UNIVERSIDADE ESTADUAL DA PARAÍBA
CURSO DE BACHARELADO EM COMPUTAÇÃO

NOEMI LEANDRO MEDEIROS DA NOBREGA

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UM PROGRESSIVE WEB APP PARA INSTITUIÇÕES RELIGIOSAS UTILIZANDO DJANGO FRAMEWORK

PATOS
2024

NOEMI LEANDRO MEDEIROS DA NÓBREGA

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UM PROGRESSIVE WEB APP PARA INSTITUIÇÕES RELIGIOSAS UTILIZANDO DJANGO FRAMEWORK

Trabalho de Conclusão de Curso apresentado à Universidade Estadual da Paraíba como requisito para obtenção do título de bacharel em computação.

Área de concentração: Engenharia de Software e Desenvolvimento Web.

Orientador(a): Esp. Giovanna Trigueiro de Almeida Araujo

PATOS

2024

N754d Nóbrega, Noemi Leandro Medeiros da.

Desenvolvimento e implementação de um *progressive web* app para instituições religiosas utilizando *django framework* [manuscrito] / Noemi Leandro Medeiros da Nóbrega. - 2024.

83 f. : il. color.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Ciência da computação) - Universidade Estadual da Paraíba, Centro de Ciências Exatas e Sociais Aplicadas, 2024.

"Orientação : Prof. Esp. Giovanna Trigueiro de Almeida Araújo, Coordenação do Curso de Computação - CCEA".

1. Django framework. 2. Progressive web app. 3. Desenvolvimento de software. 4. Tecnologia na igreja. I. Título

21. ed. CDD 004.65

Elaborada por Kaliane Eveny Martins de Oliveira - CRB - 15/986 BSC7

NOEMI LEANDRO MEDEIROS DA NOBREGA

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UM PROGRESSIVE WEB APP PARA INSTITUIÇÕES RELIGIOSAS UTILIZANDO DJANGO FRAMEWORK

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Ciência da Computação da Universidade Estadual da Paraíba, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação

Aprovada em: 22/11/2024.

Documento assinado eletronicamente por:

- **Giovanna Trigueiro de Almeida Araújo** (***.352.004-**), em **30/11/2024 19:44:16** com chave **a14dabeeaf6c11ef9f3f06adb0a3afce**.
- **José Aldo Silva da Costa** (***.862.334-**), em **01/12/2024 00:12:58** com chave **2abc267eaf9211efb6f21a1c3150b54b**.
- **Samuel Alves Medeiros** (***.187.184-**), em **01/12/2024 09:16:51** com chave **256c155cafde11efafac1a7cc27eb1f9**.

Documento emitido pelo SUAP. Para comprovar sua autenticidade, faça a leitura do QrCode ao lado ou acesse https://suap.uepb.edu.br/comum/autenticar_documento/ e informe os dados a seguir.
Tipo de Documento: Termo de Aprovação de Projeto Final
Data da Emissão: 02/12/2024
Código de Autenticação: bcd6c6

Dedico este trabalho ao meu Senhor, que me permitiu chegar até aqui e me sustentou em todo o meu caminho. "*Porque dele, e por ele, e para ele, são todas as coisas; glória, pois, a ele eternamente. Amém.*" Romanos 11:36.

AGRADECIMENTOS

Agradeço primeiramente ao meu Senhor, a quem dedico toda a minha vida, pois sem Ele nada disso seria possível.

À minha mãe, Joselanda, que sempre acreditou em mim com todo o seu coração e me criou com toda a sua dedicação e amor, sendo um dos principais pilares da minha formação.

Ao meu irmão, Samuel, por ter cuidado de nós quando precisamos em todos os momentos difíceis.

À minha avó, Iolanda, que nunca mediu esforços em nos auxiliar.

Ao meu pai, Maurílio, pelos incentivos e por sempre acreditar em mim.

À todos os meus familiares que me ajudaram com suas orações, auxílios e palavras de apoio. Em especial, a Miriam, que possibilitou a posse do meu primeiro notebook, sendo fundamental durante a minha formação.

À todos os meus amigos, que sempre acreditaram em mim e me impulsionaram em todos os momentos.

À minha amiga e colega de curso, Anna Lys, por ter sido presente em tantos momentos e ter sido uma verdadeira companheira na minha formação.

Agradeço a Neto, que esteve presente desde o ensino médio até aqui, se disponibilizando em todos os momentos, independente das circunstâncias.

À minha professora e orientadora, Giovanna, agradeço por ter acreditado em meu potencial, por todo o apoio e por ter verdadeiramente auxiliado na orientação ao longo deste trabalho.

LISTA DE ABREVIATURAS E SIGLAS

CBVs	Class Based Views
CSRF	Cross-site Request Forgery
CSS	Cascading Style Sheets
DOM	Document Object Model
GUI	Graphic User Interface
HTML	HyperText Markup Language
ID	Identificador
JIT	Just-in-Time
MVC	Model-View-Controller
ORM	Object-Relational Mapping
POO	Programação Orientada a Objetos
PWA	Progressive Web App
SGBD	Sistema Gerenciador de Banco de Dados
SQL	Structured Query Language
SSH	Secure Shell
UML	Unified Modeling Language

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE QUADROS

SUMÁRIO

RESUMO

Instituições religiosas são organizações que reúnem pessoas em torno de crenças e práticas espirituais comuns, desempenhando um papel fundamental na vida social, cultural e moral de suas comunidades. Essas instituições frequentemente dependem de serviços de gestão, administração e comunicação para operar de maneira eficaz e atender às necessidades de seus membros. Este trabalho apresenta o desenvolvimento de um sistema de gestão e comunicação para instituições religiosas, estruturado com base em metodologias exploratória e descritiva. O projeto iniciou-se com uma pesquisa de referenciais teóricos e uma coleta de dados preliminar por meio de questionários aplicados a um público específico, com o intuito de reunir informações fundamentais para a construção dos requisitos do sistema. Após o desenvolvimento e implantação do sistema, aplicou-se um questionário de avaliação utilizando o método USE para medir a usabilidade e satisfação dos usuários, a realização de análises estatísticas fundamentadas nas respostas dos participantes da pesquisa e a criação de uma matriz SWOT detalhada que representa a usabilidade geral do sistema. Os resultados das avaliações indicaram uma média de 95,3% de sucesso, refletindo a satisfação e aprovação dos usuários entrevistados quanto à eficiência e facilidade de uso do sistema. Conclui-se que o sistema atende às necessidades do público-alvo e tem potencial para aprimorar a comunicação e gestão nas instituições religiosas.

Palavras-chave: Django Framework; *Progressive Web App*; Desenvolvimento de *software*; Tecnologia na igreja.

ABSTRACT

Religious institutions are organizations that bring people together around common spiritual beliefs and practices, playing a fundamental role in the social, cultural, and moral life of their communities. These institutions often depend on management, administration, and communication services to operate effectively and meet the needs of their members. This work presents the development of a management and communication system for religious institutions, structured based on exploratory and descriptive methodologies. The project began with a review of theoretical references and a preliminary data collection through questionnaires applied to a specific audience, aiming to gather essential information for the construction of the system's requirements. After the development and implementation of the system, an evaluation questionnaire was applied using the USE method to measure usability and user satisfaction, along with the performance of statistical analyses based on the responses of the survey participants and the creation of a detailed SWOT matrix that represents the overall usability of the system. The evaluation results indicated an average success rate of 95.3%, reflecting the satisfaction and approval of the interviewed users regarding the system's efficiency and ease of use. It is concluded that the system meets the needs of the target audience and has the potential to enhance communication and management within religious institutions.

Keywords: Django Framework; Progressive Web App; Software Development; Technology in the Church.

1 INTRODUÇÃO

A implementação de sistemas para instituições religiosas cristãs, especialmente instituições religiosas, pode ser fundamentada em uma ampla gama de estudos e relatos que demonstram os benefícios do uso das tecnologias no contexto religioso. Segundo o relatório "*State of Church Tech* 2023" da *Pushpay*, a adoção de sistemas de gestão de igreja aumentou significativamente este ano, chegando a 89%. A integração de tecnologias em ministérios religiosos tem sido objeto de estudo e prática, revelando-se uma ferramenta poderosa para fortalecer a comunidade de fé, melhorar a gestão organizacional e expandir o alcance espiritual.

Nesse contexto, o desenvolvimento de soluções de *software* especializadas é essencial para atender às necessidades específicas desse público. Este trabalho visa desenvolver uma aplicação baseada no *framework* Django, utilizando a linguagem Python, para fornecer assistência abrangente a instituições religiosas. O projeto se propõe a criar uma *Progressive Web App* (PWA), permitindo que os usuários instalem a aplicação em seus dispositivos móveis, beneficiando-se de uma experiência similar à de um aplicativo nativo.

O sistema desenvolvido terá como principal objetivo a gestão eficiente de membros, cultos, eventos, reuniões, confraternizações, publicação de horários e programações, entre outras funcionalidades necessárias para o bom funcionamento de uma instituição religiosa. A escolha do Django se deve à sua robustez, segurança e flexibilidade, características fundamentais para o desenvolvimento de aplicações de alta qualidade.

Este trabalho não só busca oferecer uma solução tecnológica para problemas administrativos de instituições religiosas, mas também contribuir para a inclusão digital dessas comunidades, promovendo uma gestão mais organizada e uma comunicação mais eficaz entre seus membros. Assim, espera-se que a aplicação desenvolvida seja uma ferramenta valiosa para pastores, líderes e membros de instituições religiosas, auxiliando na administração e no engajamento da congregação.

Ao longo deste estudo, serão abordados aspectos teóricos e práticos do desenvolvimento de software, detalhando as fases de planejamento, design, implementação e

testes. Além disso, será discutida a importância da adoção de tecnologias modernas como as PWAs para a criação de soluções acessíveis e de fácil uso para um público diverso.

1.1 Problematização

Tradicionalmente, a gestão das instituições religiosas, especialmente as instituições religiosas, tem sido realizada através de métodos manuais, que muitas vezes não são eficientes para lidar com as demandas contemporâneas de comunicação, administração de membros, organização de eventos e gestão financeira. Com a crescente demanda por eficiência, surge a necessidade de um sistema de *software* dedicado que possa otimizar essas funções. De que maneira um sistema de *software* pode otimizar as práticas administrativas das instituições religiosas, como o gerenciamento de membros e eventos?

1.2 Hipóteses

A partir da realização dessa análise, este trabalho apoiará de modo considerável a implementação de um sistema de *software* dedicado, isso poderá melhorar a eficiência administrativa das instituições religiosas, reduzindo o tempo gasto e os recursos gastos em tarefas manuais.

A codificação de um aplicativo cujas características principais sejam segurança, rapidez, intuitividade, simplicidade, possibilidade de personalização, registros de programações, cadastros de instituições e membros, melhor gestão de eventos, contabilização de participantes, divulgação com maior acesso para encontros e facilitação de acesso para temas e discussões publicadas, trará maior visibilidade das programações e maior interação dos membros, alcançando a resolução das principais dificuldades atualmente encontradas.

1.3 Objetivos

1.3.1 Objetivo Geral

Esse trabalho propõe-se a desenvolver e avaliar um sistema web integrado com um *Progressive Web App* (PWA) para instituições religiosas, com funcionalidades que otimizem a gestão de membros, gerenciamento de cultos e eventos, comunicação interna e publicação de horários e programações.

1.3.2 Objetivos Específicos

- Estudar e compreender as tecnologias Django, Python e PWA.
- Realizar um levantamento dos requisitos funcionais e não funcionais do aplicativo, identificando as necessidades específicas das instituições religiosas.
- Criar protótipos de interfaces de usuários levando em consideração os requisitos levantados
- Desenvolver o aplicativo utilizando o *framework* Django, implementando os modelos de dados, *views* e templates necessários para suportar as funcionalidades planejadas e garantindo a segurança e integridade dos dados.
- Integrar os princípios do PWA no desenvolvimento do aplicativo.

1.4 Justificativa

Através das experiências religiosas atravessadas nas rotinas em instituições, há uma necessidade de organizar e tornar acessíveis as informações que precisam ser públicas, questões como administração de doações, agendas de programações, encontros e materiais de ensino, podem ser de conhecimento prévio e de fácil acesso. Esses eventos e programações das instituições podem se perder devido à ausência de membros das divulgações presenciais. Portanto, é essencial um meio de expor essas informações de forma fixa e de fácil acesso.

A implementação de um sistema de software específico para gestão de instituições religiosas pode oferecer numerosos benefícios, como otimização dos processos

administrativos, melhoria da comunicação interna e externa e aumento do engajamento comunitário.

2 METODOLOGIA

A presente pesquisa adota uma abordagem científica hipotético-dedutivo, através de uma revisão bibliográfica incluindo artigos científicos sobre os principais temas incluídos no estudo. A fundamentação teórica baseia-se em literatura acadêmica e técnica sobre HTML, CSS, JavaScript, Django, Python, MySQL, SQL e PWA.

A metodologia adotada possui uma abordagem mista, combinando métodos qualitativos e quantitativos para análise de dados, o que permite uma análise mais completa e detalhada dos fenômenos, combinando objetividade e profundidade descritiva. Essa metodologia é especialmente vantajosa em pesquisas sociais e educacionais, onde fenômenos complexos precisam ser compreendidos de maneira multifacetada (Creswell e Creswell, 2022).

A metodologia utilizada para o desenvolvimento e aplicação da pesquisa foi exploratória e descritiva, ancorada no método de estudo de caso. A abordagem exploratória permitiu investigar as percepções e expectativas dos participantes em relação ao uso de tecnologia nas instituições.

> As pesquisas exploratórias têm como propósito proporcionar maior familiaridade com o problema, com vistas a torná-lo mais explícito ou a construir hipóteses. Seu planejamento tende a ser bastante flexível, pois interessa considerar os mais variados aspectos relativos ao fato ou fenômeno estudado (Gil, 2022, p. 41).

A pesquisa descritiva "têm como objetivo a descrição das características de determinada população ou fenômeno" (Gil, 2022, p. 42). Assim possibilitando uma análise clara dos dados coletados, com foco em aspectos como o uso atual de ferramentas, as funcionalidades desejadas e as opiniões sobre a integralização da tecnologia.

O método de estudo de caso adotado tem como objetivo entender o fenômeno investigado através da realidade e do contexto de um grupo particular. Segundo Gil (2022), o estudo de caso envolve uma análise profunda e exaustiva de um ou poucos casos, possibilitando um conhecimento amplo e detalhado, o que seria praticamente inviável com outros delineamentos previamente considerados.

Foram adotados os seguintes procedimentos de pesquisa:

- **Pesquisa de campo:** Pesquisas serão realizadas diretamente nas instituições religiosas para compreender quais ferramentas e funcionalidades são desejadas para compreender quais ferramentas e funcionalidades são desejadas pelos líderes e membros. Serão coletadas informações sobre quais características eles gostariam que fossem implementadas no sistema e como elas utilizariam essas funcionalidades no dia a dia.
- **Pesquisa bibliográfica:** Uma revisão de literatura foi realizada para analisar informações sobre as tecnologias e metodologias utilizadas no desenvolvimento do sistema.

Para a coleta de dados, serão utilizados:

- **Formulário:** Um formulário será aplicado a líderes e membros das instituições religiosas para coletar dados quantitativos sobre a usabilidade e a funcionalidade do sistema;
- ***Survey*-Questionário:** Um questionário estruturado será aplicado a líderes das instituições religiosas para coletar informações sobre o uso de tecnologia, que utilidades gostariam em uma aplicação e outros aspectos relevantes para a gestão das instituições.

As técnicas de análise de dados empregadas incluem:

- **Qualitativa**: Análise qualitativa do *feedback* dos usuários, avaliando a usabilidade e aceitação do sistema.

- **De conteúdo:** Análise de conteúdo das respostas obtidas nos questionários e formulários, identificando padrões e temas recorrentes.
- **Estatísticas:** Análise quantitativa dos dados coletados, utilizando métodos estatísticos para medir a eficiência do sistema e outros indicadores-chave.

A metodologia para o desenvolvimento do sistema pode ser descrita em uma sequência de etapas principais. Essa metodologia incluir as seguintes etapas:

- **Design e Arquitetura do Sistema:** Planejamento modular em camadas para garantir escalabilidade, desempenho e segurança no desenvolvimento e uso do sistema.
- **Desenvolvimento *Front-end* e *Back-end*:** Criação de interface intuitiva com HTML, CSS e JavaScript e implementação de funcionalidades com Django e MySQL.
- **Implantação do Sistema:** Processo de configuração para disponibilizar a aplicação para uso em produção.

2.1 Questionário na Fase de Pré-Desenvolvimento

Para orientar o desenvolvimento do sistema de gestão para instituições religiosas, foi aplicado um questionário na fase de pré-desenvolvimento. Este instrumento de pesquisa teve o objetivo de explorar as necessidades e expectativas das lideranças e dos membros das instituições em relação ao uso de tecnologias para a gestão e comunicação interna, proporcionando *insights* detalhados para a definição das funcionalidades do sistema.

2.1.1 Estrutura e Organização do Questionário

Para a aplicação do questionário, optou-se por utilizar um modelo de persona específico, composto por indivíduos residentes na cidade de Patos - PB, local onde a pesquisa foi conduzida. A amostra incluiu representantes de instituições religiosas, como pastores, líderes e membros com cargos de gestão nas igrejas. Esses participantes foram selecionados por estarem em posições que permitem uma compreensão abrangente das necessidades

operacionais e administrativas de suas instituições. O questionário foi dividido em seções temáticas, estruturadas para facilitar a coleta e análise das respostas:

- **Percepção sobre Tecnologia nas Instituições Religiosas**: Perguntas que investigam como os respondentes veem a aplicação de tecnologias em contextos religiosos, incluindo seu nível de adesão e percepção de importância.
- **Preferências Funcionais**: Questões que buscam identificar quais funcionalidades específicas são mais desejadas, como a gestão de membros, controle de eventos e cultos, inscrições para atividades, e gestão financeira.

Cada seção foi organizada para fornecer uma visão ampla e detalhada das necessidades institucionais em relação à tecnologia, permitindo identificar as áreas que mais precisam de suporte digital. A seguir, são apresentadas as principais áreas de foco e as perguntas-chave incluídas em cada uma.

2.1.1.1 Percepção sobre tencologia nas instituições religiosas

Esta seção buscou entender como a tecnologia é percebida pelos líderes religiosos em termos de utilidade e importância na gestão da instituição religiosa. As perguntas visavam avaliar a frequência de uso, o impacto percebido e o grau de abertura para integrar novas tecnologias no contexto religioso. As perguntas incluídas foram:

- **Q1.** A sua instituição utiliza atualmente algum *software* ou ferramenta tecnológica para gestão?
- **Q2.** Com que frequência a sua instituição utiliza tecnologia para organizar eventos, cultos ou outras atividades?
- **Q3.** Você acredita que a tecnologia pode ser uma ferramenta útil para a gestão da instituição?
- **Q4.** Quão importante você considera a implementação de um *software* de gestão para a administração do templo?
- **Q5.** Você concorda com a integralização de tecnologia nas atividades diárias da instituição?

Essas questões forneceram um panorama da aceitação e das barreiras à implementação de um sistema de gestão digital.

2.1.1.2 Preferências Funcionais

Para definir os requisitos funcionais do sistema, esta seção do questionário focou nas necessidades práticas da gestão e comunicação eclesiástica. Os respondentes foram convidados a selecionar, dentre várias opções, aquelas funcionalidades que consideravam mais relevantes para o sistema. As funcionalidades sugeridas incluíram:

- **Gestão de membros**: Controle e atualização de dados dos membros da instituição.
- **Gestão de eventos e cultos**: Criação e divulgação de eventos, com registros e acompanhamento de presença.
- **Inscrições para eventos**: Ferramenta de inscrição para atividades, facilitando a organização e planejamento.
- **Gestão financeira e contribuições**: Registro de dízimos e ofertas e organização de informações financeiras.
- **Relatórios de participação**: Relatórios de presença em eventos e atividades, facilitando o acompanhamento de engajamento.
- **Avisos e comunicados**: Canal de comunicação entre líderes e membros.
- **Publicação de materiais**: Compartilhamento de devocionais, estudos e outros conteúdos.

Essas questões direcionaram a estrutura do sistema, permitindo que ele atendesse às demandas específicas da comunidade religiosa e agregasse valor ao dia a dia das instituições.

2.2 Formulário de avaliação e método USE

O presente estudo adotou o Método USE como abordagem para a criação de um questionário direcionado a representantes de instituições religiosas que mantêm contato com a gestão local da igreja. O Método USE, conforme delineado por Lund (2001), é uma ferramenta que visa avaliar aspectos cruciais da usabilidade, englobando quatro dimensões principais: utilidade, facilidade de uso, facilidade de aprendizado e satisfação. Para cada uma

dessas dimensões, foram elaboradas perguntas específicas que visam mensurar a usabilidade do sistema proposto, totalizando doze questões, conforme apresentado no Quadro 1.

Quadro 1 - Indicadores de dimensão do modelo USE.

MÉTRICA	QUESTÃO
Utilidade	O sistema parece útil para a organização das atividades da instituição?
	As funcionalidades principais atendem às necessidades de gestão da instituição?
	O sistema proporciona praticidade no controle e acompanhamento das atividades da instituição?
	Considero o sistema uma ferramenta que agrega valor ao cotidiano da instituição.
Facilidade de Uso	Sinto que seria fácil aprender a utilizar o sistema sem instruções adicionais.
	A navegação entre as diferentes funcionalidades é simples e intuitiva.
	Encontrei as informações e funcionalidades que preciso sem dificuldade.
	A interface do sistema é organizada e permite fácil acesso aos principais recursos.
Satisfação	Estou satisfeito(a) com a aparência visual do sistema.
	Acredito que o sistema seja agradável de usar.
	Sinto-me seguro(a) ao utilizar o sistema em termos de proteção de dados.
	Eu recomendaria este sistema para outras instituições religiosas.

Fonte: Elaborado pela Autora (2024).

Adicionalmente, optou-se pela aplicação de uma escala Likert de cinco pontos para a coleta de dados, onde as respostas dos participantes são categorizadas da seguinte forma: 1 = discordo, 2 = discordo parcialmente, 3 = indeciso, 4 = concordo parcialmente e 5 = concordo totalmente.

2.3 Análise SWOT

A Análise SWOT é uma ferramenta de planejamento estratégico amplamente utilizada em diversas áreas para identificar e avaliar os fatores internos e externos que afetam um projeto, organização ou, como no caso deste trabalho, um sistema de *software*. Segundo a Office Timeline (2023), a análise SWOT se baseia em quatro componentes fundamentais: Forças (*Strengths*), Fraquezas (*Weaknesses*), Oportunidades (*Opportunities*) e Ameaças (*Threats*). As forças e fraquezas são fatores internos que refletem, respectivamente, os aspectos que favorecem e os que limitam o sucesso da aplicação. As oportunidades e ameaças, por sua vez, estão relacionadas ao ambiente externo, representando fatores que podem impulsionar o crescimento ou, ao contrário, comprometer o desenvolvimento sustentável do sistema.

Essa metodologia permite uma visão holística do contexto de desenvolvimento e uso do sistema, facilitando a elaboração de estratégias que maximizem os pontos fortes e oportunidades, enquanto tratam as fraquezas e preparam-se para possíveis ameaças. A aplicação da análise SWOT neste projeto de desenvolvimento de um sistema de gestão para instituições religiosas visa assegurar que o sistema atenda com eficiência às necessidades essenciais, ao mesmo tempo em que se mantenha robusto e adaptável frente aos desafios externos, proporcionando um desenvolvimento mais direcionado e alinhado às expectativas dos usuários.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Adoção de tecnologias em contextos religiosos

A adoção de tecnologias por instituições religiosas tem sido objeto de estudo crescente, destacando tanto as oportunidades quanto os desafios envolvidos. De acordo com a Point Loma Nazarene University (2023), a integração de tecnologias digitais no ministério da igreja pode ajudar na adaptação à era digital, proporcionando ferramentas para melhor comunicação e administração de membros.

A pandemia de COVID-19 acelerou a adoção de tecnologia digitais em ministérios religiosos, forçando muitas instituições religiosas a adotarem transmissões ao vivo de serviços religiosos e outras formas de engajamento *online*. Lemke (2021) argumenta que essa transição digital não apenas permitiu a continuidade dos serviços religiosos durante os períodos de isolamento, mas também abriu novas possibilidades para o engajamento e a expansão das comunidades de fé. Essa rápida adaptação tecnológica pode ter efeitos duradouros na forma como os ministérios operam e interagem com suas congregações no futuro, sugerindo que a tecnologia digital continuará a desempenhar um papel importante na prática religiosa.

Conforme Christian Tech Jobs (2023), sistemas de gestão de instituições religiosas são ferramentas poderosas para centralizar e automatizar diversos processos administrativos. Esses sistemas permitem que as instituições religiosas gerenciem registros de membros, acompanhem a frequência, agendem voluntários, organizem eventos e se comuniquem com a congregação em um único lugar. Essas ferramentas economizam tempo e esforço, permitindo que os líderes religiosos se concentrem em sua missão principal.

A integração de tecnologia na gestão de instituições religiosas não é apenas uma tendência moderna, mas uma necessidade estratégica para atender às demandas de uma comunidade em constante evolução. Portanto, a implementação de um software de gestão para instituições religiosas é uma iniciativa fundamentada em evidências de benefícios significativos tanto para a administração interna quanto para o engajamento e crescimento da comunidade de fé.

3.2 Desenvolvimento Web

O conceito de “Desenvolvimento web” é tido como o processo de criação e manutenção de sites e aplicativos web (Santiago *et al*., 2020). Ele envolve a utilização de linguagens de programação, frameworks e ferramentas para a criação de sites funcionais e interativos. Dentro do desenvolvimento web, há a divisão de responsabilidades tidas como o *front-end* (*client-side*), responsável pela interface que o usuário interage diretamente, e o *back-end* (*server-side*), que lida com o servidor, o gerenciamento do banco de dados, lógica e aplicação.

Inseridos nesses contextos, há a presença de Frameworks, ferramentas muito utilizadas no desenvolvimento web para a facilitação da construção e manutenção de aplicações, por fornecerem uma base estruturada, componentes e bibliotecas de forma eficiente e padronizada. Eles podem estar presentes no desenvolvimento *front-end* ajudando na construção de interfaces dinâmicas e responsivas, ou também no back-end*,* ajudando no gerenciamento de lógica do servidor e integração com o banco de dados. Nas subseções a seguir, há o detalhamento de cada aspecto do desenvolvimento de um sistema web e suas ferramentas.

3.2.1 *Front-End*

A expressão "*frond-end"* diz respeito à interface gráfica do usuário (GUI) com o qual os indivíduos interagem, como botões, ilustrações, barras de navegação, menus e outros elementos visuais. Em termos técnicos, uma página ou tela que o usuário visualiza, contendo diversos componentes de interface, é denominada modelo de objeto de documento (DOM) (AWS, 2024).

As linguagens de programação utilizadas no *front-end* são HTML (linguagem de marcação), CSS (linguagem de estilo) e JavaScript (linguagem de *script*) (Santiago et al., 2020).

3.2.1.1 HTML

O HTML (*HyperText Markup Language*) é uma linguagem de marcação usada para desenvolver e criar páginas web. Essa linguagem utiliza etiquetas ou "*tags*" para estruturar e formatar elementos como texto, imagens e formulários, permitindo que sejam exibidos corretamente pelos navegadores (Longman, 1998). A Figura 1 exemplifica uma utilização do HTML de forma simples, onde <!*DOCTYPE*> é uma instrução para identificar a versão HTML no navegador, a *tag* “<html>” define o ponto inicial e final do documento HTML, “<head>” define o cabeçalho da página e “<title>” determinará o título que é exibido na aba no navegador web. A tag “<body>” engloba todo o conteúdo visível, como por exemplo o

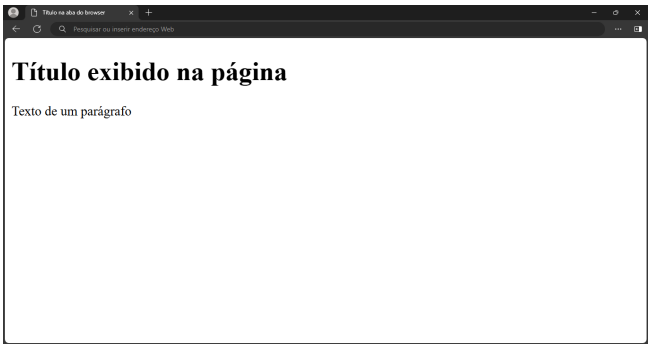
texto inserido dentro de um parágrafo “<p>”. Por fim, na Figura 2, mostra como o browser interpreta o código HTML presente na Figura 1.

Figura 1 - Exemplo de documento HTML

```
<!DOCTYPE html>
<html lang="pt-BR">
    <head>
        <meta charset="UTF-8">
        <title>Título na aba do browser</title>
    </head>
    <body>
        <h1>Título exibido na página</h1>
        <p>Texto de um parágrafo</p>
    </body>
</html>
```

Fonte: Elaborado pela autora (2024).

Figura 2 - Página resultante do código HTML da Figura 1



Fonte: Elaborado pela autora (2024).

3.2.1.2 CSS

CSS (*Cascading Style Sheets*) é uma linguagem de estilo utilizada para descrever a apresentação de documentos HTML. Sua principal funcionalidade é controlar a apresentação do layout de páginas web, como estilo e formatação, incluindo cores, fontes, tamanhos, bordas, posições e uma variedade extensa de personalizações (Santiago *et al*., 2020). De acordo com Silva (2014), há três formas distintas de integrar os estilos numa página web: *inline*, interna e externa:

- **Estilo de forma *inline***: definido dentro dos elementos através do atributo *style* dentro do próprio documento HTML.
- **Estilo de forma interna**: também definido dentro do documento HTML, porém dentro da *tag* <style> modificando os atributos da página por completo.
- **Estilo de forma externa**: é utilizado um arquivo separado do documento HTML, posteriormente identificado por meio de uma hiperligação no <head> do documento HTML. Essa utilização é a mais utilizada pela organização melhor da página pelo fato dos conteúdos estarem em documentos separados.

A sintaxe da linguagem CSS é simples e faz utilização de palavras em inglês para a especificação das propriedades de estilos de uma página. As estilizações podem ser feitas por meio de Identificadores (ID's) e Classes. Importante enfatizar que, a utilização de ID's só é feita para um único elemento por página, já a utilização de classes, pode ser usada em elementos distintos ou várias classes em um único elemento (Silva, 2014).

Na Figura 3, é exemplificado de forma simples o uso de CSS, o símbolo "#" é usado para selecionar elementos pelo ID, enquanto o símbolo "." é usado para selecionar elementos pela sua classe. A propriedade background-color serve para definir a cor do fundo do seletor ".classe", já margin serve para criar um espaço ao redor do elemento. No seletor "#id", a propriedade text-align é usada para o alinhamento do texto, font-size para o tamanho da fonte e color sua cor.

Figura 3 - Exemplo de estilização utilizando a linguagem CSS

```
#id{
    text-align: center;
    font-size: 12px;
    color: black;
}
.classe{
    background-color: red;
    margin: 10px;
}
```

Fonte: Elaborado pela autora (2024).

Silva (2014) destaca que a organização e a estruturação adequada de um ficheiro CSS são fundamentais para tornar o código mais legível. Ele ressalta que a legibilidade do CSS é crucial para a manutenção, melhoria e reformulação do código. Para isso, é importante seguir critérios como comentar o código, normalizar o conteúdo, aproveitar o efeito cascata e as heranças, organizar as propriedades em ordem alfabética, agrupar elementos com características semelhantes e escolher nomes apropriados para seletores, classes e identificadores.

6.2.1.3 JavaScript

O JavaScript é uma linguagem de scripts que suporta múltiplos paradigmas e pode ser interpretada ou compilada em tempo de execução (*Just-in-Time,* JIT). Seu nome oficial é ECMAScript, no entanto, a marca registrada pela Oracle é conhecida como JavaScript (Mozilla, 2019). Uma das principais utilidades do JavaScript é a manipulação do Document Object Model (DOM), o que permite mudar dinamicamente os elementos de uma página web. Com isso, é possível atualizar conteúdos, manipular formulários e interagir com o usuário em tempo real, sem precisar recarregar a página. (Fernando, 2023).

Além do mais, o JavaScript é essencial para validar dados em formulários e na comunicação assíncrona com servidores, através de tecnologías como Ajax (*Asynchronous* JavaScript and XML), ou do mais moderno método, o *Fetch* API.

3.2.2 *Back-End*

O *back-end* realiza o gerenciamento da funcionalidade geral da aplicação *web*. Por meio de uma interação do usuário com o *front-end*, uma solicitação é enviada ao *back-end* em formato HTTP, o *back-end* realiza o processamento da solicitação e faz o retorno de uma resposta (AWS, 2024). Nele também está presente toda a lógica de negócios, integração com o banco de dados, autenticação de usuários e execução de tarefas no servidor. Sendo a parte responsável pelas regras, permissões e gerenciamento de dados da aplicação *web*, essas necessidades de negócio devem ser aplicadas de forma eficiente e segura. Para esse fim, são usadas as linguagens como Java, C#, PHP ou Python.

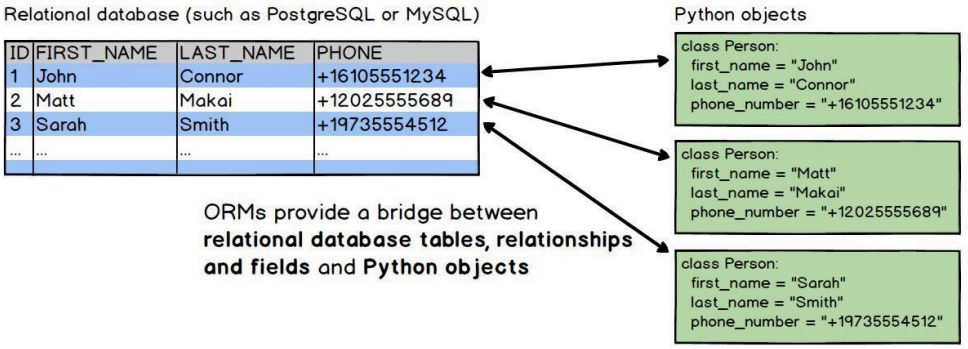
Devido à grande quantidade de responsabilidades no desenvolvimento *back-end*, como segurança, disponibilidade de serviço e desempenho, a utilização de frameworks específicos para o *back-end* pode auxiliar os desenvolvedores na garantia desses aspectos. Um componente essencial frequentemente integrado a esses *frameworks* é o ORM (Object-Relational Mapping), que consiste em uma técnica de representar as tabelas dos banco de dados como objetos (Santiago *et al*., 2020).

O ORM simplifica o acesso e a manipulação de objetos sem a necessidade de se preocupar com os detalhes das suas fontes de dados. Ele permite que os programadores mantenham uma visão consistente dos objetos ao longo do tempo, mesmo que as fontes fornecedoras, os destinos receptores e as aplicações que os acessam mudem (Zimányi *et al*., 2018).

Na ciência da computação, o ORM é uma técnica de programação que converte dados entre sistemas de tipos incompatíveis, utilizando linguagens de programação orientada a objetos. Na prática, isso cria um "banco de dados virtual" que pode ser acessado diretamente a partir da linguagem de programação (Zimányi *et al*., 2018).

A Figura 4 ilustra a representação da conversão da tabela de banco de dados em objetos Python, com o seguinte texto em inglês: "*ORMs provide a bridge between relational database tables, relationships and fields an Python objects*". Tradução nossa: "ORM fornece uma ponte entre as tabelas de banco de dados relacionais, relacionamentos e campos e os objetos Python".

Figura 4 - *Object-Relational Mapping* (ORM)



Fonte: ZIMÁNYI *et al*. (2018).

Frameworks *back-end* também oferecem recursos de segurança integrados, como proteção contra ataques *SQL Injection* e *Cross-Site Request Forgery* (CSRF), um framework *back-end* bastante utilizado em aplicações web atualmente e que oferece esses recursos é o Django, baseado na linguagem Python.

3.2.2.1 Python

Python é uma linguagem de programação conhecida pela sua versatilidade e fácil compreensão, criada na década de 80 por Guido Van Hossum, um desenvolvedor no CWI (Instituto de Pesquisa Nacional para Matemática e Ciência da Computação), com o intuito inicial de ser uma linguagem fácil e intuitiva, pois, segundo Van Hossum, os softwares programados em C eram muito complexos, e somente programadores experientes eram capazes de compreender (Silva e Silva, 2019).

Python também possui uma ampla biblioteca padrão, que inclui extensões para diversas operações, como manipulação de *strings*, expressões regulares e geração de interfaces gráficas, úteis em relação ao desenvolvimento web, como comentado por Sanner(1999), enfatizado por Baliyan *et al* (2022) .

3.2.2.2 Django *Framework*

Django é um *framework web* de código aberto, escrito em Python, criado em 2003 por Adrian Holovaty e Simon Willison no contexto do Lawrence Journal-World, para desenvolver aplicações *web* com mais rapidez e facilidade de manutenção. Sua primeira versão pública foi lançada em 2005 e, desde então, Django se tornou um dos *frameworks* mais utilizados devido ao seu suporte a estruturas de dados reutilizáveis e ao desenvolvimento seguro. (Kaplan-Moss *et al*, 2014).

É considerado um *framework* (MVC), apesar de possuir distinções desse modelo. No padrão MVC, suas responsabilidades são divididas em três camadas, *Model*, *View* e *Controller*, sendo *Model* a parte responsável pelo acesso a dados, *View* a representação visual dos dados, e o *Controller*, responsável pelo tratamento de dados entre o *Model* e a *View*. Já no Django, os desenvolvedores consideram o *framework* como *Model-Template-View* (Monsanto *et al*, 2014)*:*

- **Model** contém tudo o que for relacionado ao banco de dados, como acessá-los, validá-los e a definição de suas relações (Monsanto *et al*, 2014).
- **Template** lida com a apresentação dos dados. No caso de uma API, a "apresentação" dos dados é feita por meio de uma interface com a API usando formatos como JSON, XML ou YAML, tornando a visualização real sob responsabilidade dos clientes que a consomem (Monsanto *et al*, 2014).
- **View** é responsável pela lógica de negócios, sendo responsável pela lógica intermediária entre o *Model* e o *Template* (Monsanto *et al*, 2014).

O Django simplifica a interação com banco de dados relacionais através do Django ORM, que conecta projetos Django a diferentes tipos de banco de dados, traduzindo consultas de forma compreensível para sistemas de gerenciamento de banco de dados, independentemente de suas diferenças estruturais (Duisebekova *et al*., 2021). Assim como dito anteriormente, o ORM foi projetado para se assemelhar ao paradigma de programação orientada a objetos (POO), um exemplo de funcionalidade é o método *get_or_create*, que combina dois cenários em um: se o objeto com os parâmetro especificados não existir, ele cria uma instância com esses parâmetros; caso contrário, ele retorna o objeto.

Essa abordagem simplifica operações complexas do banco de dados, como ilustrado na Figura 5, o Django busca uma instância "Pessoa" com o nome "João Gabriel", caso não encontre, irá criar uma nova instância "Pessoa" com o nome "João Gabriel" e a idade igual a 30.

Figura 5 - Método *get_or_create* do Django

```
Pessoa.objects.get_or_create(
    nome = "João Gabriel",
    defaults={'idade': 30},
)
```

Fonte: Elaborado pela autora (2024).

Além disso, em uma aplicação Django o roteamento de URLs é um aspecto central na sua arquitetura. O arquivo "url.py" é utilizado para mapear URLs para funções de

visualização, como descrito na documentação: “o roteamento de URLs no Django é baseado em expressões regulares que mapeiam padrões URL para views correspondentes” (Django Software Foundation, 2024). Isso significa que é possível configurar URLs dinâmicas, que respondem a diferentes parâmetros e permitem a construção de URLs amigáveis e semanticamente significativas para os usuários finais. Por exemplo, URLs que incluem identificadores de recursos específicos, como “/blog/post/1/”, podem ser configuradas para serem implementadas e manipuladas pelo Django.

O Django também oferece funcionalidades para proteger as aplicações, algumas das principais medidas de segurança incluem proteção contra ataques de injeção SQL, *Cross-Site Scripting* (XSS), Cross-Site Request Forgery (CSRF) e *Clickjacking* (Django Software Foundation, 2024). O mesmo framework também oferece as *Class Based Views* (CBVs), segundo a documentação oficial, “as *class based views* permitem estruturar suas *views* em níveis de abstração e reutilizar funcionalidades comuns, reduzindo a quantidade de código duplicado” (Django Software Foundation, 2024).

As CBVs são implementadas como subclasses de várias classes base fornecidas pelo *framework*, como “View”, “TemplateView”, “ListView”, “DetailView”, entre outras. Cada classe oferece uma funcionalidade específica e métodos que podem ser sobrescritos para personalizar o comportamento da *view* de acordo com as necessidade do projeto (Django Software Foundation, 2024).

Um exemplo prático de implementação é representado na Figura 6, onde há uma definição de uma ListView para exibir uma lista de *posts* de um blog. Nesse exemplo, a “PostListView” herda da “ListView” e configurar atributos como “*model*” (tabela do banco de dados), *“paginate_by*” (quantidade de objetos para paginação) e “*template_name*” (o endereçamento do arquivo HTML da página).

Figura 6 - Exemplo de implementação de uma ListView

```python
from django.views.generic import ListView
from .models import Post

class PostListView(ListView):
    model = Post
    template_name = 'blog/post_list.html'
    paginate_by = 10
```

Fonte: Elaborado pela autora (2024).

3.3 Programação Orientada a Objetos (POO)

O principal objetivo da Programação Orientada a Objetos (POO) é representar fielmente a realidade que está sendo modelada pelos programas computacionais (Feitosa e Comarella, 2020). Essa representação ocorre através de classes, possuindo atributos (ações) e métodos (comportamentos).

Esse paradigma surgiu como uma alternativa a programação estruturada, solucionando alguns problemas e trazendo vantagens como a reutilização, facilidade de manutenção, escalabilidade e compreensão de código (Diniz e Santana, 2010).

Dentre os principais conceitos de POO estão inclusos: Atributos, Métodos, Encapsulamento, Herança, Classe e Polimorfismo (Deitel e Deitel, 2016). Abaixo estão as suas definições:

- **Atributos:** Representam os dados de um objeto, também chamados de variáveis de instância.
- **Métodos:** São as operações que manipulam os dados de um objeto e definem seu comportamento dentro do sistema.
- **Encapsulamento:** Estabelece que os diversos componentes de um sistema de *software* não devem expor detalhes de suas implementações internas, oferecendo liberdade ao programador na implementação desses detalhes.

- **Herança:** Permite a criação de classes mais especializadas a partir de classes mais genéricas, proporcionando uma estrutura hierárquica e modular para a reutilização de código.
- **Classe:** Define os tipos de objetos que compartilham o mesmo conjunto de atributos, comportamentos ou relacionamentos.
- **Polimorfismo:** Refere-se à capacidade de uma variável de objeto assumir diferentes formas, dependendo do contexto de uso.

3.4 Banco de Dados

A definição de banco de dados pode ser tida como uma coleção de dados relacionados entre si, que têm informações sobre algo no mundo real, como lojas, escritórios, bibliotecas ou bancos. A utilização desse conceito em sistemas de *software* permite a automatização de tarefas e agilização de processos. (Marins, 2019).

Para o gerenciamento do banco de dados é utilizado um SGBD (Sistema Gerenciador de Banco de Dados), um *software* que fornece uma interface para a interação com o banco de dados. Através dele, a organização, recuperação e administração de *backups*, *dumps* e outras ferramentas, são facilitadas e permitem ao usuário e aplicativos acessar e manipular os dados sem precisar conhecer os detalhes internos do armazenamento e processamento de dados. A maior parte dos bancos de dados utilizam SQL (*Structured Query Language*) para escrever e consultar dados (Oracle, 2024).

3.4.1 SQL

O SQL (*Structured Query Language*, que em português é Linguagem de Consulta Estruturada), é uma linguagem de programação amplamente utilizada por banco de dados relacionais para consultar, manipular e definir dados (Oracle, 2024). Classificada como o padrão oficial de linguagem em ambiente relacional pelo *American National Standard Institute* (ANSI), atualmente é usada em grande parte nos sistemas de Banco de Dados relacionais, como o SQLServer, PostgreSQL ou o MySQL (Marins, 2019).

3.4.2 MySQL

O MySQL é um SGBD relacional, sendo o mais conhecido em todo o mundo e utilizado em aplicações famosas como o Twitter, Netflix, Facebook, Uber e Airbnb. De acordo com uma pesquisa realizada em 2022 pelo Stack Overflow, o MySQL é o banco de dados mais popular dentre os desenvolvedores.

O banco de dados MySQL é baseado em cliente/servidor, que inclui um servidor SQL *multithread* capaz de suportar diferentes mecanismos de armazenamento, diversas ferramentas de administração, bibliotecas, programas e clientes diferentes, além de uma ampla gama de interfaces de programação de aplicações (APIs). O seu desenvolvimento tinha como objetivo o processamento rápido de banco de dados extensos e é usado em ambientes de produção com altas requisições. Dentre os benefícios da utilização do MySQL como SGBD, está a confiabilidade, alta escalabilidade e facilidade de uso. (Oracle, 2024)

3.5 *Progressive Web Apps* (PWAs)

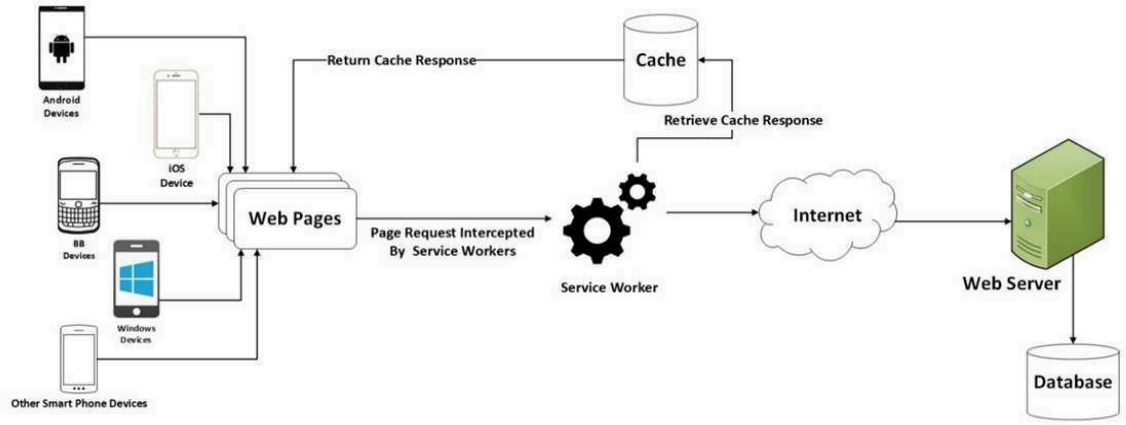
Devido a grande quantidade de aplicativos baixados e a sobrecarga de opções nas lojas de aplicativos, certa fadiga denominada como "*App Fatigue*" foi notada entre os usuários por questões como: preocupações com segurança, complexidade dos *apps* e desempenho nos dispositivos. Um relatório identificou que o número de *downloads* de aplicativos teve uma queda de 143,6 bilhões em 2021 para 142,6 bilhões em 2022 (Olalemi, 2023), essa diminuição no número de downloads de aplicativos evidencia uma mudança perceptível no comportamento dos usuários.

Uma solução eficaz para esse problema é o desenvolvimento de PWAs, que podem trazer a acessibilidade de qualquer dispositivo sem necessidade de *download* ou instalação. As PWAs são aplicações que utilizam tecnologias *web* para oferecer uma experiência semelhante à de aplicativos nativos, combinando acessibilidade em múltiplas plataformas com funcionalidades avançadas. (Mozilla, 2024). O conceito de PWAs surgiu em 2015, apresentado oficialmente pela Google. Os PWAs oferecem experiências de navegação

uniformes em laptops, tablets e outros dispositivos com variação de resolução de tela. Além disso, não é necessário distribuir os PWAs através de marketplaces como App Store, Google Play ou Microsoft Store; a instalação pode ser feita diretamente pelo navegador. (Sharma, 2019).

Na Figura 7 é possível visualizar diagramaticamente a abordagem de desenvolvimento de PWA: O *service worker* intercepta a requisição feita pelos dispositivos móveis nas páginas web do PWA, retornando as respostas utilizando um *cache* que ele gerencia ou acessando diretamente os dados do *Web Server* através da internet.

Figura 7 - Arquitetura de abordagem de desenvolvimento PWA



Fonte: Adetunji *et al*. (2020).

Dentro da estrutura dos PWAs, três componentes principais se destacam:

- ***App Shell*:** Responsável por armazenar conteúdos estáticos da aplicação, como a barra de navegação, a página inicial e outros recursos que permaneçam constantes (HTML, *CSS-Minimal* e JavaScript). Este componente fornece um esqueleto da aplicação quando uma solicitação *offline* é feita, o que ajuda a reduzir o tempo de carregamento das aplicações (Adetunji *et al*., 2020).
- ***Service Workers*:** Oferecem funcionalidades técnicas como sincronização em segundo plano e notificações *push*. Eles operam em uma *thread* separada do navegador, permitindo que o aplicativo receba mensagens e realize atualizações mesmo quando não está ativo. Essa separação também garante que os *Service*

Workers não afetem adversamente a energia dos dispositivos móveis (Adetunji *et al*., 2020).

- ***Web Application Manifest*:** Um arquivo que expõe configurações personalizáveis ao desenvolvedor, como o caminho da imagem do logotipo e o nome do aplicativo. Isto é utilizado para modificar o comportamento e estilo do PWA (Adetunji *et al*., 2020).

A fundamentação teórica deste trabalho apresentou os principais conceitos e tecnologias que serão utilizados no desenvolvimento da aplicação. Esses elementos combinados forneceram a base técnica necessária para criar uma aplicação eficiente e de fácil utilização para o gerenciamento de instituições religiosas.

4 TECNOLOGIAS UTILIZADAS

4.1 Arquitetura do sistema

A arquitetura de *software* refere-se ao processo de definição de uma solução estruturada que atenda aos requisitos técnicos e operacionais, otimizando atributos de qualidade como desempenho, segurança e gerenciabilidade. Envolve decisões que impactam significativamente a qualidade, manutenção e sucesso da aplicação como um todo, considerando diversos fatores e restrições técnicas (CONERY et al., 2009).

A arquitetura de *software* que será utilizada para o desenvolvimento do sistema será monolítica. A arquitetura monolítica é caracterizada por uma única aplicação em camadas que integra a interface do usuário, as regras de negócio e o código de acesso a dados em um único programa e plataforma. Essa abordagem torna a aplicação autônoma e independente de outras aplicações, concentrando todas as suas funcionalidades em uma única unidade de execução (VICERI, 2024).

A arquitetura monolítica oferece benefícios como simplicidade de desenvolvimento e manutenção, além de implantação direta e eficiente, sendo ideal para projetos que necessitam de integração centralizada e menor custo inicial. Após essa abordagem, será discutida a arquitetura em camadas, que organiza a aplicação em módulos distintos, promovendo maior clareza e modularidade ao sistema.

4.1.1 Arquitetura em camadas

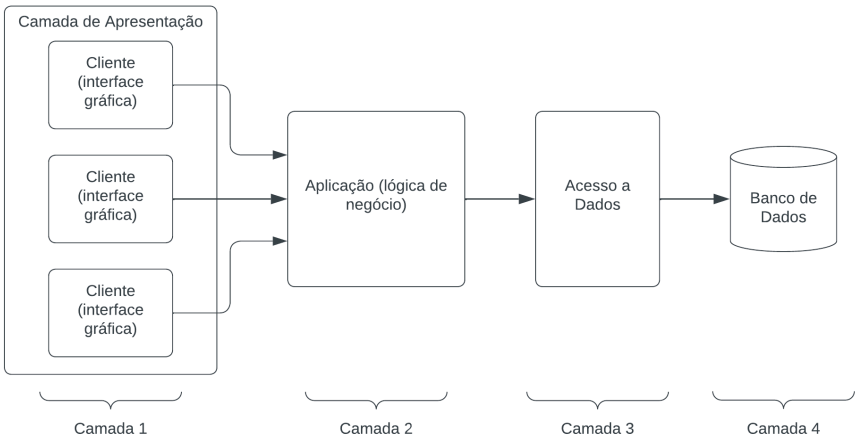
O sistema foi desenvolvido com base na arquitetura em camadas, uma abordagem que visa a organização do *software* em módulos independentes, cada qual com responsabilidades e funções específicas.

A escolha dessa arquitetura se justifica por suas características de modularidade e pela possibilidade de escalabilidade, requisitos essenciais para o sistema, que demanda gerenciamento eficiente de instituições religiosas e múltiplas funcionalidades interligadas. A arquitetura em camadas é um dos padrões arquiteturais amplamente adotado para a organização modular de sistemas, promovendo a separação de responsabilidades em componentes estruturados e hierárquicos. Conforme descrito, “arquiteturas em camadas

particiona a complexidade envolvida no desenvolvimento de um sistema em componentes menores” e disciplinam a interdependências, facilitando a manutenção e expansão do sistema (Valente, 2020).

A arquitetura do sistema foi separada em quatro camadas: Camada de Apresentação, Camada de Aplicação (lógica de negócio), Acesso a Dados e Banco de Dados. A visualização da arquitetura pode ser descrita com o diagrama presente na Figura 8.

Figura 8 - Arquitetura em camadas do sistema



Fonte: Adaptado de VALENTE (2020).

- **Camada de Apresentação:** A camada de apresentação é responsável pela interface visual do sistema, implementada por meio de *templates* HTML, com uso de CSS e JavaScript (com suporte adicional de biblioteca jQuery). Essa camada lida diretamente com a interação do usuário, sendo responsável por apresentar as informações de forma clara e acessível. Através da linguagem de templates do Django, os dados são exibidos de maneira dinâmica, favorecendo uma experiência de uso interativa e responsiva.

- **Camada de Lógica de Negócios:** A camada de lógica de negócios gerencia as operações principais do sistema, controlando o processamento e tratamento das informações. Composta pelas *views*, *models* e *forms* do Django, essa camada implementa as regras de negócio e manipula os dados conforme os requisitos funcionais do sistema, antes de enviá-los à camada de apresentação. A lógica de

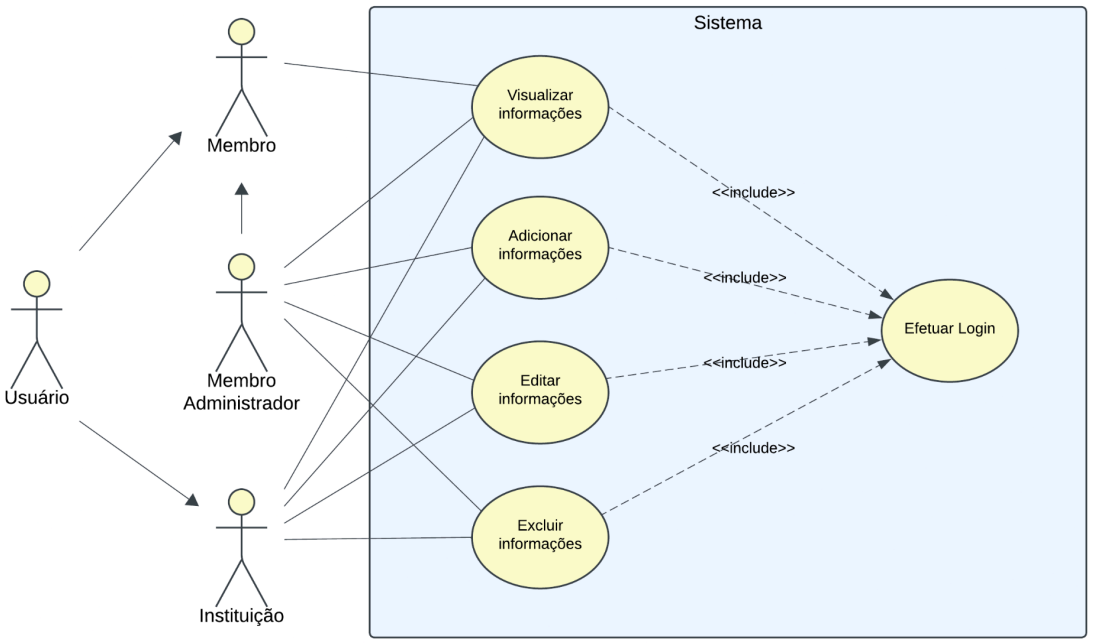
gerenciamento de eventos, contribuições, relatórios e notificações reside nesta camada, garantindo o funcionamento correto e seguro das principais operações.

- **Camada de Acesso a Dados:** A camada de acesso a dados consiste nos modelos do Django, que mapeiam as entidades principais do sistema, como membros, instituições e eventos, ao banco de dados MySQL. Essa camada é responsável pela persistência dos dados, aplicando validações e assegurando a integridade das informações.
- **Camada de Banco de Dados:** O sistema utiliza o banco de dados MySQL para armazenamento das informações, contando com tabelas e relações estruturadas para suportar a grande quantidade de dados exigida pelo sistema. Essa camada também utiliza índices e chaves estrangeiras para garantir a integridade e a eficiência das consultas.

A organização em camadas confere ao sistema uma estrutura modular e escalável, adequada às necessidades de um sistema de gerenciamento e ao crescimento contínuo. A abordagem em camadas permite que novas funcionalidades sejam integradas sem comprometer o funcionamento das demais camadas, reforçando a flexibilidade e a sustentabilidade do sistema no longo prazo.

Para ilustrar a dinâmica de funcionamento do sistema, foi elaborado um diagrama de caso de uso UML (*Unified Modeling Language*, que em português é Linguagem de Modelagem Unificada) que representa as interações entre os atores e os casos de uso principais, o diagrama está presente na Figura 9.

Figura 9 - Diagrama de Caso de Uso UML: Gestão de Usuários e Acessos



Fonte: Elaborado pela autora (2024).

No diagrama, temos quatro atores: Usuário, Membro, Instituição e Membro Administrador. O Usuário se conecta aos casos de uso do Membro e da Instituição, indicando a necessidade de acessar informações e serviços oferecidos por essas entidades.

A Instituição e o Membro Administrador desempenham papéis importantes na gestão de dados, podendo cadastrar, editar e excluir informações. Essas ações são essenciais para garantir que os dados do sistema estejam sempre atualizados. Todos os casos de uso — visualizar, cadastrar, editar e excluir — estão ligados ao caso de uso "Efetuar *login*", que é necessário para qualquer interação com o sistema.

4.2 Escolha das tecnologias

A escolha das tecnologias para o desenvolvimento do sistema foi guiada por fatores como eficiência e escalabilidade e facilidade de manutenção. O *framework* Django, em combinação com Python, foi escolhido para o desenvolvimento do *back-end*, aproveitando

seu padrão *Model-View-Template* (MVT), que facilita a organização do sistema em camadas. No *front-end*, o uso de HTML, CSS e Javascript (suportado pelo Bootstrap e pela biblioteca jQuery) permite uma interface visual responsiva e interativa, proporcionando uma experiência de usuário agradável e intuitiva.

Para otimizar o desenvolvimento e implementar funcionalidades adicionais, foram integradas bibliotecas específicas ao sistema. A biblioteca Django-Allauth foi escolhida para a autenticação de usuários, disponibilizando funções pré-configuradas, como recuperação de senha, envio de e-mail para confirmação de endereço eletrônico e códigos de recuperação, além de suporte à ativação de contas. A biblioteca também oferece "funcionalidades de autenticação social", permitindo a integração de login por meio de redes sociais, como o Google, o que amplia a flexibilidade do sistema (Django Allauth, 2024).

Para melhorar as operações de listagem e pesquisa de dados, a biblioteca Django-Extra-Views foi adotada, oferecendo maior versatilidade nas listagens e módulos de pesquisa, além de suportar *formsets*[1]. Como indicado na documentação oficial, Django-Extra-Views “oferece um conjunto de extensões de CBVs e *formsets*” que amplia as opções de manipulação de dados e proporciona flexibilidade ao trabalhar com consultas dinâmicas (Django Extra Views, 2013).

Para gerenciar as dependências e configurar o ambiente virtual do Python, utilizou-se a ferramenta Poetry, uma biblioteca que automatiza a instalação e atualização dos pacotes necessários, proporcionando um ambiente modular e atualizado conforme as necessidades do projeto (Poetry Team, 2024).

4.3 Implantação e Configuração do Sistema em Ambiente de Produção

Para o processo de implantação do sistema, foram utilizadas tecnologias amplamente reconhecidas pela segurança, escalabilidade e eficiência, incluindo DigitalOcean, Ubuntu,

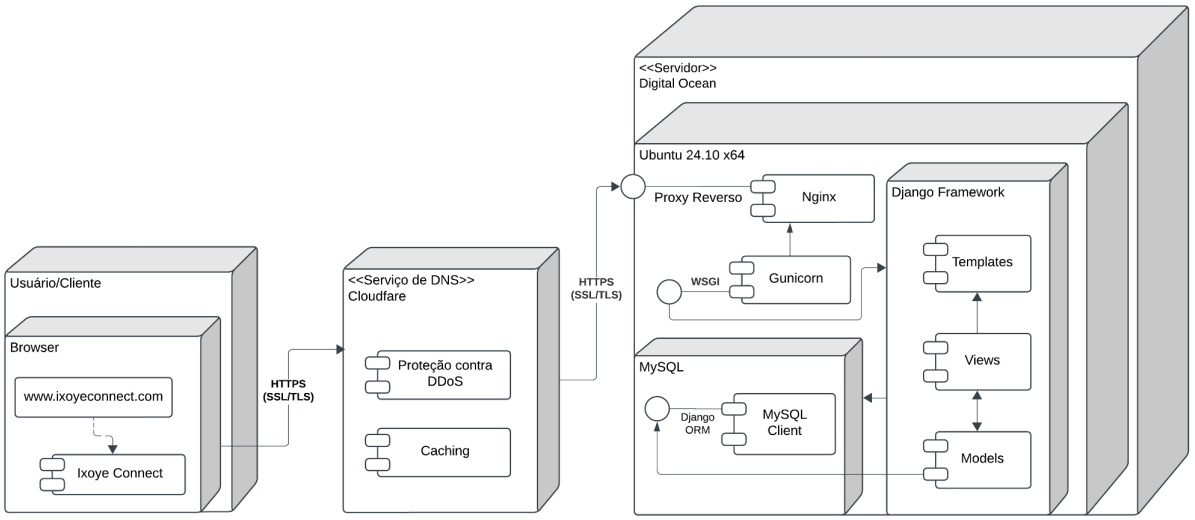
---

[1] Um *formset* é um grupo de formulários que permite a manipulação simultânea de várias instâncias de um mesmo modelo, ideal para cenários em que múltiplos registros precisam ser exibidos e editados de forma agrupada (Django Extra Views, 2013).

Nginx e Gunicorn. Esse conjunto de ferramentas e serviços é particularmente adequado para aplicações *web* que exigem robustez e alto desempenho.

Foi produzido um diagrama de implantação UML que ilustra a arquitetura do sistema em questão. Este diagrama fornece uma representação visual dos componentes do sistema, suas interações e a infraestrutura subjacente conforme apresentado na Figura 10.

Figura 10 - Diagrama de implantação UML do sistema



Fonte: Elaborada pela autora (2024).

A seguir, apresenta-se uma explicação detalhada da implantação do sistema, que descreve os elementos e as relações representadas no diagrama.

A implementação do sistema foi realizada em uma infraestrutura hospedada na Digital Ocean, que oferece uma plataforma confiável e escalável para a hospedagem de aplicações *web*. A configuração ocorreu em uma máquina virtual (*droplet*) utilizando o sistema operacional Ubuntu 24.10, uma distribuição Linux reconhecida pela sua estabilidade e segurança, amplamente utilizada em ambientes de produção (Digitalocean, 2024).

O processo de implantação iniciou-se com a clonagem do repositório hospedado no GitHub, diretamente pelo *console* do servidor Ubuntu. A comunicação entre o banco de dados e a aplicação foi possibilitada através do uso do mysqlclient, que estabelece uma interface

compatível com o Django, assegurando a persistência e integridade dos dados armazenados em MySQL.

Para atender ao volume de requisições e garantir um desempenho eficiente, o servidor foi configurado com o uso do NGINX e do Gunicorn, onde o NGINX atua como servidor *proxy* reverso, gerenciando as conexões de entrada e distribuindo as solicitações para o Gunicorn, um servidor WSGI (*Web Server Gateway Interface*) otimizado para aplicações Django, proporcionando escalabilidade e estabilidade ao sistema (Solomon, 2024).

A segurança no acesso remoto ao servidor foi garantida pelo uso do protocolo SSH (*Secure Shell*), que oferece uma conexão segura e criptografada entre o dispositivo local e o *Droplet*; tal recomendação é corroborada pela documentação oficial da Digital Ocean, que destaca o uso do SSH como essencial para proteger a comunicação com o servidor (Digitalocean, 2020). No terminal do servidor Ubuntu, foi gerada uma chave SSH local para permitir a clonagem segura do repositório no GitHub, seguindo as boas práticas para o desenvolvimento seguro (Github, 2024).

Para proporcionar uma conexão segura aos usuários, foi configurado um domínio personalizado com a plataforma Cloudflare. O Cloudflare é uma plataforma de serviços de internet que oferece soluções de segurança, desempenho e confiabilidade para websites e aplicações online. Sua principal função é atuar como um intermediário entre os usuários e os servidores de origem, proporcionando proteção contra ataques como DDoS, além de otimizar a entrega de conteúdo por meio de uma rede de distribuição global (Servcloud, 2023). Essa intermediação não apenas melhora a velocidade de carregamento das páginas, mas também garante maior disponibilidade e segurança para os sites que utilizam seus serviços.

Através do Cloudflare foi feito o gerenciamento DNS para associar o domínio do sistema ao IP do servidor, enquanto o Cloudflare aplicou uma criptografia SSL/TLS, que estabelece uma comunicação criptografada entre a Cloudflare e o cliente, assegurando que os dados trocados permaneçam protegidos contra interceptação ou adulterações.

A implementação do certificado SSL foi realizada com o *Let's Encrypt*, uma autoridade certificadora gratuita e aberta que fornece certificados digitais para sites. Para a configuração do certificado SSL, foi utilizada a ferramenta Certbot, que gerou

automaticamente o certificado e configurou o servidor Nginx para redirecionar as requisições HTTP para HTTPS e assegurando que todos os dados trafegados entre a Cloudflare e o servidor estejam protegidos por criptografia aplicando uma camada de segurança (Let's Encrypt, 2023). Assim promovendo uma comunicação ponta-a-ponta totalmente criptografada e melhorando a proteção do sistema contra ataques.

A aplicação do HTTPS foi fundamental para garantir o funcionamento do sistema como um PWA, pois a camada de segurança SSL é um requisito para a instalação de PWAs em dispositivos móveis e para o funcionamento de notificações *push*, promovendo uma experiência segura, contínua e responsiva ao usuário.

4.4 Gerenciamento de Dados com MySQL e Django *Migrations*

Para o gerenciamento de dados do sistema em ambiente de produção, foi escolhido o MySQL. No servidor, o MySQL foi configurado através do pacote mysql-server, que permite a criação e gerenciamento seguro do banco de dados. O mysql-server é uma aplicação robusta para a gestão de dados em servidores, com funcionalidades que garantem integridade e eficiência, além de suportar grandes volumes de dados e possibilitar uma configuração adaptável às necessidades do sistema (Ubuntu, 2024).

Após a instalação do MySQL no servidor, a conexão com o Django foi realizada por meio da configuração do arquivo "*settings.py*", onde os parâmetros de conexão — como nome do banco de dados, usuário, senha e *host* — foram especificados, estabelecendo uma comunicação entre o *framework* e o banco de dados. No Django, o gerenciamento do banco de dados é realizado com o auxílio das Django Migrations, uma ferramenta que permite a criação, modificação e exclusão de tabelas de forma programática. Esse sistema de migrações facilita o controle de alterações estruturais no banco de dados, mantendo um histórico de cada mudança. Segundo a documentação do Django, "as migrações são uma maneira de alterar o banco de dados de forma gradual e organizada" (Django Software Foundation, 2024) , o que garante consistência nos dados ao longo do ciclo de vida do projeto.

O uso combinado de MySQL e Django Migrations permite que o sistema realize operações complexas no banco de dados, como atualizações e consultas, de forma segura e eficiente. Enquanto o MySQL server oferece a base de dados estruturada e altamente performática, o Django Migrations permite gerenciar a estrutura desse banco diretamente a partir do código do projeto, garantindo flexibilidade e facilidade na gestão dos dados.

5 RESULTADOS E DISCUSSÕES

A implementação do sistema foi realizada com base nos levantamentos feitos por meio das pesquisas e reuniões realizadas. Essas etapas permitiram identificar os requisitos básicos e fundamentais e alinhar as funcionalidades da aplicação às necessidades específicas dos usuários. Essa seção apresenta o funcionamento do sistema de maneira detalhada.

5.1 Resultados do Questionário na Fase de Pré-Desenvolvimento

Neste capítulo, serão apresentados e discutidos os resultados obtidos a partir do questionário aplicado durante a fase de pré-desenvolvimento do sistema. Os dados foram coletados com o objetivo de identificar as percepções e preferências dos respondentes quanto ao uso de tecnologias para a gestão de instituições religiosas, bem como especificar as funcionalidades mais demandadas.

As informações extraídas desse levantamento foram de extrema importância para nortear as decisões de desenvolvimento, permitindo que o sistema se alinhasse às necessidades do público-alvo.

5.1.1 Percepção sobre Tecnologia nas instituições religiosas

Os dados coletados sobre a percepção de tecnologia nas instituições religiosas indicaram um grau variado de adesão e aceitação, refletindo a diversidade de práticas e visões sobre o papel da tecnologia na administração e organização de atividades religiosas.

- **(Q1). Utilização Atual de Software para Gestão**: Dos respondentes, 60% afirmaram que já utilizam algum tipo de *software* para a gestão da instituição,

enquanto 40% indicaram não possuir nenhuma ferramenta digital. Esses resultados evidenciam uma demanda potencial para a introdução de uma plataforma de gestão integrada para as instituições religiosas que ainda não utilizam tecnologia, enquanto as que já utilizam poderiam se beneficiar de uma solução mais personalizada e específica para o contexto religioso.

- **(Q2). Frequência de Uso de Tecnologia para Organização de Eventos**: A frequência com que a tecnologia é utilizada para a organização de eventos, cultos e outras atividades também foi investigada. Observou-se que 40% dos respondentes utilizam ferramentas digitais diariamente para essas finalidades, enquanto outros 40% afirmaram utilizá-las semanalmente. Esse dado reforça a necessidade de uma funcionalidade de fácil acesso e uso para organizar e divulgar eventos, uma vez que a maioria dos líderes já recorre à tecnologia com regularidade.
- **(Q3) e (Q4). Importância e Aceitação da Tecnologia como Ferramenta para Gestão**: Uma maioria expressiva (100%) dos líderes considera a tecnologia uma ferramenta útil para a gestão, e 80% dos respondentes classificaram a implementação de um *software* de gestão como essencial para a administração do templo. Esses resultados apontam para uma aceitação ampla da ideia de integração tecnológica, que foi uma motivação fundamental para a estruturação do sistema proposto.

5.1.2 Preferências Funcionais para o Sistema

As respostas relacionadas às preferências funcionais permitiram identificar quais módulos e funcionalidades o sistema deveria priorizar. Cada funcionalidade foi avaliada em termos de sua importância percebida para o cotidiano eclesiástico, fornecendo uma base sólida para os requisitos funcionais do sistema.

- **Gestão de Membros**: A funcionalidade de gestão de membros foi considerada essencial por 85% dos respondentes, refletindo a importância de manter um cadastro atualizado e acessível dos membros da instituição. Este recurso foi desenvolvido para permitir o acompanhamento de informações de contato, frequência e engajamento dos membros.

- **Gestão de Eventos e Cultos**: A gestão de eventos, incluindo a criação, divulgação e acompanhamento de cultos e atividades, foi apontada como uma necessidade importante para 70% dos respondentes. Esse recurso ajuda a organização de eventos ao permitir o registro de inscrições e de presença, facilitando o planejamento e o engajamento da comunidade.
- **Inscrições e Gestão Financeira**: 75% dos respondentes destacaram a importância de uma ferramenta para inscrições em eventos e a gestão financeira de contribuições como dízimos e ofertas. Esse módulo fornece um controle financeiro mais preciso e organizado, adaptado às necessidades específicas das instituições.
- **Relatório de Participação**: Apesar de ser uma funcionalidade considerada útil, apenas 60% dos respondentes a classificaram como essencial. Essa resposta indica que, embora importante, o relatório de participação pode não ser uma prioridade para todas as instituições religiosas, mas atende especialmente aquelas que desejam acompanhar de forma detalhada o engajamento dos membros.
- **Avisos e Comunicados:** Esta funcionalidade recebeu alta prioridade, sendo destacada por 90% dos respondentes como essencial para manter os membros informados. A comunicação eficiente entre líderes e congregação é um fator crucial, o que reforça a importância desse recurso para a disseminação de avisos importantes e atualizações.
- **Publicação de Materiais**: Também com 60% de respostas positivas, a funcionalidade de publicação de materiais permite o compartilhamento de conteúdos educativos e espirituais, como devocionais, estudos bíblicos e mensagens pastorais. Este recurso é relevante para instituições religiosas que visam promover o crescimento espiritual contínuo dos membros fora das atividades presenciais.

Com base nos dados coletados no questionário realizado na fase pré-desenvolvimento e nas preferências de funcionalidades indicadas pelos usuários, foram levantados os requisitos funcionais e não funcionais necessários para o sistema. A Tabela 1 a seguir apresenta os requisitos funcionais (RF), que refletem as funcionalidades essenciais que o sistema deve oferecer.

Tabela 1 - Requisitos Funcionais do Sistema.

ID	**Requisito Funcional**	**Descrição**
RF01	Cadastro de Membros	O sistema deve permitir o cadastro de membros, incluindo dados como nome, data de nascimento, e código único da igreja.
RF02	Gerenciamento de Eventos	O sistema deve permitir a criação, visualização, e edição de eventos, incluindo data, hora, título, descrição, e localização.
RF03	Notificações *Push*	O sistema deve enviar notificações push para membros sobre eventos e atividades importantes, incluindo a possibilidade de configurar preferências.
RF04	Gestão de Contribuições	O sistema deve permitir o cadastro de métodos de contribuições financeiras, incluindo tipos como dízimo, oferta, doações, e campanhas.
RF05	Relatórios de Atividades	O sistema deve gerar relatórios sobre a frequência de membros, contribuições, e eventos passados e futuros.
RF06	Cadastro de Escala de Obreiros	O sistema deve permitir a criação de escalas de obreiros para eventos e cultos, com a possibilidade de edição e visualização.
RF07	Acesso e Segurança	O sistema deve garantir que o acesso às informações seja controlado, com diferentes níveis de permissão para administradores e membros.

Fonte: Elaborado pela autora (2024).

Em seguida, a Tabela 2 de requisitos não funcionais (RNF) define as características adicionais do sistema, como desempenho, segurança, usabilidade e escalabilidade.

Tabela 2 - Requisitos Não Funcionais do Sistema.

ID	**Requisito Não Funcional**	**Descrição**
RNF01	Design Responsivo	O sistema deve se adaptar a diferentes tamanhos de tela e dispositivos (desktop, tablets, smartphones), garantindo uma experiência de usuário consistente.
RNF02	Consistência Visual	O design visual deve ser consistente em todas as telas, com paleta de cores, tipografia e elementos

		gráficos padronizados, criando uma identidade visual coesa.
RNF03	Navegação Intuitiva	O sistema deve oferecer uma navegação clara e simples, com menus e botões de fácil acesso, permitindo que os usuários encontrem rapidamente as funcionalidades principais.
RNF04	Customização do Usuário	O sistema deve permitir personalização básica da interface, como modo escuro, adaptando-se às preferências do usuário.

Fonte: Elaborado pela autora (2024).

9.1.3 Discussão dos resultados

A análise dos resultados do questionário indica uma necessidade clara e crescente de integração tecnológica nas instituições religiosas, com um foco específico em áreas que facilitam a comunicação, a gestão de membros e o controle de eventos. Os dados coletados confirmam a relevância das funcionalidades projetadas, evidenciando uma forte demanda por um sistema que centralize as atividades de gestão e comunicação em um único espaço digital.

Com base nesses resultados, o sistema Ixoye Connect foi projetado para priorizar as funcionalidades mais demandadas, garantindo que a plataforma atenda diretamente às necessidades dos usuários finais. A metodologia exploratória permitiu captar uma variedade de preferências e práticas, proporcionando um entendimento abrangente que sustentou a tomada de decisões no desenvolvimento da aplicação.

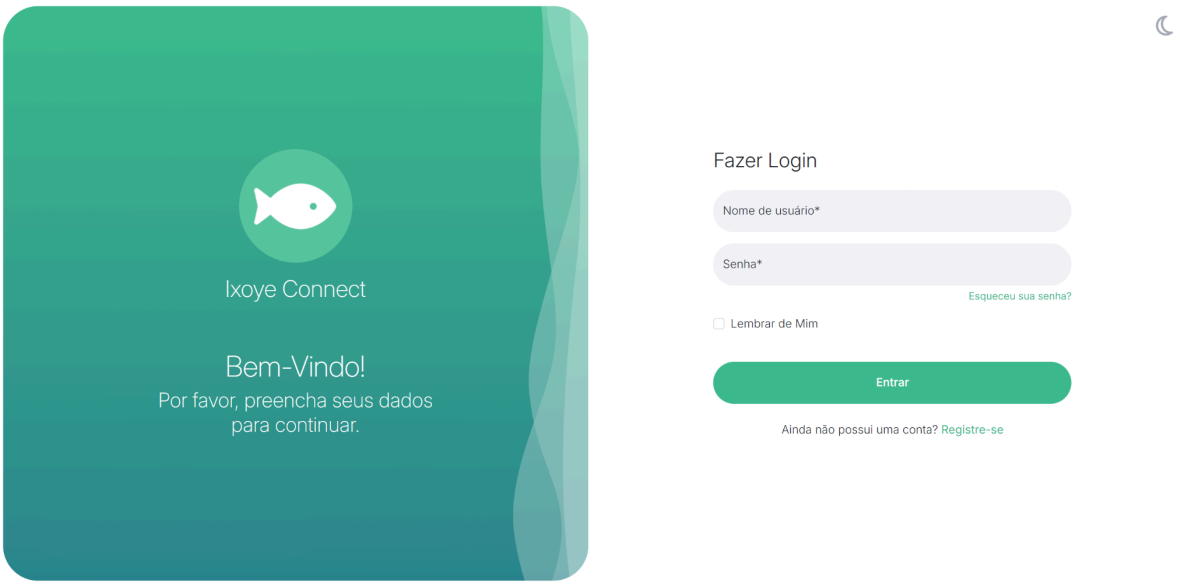
5.2 Visão geral dos resultados da aplicação

O sistema conta com oito módulos principais: agenda semanal, contribuição, escala de obreiros, eventos, notificações, conteúdos, relatórios e usuário. Todas as telas são responsivas, otimizadas para visualização no formato PWA em dispositivos móveis.

A primeira tela do sistema é a de *login*, que contém os campos essenciais para a autenticação do usuário: nome de usuário e senha. Essa tela apresenta ainda as opções de cadastro e redefinição de senha, que permitem tanto o registro de novos usuários quanto a

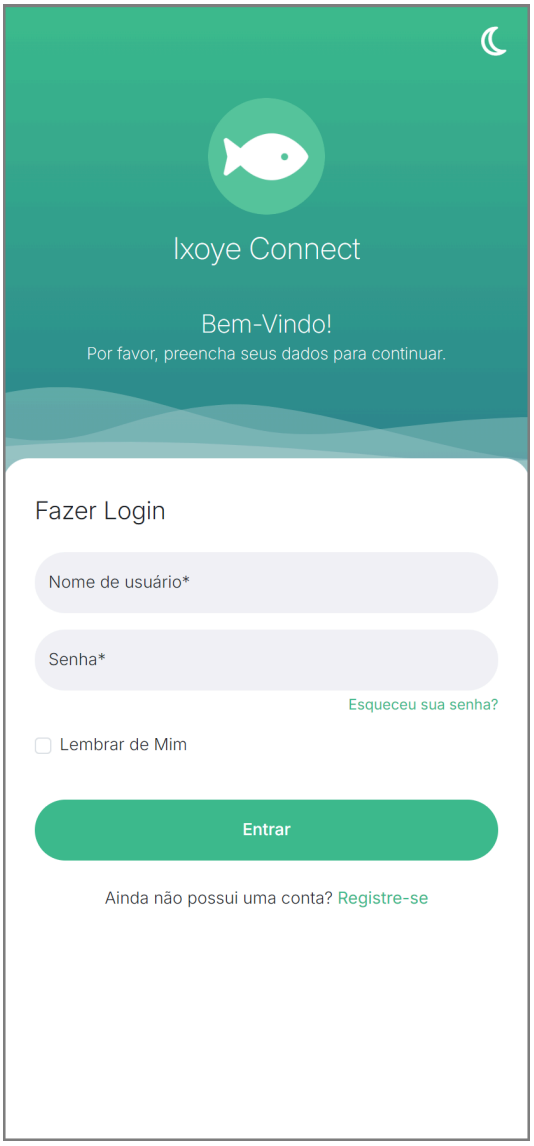
recuperação de acesso para aqueles que esqueceram suas credenciais. A interface é projetada para ser intuitiva e acessível, com uma disposição clara dos elementos e instruções de preenchimento. Essas opções iniciais de acesso estão ilustradas na Figura 11 e 12.

Figura 11 - Tela de Autenticação (*Login*)



Fonte: Elaborado pela autora (2024).

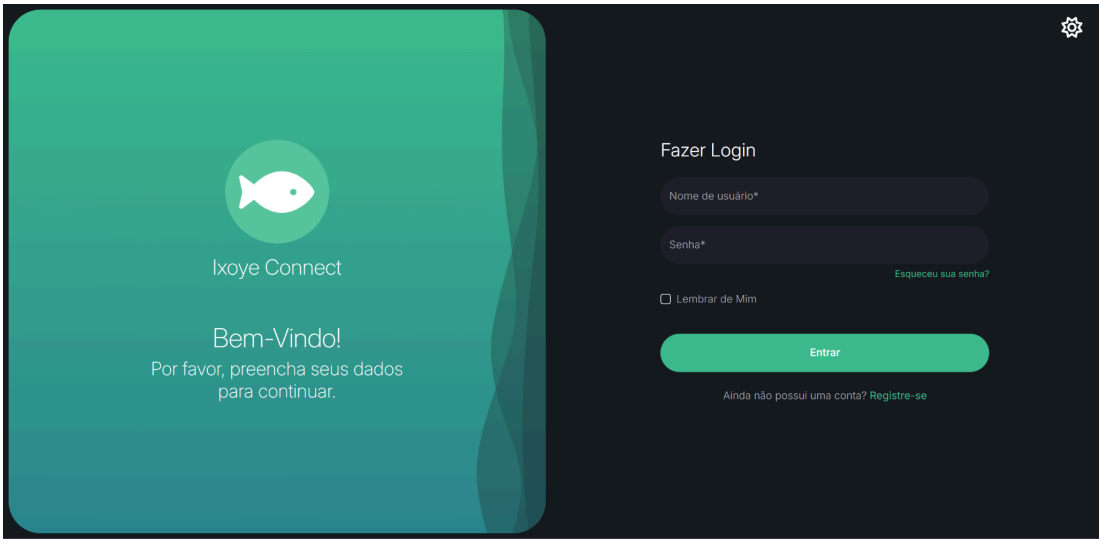
Figura 12 - Tela de Autenticação modo *mobil*e



Fonte: Elaborado pela autora (2024).

É importante destacar o ícone de lua presente na interface, que funciona como um botão para ativar o modo escuro. Ao ser clicado, o sistema ajusta as cores claras para tons escuros. Essa funcionalidade contribui para a acessibilidade e a economia de energia do dispositivo do usuário e é possível ver um exemplo dessa funcionalidade ativa na Figura 13. A opção está disponível em todo o sistema.

Figura 13 - Tela de Autenticação (*Login*) no Modo Escuro

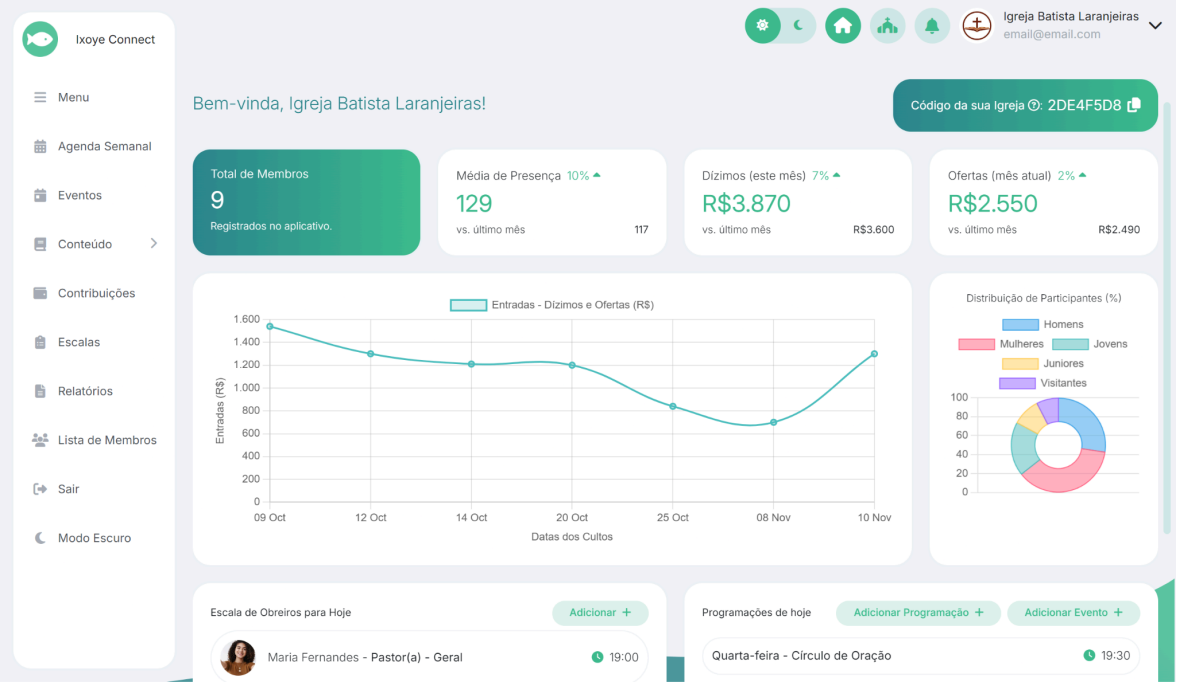


Fonte: Elaborado pela autora (2024).

Na tela de cadastro, o usuário pode se registrar como membro ou como instituição. O cadastro de instituição permite registrar uma sede e vinculá-la a uma instituição principal. Neste caso, o cadastro inclui campos específicos, como CNPJ e sigla, e a conta será vinculada à sede e cadastrada. Já o cadastro de membros inclui campos como ano de ingresso, data de nascimento e o código único da instituição, gerado automaticamente no cadastro da sede. Os demais campos, como autenticação e dados de endereço, são comuns a ambos os tipos de cadastro.

Após o login, o sistema apresenta uma tela inicial que contém um painel de controle centralizado para acesso rápido às principais funcionalidades. À esquerda, uma barra lateral fixa (*sidenav*) oferece *links* diretos para os módulos do sistema, facilitando a navegação do usuário entre as seções disponíveis. Na parte superior, o sistema conta com opções de navegação que incluem o botão de ativação do modo noturno, o ícone de página inicial, perfis tanto da instituição quanto do usuário, notificações e informações do usuário logado, organizadas para facilitar o acesso e a usabilidade. Essas informações estão presentes na Figura 14.

Figura 14 - Painel de Controle ou *Dashboard* no modo *Desktop*



Fonte: Elaborada pela autora (2024).

Na versão *mobile*, a interface foi adaptada para proporcionar uma navegação mais confortável em telas menores. A barra lateral foi substituída por um menu *hamburguer*, que economiza espaço e permite que o usuário acesse as opções do *sidenav* apenas quando necessário, ela fica localizada no cabeçalho da página, onde também está presente o título da página atual centralizado na tela.

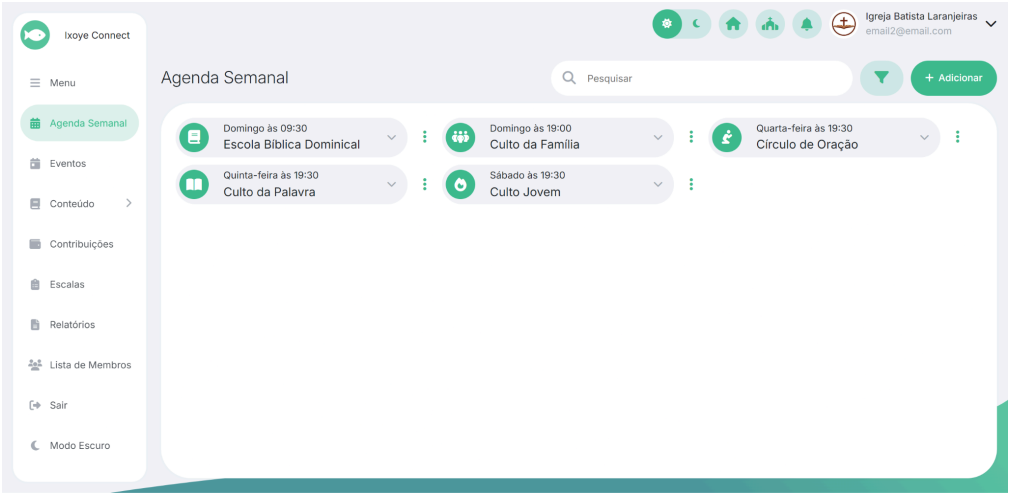
Além disso, as opções de navegação do topo foram reorganizadas em uma barra de navegação inferior (*bottom nav*) flutuante. Tanto o cabeçalho quanto a barra de navegação inferior sempre estarão fixos na tela, mantendo a experiência de uso intuitiva e funcional. Essas adaptações, que podem ser visualizadas nas Figuras 15, garantem que o sistema seja acessível e responsivo em diferentes dispositivos.

Figura 15 - Painel de controle no modo *mobile*

Fonte: Elaborada pela autora (2024).

No menu lateral, o usuário pode visualizar os módulos principais do sistema, começando pela Agenda Semanal. Ao clicar neste módulo, a página é redirecionada para a listagem das programações, onde cada agenda é apresentada em forma de *card*, incluindo ícone representativo, dia da semana, horário e título. É possível expandir cada *card* para visualizar uma descrição detalhada da programação. O padrão de listagem inclui campos de pesquisa e botões de filtro e de adição de novos registros, como demonstrado na Figura 16.

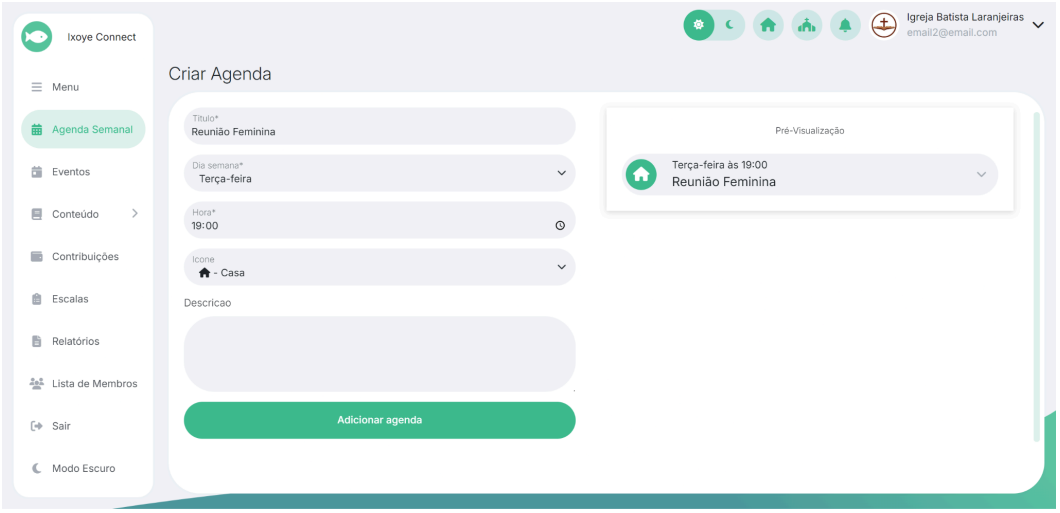
Figura 16 - Listagem de Agenda Semanal



Fonte: Elaborada pela autora (2024).

Ao clicar em “Adicionar”, o usuário é levado para a tela de cadastro da Agenda Semanal. Esta tela segue o padrão de organização de formulários, com os campos de entrada à esquerda e uma pré-visualização dinâmica à direita, que exibe os dados preenchidos em tempo real usando JavaScript. Essa disposição permite que o usuário visualize o conteúdo enquanto o cria, conforme mostrado na Figura 17. Esse padrão de organização de formulários está incluso nos módulos: Agenda Semanal, Eventos, Conteúdos e Contribuições.

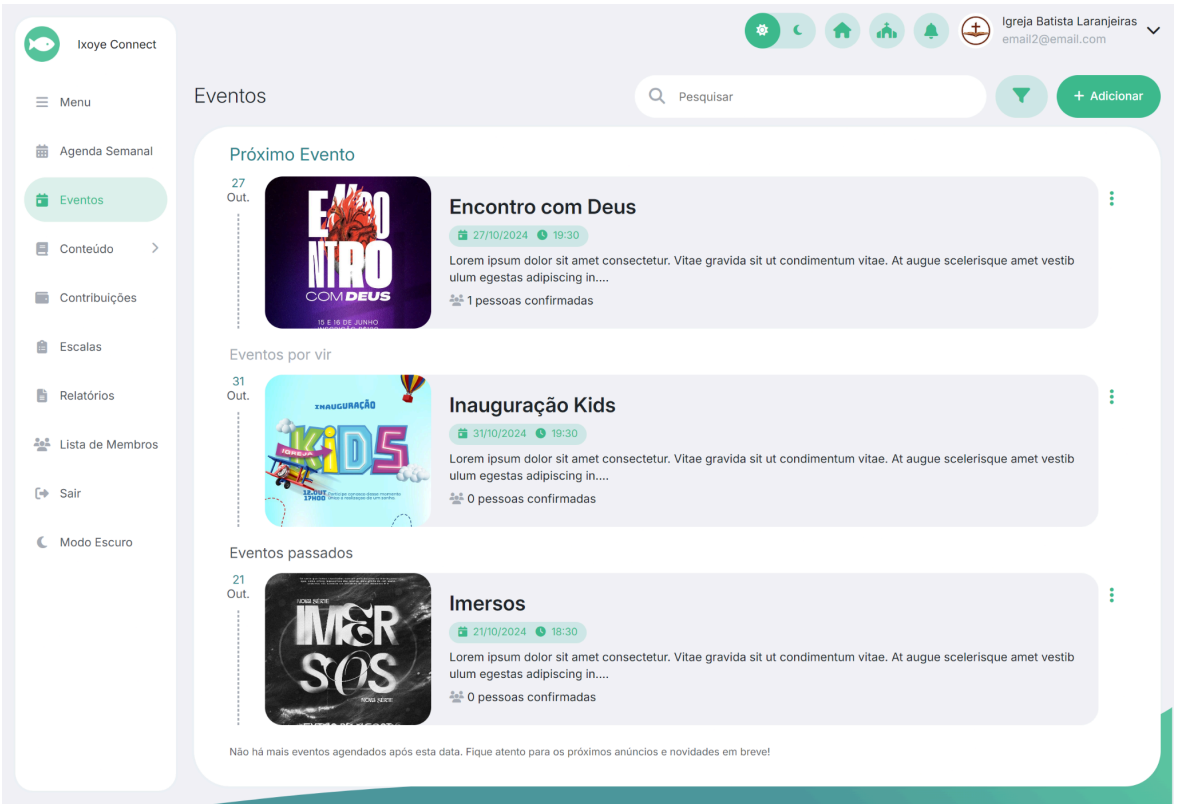
Figura 17 - Cadastro de Agenda Semanal



Fonte: Elaborada pela autora (2024).

Seguindo para o módulo de Eventos, a tela segue o padrão citado anteriormente para telas de listagem. Os *cards* de Eventos possuem as informações mais importantes de serem visualizadas na tela de listagem, que seriam: título, data, hora, uma breve parte da descrição e a quantidade de pessoas confirmadas. A listagem de Eventos é separada em três partes: Próximo Evento (que possuirá o evento futuro mais próximo da data atual), Eventos por vir (os eventos futuros após o próximo evento), e Eventos passados (eventos anteriores a data atual), isso é mostrado na Figura 18.

Figura 18 - Listagem de eventos



Fonte: Elaborada pela autora (2024).

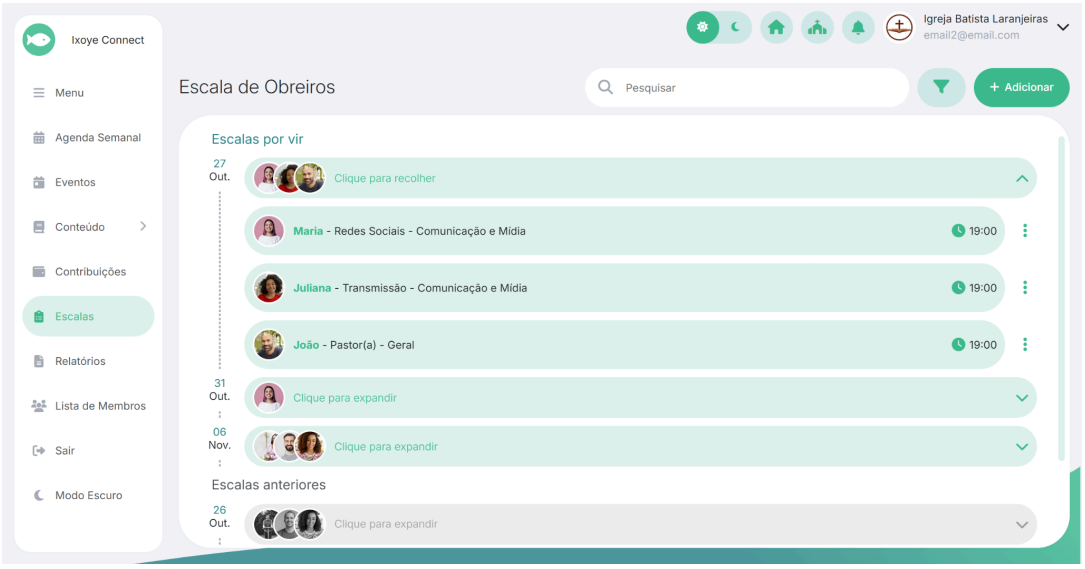
Ao clicar em “Adicionar”, o usuário será direcionado para a tela de cadastro de Eventos, seguindo o padrão de tela de cadastros citado anteriormente. No formulário possui as seguintes informações: capa, título, descrição, valor, data, hora. Também há por final duas opções para o endereço do evento, nos quais são: relacionar ao mesmo endereço cadastrado na sede ou preencher o formulário de um endereço novo.

O módulo de Conteúdos permite a publicação de diversos tipos de conteúdo, organizados em 10 categorias, como Devocional, Estudo Bíblico, Aviso e Notícias da Igreja. A separação dos conteúdos em categorias facilita o acesso ao material de interesse do usuário e contribui para a organização das publicações da instituição. Para cada conteúdo, o usuário pode adicionar uma capa opcional, título, descrição e anexar arquivos.

O módulo de Contribuições apresenta diferentes métodos de contribuição, incluindo Pix e Depósito, que podem ser vinculados a departamentos específicos da instituição. O sistema também permite configurar tipos específicos de contribuição, como oferta, dízimo e campanhas de caridade, conforme necessário. A personalização dos dados de pagamento permite uma gestão detalhada e alinhada às necessidades da instituição.

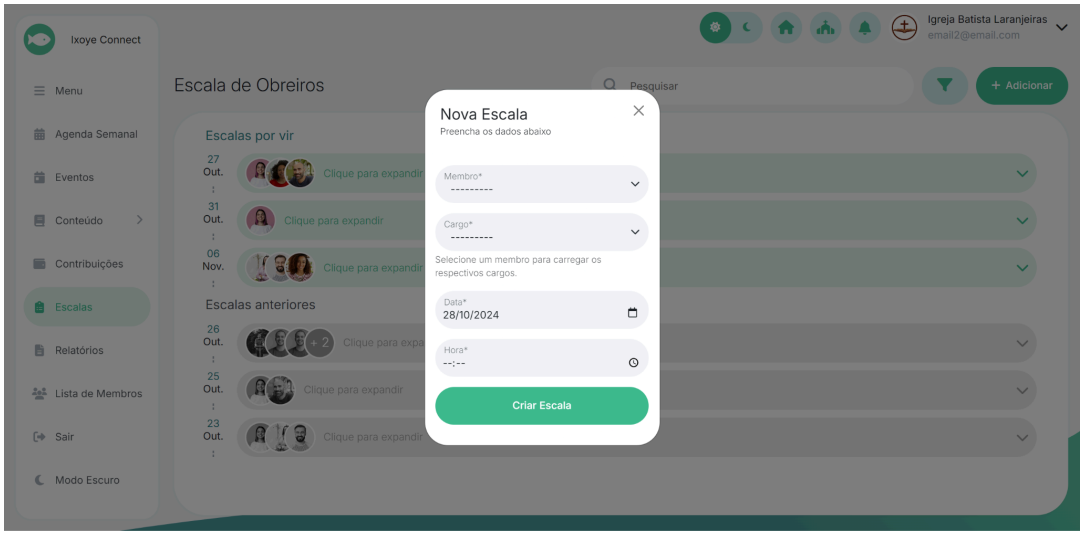
A Escala de Obreiros é organizada em botões de *accordion*, que permitem agrupar as escalas por data. Ao expandir um botão de data específica, o conteúdo exibido inclui os detalhes das escalas de obreiros para aquele dia, como nome do membro, função e horário. O cadastro de novas escalas é feito por meio de um modal, o que facilita a inserção de dados e permite uma interação rápida, sem a necessidade de uma tela de cadastro exclusiva. As Figuras 19 e 20 ilustram esses aspectos.

Figura 19 - Listagem de Escala de Obreiros



Fonte: Elaborada pela autora (2024).

Figura 20 - Modal de cadastro de Escala de Obreiros



Fonte: Elaborada pela autora (2024).

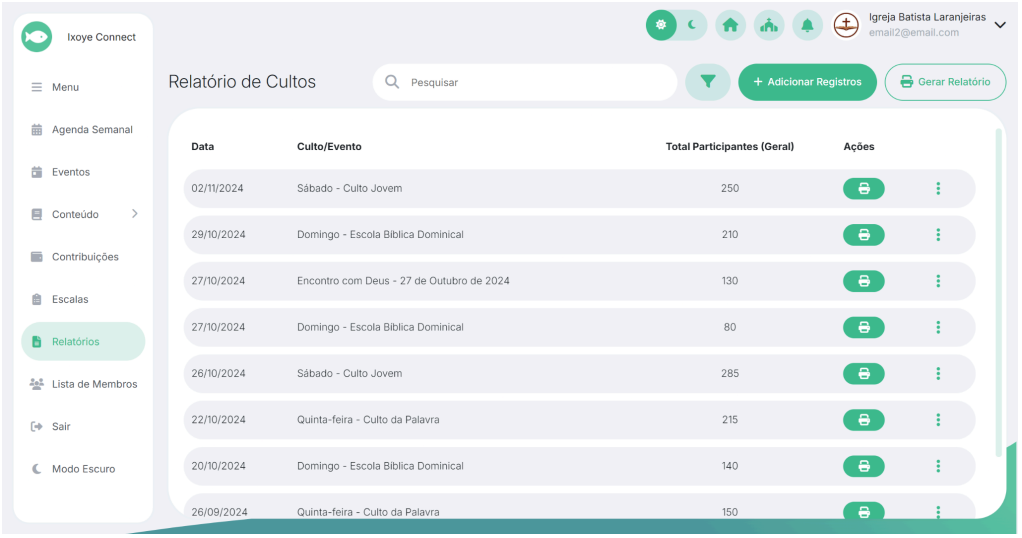
A parte de relatórios inclui uma quantidade de dados mais extensa que os demais módulos do sistema. Ele serve para fazer o controle de dados da instituição, na produção de relatórios específicos, gerais ou por período. Além disso, também serve para a comparação com o mês anterior, como entradas financeiras (dízimos e ofertas) e também a média de presença de membros por culto, isso é exibido no Painel de Controle.

A listagem é gerada a partir de uma tabela que inclui a data, o culto ou evento, o total de participantes e um botão para impressão, que permite gerar um relatório específico para aquela data. No cadastro de relatórios de cultos, é possível inserir as seguintes informações: data, horário de início e término, mensagem ou tema, e o nome do ministro (que pode ser selecionado entre os já cadastrados no sistema ou inserido manualmente em um campo de texto). Além disso, é necessário registrar o total de participantes, discriminando entre o público geral, mulheres, homens, jovens, juniores e visitantes. Também é possível incluir a programação do dia (que deve ser inserida na agenda semanal) ou eventos (cadastrados na seção de eventos), além de informações sobre dízimos e ofertas, o total de novos convertidos e batizados, e as atividades realizadas durante o culto.

Ao clicar em “Gerar Relatório”, o sistema exibe um modal com as seguintes opções: inserir uma data específica ou período, inserir um ano específico ou período por ano, filtrar

apenas cultos ou eventos, e incluir as escalas de obreiros no relatório, a tela de listagem de relatórios pode ser visualizada na Figura 21.

Figura 21 - Tabela de listagem de relatórios



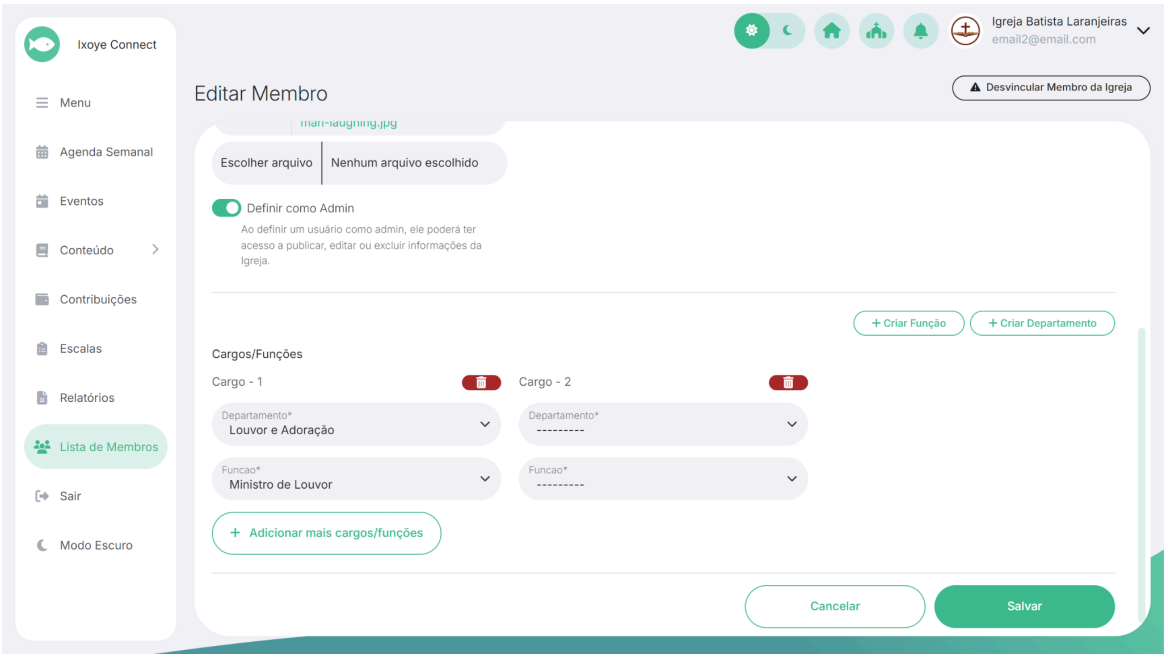
Fonte: Elaborada pela autora (2024).

No último módulo presente no menu lateral, está presente a Listagem de Membros, onde há uma tabela contendo o nome, idade, tempo que é membro daquela instituição, celular e data de aniversário, a aparência é semelhante à tabela de relatórios, mudando apenas os dados exibidos. O cadastro de novos membros na instituição também pode ser feito a partir deste módulo com as mesmas informações contidas na tela de cadastro inicial do sistema, ele já será vinculado automaticamente à instituição sem precisar inserir o código único.

A tela de edição possui uma seção extra em comparação a tela de cadastro, onde a instituição pode selecionar cargos ou funções de um departamento para o membro. Também é importante citar que a instituição pode definir um membro como administrador, assim o usuário do membro poderá também cadastrar, editar ou excluir dados daquela Instituição. O cadastro de funções e departamentos podem ser feitos nessa tela a partir dos botões “Criar Função” e “Criar Departamento” respectivamente. O cadastro de departamento exige apenas o nome ou uma descrição opcional, já o cadastro de funções exige relacionar à um departamento específico. Por fim, a instituição pode desvincular a conta de um membro de sua instituição ao clicar no botão “Desvincular Membro da Igreja”, que exibirá um modal de

confirmação ou cancelamento da ação, a Figura 22 exibe as informações descritas neste parágrafo.

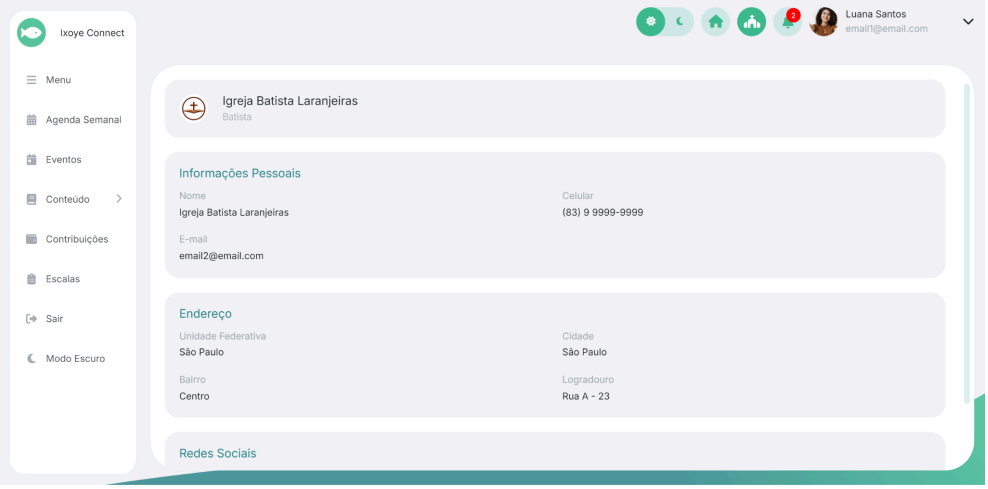
Figura 22 - Edição de Membro



Fonte: Elaborada pela autora (2024).

A página de Perfil da Instituição, que pode ser acessada pelo botão na parte de navegação superior com o ícone de igreja, contém as informações da Instituição, caso o usuário logado seja a própria instituição, haverá a opção de editar dados e também adicionar redes sociais que serão visíveis para os membros que visualizarem esta seção, na Figura 23 há a tela de Perfil da Instituição (caso logado como membro).

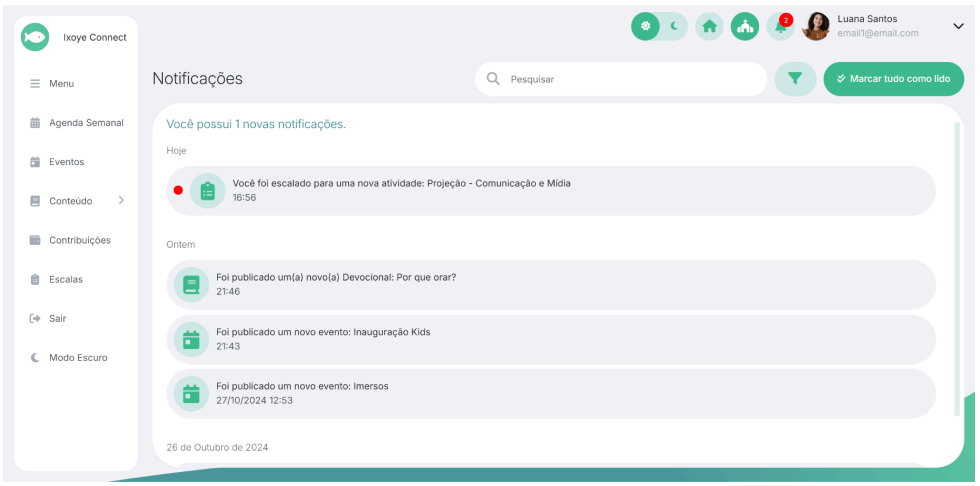
Figura 23 - Perfil da Instituição



Fonte: Elaborada pela autora (2024).

Após o botão de Perfil da Instituição, há o botão para a listagem de notificações, elas são geradas automaticamente ao cadastrar um evento, conteúdo ou escala. No cadastro de eventos ou conteúdos elas são enviadas para todos os membros relacionados à instituição, já no caso de escala a notificação é enviada apenas ao membro escalado. Ao clicar na notificação o usuário é redirecionado ao módulo respectivo daquela notificação, também é possível ler todas as notificações ao clicar no botão “Marcar tudo como lido”, como mostrado na Figura 24.

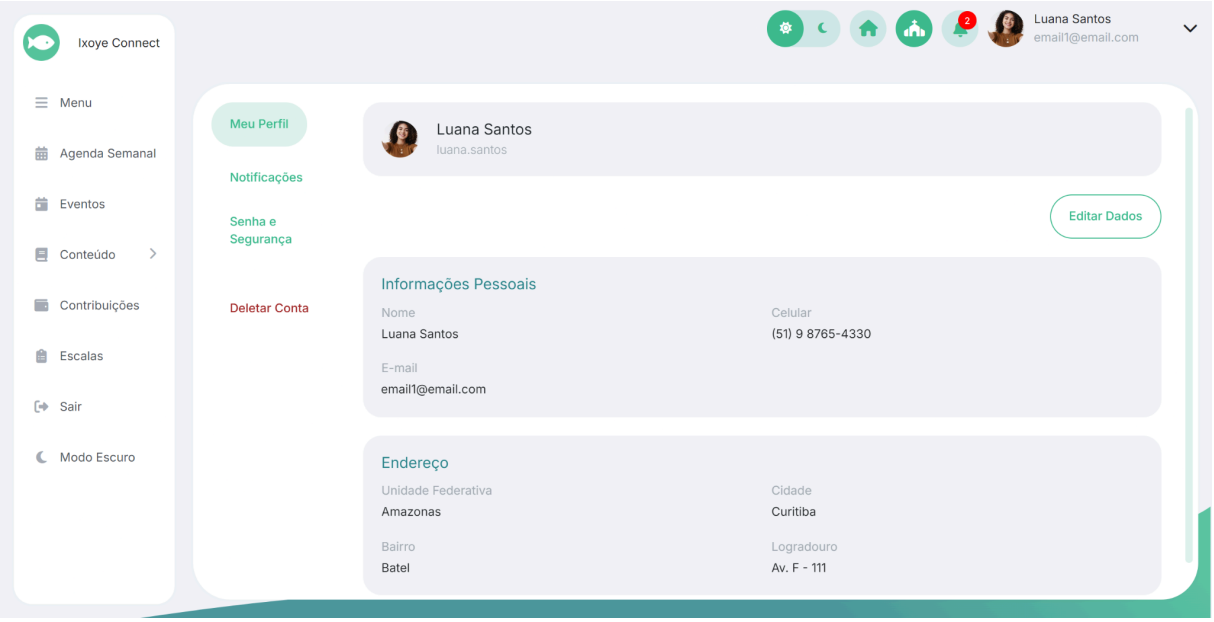
Figura 24 - Listagem de notificações



Fonte: Elaborada pela autora (2024).

Por fim, a tela de Perfil do Usuário exibe as informações de forma semelhante a tela de Perfil da Instituição diferenciando apenas os dados da conta, porém na lateral há uma seção de navegação para configurações do usuário, contendo “meu perfil” como primeira opção, em seguida configuração de notificações, senha e segurança e deletar, como presente na Figura 25. Essa seção lateral também é exibida no Perfil da Instituição caso a conta logada seja a própria instituição.

Figura 25 - Perfil do usuário e configurações



Fonte: Elaborada pela autora (2024).

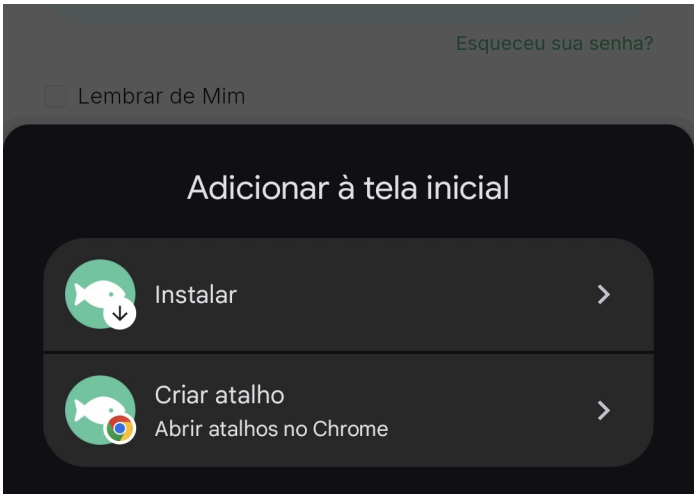
A configuração de notificações permite habilitar notificações *push*, desativar notificações, configurar período que prefere receber notificações, e silenciar notificações por um período de tempo. Em “Senha e segurança” há um formulário para mudar a senha do usuário, tendo que informar a senha atual, uma nova senha e confirmar a nova senha. E na opção final “Deletar Conta” exibe uma explicação para o usuário sobre a exclusão da conta e sobre a perda de seus dados, ao clicar no botão ainda haverá um modal de confirmação se realmente deseja realizar a exclusão e que seus dados serão perdidos sem possibilidade de desfazer a ação.

5.2.1 Resultados da Aplicação do PWA

A implementação do PWA aproxima o sistema de um aplicativo nativo e amplia suas funcionalidades. As funcionalidades implementadas pelo PWA incluem a opção de instalação do sistema diretamente na tela inicial dos dispositivos móveis e *desktops*, além de suporte a notificações *push*, oferecendo uma experiência mais interativa e acessível aos usuários. A seguir, são apresentados os resultados obtidos com a aplicação do PWA:

- **Opção de Instalação no Dispositivo *Mobile*:** Ao acessar o sistema por meio do navegador em um dispositivo *mobile*, o usuário é apresentado com a opção de instalar o sistema na tela inicial. Essa funcionalidade permite que o sistema seja acessado de maneira semelhante a um aplicativo nativo (Figura 26).

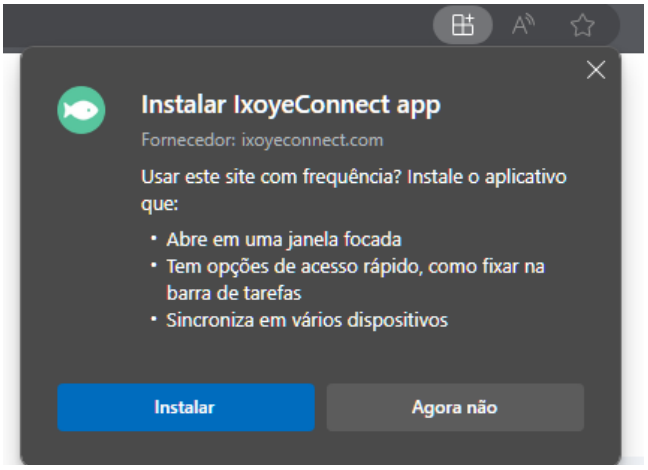
Figura 26 - Opção de instalação no dispositivo *mobile*



Fonte: Elaborado pela autora (2024).

- **Opção de Instalação no Desktop:** Em dispositivos *desktop*, a funcionalidade de instalação permite que o sistema funcione como um aplicativo independente, sem depender do navegador para ser executado, conforme ilustrado na Figura 27.

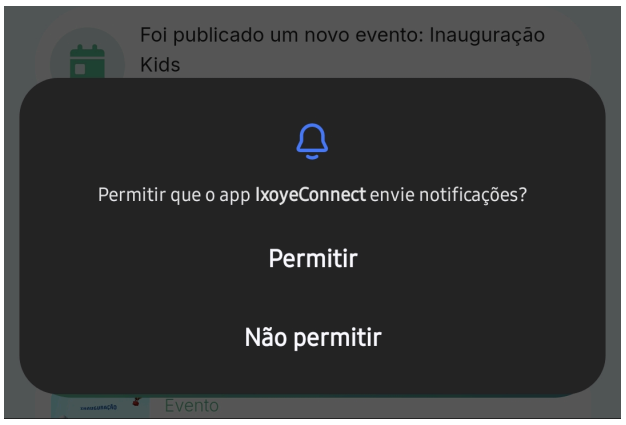
Figura 27 - Opção de instalação no *Desktop*



Fonte: Elaborado pela autora (2024).

- **Solicitação de Permissão para Notificações:** Ao acessar o sistema pela primeira vez, o usuário é solicitado a permitir notificações. Esta funcionalidade assegura que os usuários possam receber atualizações em tempo real sobre novos conteúdos, eventos ou escalas, promovendo uma comunicação mais direta e eficiente com o sistema (Figura 28).

Figura 28 - Solicitação de notificação (dispositivo *mobile*)

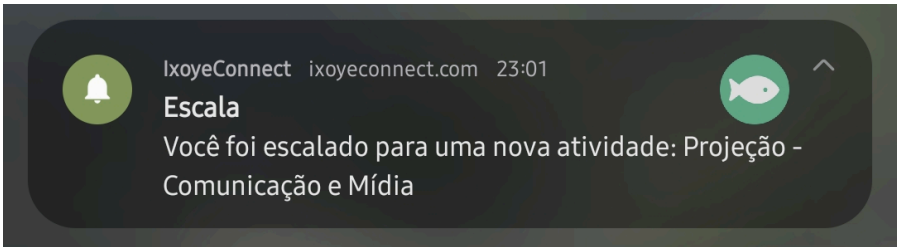


Fonte: Elaborado pela autora (2024).

- **Exibição de Notificações *Push*:** As notificações *push* são enviadas diretamente ao dispositivo sempre que uma nova atividade ocorre no sistema. Além disso, o ícone do aplicativo apresenta um indicador vermelho que informa ao usuário sobre novas

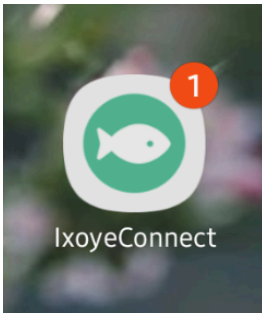
notificações, garantindo que o usuário perceba as atualizações de maneira visual (Figuras 29 e 30).

Figura 29 - Notificação *push* em um dispositivo *mobile*



Fonte: Elaborado pela autora (2024).

Figura 30 - Ícone do PWA com notificação *push*



Fonte: Elaborado pela autora (2024).

Essas funcionalidades PWA elevam a usabilidade e engajamento do sistema, permitindo que os usuários interajam com o sistema de forma rápida e independente, com acesso *offline* e notificações personalizadas. A adoção do PWA otimiza a experiência do usuário, oferecendo um sistema com características de aplicativo, que melhora o engajamento e proporciona uma experiência de navegação integrada e completa.

Os resultados apresentados demonstram o funcionamento completo do sistema, com cada módulo atendendo as necessidades específicas. O sistema oferece uma solução prática e eficiente para a administração de instituições religiosas, com uma interface intuitiva e responsiva.

5.3 Resultados do Formulário de avaliação

Neste capítulo, iremos explorar todos os resultados obtidos em campo, incluindo dados, experiências e *feedbacks* recebidos. Ao todo, foram coletadas 14 respostas ao questionário proposto. Além disso, responderemos a todas as questões levantadas na fase de metodologia, que foram aplicadas individualmente através de um formulário, seguindo os questionamentos do método USE.

O *software* Ixoye Connect foi avaliado pelos usuários para que diversas variáveis de usabilidade pudessem ser examinadas, as quais serão apresentadas nas subseções seguintes, acompanhadas das tabelas dos resultados obtidos.

5.3.1 Indicadores de Utilidade

O objetivo da avaliação dos indicadores de utilidade é mensurar o quanto o sistema agrega valor às atividades de gestão da igreja, verificando se ele cumpre a função para a qual foi desenvolvido. Esses indicadores buscam entender se o sistema realmente atende às necessidades cotidianas das instituições religiosas, facilitando a organização e o acompanhamento das atividades.

Ao verificar a utilidade prática das funcionalidades oferecidas, essa análise visa confirmar que o sistema não só possui ferramentas relevantes, mas também que estas são eficazes e realmente utilizadas pelos membros e líderes das igrejas. Os resultados dessa análise estão organizados e informados na Tabela 3.

Tabela 3 - Análise de Resultados sobre a Opinião dos Respondentes para Utilidade do Sistema

Característica	**Sucesso%**	**Insucesso%**	**Média**	**Desvio Padrão**	**P-value**
Total indicador de Utilidade	**98,6%**	**1,4%**	**4,98**	**0,09**	**< 0,05**
O sistema parece útil para a organização das atividades da igreja	100%	0%	5	0	-

As funcionalidades principais atendem às necessidades de gestão da igreja	100%	0%	5	0	-
O sistema proporciona praticidade no controle e acompanhamento das atividades da instituição	100%	0%	5	0	-
Considero o sistema uma ferramenta que agrega valor ao cotidiano da igreja	92,9%	7,1%	4,93	0,27	< 0,05

Fonte: Elaborado pela autora (2024).

O indicador de utilidade obteve uma média de 4,98 e uma taxa de sucesso de 98,6%, refletindo uma percepção extremamente positiva dos usuários sobre o valor e a praticidade do sistema para as atividades da igreja. Todos os participantes concordaram que o sistema oferece funcionalidades e organização adequadas para a gestão e controle institucional. O valor de p (inferior a 0,05) para o total do indicador sugere que as percepções positivas são estatisticamente significativas, reafirmando a importância do sistema no cotidiano da igreja. O desvio padrão baixo reforça a consistência nas respostas.

5.3.2 Indicadores de Facilidade de Uso

A avaliação dos indicadores de facilidade de uso tem como foco compreender a experiência do usuário em termos de simplicidade e intuitividade ao interagir com o sistema. Este aspecto é fundamental para garantir que o sistema possa ser utilizado sem barreiras ou dificuldades excessivas, promovendo uma curva de aprendizado rápida e um acesso facilitado às funcionalidades principais. O interesse dessa análise está em identificar o quanto os usuários conseguem operar o sistema de maneira autônoma e descomplicada, essencial para maximizar a adesão e a satisfação no uso contínuo do sistema. Os resultados dessa análise estão dispostos e apresentados na Tabela 4.

Tabela 4 - Análise de Resultados sobre a Opinião dos Respondentes para Facilidade de Uso

Característica	**Sucesso%**	**Insucesso%**	**Média**	**Desvio**	**P-**

				Padrão	value
Total Indicador de Facilidade de Uso	**91,1%**	**8,9%**	**4,86**	**0,23**	**< 0,05**
Sinto que seria fácil aprender a utilizar o sistema sem instruções adicionais	92,8%	7,1%	4,71	0,57	< 0,05
A navegação entre as diferentes funcionalidades é simples e intuitiva	92,9%	7,1%	4,93	0,27	< 0,05
Encontrei as informações e funcionalidades que preciso sem dificuldade	100%	7,1%	5	0	-
A interface do sistema é organizada e permite fácil acesso aos principais recursos	100%	0%	5	0	-

Fonte: Elaborado pela autora (2024).

Com uma taxa de sucesso de 91,1% e média de 4,86, os indicadores de facilidade de uso apontam para uma percepção majoritariamente positiva. A maioria dos usuários relatou facilidade em aprender a usar o sistema, navegar entre funcionalidades e localizar informações. O desvio padrão médio baixo indica respostas consistentes, e o p-value significativo sugere que a facilidade de uso foi um ponto forte do sistema. Pequenos ajustes podem ser feitos para abordar os poucos casos de insucesso, garantindo uma navegação mais intuitiva para novos usuários.

5.3.3 Indicadores de Satisfação

Os indicadores de satisfação têm a função de avaliar o contentamento dos usuários com o sistema, levando em conta a aparência visual, o conforto ao utilizar as funcionalidades, a segurança percebida e a recomendação para outras igrejas. Esse indicador é relevante para captar a percepção emocional e prática do usuário em relação ao sistema, indo além da

simples funcionalidade para considerar a experiência geral. Através dessa análise, busca-se entender o grau de aceitação do sistema e identificar aspectos que possam ser melhorados para aumentar a aprovação e confiança dos usuários, assegurando uma experiência completa e satisfatória. Os dados dessa análise estão organizados e exibidos na Tabela 5.

Tabela 5 - Análise de Resultados sobre a Opinião dos Respondentes para Satisfação do Usuário

Característica	**Sucesso%**	**Insucesso%**	**Média**	**Desvio Padrão**	**P-value**
Total Indicador de Satisfação	**96,4%**	**3,6%**	**4,96**	**0,13**	**< 0,05**
Estou satisfeito(a) com a aparência visual do sistema	85,7%	14,3%	,86	0,36	< 0,05
Acredito que o sistema seja agradável de usar	100%	0%	5	0	-
Eu recomendaria este sistema para outras igrejas ou instituições religiosas	100%	0%	5	0	-
Sinto-me seguro(a) ao utilizar o sistema em termos de proteção de dados	100%	0%	5	0	-

Fonte: Elaborado pela autora (2024).

O indicador de satisfação obteve 96,4% de sucesso e uma média de 4,96, mostrando que a maioria dos usuários ficou satisfeita com o sistema em aspectos como aparência, segurança de dados e agradabilidade de uso. As avaliações de segurança e recomendação tiveram 100% de aprovação, enquanto a aparência visual, apesar de bem avaliada, apresentou algumas sugestões de melhoria. O desvio padrão próximo de zero e o p-value significativo confirmam uma percepção sólida e consistente da satisfação geral dos usuários.

5.4 Análise SWOT

5.4.1 Forças (*Strengths*)

As principais forças do sistema são derivadas de sua alta avaliação em indicadores de utilidade e satisfação, como mostrado nos resultados do formulário. O sistema foi percebido como uma ferramenta extremamente útil para organização e gestão de atividades, com uma taxa de 100% de concordância total em itens relacionados à organização e atendimento das necessidades das instituições religiosas. Isso demonstra que ele foi bem recebido como um recurso facilitador para o controle das atividades, simplificação dos processos de inscrição para eventos e registro de atividades.

Outro ponto forte é a interface intuitiva e a facilidade de uso do sistema, que receberam altas avaliações nos indicadores de facilidade, com a maioria dos usuários afirmando que conseguem utilizar o sistema sem dificuldade e que a navegação é intuitiva. Esse fator é essencial para aumentar a adesão e tornar a ferramenta acessível mesmo para usuários menos familiarizados com tecnologia. A experiência visual e a organização da interface foram também muito bem avaliadas, sugerindo que o *design* atende tanto às expectativas de funcionalidade quanto às de estética e usabilidade.

Adicionalmente, o sistema foi bem avaliado em segurança de dados, com 100% dos usuários se sentindo seguros ao utilizá-lo, o que é essencial para qualquer aplicação de gestão, especialmente no contexto de instituições religiosas, que lidam com dados sensíveis dos membros. A satisfação geral, com 100% dos respondentes indicando que recomendariam o sistema a outras igrejas, reforça a sua aceitação e o potencial para se estabelecer como uma solução de referência no segmento.

5.4.2 Fraquezas (*Weaknesses*)

Apesar das avaliações positivas, alguns pontos fracos foram identificados a partir das respostas. Uma fraqueza observada é que, embora a interface seja avaliada como intuitiva, 7,1% dos respondentes ainda se sentem indecisos sobre a facilidade de uso, sugerindo que

alguns ajustes poderiam tornar a experiência ainda mais acessível, especialmente para públicos que podem não ser tão familiarizados com tecnologia.

Outro ponto a considerar é que, embora a funcionalidade de relatórios de participação tenha sido solicitada e integrada, ela foi uma das características com menor taxa de adesão, sugerindo que pode ser subutilizada ou que talvez precise de ajustes para melhor atender às expectativas. Isso aponta uma oportunidade de refinamento, seja na apresentação das funcionalidades ou no treinamento para garantir que os usuários percebam o valor dessa ferramenta específica.

Além disso, um aspecto menor, mas relevante, é a taxa de satisfação visual, em que 85,7% dos respondentes concordaram totalmente, enquanto 14,3% indicaram uma satisfação parcial. Isso pode sinalizar uma necessidade de aprimoramento da interface, talvez por meio de personalizações estéticas, que atendam melhor às preferências dos usuários e aumentem a percepção de modernidade e atratividade do sistema.

5.4.3 Oportunidades (*Opportunities*)

O sistema de gestão tem um grande potencial de crescimento e desenvolvimento. Com base no *feedback* positivo de utilidade e satisfação, uma oportunidade clara é a expansão do sistema para outras instituições religiosas que busquem soluções tecnológicas para gestão e organização. A alta taxa de recomendação sugere que o sistema pode ganhar tração de forma orgânica, com usuários atuais incentivando sua adoção por outras instituições religiosas.

Além disso, há uma oportunidade para desenvolver treinamentos ou materiais de apoio, como tutoriais e guias, que possam facilitar ainda mais o uso do sistema e ajudar a maximizar o aproveitamento de funcionalidades como os relatórios de participação. Com o suporte adequado, essas funcionalidades poderiam ser mais bem aproveitadas, aumentando o valor percebido pelo usuário.

Outra oportunidade está na contínua adição de funcionalidades baseadas em demandas identificadas, como a publicação de materiais religiosos e a comunicação de avisos, ambas características muito bem avaliadas e que podem ser estendidas. Ampliar essas

funcionalidades pode criar um ecossistema de soluções que suporte mais aspectos do cotidiano das instituições e, com isso, tornar o sistema ainda mais indispensável para o público-alvo.

5.4.4 Ameaças (*Threats*)

Um dos principais riscos ao sistema está na evolução das exigências tecnológicas e nas necessidades de adaptação contínua. Com a crescente demanda por ferramentas mais robustas e adaptativas, há o risco de que funcionalidades consideradas úteis hoje se tornem obsoletas ou insuficientes. Isso implica a necessidade de manutenção constante e desenvolvimento contínuo para acompanhar as mudanças tecnológicas e responder às novas demandas das igrejas e dos usuários.

Outro desafio é a concorrência potencial de outras plataformas e *softwares* voltados para gestão de instituições religiosas, que possam oferecer soluções similares. Caso outras ferramentas ofereçam funcionalidades diferenciadas, como maior customização ou recursos mais sofisticados de análise de dados, isso poderia diminuir o interesse pelo sistema atual. Assim, manter um processo de atualização e inovação contínua é essencial para preservar a competitividade.

Além disso, como o sistema lida com dados pessoais e financeiros, existe uma ameaça relacionada à segurança cibernética. Com o aumento dos incidentes de ataques e invasões, garantir a segurança dos dados será um ponto de atenção constante, pois qualquer incidente de vulnerabilidade ou violação de dados poderia comprometer a confiança dos usuários e prejudicar a reputação do sistema.

5.4.5 Ilustração da matriz SWOT

Figura 32 - Matriz SWOT

Fonte: Elaborado pela autora (2024).

6 CONCLUSÃO

Neste capítulo, apresentaremos as considerações finais do projeto, destacando os principais resultados e suas implicações. Os próximos subtópicos oferecerão uma visão clara das conclusões e direções para trabalhos futuros.

6.1 Conclusões Finais

O desenvolvimento deste sistema de gestão para instituições religiosas atingiu com sucesso os objetivos propostos. Desde o início, buscou-se a criação de uma ferramenta intuitiva e eficiente para facilitar a organização e comunicação nas instituições. Os resultados da avaliação final, obtidos através de um formulário com os usuários, mostraram uma taxa elevada de satisfação com as funcionalidades e a interface do sistema. Os indicadores de utilidade, facilidade de uso e satisfação apresentaram respostas amplamente positivas, com destaque para a recomendação unânime do sistema por parte dos representantes das igrejas. Essa aceitação confirma que o sistema atende bem às necessidades das igrejas e demonstra o impacto positivo da integração de tecnologias no cotidiano das instituições religiosas.

6.2 Contribuições da Pesquisa

Este trabalho contribui tanto para o campo acadêmico quanto para a prática, com ênfase na ciência da computação aplicada a contextos religiosos, uma área pouco explorada. A pesquisa possibilitou avanços na concepção e implementação de sistemas de gestão voltados para um público com necessidades e estruturas específicas. A utilização de métodos de análise, como a análise SWOT e o Método USE, para avaliar um *software* em ambiente real, traz uma perspectiva valiosa sobre as demandas de usuários com perfis distintos. Além disso, a pesquisa preenche uma lacuna existente em estudos voltados à tecnologia em contextos religiosos, fornecendo uma base para futuras investigações e implementações na área.

Outro aspecto relevante é a abordagem iterativa e ágil empregada durante o desenvolvimento, que permitiu ajustar o sistema conforme o *feedback* dos usuários,

reforçando a importância do desenvolvimento incremental em projetos voltados para comunidades com diferentes graus de familiaridade com tecnologia.

6.3 Desafios e Limitações do Estudo

Durante o desenvolvimento e a avaliação do sistema, alguns desafios foram identificados. A dificuldade de localizar representantes de igrejas que pudessem avaliar o sistema presencialmente foi uma limitação significativa, uma vez que esses participantes representam a diversidade de necessidades no uso de tecnologia nas instituições religiosas. Além disso, notou-se que parte dos entrevistados possuía baixo conhecimento tecnológico, o que gerou a necessidade de uma interface ainda mais intuitiva e de instruções adicionais para facilitar o uso do sistema. Outra limitação é a escassez de trabalhos acadêmicos semelhantes, o que dificultou o embasamento teórico e a comparação de resultados. A pouca disponibilidade de pesquisas sobre sistemas de gestão para igrejas destacou a importância de ampliar as investigações no tema.

Além desses fatores, a infraestrutura de hospedagem também trouxe dificuldades. O sistema inicialmente foi implantado em um servidor AWS, mas a constante indisponibilidade, incluindo quedas frequentes e problemas técnicos, impactou o fluxo de produção, forçando a migração do sistema para um servidor na Digital Ocean. Essa mudança foi essencial para garantir maior estabilidade, mas resultou em atrasos e ajustes adicionais. Esses desafios destacam a complexidade de se desenvolver uma solução tecnológica robusta para um público específico, ressaltando a necessidade de adaptações ao longo do processo para assegurar a viabilidade do projeto.

6.4 Sugestão de Trabalhos Futuros

A pesquisa e as entrevistas com os usuários trouxeram sugestões relevantes que podem direcionar trabalhos futuros e expandir as funcionalidades do sistema. Algumas das melhorias sugeridas incluem a criação de novos módulos, como Carteira de Membro, Carta de Recomendação, Certificado de Batismo e um módulo exclusivo para gestão financeira. Essas

funções adicionais poderão tornar o sistema mais completo e atender a outras necessidades específicas das igrejas. Trabalhos futuros poderiam também explorar integrações com outras plataformas e recursos de segurança aprimorados, visando aumentar a robustez e o alcance do sistema. Esses aprimoramentos e novas funções potencializam ainda mais a aplicabilidade do sistema em contextos religiosos, fornecendo uma plataforma multifuncional e adaptada para diferentes atividades e necessidades das instituições.

11 CRONOGRAMA

Quadro 2 - Cronograma de Atividades de 2024

Período	2024								
Atividades	Mar	Abr	Maio	Jun	Jul	Ago	Set	Out	Nov
Pesquisa bibliográfica	X	X	X	X					
Determinação dos Objetivos	X								
Coleta de dados					X	X			
Levantamento de Requisitos					X	X			
Design e Arquitetura do Sistema						X	X		
Desenvolvimento *Front-end* e *Back-end*						X	X	X	
Aplicação de									X

Questionário									
Redação do TCC							X	X	X

Fonte: Elaborada pela autora (2024).

REFERÊNCIAS

ADETUNJI, O.; AJAEGBU, C.; OTUNEME, N.; OMOTOSHO, O. Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development. **American Scientific Research Journal for Engineering, Technology, and Sciences**, v. 68, n. 1, p. 85-99, 2020.

AMAZON. **O que é o Amazon EC2? - Amazon Elastic Compute Cloud**. 2024. Disponível em: https://docs.aws.amazon.com/pt_br/AWSEC2/latest/UserGuide/concepts.html>.

AMAZON. **New: Using Amazon EC2 Instance Connect for SSH access to your EC2 Instances**. 2019. Disponível em: https://aws.amazon.com/pt/blogs/compute/new-using-amazon-ec2-instance-connect-for-ssh-access-to-your-ec2-instances. Acesso em: 5 nov. 2024.

AMIT M.; ADITYA B.; PRIYANKA K.; RADHIKA M. Progressive Web App for Educational System. **International Research Journal of Engineering and Technology (IRJET)**, 2018.

AWS. **Front-end x back-end — Diferença entre desenvolvimento de aplicativos.** 2024. Disponível em: https://aws.amazon.com/pt/compare/the-difference-between-frontend-and-backend. Acesso em: 30 maio. 2024.

AWS. **AWS introduz o Ubuntu Desktop para Amazon WorkSpaces**. 2022. Disponível em: https://aws.amazon.com/pt/about-aws/whats-new/2022/09/aws-introduces-ubuntu-desktop-amazon-workspaces. Acesso em: 5 nov. 2024.

BALIYAN, A.; KASWAN, K. S.; DHATTERWAL, J. S. An Empirical Analysis of Python Programming for Advance Computing. **2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)**. Noida, India. IEEE. 2022.

CHRISTIAN TECH JOBS. **What is Church Tech? A Comprehensive Guide to Church Technology.** 2023. Disponível em: https://www.christiantechjobs.io/blog/what-is-church-tech-a-comprehensive-guide-to-church-technology>. Acesso em: 19 jun. 2024.

CONERY, R.; HANSELMAN, S.; HAACK, P.; GUTHRIE, S. **Microsoft Application Architecture Guide, 2nd Edition: Designing Applications on the .NET Platform.** Microsoft Press, 2. ed., 2009.

CRESWELL, J. W.; CRESWELL, J. D. Research design: qualitative, quantitative, and mixed methods approaches. 6. ed. **Thousand Oaks, CA: Sage**, 2022.

DEITEL, H. M.; DEITEL, P. J. **Java: como programar** (10th ed.). São Paulo, 2016.

DIGITALOCEAN. **Learning paths: Ubuntu.** 2024. Disponível em: https://www.digitalocean.com/community/learning-paths/ubuntu. Acesso em 1 nov. 2024.

DIGITALOCEAN. **SSH. Documentação DigitalOcean.** 2024. Disponível em: https://docs.digitalocean.com/glossary/ssh. Acesso em: 01 nov. 2024.

DINIZ, S.; SANTANA, B. **Interação humano-computador**. Rio De Janeiro: Elsevier, 2010.

DJANGO SOFTWARE FOUNDATION. **Documentação do Django: Class-based views.** 2024. Disponível em: https://docs.djangoproject.com/pt-br/5.0/topics/class-based-views. Acesso em: 28 jun. 2024.

DJANGO SOFTWARE FOUNDATION. **Documentação do Django: Despachante de URL.** 2024. Disponível em: https://docs.djangoproject.com/pt-br/5.0/topics/http/urls. Acesso em: 28 jun. 2024.

DJANGO SOFTWARE FOUNDATION. **Documentação do Django: Migrations.** 2024. Disponível em: https://docs.djangoproject.com/en/5.1/topics/migrations. Acesso em: 27 out. 2024.

DJANGO SOFTWARE FOUNDATION. **Documentação do Django: versão 5.0.** 2024. Disponível em: https://docs.djangoproject.com/pt-br/5.0. Acesso em: 28 jun. 2024.

DJANGO ALLAUTH. **Documentação do Django All-Auth: versão 65.2.0.** 2024. Disponível em: https://docs.allauth.org/en/latest. Acesso em: 3 out. 2024.

DJANGO EXTRA VIEWS. **Django Extra Views 0.15.0 documentation**. 2013. Disponível em: https://django-extra-views.readthedocs.io/en/latest. Acesso em: 3 out. 2024.

DUISEBEKOVA, K. *et al*. DJANGO AS SECURE WEB-FRAMEWORK IN PRACTICE. **The Bulletin of KazATC Вестник КазАТК**, v. 1, n. 116, p. 275–281, 2021.

FEITOSA, S.; COMARELLA, R. Aprendendo Conceitos de Orientação a Objetos Usando as Ferramentas Scratch e Snap!. **XI Computer on the Beach.** Camboriú, SC. 2020.

FERNANDO, L**. JavaScript: A chave para desenvolver sites dinâmicos e interativos.** Disponível em: https://carreiraemti.com.br/blog/javascript-a-chave-para-desenvolver-sites-dinamicos-e-interativos. Acesso em: 17 jun. 2024.

GIL, A, C. **Como Elaborar Projetos de Pesquisa.** [s.l.] Atlas, 2022. *E-book*.

GITHUB. **Sobre o SSH - GitHub Docs.** 2024. Disponível em: https://docs.github.com/pt/authentication/connecting-to-github-with-ssh/about-ssh>.

KAPLAN-MOSS, J.; HOLOVATY, A.; CUNNINGHAM, K. **The definitive guide to Django: web development done right.** 3. ed. New York: Apress, 2014.

LEMKE, W. **Technology and Ministry during a Time of Pandemic: An Annotated Bibliography.** 2021.

LET'S ENCRYPT. **Let's Encrypt - How it works.** Disponível em: https://letsencrypt.org/pt-br/how-it-works/. Acesso em: 27 out. 2024.

LONGMAN, A. **Chapter 2**. A history of HTM . Disponível em: <https://www.w3.org/People/Raggett/book4/ch02.html>.

LUND, A. M. **Measuring usability with the use questionnaire.** Usability and User Experience Newsletter of the STC Usability SIG, v. 8, n. 2, p. 3–6, 2001.

MARINS, J.; GUARIENTI, G. **Introdução a Banco de Dados.** Apostila do curso de Graduação em Tecnologia Educacional e Licenciatura da Universidade Aberta do Brasil, 2019.

MONSANTO, F.; MESTRE, H.; OLIVEIRA, C. **Management Information System (Django).** Mestrado em Engenharia Informática Relatório Final de Estágio, 2014.

MOZILLA. **JavaScript Basics.** Disponível em: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics>.

MOZILLA. **Progressive Web Apps.** 2024. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/Progressive_web_apps.

OFFICE TIMELINE. **SWOT analysis: how to plan for success.** 2023. Disponível em: https://www.officetimeline.com/blog/swot-analysis-how-to-plan-for-success. Acesso em: 08 nov. 2024.

OLALEMI, S. **App Fatigue: Why people don't want to download your mobile apps and what you can do differently.** Disponível em: https://olalemis.medium.com/app-fatigue-why-people-dont-want-to-download-your-mobile-apps-and-what-you-can-do-diferentemente-afc01e12d47e.

ORACLE. **O que é o MySQL?.** 2024. Disponível em: https://www.oracle.com/br/mysql/what-is-mysql. Acesso em: 20 maio. 2024.

ORACLE. **O que é um banco de dados?.** 2024. Disponível em: https://www.oracle.com/br/database/what-is-database.

POINT LOMA NAZARENE UNIVERSITY. **Integrating Technology in Ministry: Adapting for the Digital Age.** 2023. Disponível em: https://www.pointloma.edu/resources/theology-christian-ministry/technology-ministry-adapting-digital-age>. Acesso em: 19 maio. 2024.

PUSHPAY. **Pushpay's 2023 State of Technology Report Reveals Hybrid Church Is Here to Stay.** 2023. Disponível em: https://www.globenewswire.com/news-release/2023/03/01/2618281/0/en/Pushpay-s-2023-Sta

te-of-Technology-Report-Reveals-Hybrid-Church-Is-Here-to-Stay.html. Acesso em: 20 jun. 2024.

POETRY TEAM. **Python Poetry Documentation.** 2024. Disponível em: https://python-poetry.org/docs. Acesso em 1 nov. 2024.

SANNER, M. F. Python: a programming language for software integration and development. Journal of Molecular Graphics & Modelling, **The Scripps Research Institute**, 1999.

SANTIAGO, C. *et al*. Desenvolvimento de sistemas Web orientado a reuso com Python, Django e Bootstrap. **Sociedade Brasileira de Computação**, 2020.

SERVCLOUD. **O que é CloudFlare e porque usá-lo?.** 2023. Disponível em: https://servcloud.com.br/o-que-e-cloudflare-e-porque-voce-deve-usa-lo.

SHARMA, V. *et al*. Progressive Web App (PWA) - One Stop Solution for All Application Development Across All Platforms. **International Journal of Scientific Research in Computer Science, Engineering and Information Technology**, p. 1120–1122, 20 mar. 2019.

SILVA, I. F. B. DA. **Utilização de folhas de estilo para definir a apresentação de páginas Web.** Mestrado em Ensino, 2014.

SILVA, R. O. da; SILVA, I. R. S. **Linguagem de programação Python.** Tecnologias em projeção, 2019.

SOLOMON, B. **How to deploy Django applications with NGINX and Gunicorn.** 2024. Disponível em: https://realpython.com/django-nginx-gunicorn. Acesso em 1 nov. 2024.

STACK OVERFLOW. **Stack Overflow Developer Survey 2022.** 2022. Disponível em: https://survey.stackoverflow.co/2022/#most-popular-technologies-database>. Acesso em: 20 maio. 2024.

INSPER. **Stack Overflow Developer Survey aponta tecnologias mais populares em 2023.** Disponível em: https://www.insper.edu.br/noticias/stack-overflow-developer-survey-aponta-tecnologias-mais-populares-em-2023. Acesso em: 1 jun. 2024.

UBUNTU. **Install and configure a MySQL server.** 2024. Disponível em: https://ubuntu.com/server/docs/install-and-configure-a-mysql-server. Acesso em: 27 out. 2024.

VALENTE, Marco Tulio. **Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade.** 1. ed. [S. l.]: Independente, 2020.

VICERI. **Aplicações serverless: quais as vantagens de usar esta arquitetura?.** 2020. Disponível em:

https://viceri.com.br/insights/aplicacoes-serverless-quais-as-vantagens-de-usar-esta-arquitetura/.

ZIMÁNYI, E.; JALLOW, B.; SHAFAGH, K. Object Relational Mapping and Entity Framework. **Université Libre de Bruxelles**, 2018.