas áreas de trabalho (SABINO, 2023).

3.1.3 Desenvolvimento Orientado a Testes

O Desenvolvimento Orientado a Testes, do inglês *Test Driven Development* (Desenvolvimento Orientado a Testes) (TDD) é uma abordagem no processo de desenvolvimento de software que enfatiza a criação de testes automatizados antes da implementação real do código. Tem como principal objetivo garantir a qualidade do software desde o início do processo de desenvolvimento. Ao criar testes antes da implementação do código, os desenvolvedores buscam detectar precocemente erros, melhorar a estrutura do código, facilitar a manutenção futura e manter o foco nas necessidades do usuário.

A abordagem visa criar uma suíte de testes abrangente que promova a confiança na funcionalidade e estabilidade do software, permitindo mudanças e atualizações com maior segurança. O ciclo do TDD segue as seguintes etapas *Red-Green-Refactor*, como pode ser visualizado na Figura 4. Na etapa *Red* os desenvolvedores escrevem um teste automatizado que inicialmente falha, representando a necessidade de uma nova funcionalidade ou alteração. Na fase *Green*, o código é implementado de maneira mínima para fazer o teste passar, garantindo que a funcionalidade desejada seja alcançada. Por fim, na fase *Refactor*, os desenvolvedores aprimoram o código sem alterar o comportamento, mantendo a integridade dos testes. Esse ciclo contínuo promove o desenvolvimento incrementado e aprimora a qualidade do código ao longo do tempo (GUEDES, 2019).

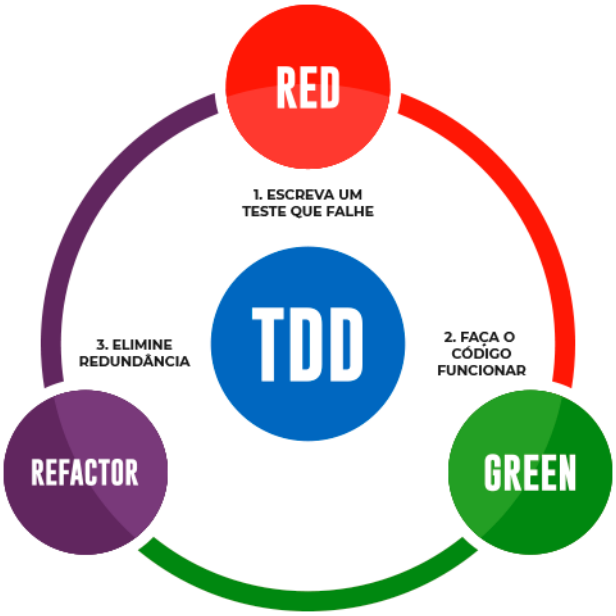


Figura 4 – Ciclo do TDD
Fonte: Adaptado de Guedes (2019).

3.1.4 Técnica de Priorização MoSCoW

A priorização de requisitos é uma etapa fundamental no gerenciamento de projetos, pois permite definir quais funcionalidades devem ser implementadas com maior urgência e quais podem ser postergadas. Entre as diversas abordagens disponíveis, a técnica MoSCoW (*Must have, Should have, Could have, Won't have* – Deve ter, Deveria ter, Poderia ter, Não terá) se destaca por sua simplicidade e eficácia na categorização dos requisitos de um sistema (PM3, 2022).

A técnica MoSCoW é amplamente utilizada em metodologias ágeis, como o *Scrum* (ver Seção 3.1.1), pois auxilia na tomada de decisões ao priorizar as entregas mais críticas. Essa abordagem propõe a classificação dos requisitos em quatro categorias principais:

- ***Must have*** **(Deve ter)**: São requisitos essenciais para o funcionamento do sistema. Sem esses elementos, o projeto não pode ser considerado viável. Geralmente, incluem funcionalidades básicas ou exigências regulatórias que não podem ser ignoradas.
- ***Should have*** **(Deveria ter)**: Representam requisitos importantes, mas não essenciais. Embora sua ausência não impeça a operação do sistema, podem afetar a experiência do usuário ou a eficiência do processo. São priorizados logo após os requisitos *Must have*.
- ***Could have*** **(Poderia ter)**: São requisitos opcionais que agregam valor ao sistema, mas que não são críticos para seu funcionamento. Esses itens podem ser implementados caso haja tempo e recursos disponíveis, mas sua ausência não compromete a entrega do produto.
- ***Won't have*** **(Não terá)**: Requisitos que foram identificados, mas que não serão implementados no ciclo atual do projeto. Podem ser considerados para versões futuras ou descartados caso percam relevância.

A aplicação da técnica MoSCoW permite que as equipes de desenvolvimento concentrem esforços nas entregas mais relevantes, evitando desperdício de tempo e recursos com funcionalidades de menor impacto. Além disso, ao adotar essa abordagem, os *stakeholders* (partes interessadas) conseguem alinhar expectativas de maneira objetiva, garantindo que os itens críticos sejam desenvolvidos prioritariamente.

A priorização MoSCoW também pode ser ajustada conforme a necessidade do projeto. Em alguns casos, subdivisões dentro das categorias podem ser criadas para refinar ainda mais a seleção de requisitos. Por exemplo, pode-se dividir os itens *Could have* em níveis diferentes, considerando sua viabilidade e impacto no sistema.

Ao longo do desenvolvimento deste projeto, a técnica MoSCoW foi utilizada para classificar as histórias de usuário e determinar a ordem de implementação das funcionalidades.

Essa abordagem contribuiu para um planejamento mais eficiente e estruturado, permitindo que os requisitos fossem organizados conforme sua real necessidade e disponibilidade de recursos (PM3, 2022).

3.2 Ferramentas de desenvolvimento

As ferramentas de desenvolvimento consistem em utilizar ferramentas apropriadas para aumentar a eficiência do desenvolvimento, reduzindo a necessidade de realizar tarefas manuais e entregando a organização e qualidade do sistema em construção. Neste contexto, para que isso possa ser possível, serão apresentadas ferramentas que facilitam o desenvolvimento de aplicações, contribuindo para organização, uso de boas práticas e manutenção do sistema.

3.2.1 GitHub

O *GitHub* é uma plataforma de desenvolvimento colaborativo baseado em nuvem que hospeda um *Version Control System* (Sistema de Controle de Versão) (VCS) chamado *Git* que fornece um ambiente para que desenvolvedores possam colaborar em projetos de software, rastrear mudanças no código-fonte, gerenciar problemas e realizar diversas outras tarefas relacionadas ao desenvolvimento de software. Tem como principal funcionalidade permitir que os desenvolvedores armazem seus projetos em reposiórios públicos ou privados, facilitando, assim, trabalho em equipe, pois dessa forma diferentes desenvolvedores podem colaborar em um projeto simultaneamente, fazendo alterações que ficam todas registradas pelo *GitHub* (LONGEN, 2023b).

Algumas ferramentas de desenvolvimento integradas ao *GitHub* serão de suma importância para concretização do projeto, além de centralizar todas as ideias em apenas um local. Estas serão discutidas nos seguintes subtópicos.

3.2.2 Git e GitFlow

O *Git* é um sistema de controle de versão distribuído amplamente utilizado no desenvolvimento de software, criado por Linus Torvalds em 2005 para ajudar no desenvolvimento do kernel do Linux, mas desde então tornou-se uma ferramenta essencial para projetos de software em geral. Por meio dele, qualquer desenvolvedor autorizado numa equipe pode gerenciar o código-fonte e seu histórico de mudanças usando ferramentas de linha de comandos, mas diferente dos sistemas de controle de versão centralizados, ele oferece ramificações de recursos, também conhecidos como *feature branches*, o que permite que cada desenvolvedor separe seu trabalho em ramificações locais para fazer suas mudanças no código (LONGEN, 2023b).

Para controle e uso dessas ramificações foi desenvolvido o modelo alternativo de ramificações denominado *Gitflow* que consiste no uso de ramificações de recursos e várias ramificações primárias, fornecendo um conjunto de diretrizes para organizar o fluxo de trabalho ao utilizar o *Git*, principalmente em projetos com um ciclo de vida mais complexo que venham a envolver desenvolvimento de novas funcionalidades, correções de bugs e manutenções. Comparado ao desenvolvimento baseado em troncos, o *Gitflow* tem mais ramificações de vida longa e commits maiores, pois neste modelo, os desenvolvedores criam uma ramificação de recurso e retardam a mescla com a ramificação de tronco principal até que o recurso esteja completo. Esse fluxo trabalho é constituído por uma ramificação principal, denominada de *main*, onde armazena o histórico da versão oficial do sistema, seguido pelas ramificações de desenvolvimento, denominada de *develop*, que contém os códigos para a próxima versão do sistema. Cada nova funcionalidade deverá ser desenvolvida a partir da *develop* e quando a funcionalidade for concluída, deve ser feito a mesclagem, ou *merge*, com a *develop* e quando esta tiver recursos suficiente para ir para ramificação principal, ela deve ser bifurcada em uma ramificação de lançamento, denominada como *release*. Quando a mesma estiver pronta para ser lançada é mesclada com a *main*, mas se durante esse processo for preciso corrigir problemas com rapidez, são criadas ramificações denominadas de *hotfix*, sendo estes baseados na *main* já que reflete ao código em produção. Todo o ciclo de trabalho pode ser visto na Figura 5 (ATLASSIAN, 2023).

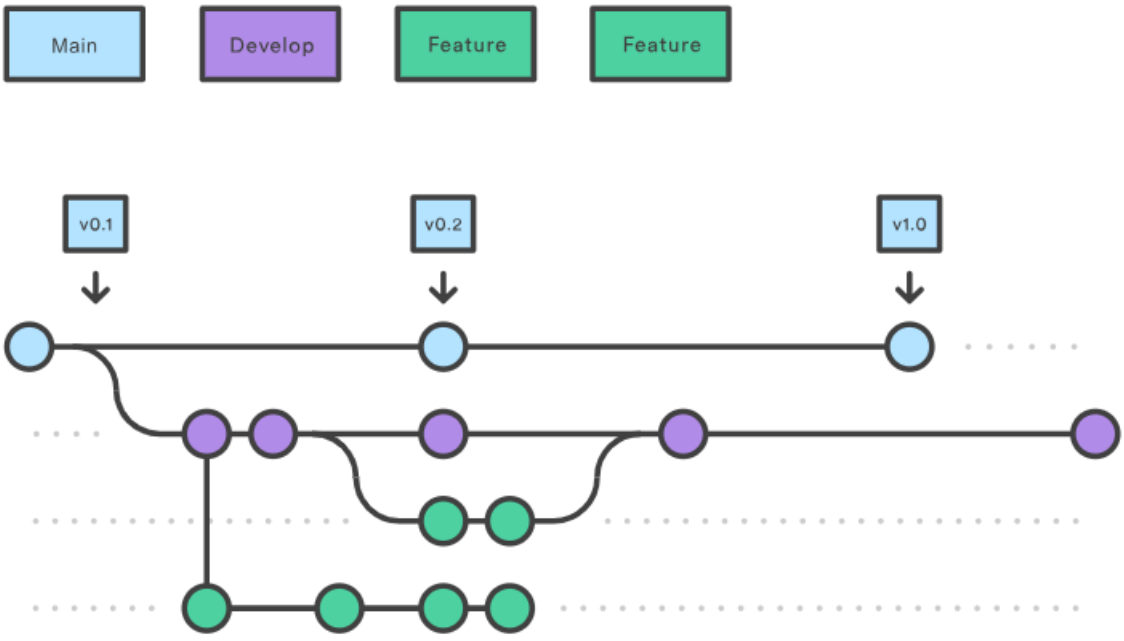


Figura 5 – Fluxo de Trabalho
Fonte: Adaptado de Atlassian (2023).

3.3 Tecnologias do lado do cliente

O lado do cliente, caracterizado como *Frontend*, desempenha um papel crucial na criação da interface visual de uma aplicação, pois esse possibilita a interação entre o usuário e o sistema, constituída por elementos visuais como layouts, botões e elementos interativos. Além disso, o *Backend* é responsável em interagir com o banco de dados por meios de requisições com o servidor para controle de informações, sendo para criar novos dados, editar ou exibir dados já existentes. Neste contexto, para que isso possa ser possível, serão apresentadas ferramentas que permitam a criação de uma interface visual adequada para interação com o usuário.

3.3.1 Angular

Angular é um *framework* de desenvolvimento da web criado pelo Google, usado para construir aplicações web e mobile interativas e dinâmicas. Por utilizar a arquitetura *Model-View-Controller* (Modelo-Visão-Controlador) (MVC), simplifica o desenvolvimento ao oferecer estruturas organizacionais, como componentes reutilizáveis e módulos, que facilitam a construção de interfaces de usuário complexas, além disso, oferece recursos como data binding, injeção de dependência e um sistema de roteamento integrado, o que permite aos desenvolvedores criar aplicações web robustas de forma mais eficiente. Na versão Angular, e não AngularJs, utiliza TypeScript, uma linguagem que contém todos os elementos e funcionalidades do JavaScript, mas com algumas funcionalidades a mais, como a tipagem estática, interfaces e outras construções que não estão presentes no JavaScript padrão. Com sua *Command Line Interface* (Interface de Linha de Comando) (CLI), sendo uma ferramenta de linha de comando, os desenvolvedores podem gerenciar facilmente tarefas como compilação, teste e implantação, tornando o processo de desenvolvimento mais ágil e consistente (LONGEN, 2023a).

3.4 Tecnologias do lado do servidor

O lado do servidor, caracterizado como *Backend*, desempenha o papel de cérebro da aplicação, onde se torna o responsável por receber e executar processos por requisições via *API*, sendo responsável por armazenar os dados do sistema, manipular os dados armazenados, definir rotas, processos e tratamento de dados, além da parte de segurança do sistema. Neste contexto, para que isso possa ser possível, serão apresentadas ferramentas que permitam a realização desse tratamento de dados do sistema.

3.4.1 Laravel

O Laravel é um popular *framework* de desenvolvimento web em PHP que foi criado por Taylor B. Otwell e é conhecido por sua elegância e facilidade de uso, sendo projetado para tornar o desenvolvimento web mais eficiente e agradável, oferecendo uma variedade de recursos com um sistema robusto de gerenciamento de banco de dados e uma sintaxe expressiva denominada *Eloquent* para interação com o banco de dados, sendo um poderoso sistema de roteamento. Além disso, integra o *Composer* para gerenciamento de dependências facilitando a inclusão e atualização de bibliotecas de terceiros. Possui uma sintaxe limpa e recursos avançados que simplificam as tarefas comuns, além de incentivar boas práticas de desenvolvimento por sua estrutura eloquente, promovendo a legibilidade e manutenção do código. É um *framework* com uma excelente curva de aprendizagem por ser fácil e aprender, além de poupar tempo por permitir que tarefas complexas sejam executadas em pouco tempo, acompanhado de uma documentação simples e direta frequentemente atualizada devido a sua comunidade ativa está sempre auxiliando (CARDOSO, 2023).

3.4.2 Docker

O Docker é uma plataforma de virtualização de contêineres que simplifica o processo de empacotamento, distribuição e execução de aplicativos em ambientes isolados, chamados contêineres. A conteinerização permite que vários aplicativos funcionem em diferentes ambientes complexos, pois cada contêiner contém todos os elementos necessários para executar uma aplicação. Isso elimina as diferenças entre ambientes em grupos de desenvolvimento.

A arquitetura do Docker é composta por quatro componentes: Cliente Docker, Servidor Docker, Imagens do Docker e Registro do Docker. O cliente Docker é o principal componente para criar, gerenciar e executar aplicativos em contêiner, sendo o principal método de controle do servidor Docker. O servidor Docker, também conhecido como o daemon do Docker, aguarda as solicitações da *API REST* feitas pelo cliente Docker e gerencia imagens e containers.

As imagens do Docker consistem nas instruções dadas ao servidor Docker com os requisitos sobre como criar um contêiner Docker. Por fim, o registro do Docker é um aplicativo do lado do servidor de código aberto usado para hospedar e distribuir imagens. É extremamente útil para armazenar imagens localmente e manter controle total sobre elas (CHAN, 2023).

Com o Docker e sua arquitetura, os desenvolvedores podem criar contêineres a partir de imagens padronizadas, compartilhando facilmente suas aplicações e suas dependências com outros membros da equipe, com contêineres leves que inicializam rapidamente, otimizando os recursos do sistema. Essa flexibilidade e portabilidade fazem do Docker uma ferramenta valiosa para orquestração de aplicativos em ambientes de nuvem, permitindo escalabilidade eficiente e facilitando a integração contínua e a implantação contínua (CI/CD) (CHAN, 2023).

3.4.3 MySQL

O MySQL é um sistema código aberto de gerenciamento de base de dados relacional *Structured Query Language* (Linguagem de Consulta Estruturada) (SQL), ou seja, ele armazena essas informações em tabelas separadas e as conecta com chaves, tendo sido desenvolvido e suportado pela Oracle, amplamente utilizado, conhecido por sua confiabilidade e desempenho. Permite aos desenvolvedores armazenar, organizar e recuperar dados de maneira eficiente. Com suporte a SQL, facilita a criação e a manutenção de bancos de dados relacionais para uma variedade de aplicativos, desde sistemas simples até sistemas mais complexos (KINSTA, 2021).

Os recursos do MySQL incluem a capacidade de lidar com grandes volumes de dados, suporte a transações *Atomicity, Consistency, Isolation and Durability* (Atomicidade, Consistência, Isolamento e Durabilidade) (ACID) para garantir a integridade dos dados, e a capacidade de ser integrado a várias linguagens de programação. A atomicidade é uma propriedade que garante que cada transação seja tratada como uma entidade única, ou seja, para a transação ter sucesso necessita que todas as operações a ser executadas passem com sucesso.

A consistência permite assegurar que uma transação somente leve o banco de dados se todos os dados forem válidos, mantendo a estabilidade do banco. O Isolamento permite que várias tabelas sejam lidas ou alteradas por vários usuários simultaneamente. A durabilidade garante que uma transação, uma vez executada, permanecerá neste estado mesmo que haja um problema grave no sistema (REIS, 2018).

3.4.4 Digital Ocean

A *Digital Ocean* é uma plataforma de computação em nuvem voltada para desenvolvedores, oferecendo serviços escaláveis e de alto desempenho para hospedagem de aplicações. Com uma interface simplificada e suporte a diversos sistemas operacionais, a plataforma permite a criação e gerenciamento de servidores virtuais (*droplets*), armazenamento em nuvem e banco de dados gerenciados (OCEAN, 2023).

3.5 Considerações Finais

A escolha das ferramentas deste sistema exige a seleção criteriosa de metodologias e tecnologias que garantam sua eficiência, organização e qualidade ao longo do processo. A adoção do *Scrum* e do *Kanban* permite estruturar o trabalho de forma iterativa e visual, enquanto o uso de *TDD* assegura maior confiabilidade ao código desde as primeiras etapas.

Além disso, ferramentas como *GitHub*, *GitFlow* e *Docker* viabilizam um fluxo de desenvolvimento mais colaborativo e estável. A escolha de *Angular* para o *frontend* e *Laravel* com

MySQL para o *backend* proporciona uma base sólida para a construção da aplicação e facilidade de manutenção.

4 MATERIAIS E MÉTODOS

4.1 Materiais

Para a organização e gestão do projeto, foram empregadas diversas ferramentas que desempenharam papéis fundamentais na implementação e no acompanhamento das tarefas.

A primeira delas foi o *GitHub*, uma plataforma de controle de versão. Esse sistema de versionamento permite o acompanhamento e registro das alterações no código-fonte, facilitando a colaboração entre os membros da equipe. Além disso, o *Git Flow* foi adotado como metodologia de trabalho, o que implica a criação de branches para desenvolvimento, revisão e implementação de novas funcionalidades. Essa abordagem contribui para a organização e rastreamento eficaz das alterações no projeto.

O *GitHub Projects* foi usada como a ferramenta para o gerenciamento de tarefas. Esse recurso permite a criação de painéis de tarefas, organizadas em colunas que representam etapas do projeto, tornando mais claro o andamento das atividades. O modo roadmap proporciona uma visão geral do progresso e das metas a serem alcançadas, auxiliando na definição de prioridades e prazos. Portanto, foi usado o modo roadmap.

No processo de prototipagem, o Figma foi empregado na criação de protótipos de telas. Essa ferramenta facilita a representação visual dos elementos que comporão o sistema, permitindo a visualização e a validação das interfaces de forma mais prática.

Para a modelagem do banco de dados, foi adotado o MySQL Workbench, uma ferramenta que possibilita a criação de diagramas e a definição das tabelas e relações necessárias para o funcionamento do sistema. Isso proporciona uma compreensão mais clara da estrutura de dados do projeto.

No desenvolvimento propriamente dito, para a parte frontend, o framework Angular foi utilizado. Sua escolha se deve, primeiramente a questão pessoal, por costume em utilizar o framework e gostar de sua usabilidade além da capacidade de criar interfaces de usuário interativas e responsivas, essenciais para a usabilidade do sistema. No que diz respeito ao backend, foi empregado o framework Laravel, que facilita o desenvolvimento de APIs e a integração com o banco de dados MySQL. O acesso ao banco foi realizado por meio do MySql Workbench, proporcionando uma interface amigável para gerenciar os dados.

Para garantir a consistência e compatibilidade dos componentes necessários, foi desenvolvido um ambiente Docker. Perimitindo criar um ambiente com todas as versões necessárias para funcionamento desse sistema.

O ambiente de desenvolvimento escolhido para a elaboração do código foi o Visual Studio Code, uma ferramenta que oferece recursos essenciais para a programação, como depuração e extensões que auxiliam no desenvolvimento.

Para a hospedagem do sistema online, foi utilizada a *Digital Ocean*, garantindo estabilidade, segurança e flexibilidade para a aplicação. A escolha dessa infraestrutura possibilita

uma implantação ágil e eficiente com monitoramento contínuo, essenciais para a manutenção do sistema em ambiente de produção.

4.2 Métodos

Neste projeto, foi adotada uma metodologia de desenvolvimento incremental, inspirada no Scrum. Embora o Scrum seja comumente utilizado por equipes, ele foi adaptado para atender às necessidades de um projeto desenvolvido individualmente. O desenvolvimento contou com o apoio do orientador e reuniões regulares com os líderes do ministério infantil "Amigos do Rei". O orientador e os líderes do ministério assumiram papéis importantes na metodologia Scrum, atuando como *Product Owners*, enquanto o aluno desempenhou o papel de *Developer*. O Professor orientador também desempenhou o papel de *Scrum Master*.

O desenvolvimento das atividades foi dividido em *sprints* quinzenais, que representam ciclos de trabalho. Para o gerenciamento de tarefas e acompanhamento do progresso, foi utilizado o recurso Projects do GitHub. Isso proporcionou uma visão geral do sistema, seus requisitos e o acompanhamento efetivo do que está sendo realizado, com graus definidos de importância para as tarefas.

Para identificação das tarefas, foi adotado um número de ID único, seguindo um padrão de cinco dígitos. Esse ID foi replicado na *branch* do projeto, facilitando a compreensão de qual tarefa está relacionada a cada *commit*. A estrutura da tarefa incluiu um título composto pelo ID e uma breve descrição do que deve ser feito. Em seguida, a tarefa foi detalhada de forma clara, incluindo a prioridade atribuída para indicar a urgência de sua realização. As prioridades podem ser:

- **Normal**: tarefa que pode ser desenvolvida quando possível.
- **Importante**: tarefa que precisa ser desenvolvida sem tanta urgência.
- **Urgente**: tarefa que precisa ser desenvolvida de forma prioritária.
- **Bug**: tarefa que precisa ser resolvida urgentemente para corrigir falhas no sistema causadas por bugs.

Cada tarefa tem um tempo previsto para conclusão, alinhado com a carga horária semanal de 15 horas dedicadas ao projeto pelo aluno. A organização das tarefas em sprint iniciou sendo subdividas e definidas dentro do *Backlog* para serem distribuídas a cada *sprint* seguindo o total de 30 horas, subdivido por listas identificadas pelo número da sprint e período definido. Cada coluna dessas views foram dividas em:

- **To Do**: tarefas que precisam ser desenvolvidas ao longo da sprint.
- **In Progress**: tarefa está em fase de desenvolvimento.

- **Review**: tarefa finalizada pronta para ser revisada ao fim da sprint e testada.
- **Done**: tarefa finalizada e aprovada.

Quanto à construção do projeto, foi empregado o Docker para criar contêineres controlados, o que facilita o desenvolvimento, evita problemas de versionamento de componentes e permite manutenções mais eficientes. Os contêineres foram divididos em:

- **amigos-back** : responsável por executar o framework Laravel e com todos os componentes necessários para sua utilização nas versões corretas, onde está concentrada toda a aplicação backend do sistema, como as APIs que serão consumidas, toda a lógica de tratamento dos dados enviados, além de segurança dos dados do sistema.
- **amigos-front** : responsável por executar o framework Angular e com todos os componentes necessários para sua utilização nas versões corretas, onde está concentrada toda a aplicação frontend do sistema, como as telas que serão acessadas pelo cliente.
- **amigos-db** :responsável por executar o MySQL, onde serão armazenados todos os dados gerados pelos usuários que servem para a funcionalidade do sistema.

A adoção das metodologias e materiais descritos foi fundamental para a organização e o sucesso deste projeto. Essa abordagem proporcionou um sistema funcional e alinhado às expectativas do cliente, garantindo uma distribuição eficiente das tarefas. Dessa forma, o desenvolvimento ocorreu dentro do prazo estipulado, com um acompanhamento claro de cada etapa, possibilitando tempo hábil para testes e aprimoramentos.

5 ANÁLISE E PROJETO DO SISTEMA

5.1 Análise do sistema

Com base na metodologia *Scrum* explanada na Seção 3.1.1, foram definidas as histórias dos usuários utilizando a estrutura *"COMO [persona], QUERO [funcionalidade], PARA [algum fim]."* e os critérios de aceitação.

Os níveis de acesso de usuários ao sistema são: Administrador, Professor e Pais. Assim, estes usuários podem utilizar recursos específicos de acordo com suas categorias, contribuindo assim para a melhoria do aprendizado das crianças participantes da EBD e proporcionando uma melhor interação entre professores e os pais no ensino bíblico das crianças como objetivo focal do sistema.

As histórias do usuário estão apresentadas na Tabela 1, formando o Backlog do sistema. A tabela possui as seguintes colunas: identificador (ID) numérico da história do usuário, o peso para representar o grau de prioridade e a própria descrição da história do usuário.

Tabela 1 – Histórias de Usuário (Ordenado por Peso)

ID	Peso	Descrição
1	5	COMO administrador QUERO cadastrar usuários PARA acessarem ao sistema
6	5	COMO professor QUERO acessar calendário PARA me programar para aula
2	4	COMO administrador QUERO inserir o cronograma PARA aulas dos professores
7	4	COMO professor QUERO acessar materiais inseridos pelo administrador PARA preparar aula
8	4	COMO professor QUERO documentar aula PARA mostrar o que foi feito
14	4	COMO administrador QUERO alterar cronogramas PARA quanto trocar professores
3	3	COMO administrador QUERO salvar apostila em aula PARA professores acessarem
9	3	COMO professor QUERO confirmar aula PARA mostrar que eu dei a aula marcada
11	3	COMO professor QUERO alterar perfil PARA definir senha e atualizar dados
15	4	COMO pais QUERO saber o que meu filho aprendeu PARA reforçar em casa
5	2	COMO administrador QUERO cadastrar eventos PARA professores acessarem
4	2	COMO administrador QUERO emitir relatórios PARA professores acessarem
10	2	COMO professor QUERO acessar grupo do WhatsApp PARA dúvidas e avisos
12	1	COMO professor QUERO enviar conteúdos PARA auxiliar a criar aulas por tema
13	1	COMO administrador QUERO aprovar conteúdos PARA disponibilizar a todos

Fonte: Autoria própria (2025).

Durante o desenvolvimento do sistema, algumas histórias de menor peso foram removidas do planejamento, conforme mostrado na Tabela 2. Essa decisão foi tomada para priorizar

funcionalidades essenciais, garantindo que os recursos mais críticos estivessem plenamente operacionais e alinhados às necessidades do cliente para a utilização do sistema.

Tabela 2 – Histórias Removidas

ID	Peso	Descrição
8	4	COMO professor eu QUERO documentar aula PARA mostrar o que foi feito
3	3	COMO administrador eu QUERO salvar apostila em aula PARA professores acessarem
9	2	COMO professor eu QUERO confirmar aula PARA mostrar que eu dei a aula marcada
4	2	COMO administrador eu QUERO emitir relatórios PARA professores acessarem
12	1	COMO professor eu QUERO enviar conteúdos PARA auxiliar a criar aulas por tema
13	1	COMO administrador eu QUERO aprovar conteúdos PARA disponibilizar a todos

Fonte: Autoria própria (2025).

Em relação aos pesos, foram utilizadas a técnica de Moscow, explicada na Seção 3.1.4, onde foi conversado com o cliente para entender todas as suas necesidades e assim dividir cada uma das histórias desenvolvidas de acordo com o grau de importância para o cliente. Sendo assim, a categoria *Could have* foi subdividida com base no grau de necessidade mais elevado (PM3, 2022). A seguir, apresenta-se a distribuição desses pesos divido de 1-5 onde 1 o menor grau de importância e o 5 o maior:

- **5 - *Must have* (Deve ter)**: sendo como requisito de maior grau de importância e precisa ter no sistema, algo indispensável.
- **4 - *Should have* (Deveria ter)**: sendo como requisito do sistema que deveriam ter no sistema, logo podem não ter a importância do anterior, mas está logo em seguida.
- **3 - *Could have* (Poderia ter)**: sendo um requisito do sistema que pode vir a ter mas sendo melhor que tenha do que o de peso 2.
- **2 - *Could have* (Poderia ter)**: sendo um requisito do sistema que pode vir a ter, mas menos importante que o de peso 3.
- **1 - *Won't have* (Não terá)**: sendo um requisito do sistema que não crítico para o sistema funcionar e caso não possa vir a ser implementado não seria realmente um problema.

5.2 Prototipação das Telas

A aplicação apresentada nesse trabalho busca entregar ao usuário uma experiência de fácil utilização tanto para obter informações de aulas, materiais e cronogramas, como para for-

necer dados de suas aulas que foram ministradas. Todo acesso ao sistema precisa ser apenas por meio de login, onde este seria divido pelas seguintes categorias:

- **Admin**: qualidade de administrador do sistema, tendo acesso a todas as áreas do sistema, além de áreas exclusivas que permitem com que ele possa gerar relatórios.
- **Professor**: o acesso de professor vem com mais uma categoria de restrição, sendo por turmas as quais ministra suas aulas.
- **Pais**: o acesso dos pais possibilita que eles entrem em uma página de *feed*, proporcionando uma experiência simplificada e direta para acompanhar o que o filho aprendeu no último domingo. Isso inclui a visualização de fotos da aula e a leitura de recados importantes.

Para efetuar o login, e também sendo a tela inicial do sistema, foi feito um protótipo com auxílio do Figma, como apresentado na seção 4.1 e pode ser visto na Figura 6. Em caso de ao mesmo tempo ser professora e mãe de alguma criança que frequenta o ministério, ao logar, o usuário escolhe qual tipo de perfil quer acessar.

Figura 6 – Tela de Login
Fonte: Autoria Própria (2023).

Ao realizar o login, tanto professoras quanto administradores têm acesso direto à tela inicial do sistema, onde encontram informações cruciais sobre as próximas aulas, conforme evidenciado nas Figuras 7 *a* e *b*. Essas informações incluem o dia da aula (retângulo azul), o tema a ser abordado (retângulo verde), a turma responsável pela aula (retângulo rosa) e o texto base para a próxima aula (retângulo lilás). Além disso, ambos têm acesso a um menu lateral (círculo com o logo e retângulo amarelo) para explorar outras áreas do sistema. Esse menu é flexível, podendo ser ocultado ou exibido conforme o clique do usuário.

Na barra de navegação (*navbar*), destacada na parte superior em degradê rosa nas Figuras 7 *a* e *b*, os usuários podem acessar a área de calendários, onde visualizam os cronogramas relacionados ao seu perfil. Para o administrador, o calendário exibirá informações de todos os professores. Adicionalmente, a barra de navegação oferece acesso à área de mensagens, direcionando para o *Whatsapp* atual das professoras. Isso facilita a comunicação em caso de trocas de escala, necessidade de assistência ou dúvidas. Os usuários também podem acessar a tela de perfil e efetuar logout do sistema. Clicar no logo sempre redireciona para a tela inicial, independentemente da parte do sistema em que estejam. As representações visuais dessas funcionalidades podem ser visualizadas nos protótipos das Figuras 7 *a* e *b*.

(a) Sem Menu **(b) Com Menu**

Figura 7 – Tela Inicial

Fonte: Autoria Própria (2023).

Na tela de perfil, o usuário encontra informações sobre o usuário logado, como nome, foto, formas de contato e para qual turma ou turmas a professora está no cronograma para dar suas aulas. Na Figura 8, representado pelo retângulo azul terão todas as informações da professora um breve comentário sobre quem é a professora, explicando se ela tem filhos do ministério e quais são suas faixas etárias que mais se identifica, além de informar se tem alguma limitação ou auxilia em mais partes do ministério, como em cultos noturnos, teatro ou música. No retângulo lilás, terão informações resumidas das últimas aulas dadas pela professora em questão.

Além disso, através do menu terá acesso a telas de multimídias, onde professores irão encontrar vídeos, atividades e músicas como materiais de auxílios para ajudar professoras a dar suas aulas e até fazer a hora da música inicial que serão cadastrados previamente pelos administradores ao criarem as aulas e inserirem os materiais de apoio. Terão também acesso para a página da apostila, nela terá todas as aulas da apostila separadas por temas, onde dentro terá os capítulos respectivos àquele tema e qualquer arquivo auxiliar. Outra página será a de registro de aula, onde a professora depois de dar a aula, ela nessa página registra qual foi a aula que ela deu naquele dia, dá uma breve explicação do que foi feito, incluindo atividade e anexa fotos dos alunos em aula. Todas essas fotos receberão marcas d'água e apenas o administrador terá acesso a essas fotos sem marcas para respeito de imagem das crianças e

Figura 8 – Tela de Perfil
Fonte: Autoria Própria (2023).

professores. Pelo menu lateral, também terá acesso ao *feed* explicado posteriormente nesta seção.

5.3 Modelagem do Banco de Dados

O banco de dados deste sistema passou por ajustes ao longo do desenvolvimento para melhor atender às necessidades do projeto, tendo sido desenvolvido pensado em uma forma de entregar todas as funcionalidades necessárias apresentadas nas histórias de maneira organizada e com uso de chaves estrangeiras que simplifiquem as buscas deixando de maneira mais rápida as buscas no banco. A Figura 9 apresenta a modelagem inicial, enquanto a Figura 10 mostra a versão final, refinada com base nos aprendizados do processo. A modelagem do banco de dados inicial consiste nas seguintes tabelas principais:

- **users**: tabela que contém os dados de todos os usuários do sistema, com suas informações de contato, como nome, e-mail e telefone, tipo de login (administrativo, professor ou pais), status de ativo ou inativo, além de login e senha para acessar o sistema.
- **children**: tabela que contém o nome da criança e sua data de nascimento para determinar sua turma vigente.
- **classes**: tabela que contém as divisões de classes, sendo esta uma tabela de valores já pré-determinados como *babies*, primário, juniores e pré-adolescentes, contendo quais idades pertecem a essa turma e se o turno é da manhã ou da noite.

- **schedule**: tabela que contém as aulas e programações do ministério, sendo subdivida em tipos (aula, reunião, confraternização e eventos externos), data que irá acontecer, para qual usuário representa aquela programação e qual classe se refere.
- **multimedia**: tabela que contém os arquivos de multimídia do banco referente a arquivos de aula e relacionados com temas definidos, contendo o nome, url, tipos de arquivos (documento, vídeo ou foto), usuário que adicionou e seu status (aprovado, reprovado e aguardando).
- **themes**: tabela que contém os temas que serão encaixados os arquivos de multimídia, sendo definida apenas por seu id e nome.

As tabelas de conexão auxiliares serão:

- **professor-has-class**: tabela com cardinalidade 1 para N de professores com classes, onde um professor pode esta em uma ou mais classes diferentes.
- **parent-has-children**: tabela de conexão entre os pais e seus filhos com cardinalidade N para N, uma vez que o casal pode ter acesso individualizado e ter mais de um filho.
- **schedule-has-photos**: tabela que conecta a programação com fotos que serão inseridas para demonstrar o que aconteceu, sendo com cardinalidade 1 para N, onde cada programação pode ter de uma ou mais fotos.
- **theme-has-multimedia**: tabela com cardinalidade de N para N de temas com os arquivos adicionados, onde um tema pode ter diversos arquivos multimídia e um arquivo multimídia pode ter mais de um tema.
- **class-has-multimedias**: tabela com cardinalidade de 1 para N de classe com arquivos multimídia, onde uma classe pode ter um ou vários arquivos multimídia de referência.

Por fim, a tabela de histórico será:

- **log**: tabela que irá conter tudo que aconteceu no sistema, gerando histórico que será utilizado para gerar relatórios no sistema sobre tudo que aconteceu ao longo do período. Contendo descrição do que aconteceu, qual tipo de ação foi feita (adicionar, editar, deletar, trocar, dentre outras), qual tabela se refere a informação e qual o id nessa tabela se refere, além disso, que usuário realizou essa ação.

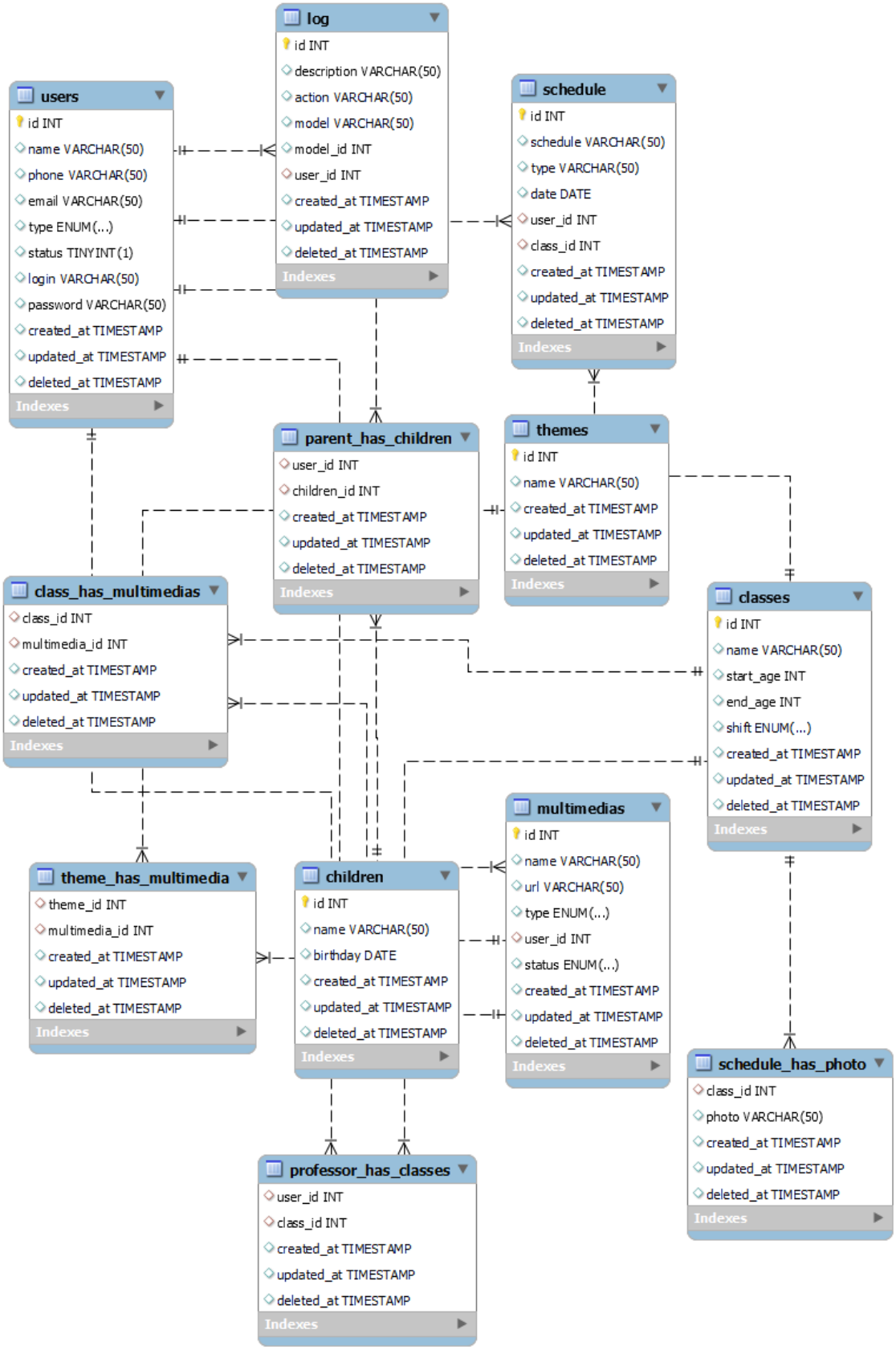


Figura 9 – Modelagem do Banco de Dados

Fonte: Autoria Própria (2023).

Durante o desenvolvimento, a estrutura do banco de dados foi revisada para se adequar às novas funcionalidades do sistema, garantindo consistência e eficiência. As principais alterações incluem:

- **Tabela `users`:** Adicionado o campo `gender` para armazenar o gênero do usuário e a coluna `remember_token` para suporte a sessões persistentes.
- **Tabela `log`:** Padronização do tamanho dos campos e manutenção da chave estrangeira `user_id`.
- **Tabela `classes`:** Removido o campo `user_id`, agora gerenciado na tabela intermediária `professor_has_classes`.
- **Tabela `schedule`:** Adicionado o campo `theme_id` para relacionar cronogramas às aulas.
- **Tabela `themes`:** Incluídos os campos `content` e `resume` para detalhar os temas das aulas.
- **Tabela `multimedias`:** Padronização do campo `type` para classificar conteúdos como `video` ou `image`.
- **Remoção de tabelas:** `password_reset_tokens` e `personal_access_tokens` foram eliminadas por não serem essenciais ao novo modelo de dados.

Essas mudanças otimizam o banco de dados, tornando-o mais alinhado às necessidades do sistema e facilitando sua manutenção e escalabilidade.

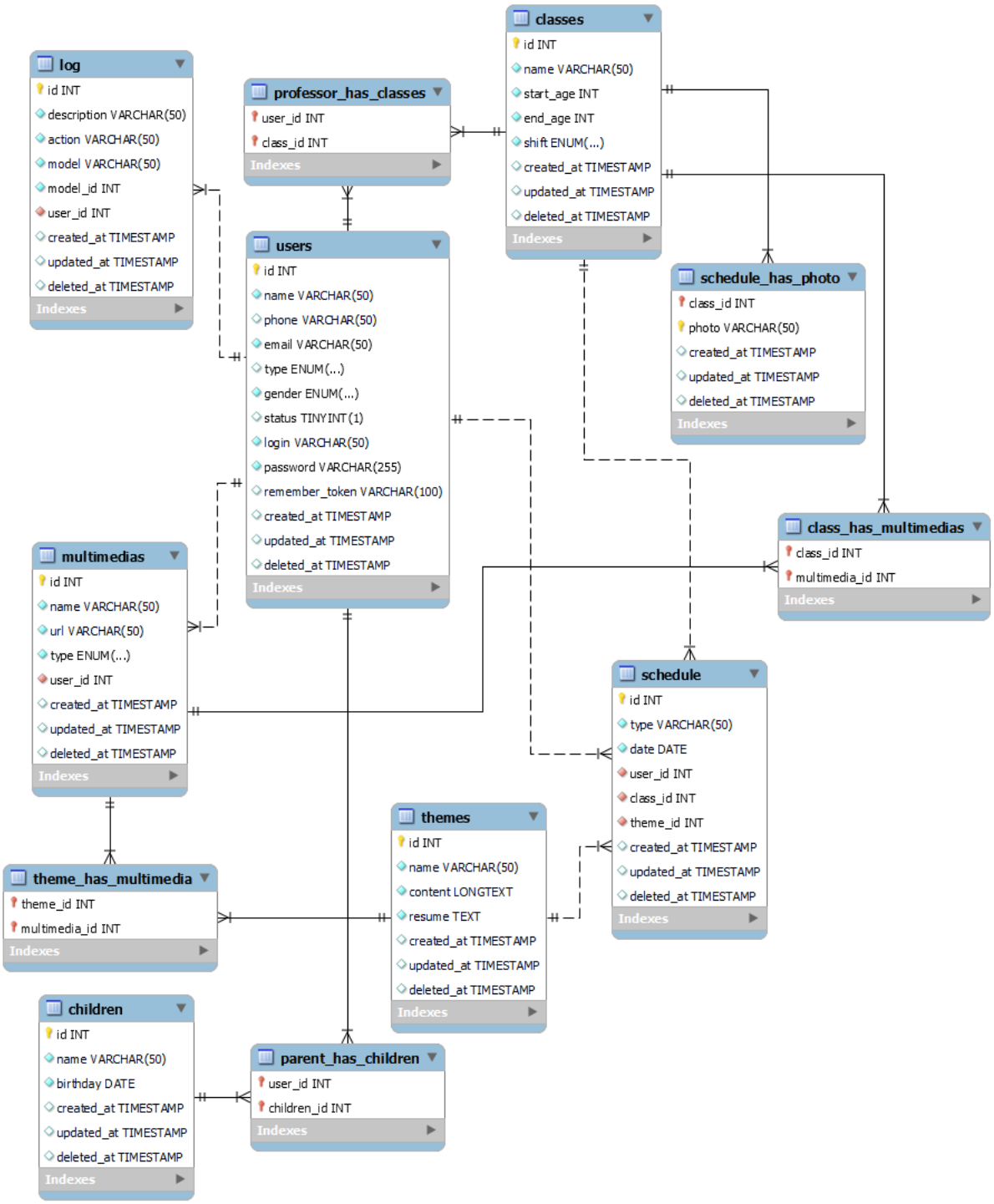


Figura 10 – Modelagem do Banco de Dados Final

Fonte: Autoria própria (2025).

6 DESENVOLVIMENTO

O desenvolvimento deste projeto foi estruturado em doze sprints, cada uma com duração de quinze dias, ao longo do semestre dedicado à sua implementação. Seguindo alguns princípios do Scrum, conforme descrito na Seção 3.1.1, essa abordagem permitiu um processo iterativo e flexível, facilitando ajustes e aprimoramentos contínuos ao longo do desenvolvimento.

6.1 Fluxo de Desenvolvimento

O desenvolvimento do projeto seguiu uma metodologia estruturada, visando garantir a organização, a estabilidade e o acompanhamento contínuo de todas as atividades, buscando seguir as diretrizes previamente apresentadas na Seção 4.2.

Ao garantir que a base do projeto estivesse estável e funcional na base local e na configuração do ambiente de desenvolvimento com Docker, todo o código foi adicionado em um repositório público no GitHub, permitindo o controle de versão e o compartilhamento do progresso com todos os envolvidos no projeto.

Para organizar o trabalho de forma eficaz, todas as histórias de usuários descritas na Tabela 1 foram estruturadas na ferramenta GitHub Project, utilizando a metodologia Kanban, apresentada na Seção 3.1.2. Isso permitiu o planejamento, acompanhamento e visualização clara das atividades de cada Sprint, facilitando a definição das tarefas a serem realizadas e o gerenciamento eficiente do projeto por meio da divisão das atividades em projetos Kanban.

Ao longo de cada *sprint*, os *commits* foram realizados ao final de cada projeto, com o objetivo de registrar o progresso e garantir que todos os desenvolvedores contribuíssem de forma contínua. Durante o processo, foi dada atenção especial à resolução de conflitos e erros que pudessem surgir.

A utilização do GitHub, que por meio do Git, permitiu a organização das tarefas e versionamento do código proporcionou uma boa prática de desenvolvimento, permitindo uma abordagem colaborativa e eficaz, com a *master* sempre atualizada antes de iniciar uma nova *sprint*.

6.2 Detalhamento das Sprints

6.2.1 Sprints 1 a 4 - Configuração do Ambiente de Desenvolvimento

A configuração do ambiente foi uma das etapas mais complexas do projeto, pois envolveu a definição e organização da infraestrutura necessária para garantir um ambiente estável e eficiente para o desenvolvimento. Essa fase foi dividida em duas partes principais: a configuração do ambiente local e a containerização da aplicação utilizando Docker.

A primeira etapa consistiu na configuração do ambiente de desenvolvimento local, garantindo que todas as dependências e extensões essenciais estivessem devidamente instaladas. O projeto foi estruturado em duas aplicações principais: o frontend, desenvolvido em Angular, e o backend, implementado em Laravel. Estes sendo o framework selecionados para realização desse projeto.

Para assegurar o funcionamento adequado de ambas as aplicações e garantir que estivessem atualizadas para suas versões mais recentes, foi realizada a instalação e configuração dos pacotes necessários. Além disso, a organização inicial dos arquivos foi definida, incluindo o gerenciamento de dependências por meio do Composer (para Laravel) e do npm (para Angular). Essa estruturação foi fundamental para estabelecer uma base sólida que facilitasse a posterior containerização do projeto.

O objetivo dessa etapa foi proporcionar um ambiente de desenvolvimento funcional e eficiente, adequado tanto às necessidades do contexto acadêmico quanto ao ambiente eclesiástico, seguindo boas práticas modernas de desenvolvimento web.

Após a configuração do ambiente local, iniciou-se o processo de containerização do projeto, permitindo a integração de todos os serviços necessários dentro de um ambiente isolado e replicável. Para isso, foi utilizada a ferramenta Docker, juntamente com o Docker Compose, para definir e gerenciar múltiplos contêineres.

A estrutura dos contêineres foi definida da seguinte maneira:

- `amigos-frontend/` - contém a aplicação Angular;
- `amigos-backend/` - contém a API Laravel;
- `docker-compose.yml` - define e configura os contêineres;
- `Dockerfile` - contém as instruções para a construção das imagens Docker.

Os contêineres foram organizados da seguinte forma:

- **Frontend**
  - Imagem: construída a partir do Dockerfile do Angular;
  - Nome do serviço: `frontend`;
  - Porta: 8082 (Nginx).
- **Backend**
  - Imagem: construída a partir do Dockerfile do Laravel;
  - Nome do serviço: `backend`;
  - Porta: 8081 (Laravel).
- **Banco de Dados**

  - Imagem: MariaDB;
  - Nome do serviço: `db`;
  - Porta: 3307 (MariaDB).

- **phpMyAdmin**
  - Imagem: phpMyAdmin;
  - Nome do serviço: `phpmyadmin`;
  - Porta: 8083 (phpMyAdmin).

Após a configuração e validação dos contêineres, os serviços foram integrados e testados, garantindo a comunicação entre frontend, backend e banco de dados. Essa etapa foi essencial para assegurar que o ambiente de desenvolvimento fosse portátil, escalável e de fácil replicação, permitindo o isolamento de cada serviço, o que minimiza o impacto de possíveis falhas individuais.

Um dos principais desafios enfrentados nessa fase foi garantir a comunicação eficiente entre os contêineres, bem como a sincronização dos dados locais com aqueles armazenados nos contêineres. A cada nova implementação, era necessário assegurar que todas as alterações fossem corretamente replicadas no ambiente containerizado. Além disso, foi fundamental garantir que que inicialmente foi completamente construído (*build*), pudesse funcionar corretamente sempre que os contêineres fossem iniciados, incluindo a instalação de todas as bibliotecas e dependências essenciais para o correto funcionamento da aplicação.

6.2.2 Sprint 5 - Estruturação do Banco de Dados

A estruturação do banco de dados seguiu a lógica apresentada na Seção 5.3, sendo desenvolvida utilizando o Laravel auxiliar o desenvolvedor a garantir uma implementação eficiente e escalável. Para isso, foram criadas todas as *migrations* necessárias, definindo as tabelas e seus relacionamentos. Além disso, foram elaborados *seeders* para popular o banco de dados com dados iniciais, permitindo a utilização do sistema de forma adequada desde as primeiras interações.

A modelagem inicial do banco de dados foi estruturada com tabelas principais para usuários, crianças, classes, cronograma de atividades, arquivos multimídia e temas. Também foram criadas tabelas auxiliares para representar relações como professores e suas classes, além da vinculação entre pais e filhos. Essa abordagem permitiu organizar e estruturar os dados de forma clara, refletindo as necessidades do sistema.

Durante o desenvolvimento, diversas adaptações foram necessárias à medida que o sistema evoluía e novas funcionalidades eram integradas. Algumas das mudanças implementadas incluem:

- **Adição de novas colunas** em tabelas existentes para armazenar informações essenciais que não haviam sido previstas inicialmente, como status de usuários, permissões específicas e categorias adicionais para os arquivos multimídia.
- **Ajustes nos relacionamentos** entre as tabelas para melhor refletir a lógica do sistema, garantindo que as conexões entre usuários, classes e eventos estivessem corretas e eficientes.
- **Modificação da estrutura de algumas tabelas**, como a de classes, para incluir mais detalhes sobre os períodos de aulas e as faixas etárias de cada turma.
- **Otimização dos seeders** para que o preenchimento inicial do banco de dados fosse mais completo e refletisse melhor o uso real da aplicação.

Essas mudanças foram fundamentais para garantir que o banco de dados atendesse plenamente às necessidades do sistema, permitindo a evolução do projeto sem comprometer sua integridade e escalabilidade. A implementação dessas alterações será detalhada nos próximos tópicos, destacando os ajustes específicos realizados em cada uma das tabelas e seus impactos no funcionamento da aplicação.

6.2.3 Sprint 6 - Autentificação e Autorização para acesso de dados

A implementação da segurança do sistema foi um dos principais desafios deste projeto, exigindo uma abordagem cuidadosa para garantir a proteção das rotas e a comunicação segura entre o *frontend* e o *backend*. O objetivo principal foi assegurar que as rotas fossem acessíveis apenas para usuários autenticados, retornando um erro de acesso negado caso contrário.

Para isso, foi realizado um estudo aprofundado sobre o uso do *storage* e dos *cookies* do navegador, permitindo a integração segura entre os dois *frameworks* utilizados. O foco foi garantir que toda a lógica de segurança fosse centralizada no *backend*, aplicando técnicas de criptografia para reforçar a proteção dos dados e impedir acessos não autorizados.

As principais medidas implementadas foram:

- **Estruturação para divisão de rotas públicas e autenticadas**, garantindo que apenas usuários logados pudessem acessar determinadas áreas do sistema a medida que as rotas fossem adicionadas.
- **Ajustes nas *controllers***, aplicando verificações constantes para validar a autenticação do usuário antes de processar qualquer requisição.
- **Implementação de *middlewares* de segurança** para interceptar as requisições e assegurar que apenas usuários devidamente autenticados pudessem acessar os recursos protegidos.

- **Refinamento dos repositórios de dados**, garantindo que os retornos fossem sempre personalizados para o usuário logado, evitando exposição indevida de informações.

Com essas melhorias, foi possível estabelecer um ambiente mais seguro para os usuários, protegendo tanto os dados sensíveis quanto a integridade da comunicação entre o *frontend* e o *backend*. Essa estruturação sólida da segurança permitirá maior confiabilidade e estabilidade ao longo do desenvolvimento contínuo do sistema.

Um dos principais desafios enfrentados nessa fase foi garantir a segurança e a passagem do token correto pelas rotas, enfrentando problemas de autentificação com as bibliotecas fornecidas pelo Laravel e garantir que os *middlewares* recebecem os dados corretos para permitir o acesso ao sistema. Para isso foi feito por meio de *interceptors* envios de *tokens* armazenados em sessões do Angular para serem enviados ao Lavarel, onde ele iria decodificar com auxílio da biblioteca de autentificação do framework *sanctum* para análise e conferência dos dados corretos a serem fornecidos.

6.2.4 Sprint 7 - Desenvolvimento das Telas de Login e Página Inicial

Durante esta sprint, foram desenvolvidas as histórias de IDs 1[1] e 11[2].

Seguindo a estrutura previamente estabelecida para a segurança do sistema, a comunicação entre o *frontend* e o *backend* foi um aspecto fundamental nesta etapa do desenvolvimento. O principal objetivo foi garantir que o processo de autenticação ocorresse de forma segura e eficiente, validando os dados do usuário no *backend* e retornando as informações necessárias para a interface.

Durante o processo de login, o sistema verifica as credenciais do usuário, e caso estejam corretas, um *token* de autenticação é gerado e armazenado, estabelecendo uma sessão segura. Caso contrário, o usuário recebe um alerta informando o erro. Esse *token* é utilizado para manter a conexão ativa e permitir que a página inicial carregue corretamente as informações do usuário logado.

Na página inicial, foram implementadas funcionalidades para exibir dados personalizados, garantindo que cada usuário visualizasse apenas suas próprias informações. Entre os principais dados apresentados, destacam-se:

- Nome e informações básicas do usuário.
- Eventos vinculados ao usuário, organizados de forma acessível.
- Foto de perfil, permitindo uma identificação visual rápida.
- Outras configurações personalizadas relacionadas ao perfil do usuário.

[1] COMO administrador QUERO cadastrar usuários PARA acessarem ao sistema
[2] COMO professor QUERO alterar perfil PARA definir senha e atualizar dados

Além da lógica de funcionamento, a tela de login foi uma das que demandou mais tempo no desenvolvimento do *CSS* para ficar mais similar ao protótipo e ainda interativa. A necessidade de criar uma interface visualmente atrativa e responsiva exigiu atenção especial à estética, garantindo uma experiência de usuário intuitiva e agradável. Ao mesmo tempo, toda a lógica de segurança foi cuidadosamente implementada para proteger os dados sensíveis e evitar vulnerabilidades no processo de autenticação.

Com essas implementações, foi possível estabelecer um fluxo de login e a exibição inicial das informações do usuário, mantendo tanto segurança quanto usabilidade no sistema.

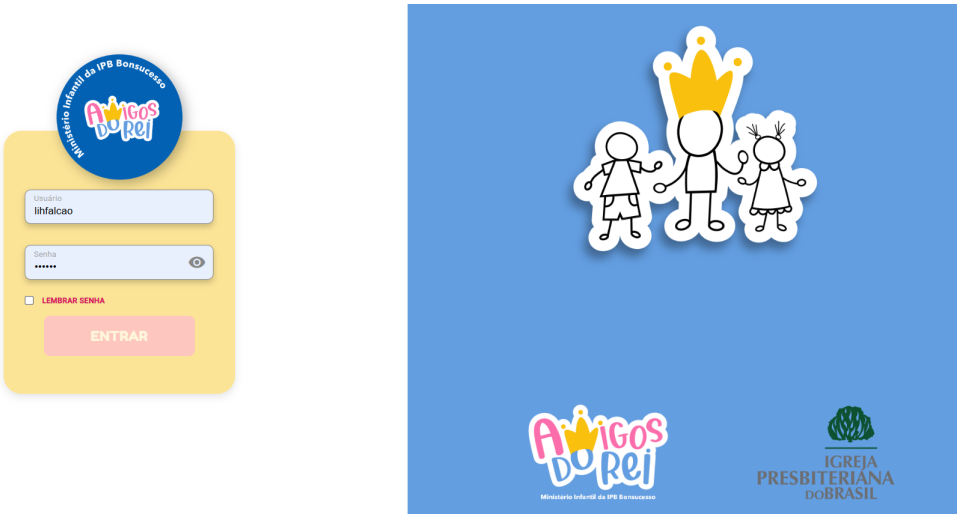


Figura 11 – Tela de Login Final
Fonte: Autoria própria (2025).

Figura 12 – Tela Inicial Final
Fonte: Autoria própria (2025).

6.2.5 Sprint 8 e 9 - Criação da Tela de Perfil e Menu

Nesta sprint, foi desenvolvida a história de IDs 10[3] e 11[4].

A implementação da tela de perfil apresentou diversos desafios, especialmente na manipulação e atualização dos dados do usuário. Um dos principais recursos dessa funcionalidade foi a possibilidade de alteração das informações do perfil, permitindo que o próprio usuário pudesse modificar seus dados pessoais diretamente pela interface do sistema. Além disso, foi necessário implementar o gerenciamento de imagens de perfil, o que envolveu o desenvolvimento de um mecanismo de upload e manipulação de imagens através de uma API, armazenando-as na pasta pública do Laravel.

Nesta fase do projeto, também foram criados dois componentes fundamentais para a estrutura da interface: o ***footer*** e o ***navbar***, projetados como componentes isolados para serem utilizados em praticamente todas as páginas do sistema, o que permitiu padronizar a navegação e melhorar a experiência do usuário. Para garantir a integridade e a segurança das informações do perfil, os dados foram estruturados no *backend* e enviados ao *frontend*, identificando via token qual era o usuário logado naquele momento, assegurando que cada usuário tivesse acesso apenas às suas próprias informações. Entre os dados exibidos, destacam-se:

- Informações pessoais do usuário, garantindo que somente o dono do perfil pudesse editá-las.
- Resumo da agenda de eventos do usuário, exibindo as aulas e compromissos vinculados ao seu perfil.
- Formatação e exibição dos dados utilizando *pipes*[5] no Angular para formatação de dados de telefone, datas de maneira agradável para o usuário.

Outro grande desafio foi a estilização da interface, afim de garantir que a tela de perfil mantivesse uma aparência amigável e intuitiva, ao mesmo tempo em que seguisse o design planejado. Como a lógica de segurança já havia sido ajustada nas *sprints* anteriores, essa parte do desenvolvimento foi menos problemática, permitindo que o foco principal estivesse na organização e apresentação dos dados.

Por fim, essa fase foi a maior responsável pelas modificações no banco de dados, com ajustes na estrutura das tabelas, como a adição de novas colunas, a remoção de colunas obsoletas e a redefinição de alguns relacionamentos. Essas mudanças foram essenciais para garantir que o banco de dados estivesse alinhado às novas funcionalidades implementadas, tornando a interface do sistema mais acessível e organizada e proporcionando uma experiência de uso aprimorada e intuitiva para os usuários. Podendo ser vistas nas Figuras 13 e 14

[3] COMO professor QUERO acessar grupo do WhatsApp PARA dúvidas e avisos
[4] COMO professor QUERO alterar perfil PARA definir senha e atualizar dados
[5] Funções que transformam valores de entrada em templates HTML

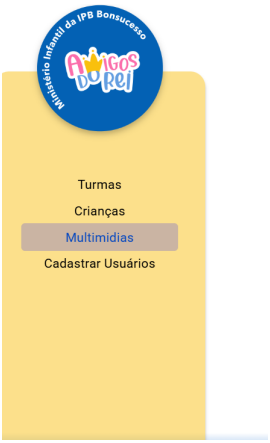


Figura 13 – Menu

Fonte: Autoria própria (2025).

Figura 14 – Tela Perfil Final

Fonte: Autoria própria (2025).

6.2.6 Sprint 10 - Implementação da Tela de Calendário

Durante esta sprint, foram desenvolvidas a história de IDs 2 [6] e 6[7]. Foi implementada a tela de calendário utilizando a biblioteca *FullCalendar* do Angular, com o objetivo de exibir a agenda de eventos como pode ser visto na Figura 15. Ao acessar essa tela, o usuário tem como visão inicial o dia e o mês atual, com todos os eventos agendados já visíveis no calendário. A interface foi projetada para que a navegação fosse simples e eficiente, permitindo que os usuários visualizassem a agenda do mês de forma clara e sem sobrecarga de informações.

Ao clicar em qualquer item do calendário, o usuário será direcionado para uma tela detalhada, onde todas as informações do evento selecionado são exibidas, incluindo os professores vinculados à ele e as turmas associadas, onde além de visualizar pode-se editar nessa mesma

[6] COMO administrador QUERO inserir o cronograma PARA aulas dos professores
[7] COMO professor QUERO acessar calendário PARA me programar para aula

tela o evento ou até deletar o mesmo. Isso proporcionou uma experiência fluida e prática na consulta e edição de eventos, com um layout organizado e de fácil compreensão.

Também foi adicionado um botão para criar novos eventos. Caso o usuário não encontrasse um tema já existente ao tentar adicionar um evento, a é possível que ele crie um novo tema diretamente na tela de adição do evento, o que aumentou a flexibilidade e a personalização da plataforma. Essa característica garantiu que a plataforma atendesse de forma eficiente a diferentes necessidades dos usuários, tornando a tela de calendário uma ferramenta poderosa e prática dentro do sistema. E estes podem ser vistos na Figura 16.

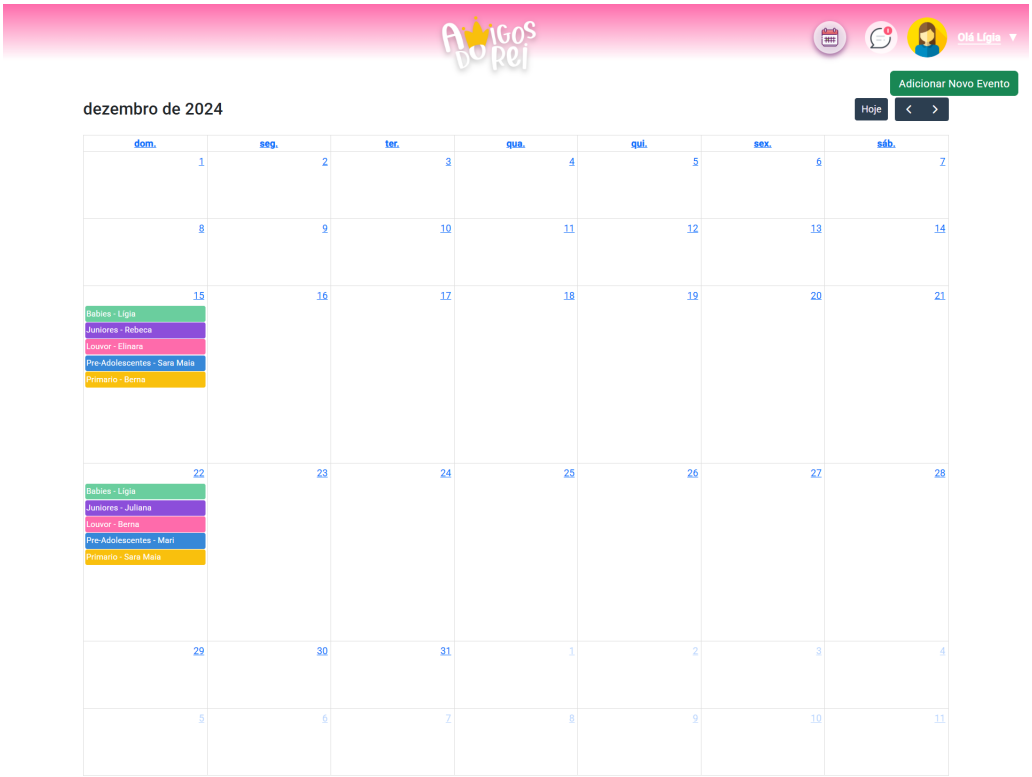
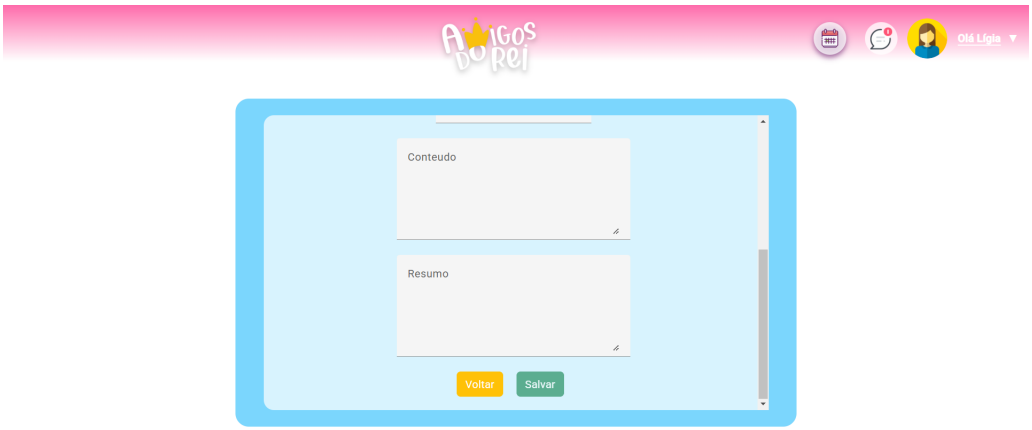


Figura 15 – Calendário

Fonte: Autoria própria (2025).

(a) Adicionar Eventos



(b) Adicionar Eventos
Figura 16 – Adicionar Eventos
Fonte: Autoria própria (2025).

6.2.7 Sprint 11 - Desenvolvimento das Telas de Turmas e Eventos

Nesta sprint, foram desenvolvidas as histórias de IDs 5[8], 7[9], 9[10] e 14 [11]. Foram desenvolvidas as telas de **Turmas** e **Eventos**, proporcionando uma interface organizada e intuitiva para a gestão dessas informações dentro do sistema.

A tela de **Turmas** exibe uma listagem completa de todas as turmas disponíveis, organizadas de acordo com seus respectivos nomes e faixas etárias. Além da visualização, foi implementada a funcionalidade de gerenciamento dos professores, permitindo que novos professores fossem adicionados ou removidos de cada turma conforme necessário. Além disso, foi disponibilizada a opção de criação e remoção de turmas, garantindo maior flexibilidade na administração do sistema. As turmas foram categorizadas por turnos, facilitando a organização

[8] COMO administrador QUERO cadastrar eventos PARA professores acessarem
[9] COMO professor QUERO acessar materiais de aulas PARA preparar aula
[10] COMO professor QUERO confirmar aula PARA mostrar que eu dei a aula marcada
[11] COMO administrador QUERO alterar cronogramas PARA quando trocar professores

e a alocação dos alunos e professores. Essa estrutura permitiu que a plataforma se tornasse mais dinâmica e adaptável às necessidades específicas dos usuários, garantindo um controle eficiente sobre as turmas disponíveis.

Na tela de **Eventos**, seguindo a lógica previamente estabelecida na implementação do calendário, foram desenvolvidas novas interfaces para o gerenciamento de classes e eventos. Já era possível adicionar um novo evento, definir seus temas e atribuí-los a um professor, além de visualizar detalhes sobre cada tema. No entanto, nesta *sprint*, foi criada uma nova funcionalidade que permite a edição desses dados, oferecendo aos usuários a possibilidade de modificar informações de uma classe existente, adicionar mais professores a uma classe ou até mesmo excluir classes que não sejam mais necessárias. Essas melhorias ampliaram significativamente a capacidade de gerenciamento dentro do sistema, tornando a administração de eventos mais eficiente e acessível.

As implementações dessa *sprint* podem ser vistas nas Figuras 17 e 18.

(a) Ver Turma

(b) Editar Turma

Figura 17 – Turmas

Fonte: Autoria própria (2025).

(a) Ver Classe

(b) Editar Classe

Figura 18 – Classes

Fonte: Autoria própria (2025).

6.2.8 Sprint 12 - Cadastro de Usuários

Na última *sprint* de desenvolvimento, foi desenvolvida a história de ID 1[12]. Foi implementada a tela de **Cadastro de Usuários**, permitindo que novos usuários fossem registrados no sistema de forma simples e intuitiva. Durante o processo de cadastro, o usuário pode inserir seus principais dados pessoais, como **nome** e **e-mail**, além de definir suas credenciais de acesso, incluindo **login** e **senha**.

Além dessas informações básicas, essa tela também inclui a funcionalidade de definição do **tipo de usuário**, categorizando cada novo cadastro como **professor**, **administrador** ou **responsável**. Essa categorização é essencial para determinar os níveis de acesso e as permissões dentro do sistema, garantindo que cada usuário tenha acesso apenas às funcionalidades correspondentes ao seu perfil.

[12] COMO administrador QUERO cadastrar usuários PARA acessarem ao sistema

A implementação dessa funcionalidade foi fundamental para estruturar o gerenciamento de usuários no sistema, assegurando um controle sobre os diferentes papéis ao longo do sistema. Podendo ser visto sua implementação na Figura 19.

Figura 19 – Cadastro de Usuários
Fonte: Autoria própria (2025).

6.3 Considerações sobre o Desenvolvimento

O desenvolvimento do sistema demandou um planejamento e uma execução alinhada às necessidades dos usuários. As tecnologias e metodologias empregadas possibilitaram um fluxo de trabalho estruturado, assegurando a implementação progressiva das funcionalidades essenciais. O processo de desenvolvimento passou por diversas *sprints*, seguindo as histórias previamente selecionadas pelo seu peso, garantindo que a aplicação estivesse funcional.

Ao longo do desenvolvimento, foram implementadas **12 sprints**, cada uma abordando diferentes aspectos do sistema, desde a configuração inicial do ambiente até a implementação de funcionalidades específicas para os usuários. No total, foram desenvolvidas 9 histórias de usuário, atendendo aos requisitos previamente estabelecidos. Esse resultado demonstra que as histórias planejadas para implementação foram devidamente desenvolvidas, garantindo que o sistema final contemplasse as necessidades propostas. Dessa forma, o projeto foi entregue com sucesso, proporcionando uma base sólida para futuras expansões e aprimoramentos.

A organização das tarefas e a priorização das entregas foram fatores determinantes para o sucesso do projeto, possibilitando a adaptação contínua às exigências do sistema e garantindo um resultado final entregasse um sistema que atende as necessidades principais do cliente e de fácil manutenção.

Durante o desenvolvimento, enfrentaram-se desafios na instalação e comunicação entre os contêineres, exigindo a reinstalação e atualização para versões mais recentes a fim de corrigir inconsistências e garantir que o sistema passasse pelo processo de *build* sem falhas.

Um dos principais obstáculos foi o *deploy* da aplicação, pois a configuração da rede entre os contêineres apresentou dificuldades que impactaram a implementação. Para solucionar esse problema, optou-se por estruturar a aplicação com *Docker*, assegurando um ambiente replicável e isolado. Essa decisão exigiu a migração para uma nova plataforma de *hosting* e a reconfiguração das máquinas virtuais, entretanto, como haviam diferenças no código de produção e de desenvolvimento, não pode ser colocado no ar até o presente momento.

7 CONCLUSÃO

O presente trabalho teve como objetivo o desenvolvimento de uma aplicação web para a organização da EBD voltada para crianças de zero a doze anos, proporcionando um ambiente mais estruturado para professores e pais. O sistema buscou solucionar as dificuldades enfrentadas pelos professores do Ministério Infantil da IPB na organização de escalas, conteúdos de aula e compartilhamento de informações, além de permitir um acompanhamento mais próximo dos pais ao aprendizado de seus filhos.

Com a implementação do sistema, foi possível centralizar os dados da EBD, garantindo que os professores tivessem acesso facilitado às escalas de aula e ao material didático necessário para cada encontro. A interface intuitiva permite que os usuários visualizem sua próxima escala diretamente na página inicial e consultem todas as datas importantes no calendário integrado. Além disso, o sistema disponibiliza conteúdos previamente organizados para cada aula, incluindo textos da apostila, sugestões de atividades, vídeos e músicas, garantindo que os professores possam preparar suas aulas de forma mais eficiente.

Outra funcionalidade relevante foi a disponibilização de um acesso específico para os pais, permitindo que acompanhem as aulas ministradas por meio do calendário do sistema. Assim, os responsáveis podem visualizar o resumo e o conteúdo completo das aulas dadas, bem como identificar quais professoras estiveram responsáveis por cada classe. Essa funcionalidade tem como objetivo incentivar o acompanhamento do aprendizado em casa e reforçar os conteúdos abordados em sala.

Os objetivos específicos estabelecidos na introdução foram integralmente atendidos, contribuindo para uma gestão mais eficiente da EBD. O desenvolvimento seguiu um planejamento estruturado em *sprints*, aplicando metodologias ágeis para garantir a implementação progressiva das funcionalidades essenciais. Com isso, o sistema foi projetado para ser intuitivo e acessível, reduzindo a dependência do WhatsApp para organização e comunicação, consolidando todas as informações relevantes em um único local.

Embora a aplicação já esteja desenvolvida, sua implementação efetiva pelo Ministério Infantil ainda não ocorreu devido aos problemas na hospedagem do sistema. No entanto, espera-se que, no futuro, sua adoção traga benefícios significativos para a organização da EBD, proporcionando um ensino mais estruturado e acessível. Assim, a Igreja Presbiteriana Bonsucesso reafirma seu compromisso com a formação cristã das crianças, utilizando a tecnologia como um meio para aprimorar o ensino e a comunicação entre professores, alunos e pais.

7.1 Trabalhos Futuros

Durante o desenvolvimento do sistema, foram priorizadas as funcionalidades essenciais para garantir seu funcionamento adequado. No entanto, algumas melhorias e expansões podem

ser consideradas para versões futuras, caso surjam novas demandas. A Tabela 3 apresenta algumas histórias de usuário que podem ser implementadas para aprimorar o sistema.

Tabela 3 – Sugestões de Histórias Futuras

ID	Peso	Descrição
8	4	COMO professor eu QUERO documentar aula PARA mostrar o que foi feito
4	2	COMO administrador eu QUERO emitir relatórios PARA professores acessarem
12	1	COMO professor eu QUERO enviar conteúdos PARA auxiliar a criar aulas por tema
13	1	COMO administrador eu QUERO aprovar conteúdos PARA disponibilizar a todos
16	1	COMO pai eu QUERO ver as imagens do meu filho em aula

Fonte: Autoria própria (2025).

Além dessas melhorias, o sistema pode ser expandido para incluir recursos adicionais, como uma área de mídias mais robusta, contendo sugestões de vídeos, músicas e atividades interativas para auxiliar na preparação das aulas. Outra possibilidade seria a implementação de um sistema de relatórios para análise do que foi ensinado aos alunos, fornecendo métricas úteis para a gestão pedagógica da EBD.

Por fim, a adoção de novas funcionalidades dependerá da necessidade identificada pelos usuários e da evolução do sistema ao longo do tempo. O projeto, portanto, representa um primeiro passo para modernizar a estrutura de ensino da EBD, garantindo uma experiência mais organizada e enriquecedora para os professores e permitir a aproximação maior dos pais com o ministério sobre o que está sendo ensinado aos seus filhos.

REFERÊNCIAS

ATLASSIAN. **Saiba tudo sobre o Gitflow Workflow**. 2023. Disponível em: https://www.atlassian.com/br/git/tutorials/comparing-workflows/gitflow-workflow. Acesso em: 17 nov. 2023.

BERTINATTI, N.; NASCIMENTO, E. Fraga Vilas-Bôas Carvalho do. **Os modelos de organização das escolas dominicais presbiterianas no Brasil: o ideal e o possível**. 2011. Disponível em: https://www.snh2011.anpuh.org/resources/anais/14/1300850691_ARQUIVO_TRABALHOCOMPLETOANPUH2011.pdf. Acesso em: 08 out. 2023.

BÍBLIA Sagrada. Nova almeida atualizada. [*S.l.*]: Sociedade Bíblica do Brasil, 2017.

CARDOSO, R. **Laravel: Conheça o framework PHP mais utilizado hoje em dia!** 2023. Disponível em: https://www.locaweb.com.br/blog/temas/codigo-aberto/laravel-framework-php/. Acesso em: 17 nov. 2023.

CHAN, D. **O Que é Docker e Como Ele Funciona? – Docker Explicado**. 2023. Disponível em: https://www.hostinger.com.br/tutoriais/o-que-e-docker. Acesso em: 17 nov. 2023.

FREITAS, A. V. **Kanban: o método de produção da Toyota que se tornou aliado da gestão de projetos**. 2022. Disponível em: https://www.promadjr.com/post/kanban-o-m%C3%A9todo-de-produ%C3%A7%C3%A3o-da-toyota-que-se-tornou-aliado-da-gest%C3%A3o-de-projetos. Acesso em: 17 nov. 2023.

GOOGLE. **Google Drive**. 2012. Disponível em: https://www.google.com/intl/pt-BR/drive/. Acesso em: 08 out. 2023.

GUEDES, M. **Afinal, o que é TDD?** 2019. Disponível em: https://www.treinaweb.com.br/blog/afinal-o-que-e-tdd. Acesso em: 17 nov. 2023.

KINSTA. **O Que é MySQL? Uma Explicação Simples para Quem Está Começando**. 2021. Disponível em: https://kinsta.com/pt/base-de-conhecimento/o-que-e-mysql/. Acesso em: 17 nov. 2023.

LONGEN, A. S. **O Que É Angular: O Framework Que Transformará Seu Código**. 2023. Disponível em: https://www.hostinger.com.br/tutoriais/o-que-e-angular. Acesso em: 17 nov. 2023.

LONGEN, A. S. **O Que é GitHub, Para Que Serve e Como Usar**. 2023. Disponível em: https://www.hostinger.com.br/tutoriais/o-que-github. Acesso em: 17 nov. 2023.

LÜCKE, N. C. F. S. A importância do estímulo no desenvolvimento da criança. **Revista Científica Multidisciplinar Núcleo do Conhecimento**, São Paulo, v. 12, n. 06, p. 33–44, jun 2019. ISSN 2448-0959.

MINDMASTER. **Scrum: A Metodologia Ágil Explicada de forma Definitiva**. 2023. Disponível em: https://mindmaster.com.br/scrum/. Acesso em: 17 nov. 2023.

MOODLEDOCS. **Sobre o Moodle**. 2021. Disponível em: https://docs.moodle.org/all/pt_br/Sobre_o_Moodle. Acesso em: 08 out. 2023.

OCEAN, D. **Cloud computing simplified: The DigitalOcean developer cloud**. 2023. Disponível em: https://www.digitalocean.com. Acesso em: 08 fev. 2025.

PM3. **Método MoSCoW: framework para ajudar a priorizar tarefas**. 2022. Disponível em: https://www.cursospm3.com.br/blog/metodo-moscow-framework-para-priorizar-tarefas/?gad_source=1&gclid=CjwKCAiA-P-rBhBEEiwAQEXhHyRKjkfx4Owjvd9E7zIlTLOVxVBgJ89BHESM78bf2DTWLXYBB-hyKhoCR7BwE. Acesso em: 15 dez. 2023.

REIS, F. dos. **Conceitos de Bancos de Dados – O que significa ACID**. 2018. Disponível em: http://www.bosontreinamentos.com.br/bancos-de-dados/conceitos-de-bancos-de-dados-o-que-significa-acid/. Acesso em: 17 nov. 2023.

SABINO, R. **Kanban: o que é, o Método Kanban, principais conceitos e como funciona no dia a dia**. 2023. Disponível em: https://www.alura.com.br/artigos/metodo-kanban. Acesso em: 17 nov. 2023.

TRAJANO, R. **Material didático de estudo bíblico para crianças**. 2023. Disponível em: https://rosangelatrajano.com.br/material-didatico-de-estudo-biblico-para-criancas/. Acesso em: 08 out. 2023.

YOUVERSION. **Bíblia para Crianças**. 2013. Disponível em: https://play.google.com/store/apps/details?id=com.bible.kids&hl=pt_BR&gl=US&pli=1. Acesso em: 08 out. 2023.