Centro Universitário Senac - Santo Amaro

Enrico Andrade

Gabriel Jesus

Italo Daniel

Kaio Paulo

**Projeto Integrador**

Life And Peace

São Paulo

2024

Enrico Andrade

Gabriel Jesus

Italo Daniel

Kaio Paulo

**Projeto Integrador**

Life And Peace

Projeto Integrador apresentado para o quarto semestre da graduação de Sistemas de Informação.

Orientador pelo professor Calyton Mendonça Feliciano

São Paulo

2024

Sumário

[**EXEMPLO DE DOCUMENTO DE ESPECIFICAÇÃO DE SOFTWARE 4**](#_gjdgxs)

[**1. INTRODUÇÃO 4**](#_30j0zll)

[**2. VISÃO GERAL 6**](#_1fob9te)

[**2.1. O Projeto e a contribuição à comunidade 7**](#_3znysh7)

[**2.2. ESG e as ODS contempladas no projeto 7**](#_2et92p0)

[**2.3. Justificativa 8**](#_tyjcwt)

[**3. ESCOPO DO PROJETO 9**](#_3dy6vkm)

[**3.1. Matriz de papéis e responsabilidades 9**](#_1t3h5sf)

[**4. CLIENTE 10**](#_4d34og8)

[**5. DESCRIÇÃO DOS REQUISITOS 10**](#_2s8eyo1)

[**5.1 REQUISITOS FUNCIONAIS: 10**](#_17dp8vu)

[**5.2 REQUISITOS NÃO FUNCIONAIS 11**](#_3whwml4)

[**5.3 REGRAS DE NEGÓCIO 11**](#_3rdcrjn)

[**6. PRODUCT BACKLOG INICIAL 12**](#_26in1rg)

[**6.1. Divisão de Sprints ou Gantt 12**](#_lnxbz9)

[**6.2. Histórias de Usuário 13**](#_35nkun2)

**6.3 BDD 14**

[**7. ARQUITETURA DE SOFTWARE 15**](#_1ksv4uv)

[**7.1. Padrão de Projeto e/ou Estilo Arquitetural 16**](#_2bn6wsx)

[**8. Design centrado no usuário 17**](#_44sinio)

[**8.1. Mapa de Empatia 21**](#_2jxsxqh)

[**8.2. PERSONAS / Protopersonas 21**](#_z337ya)

[**9. MODELO DE CASOS DE USO 22**](#_3j2qqm3)

[**9.1 IDENTIFICAÇÃO DOS ATORES E SUAS RESPONSABILIDADES 22**](#_1y810tw)

[**9.2 DEFINIÇÃO DE PRIORIDADE DE DESENVOLVIMENTO DOS CASOS DE USO 22**](#_4i7ojhp)

**9.3 DESCRIÇÃO DETALHADA DO CASO DE USO 23**

**10 DIAGRAMA DE ATIVIDADE 27**

**11 DIAGRAMA DE SEQUÊNCIA 28**

**12 DIAGRAMA DE CASO DE USO 29**

**13 DIAGRAMA DE PROCESSO DE NEGÓCIO 29**

**14 DIAGRAMA UML 30**

**13 METODOLOGIA 31**

**14 DESENVOLVIMENTO DO TRABALHO DO GRUPO 33**

**15 PROJETOS FUTUROS 34**

**16 CONCLUSÃO 35**

Sumário de Imagens

**FIGURA 1: ODS 16** [**7**](#_gjdgxs)

**FIGURA 2: ODS 12** [**8**](#_30j0zll)

**FIGURA 3: TELA INICIO** [**17**](#_1fob9te)

**FIGURA 4: FUNCIONALIDADES 17**

**FIGURA 5: TELA SOBRE** [**18**](#_2et92p0)

**FIGURA 6: TELA BATIZADO** [**19**](#_tyjcwt)

**FIGURA 7: TELA CASAMENTO** [**20**](#_3dy6vkm)

**FIGURA 8: TELA CONTATO** [**21**](#_1t3h5sf)

**FIGURA 9: TELA DOAÇAO 21**

**FIGURA 10: DIAGRAMA DE ATIVIDADE 28**

**FIGURA 11: DIAGRAMA DE SEQUÊNCIA 29**

**FIGURA 12 DIAGRAMA DE CASO DE USO 30**

**FIGURA 13: DIAGRAMA DE PROCESSO DE NEGOCIOS 31**

**FIGURA 14: DIAGRAMA UML 32**

**FIGURA 15: API DE EMAIL 33**

**FIGURA 16: API DE EMAIL 34**

**FIGURA 17: API DE CALENDARIO 34**

**FIGURA 18: API DE REDES SOCIAS 35**

# EXEMPLO DE DOCUMENTO DE ESPECIFICAÇÃO DE SOFTWARE

Este projeto será um site para uma igreja com cadastro e controle de usuários, atualização de informações e avisos, marcação de eventos como casamentos e batismos online, com garantia de segurança de dados e acesso controlado pelos membros administrativos da igreja, como aprovação e não aprovação de eventos.

Neste projeto serão contempladas duas ODS, sendo elas:

**ODS N°16. Paz, Justiça e Instituições Eficazes**

“Promover sociedades pacíficas e inclusivas para o desenvolvimento sustentável, proporcionar o acesso à justiça para todos e construir instituições eficazes, responsáveis e inclusivas em todos os níveis”

**ODS N° 12. Consumo e produção responsáveis**

“Garantir padrões de consumo e de produção sustentáveis”

# 1. INTRODUÇÃO

Este documento tem como objetivo coletar, analisar e definir as características e necessidades de alto nível da igreja para o desenvolvimento de um sistema de gestão.

O foco do trabalho está nas capacidades necessárias para os envolvidos e usuários finais, e por que estas necessidades existem. Os detalhes de como a igreja atende estas necessidades estão detalhados ao longo deste documento.

Este trabalho apresenta o desenvolvimento de um sistema de gestão para uma igreja, focado em facilitar o gerenciamento de membros, agendamentos de eventos e comunicação por meio de avisos.

Tratando-se de uma igreja pequena, o sistema terá como diferencial o tratamento mais próximo entre os administradores e os membros. Além disso, o sistema permitirá avisos na tela inicial, consulta de horários e datas para agendamento dos eventos, assinatura digital de documentos necessários para o agendamento, rápido acesso às redes sociais e um player na tela inicial onde haverá uma live ou vídeo passando.

Realizada a pesquisa em alguns sites semelhantes, espera-se que inicialmente havendo um tráfego pequeno no site da igreja Life And Peace, os administradores terão um contato mais próximo com os membros, além disso, percebe-se a falta de um agendador com validação eletrônica nesses sites e como principal diferencial decidimos incluir no site Life And Peace.

# 2. VISÃO GERAL

O objetivo deste trabalho é apresentar o processo de desenvolvimento de um sistema de gestão para uma igreja, abordando desde a identificação das necessidades até a implementação e teste do sistema. Além disso, busca-se fornecer uma visão clara dos benefícios e funcionalidades oferecidos pelo sistema.

**Segurança de Dados**

Restrições de segurança devem ser rigorosamente observadas para proteger os dados dos usuários da igreja. Isso inclui criptografia de senhas, controle de acesso e medidas para proteção contra ameaças de segurança.

**Conexão à Internet**

O sistema requer uma conexão constante à internet para a maioria das operações.

**Escalabilidade**

Embora o sistema seja de pequeno porte, será projetado de forma a permitir futuras expansões e escalabilidade, caso a igreja cresça ou necessite de mais recursos.

**Manutenção e Suporte**

O sistema deve ser documentado adequadamente para permitir sua manutenção contínua e prestação de suporte aos administradores da igreja.

**Interface Amigável e Intuitiva**

O sistema deverá ter uma interface agradável que não confunda o usuário e que o guie de forma didática durante sua experiência, contribuindo com a inclusão digital

**Digitalização de Documentos**

Para facilitar o processo de agendamento de eventos, o sistema incluirá a funcionalidade de digitalizar documentos necessários, como contratos para casamentos e batismos, garantindo a validade legal e a segurança das transações.

**2.1 O Projeto e a Contribuição à Comunidade**

Através do site, as pessoas poderão marcar um casamento ou batizado sem precisar ir até o local. A igreja poderá anunciar seus eventos e avisos com todas as informações importantes para o público de forma rápida e direta, contribuindo para a diminuição do uso de papel e do impacto ambiental, podendo também receber doações voluntárias dos membros.

A construção do projeto envolveu etapas como levantamento de requisitos, análise de viabilidade, definição de arquitetura e desenvolvimento do sistema. Foram utilizadas tecnologias como banco de dados no mySql, linguagens de programação como HTML, CSS e JavaScript e frameworks para garantir a eficiência e segurança do sistema.

# 2.2. ESG e as ODS contempladas no projeto

**ODS contempladas no projeto:**

Figura 1

ODS N°16. Paz, Justiça e Instituições EficazesAplicativo

Descrição gerada automaticamente com confiança baixa

“Promover sociedades pacíficas e inclusivas para o desenvolvimento sustentável, proporcionar o acesso à justiça para todos e construir instituições eficazes, responsáveis e inclusivas em todos os níveis”

“16.7 Garantir a tomada de decisão responsiva, inclusiva, participativa e representativa em todos os níveis”

Fonte: www.embrapa.com

Figura 2

ODS N° 12. Consumo e produção responsáveisUma imagem contendo Ícone

Descrição gerada automaticamente

“Garantir padrões de consumo e de produção sustentáveis”

“12.5 Até 2030, reduzir substancialmente a geração de resíduos por meio da prevenção, redução, reciclagem e reuso”

Fonte: www.embrapa.com

ESG contempladas no projeto:

**AMBIENTAL**: Para minimizar o impacto no meio ambiente, a igreja reduzira o uso de papel, como por exemplo em panfletos com informações sobre as próximas missas etc, marcação de eventos como casamentos e batismos serão feitos online, assim diminuindo a produção a produção de resíduos.

**SOCIAL:** Visando um ambiente inclusivo e tratamento justo para todos os colaboradores, sera implementado um sistema interface inituitiva com fonte em alto contraste e legível para todos, junto com um sistema de acessibilidade em libras para deficientes visuais.

**2.3. Justificativa**

Igrejas nem sempre tem um sistema virtual e na maioria dos casos seus colaboradores não possuem conhecimento para criação de tal coisa, fazendo com que seus membros tenham que ir pessoalmente a igreja para busca de informações e marcações de eventos, se limitando sempre ao uso do papel.

Com isso em mente decidimos criar um sistema para auxílio tanto de membros que trabalham na igreja, quanto para colaboradores que buscam informações da igreja sem precisar sair de casa.

O levantamento do problema revelou que a igreja enfrentava dificuldades no gerenciamento de informações dos membros, agendamento de eventos e comunicação interna.

Com base nessa pesquisa, foram identificadas as funcionalidades necessárias para solucionar esses problemas, como cadastro de membros, agendamento de eventos e gerenciamento de avisos.

**ODS N°16. Paz, Justiça e Instituições Eficazes**

Atraves de nosso site, sistema será responsivo para que as decisões possam ser tomadas de forma rápida e adaptativa. Teremos uma interface inclusiva e acessível para todos os públicos, com fontes com boas legibilidade e alto contraste com uma ferramenta em libras para deficientes visuais.

**ODS N° 12. Consumo e produção responsáveis**

Com a criação do site a igreja reduzira o consumo de papel de papel em seu dia a dia, já que informações sobre missas e avisos de eventos, marcações de casamentos e batismos serão todos feitos virtualmente.

# 3. ESCOPO DO PROJETO

O escopo deste projeto inclui o desenvolvimento de um sistema de gerenciamento de usuários da igreja e avisos para web. As funcionalidades principais que farão parte do produto são:

* Cadastro de usuários da igreja com informações detalhadas;
* Atualização e exclusão de informações de usuários da igreja;
* Listagem de usuários da igreja para consulta;
* Cadastro de avisos para comunicação interna;
* Atualização e exclusão de avisos;
* Visualização de avisos pelos usuários da igreja;
* Agendamento de eventos pelos usuários;
* Aprovação de agendamento pelo administrador;
* Digitalização de documentos necessários para agendamentos;
* Garantia de segurança de dados e acesso controlado;
* Streaming de uma missa;
* Atalhos para redes sociais da igreja.

**3.1. Matriz de papéis e responsabilidades**

Documentação: Gabriel Jesus e Kaio Paulo

Desenvolvimento Front-end: Italo Daniel e Kaio Paulo

Desenvolvimento Back-end: Enrico Andrade e Gabriel Jesus

# 4. CLIENTE

Nome: SENAC - SP

Ramo de Atividade: Estabelecimento de ensino em nível médio e superior tanto graduação quanto pós graduação, com cursos em todas as áreas do conhecimento.

# 5. DESCRIÇÃO DOS REQUISITOS

**5.1 REQUISITOS FUNCIONAIS:**

Os seguintes requisitos funcionais foram encontrados após o levantamento de requisitos inicial do Sistema da <Nome da Igreja>

* **RF01:** Sistema deve implementar criptografia nos campos de senhas para garantir a segurança dos dados dos usuários.
* **RF02:** Sistema deve permitir o cadastro de usuários da igreja com informações detalhadas, como: nome, endereço, CPF e informações de contato.
* **RF03:** Administradores devem poder aprovar ou rejeitar os agendamentos feitos pelos membros da igreja.
* **RF04:** Administradores devem conseguir atualizar as informações cadastrais dos membros e excluir usuários da igreja quando necessário.
* **RF05:** Deve ser possível cadastrar avisos de fácil visualização para os membros da igreja.
* **RF06:** Controle de acesso permitindo diferentes níveis de permissões para membros e administradores..
* **RF07:** Avisos devem ser atualizáveis, permitindo modificações e exclusões quando não forem mais relevantes.
* **RF08:** Sistema deve permitir ao anexo de documentos necessários para agendamentos de eventos.

**5.2 REQUISITOS NÃO FUNCIONAIS**

* **RNF01:** Desempenho
  + Tempo de resposta de transação médio inferior a 2 segundos para todas as operações.
  + Taxa de transferência mínima de 10 transações por segundo.
* **RNF02:** Sistema Inoperante
  + Se houver falha no servidor, o sistema deve entrar em modo de leitura, permitindo apenas visualização de dados até que o servidor seja restaurado.
* **RNF03:** Necessidade Computacional
  + O sistema deve utilizar menos de 500 MB de memória RAM durante a operação normal.
  + O espaço em disco necessário para armazenamento de dados e logs não deve exceder 1 GB.
* **RNF04:** Treinamento
  + 2 horas para funcionários administrativos (pré-requisito: conhecimentos básicos de informática).
  + Ajuda on-line: será disponibilizado um tutorial básico com as principais funcionalidades do sistema.

**5.3 REGRAS DE NEGÓCIO**

* **RN01:** Pré-requisito para acesso às funcionalidades do sistema: estar conectado ao sistema utilizando o usuário cadastrado e senha.
* **RN02:** Pré-requisito para agendamento de eventos: o usuário deve ser identificado como membro ou administrador.
* **RN03:** Disponibilidade de agendamento de eventos: um agendamento de evento só poderá acontecer se houver horário disponível para a data selecionada.

**6. PRODUCT BACKLOG INICIAL**

O Product Backlog inicial consiste em uma lista de tarefas planejadas para o projeto, sujeita a alterações e evolução ao longo do tempo.

1. Configuração do Ambiente de Desenvolvimento
2. Análise de Requisitos
3. Estruturação da Arquitetura do Sistema
4. Implementação do Cadastro de Usuários
5. Desenvolvimento do Controle de Acesso
6. Criação do Módulo de Avisos
7. Integração com Bancos de Dados
8. Implementação do Agendamento de Eventos
9. Implementação da Segurança dos Dados
10. Implementação de anexo Digital
11. Testes de Unidade e Integração

**6.1. Divisão de Sprints ou Gantt**

As tarefas do projeto serão divididas em sprints para um melhor planejamento e acompanhamento do progresso. Abaixo está a divisão das tarefas em duas sprints iniciais:

**Sprint 1 (Semana 1-4)**

* Configuração do Ambiente de Desenvolvimento
* Análise de Requisitos
* Estruturação da Arquitetura do Sistema

**Sprint 2 (Semana 4-8)**

* Implementação do Cadastro de Usuários
* Desenvolvimento do Controle de Acesso
* Criação do Módulo de Avisos

Após a conclusão dessas duas sprints iniciais, novas tarefas serão adicionadas ao backlog de acordo com as necessidades e prioridades identificadas durante o desenvolvimento.

**6.2. Histórias de Usuário**

1. **Configuração do Ambiente de Desenvolvimento:** Como desenvolvedor, quero configurar o ambiente de desenvolvimento com as ferramentas necessárias para garantir um fluxo de trabalho eficiente desde o início do projeto.
2. **Análise de Requisitos:** Como analista de requisitos, quero realizar uma revisão detalhada dos requisitos funcionais e não funcionais do sistema da igreja para garantir que todas as necessidades dos usuários e do negócio sejam atendidas de forma adequada.
3. **Estruturação da Arquitetura do Sistema:** Como arquiteto de software, quero definir a arquitetura em camadas e o estilo arquitetural REST para o sistema da igreja, para garantir uma estrutura sólida e escalável que atenda aos requisitos de segurança e desempenho.
4. **Implementação do Cadastro de Usuários:** Como administrador, quero cadastrar novos membros da igreja com informações detalhadas, como nome, endereço, CPF e informações de contato, para manter um registro atualizado e seguro dos membros.
5. **Desenvolvimento do Controle de Acesso:** Como administrador, quero implementar um sistema de controle de acesso com diferentes níveis de permissões para membros e administradores, para garantir a segurança e privacidade dos dados dos usuários.
6. **Criação do Módulo de Avisos:** Como administrador, quero cadastrar avisos de fácil visualização para os membros da igreja, permitindo a atualização e exclusão dos avisos quando necessário, para manter os membros informados sobre eventos e comunicados importantes.
7. **Integração com Bancos de Dados:** Como desenvolvedor, quero integrar o sistema com um banco de dados para armazenamento seguro e eficiente das informações dos usuários e dos avisos da igreja, garantindo a integridade e disponibilidade dos dados.
8. **Implementação do Agendamento de Eventos:** Como membro da igreja, quero agendar eventos online, como casamentos e batismos, para facilitar o processo de marcação de eventos e evitar deslocamentos desnecessários até a igreja.
9. **Implementação da Segurança dos Dados:** Como desenvolvedor, quero implementar criptografia nos campos sensíveis, como senhas, para garantir a segurança e proteção dos dados dos usuários contra acessos não autorizados.
10. **Implementação da Digitalização de Documentos:** Como administrador, quero permitir a digitalização de documentos necessários para agendamentos, para validar legalmente os eventos e garantir a segurança das transações.
11. **Testes de Unidade e Integração:** Como testador, quero realizar testes de unidade e integração para garantir o funcionamento correto das funcionalidades desenvolvidas no sistema da igreja, identificando e corrigindo possíveis erros ou falhas.

**6.3. BDD**

1. **Cadastro de Novo Usuário:**

* **Dado** que o usuário está na página de cadastro.
* **Quando** o usuário preenche corretamente todos os campos obrigatórios.
* **Então** o sistema deve permitir o cadastro do novo usuário e exibir uma mensagem de confirmação.

1. **Login de Usuário Registrado:**

* **Dado** que o usuário está na página de login.
* **Quando** o usuário insere suas credenciais corretas (e-mail e senha).
* **Então** o sistema deve permitir o acesso à área restrita e redirecionar o usuário para a página principal.

1. **Cadastro de Novo Aviso:**

* **Dado** que o usuário está logado como administrador.
* **Quando** o usuário preenche o formulário de cadastro de aviso.
* **Então** o sistema deve permitir o cadastro do aviso e exibir o aviso na página principal para os membros**.**

1. **Agendamento de Evento:**

* **Dado** que o usuário está logado como membro.
* **Quando** o usuário acessa a página de agendamento de eventos e seleciona a data e horário desejados para o evento.
* **Então** o sistema deve confirmar o agendamento do evento e exibir o evento na agenda do usuário.

1. **Controle de Acesso Administrativo:**

* **Dado** que o usuário está logado como administrador.
* **Quando** o administrador tenta acessar uma funcionalidade restrita.
* **Então** o sistema deve permitir o acesso **sem restrições e registrar a ação de**

**7. ARQUITETURA DE SOFTWARE**

A arquitetura mais adequada para o sistema da igreja, considerando os requisitos apresentados, é a Arquitetura em Camadas (Layered Architecture).

**Camada de Apresentação (Interface do Usuário):**

Responsável pela interação com o usuário. Inclui interfaces para cadastro, agendamento, listagem, controle de acesso, exibição de avisos e assinatura digital de documentos.

**Camada de Aplicação:**

Gerencia a lógica de negócios e os processos do sistema. Responsável por validar os dados, controlar acesso, aprovações, exclusões e atualizações. Implementa as regras de negócio (RN01, RN02, RN03).

**Camada de Persistência de Dados:**

Armazena os dados de forma segura e eficiente. Implementa a criptografia para senhas (RF01). Garante integridade e disponibilidade dos dados.

**Camada de Serviços:**

Gerencia a comunicação entre as camadas e outros sistemas externos. Implementoperações offline limitadas (RNF02). Coordena a troca de informações com dispositivos clientes

**7.1. Padrão de Projeto e/ou Estilo Arquitetural**

**Estilo Arquitetural:**

Dentro da Arquitetura em Camadas, um estilo arquitetural complementar que pode ser adotado é o Estilo Arquitetural REST (Representational State Transfer) para as comunicações entre as camadas e serviços externos.

**Descrição do Estilo Arquitetural REST:**

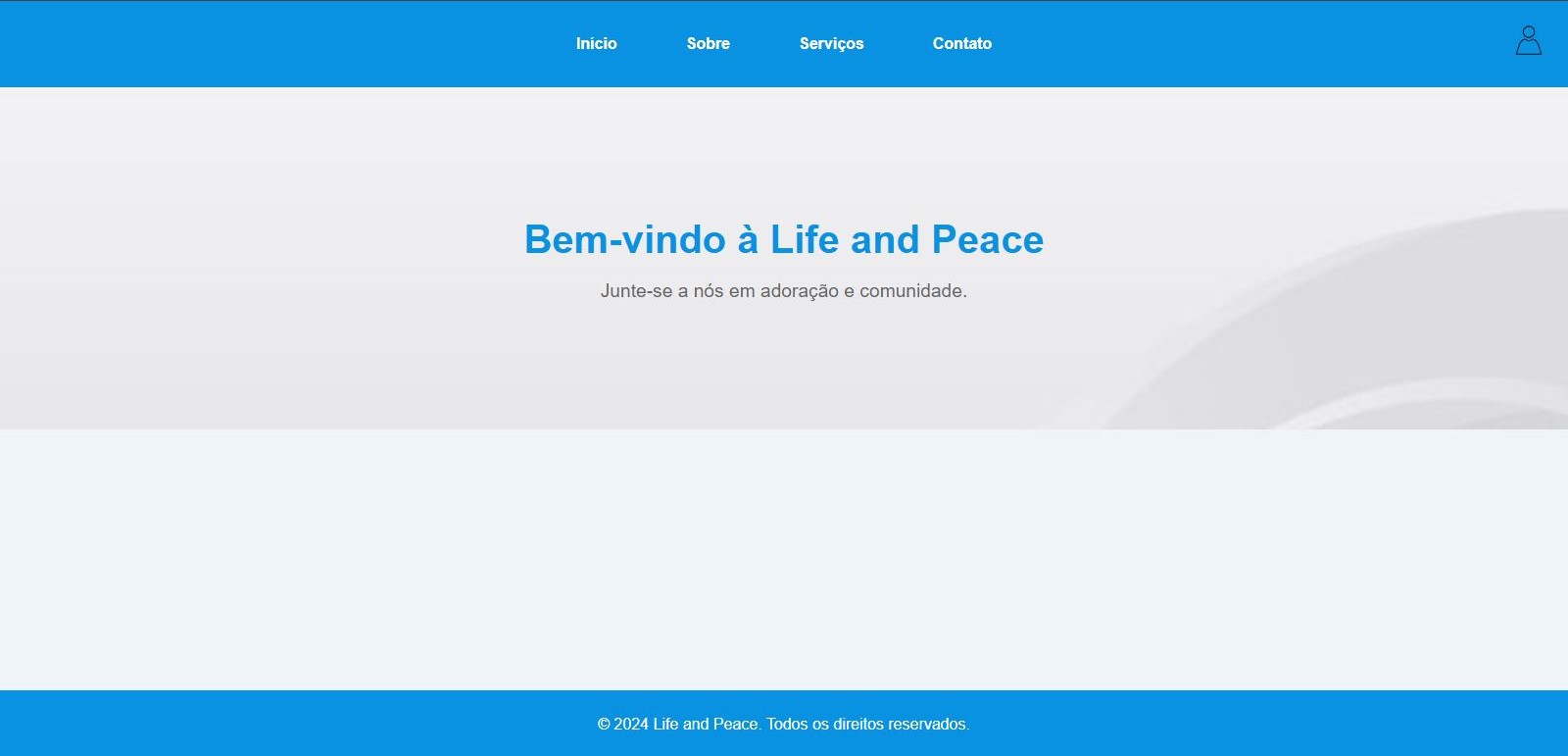
* Utiliza recursos (endpoints) para representar entidades e operações.
* Comunicação baseada em protocolo HTTP.
* Fácil integração com sistemas web e dispositivos móveis.
* Suporta as operações CRUD (Create, Read, Update, Delete) necessárias para o sistema da igreja.

Essa arquitetura em camadas, combinada com o estilo arquitetural REST, permite uma separação clara de responsabilidades, facilita a manutenção, escalabilidade e integração com outros sistemas, além de atender aos requisitos funcionais, não funcionais e regras de negócio identificados para o sistema da igreja.

**8.** [**Design**](#_qsh70q) **centrado no usuário**

O princípio da UX no projeto visa garantir que a experiência do usuário seja intuitiva, agradável e eficiente em todas as interações com o sistema. Isso inclui a criação de interfaces amigáveis, navegação intuitiva e design responsivo para diferentes dispositivos.

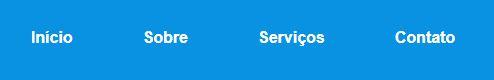
Figura 3



Fonte: Projeto Life and Peace

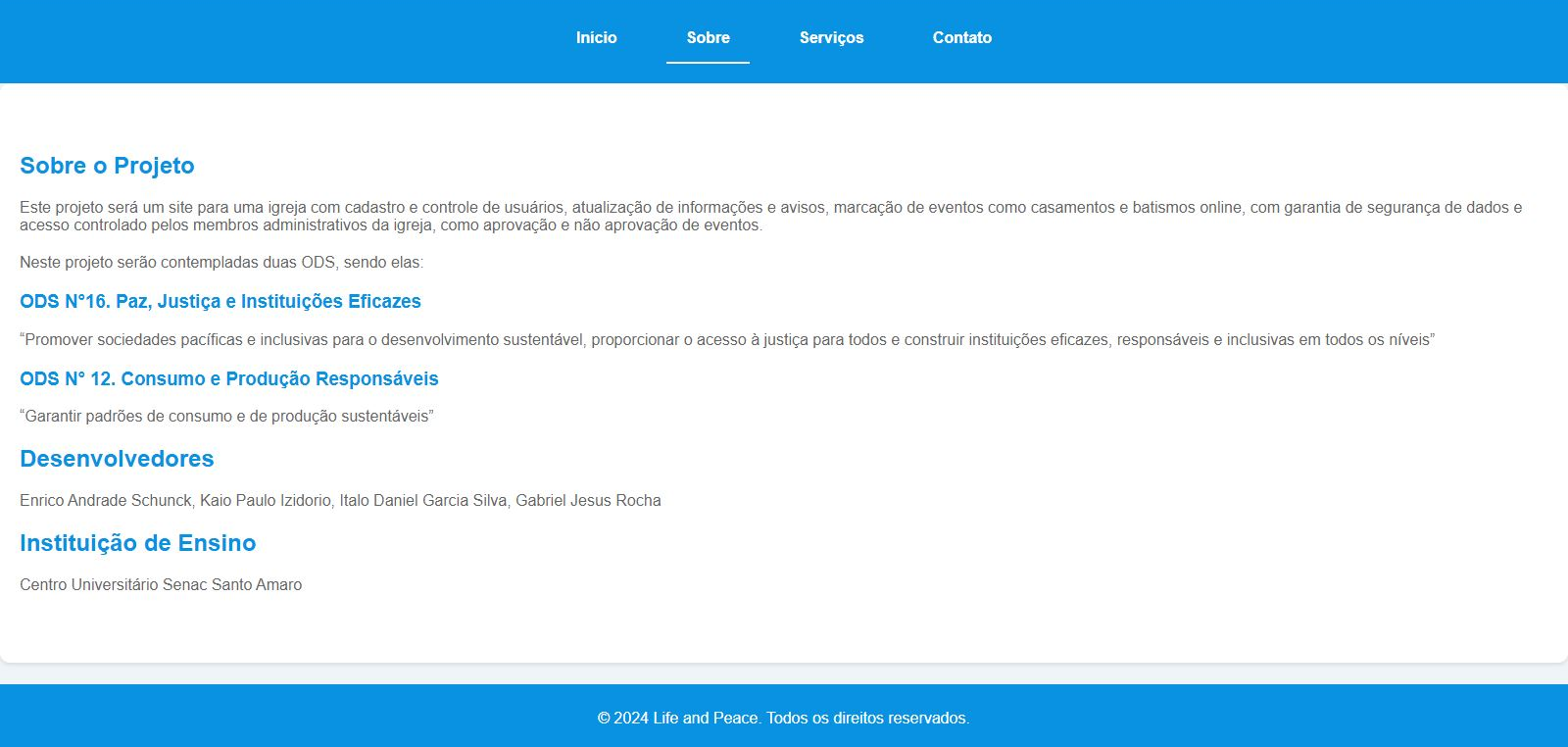
Acima consta o exemplo da tela de início do sistema, incluindo na parte superior da aba todas as funcionalidades do sistema.

Figura 4



Fonte: Projeto Life and Peace

Foi utilizado uma paleta de cores agradável baseando se no intuito de ser um sistema para uma igreja.

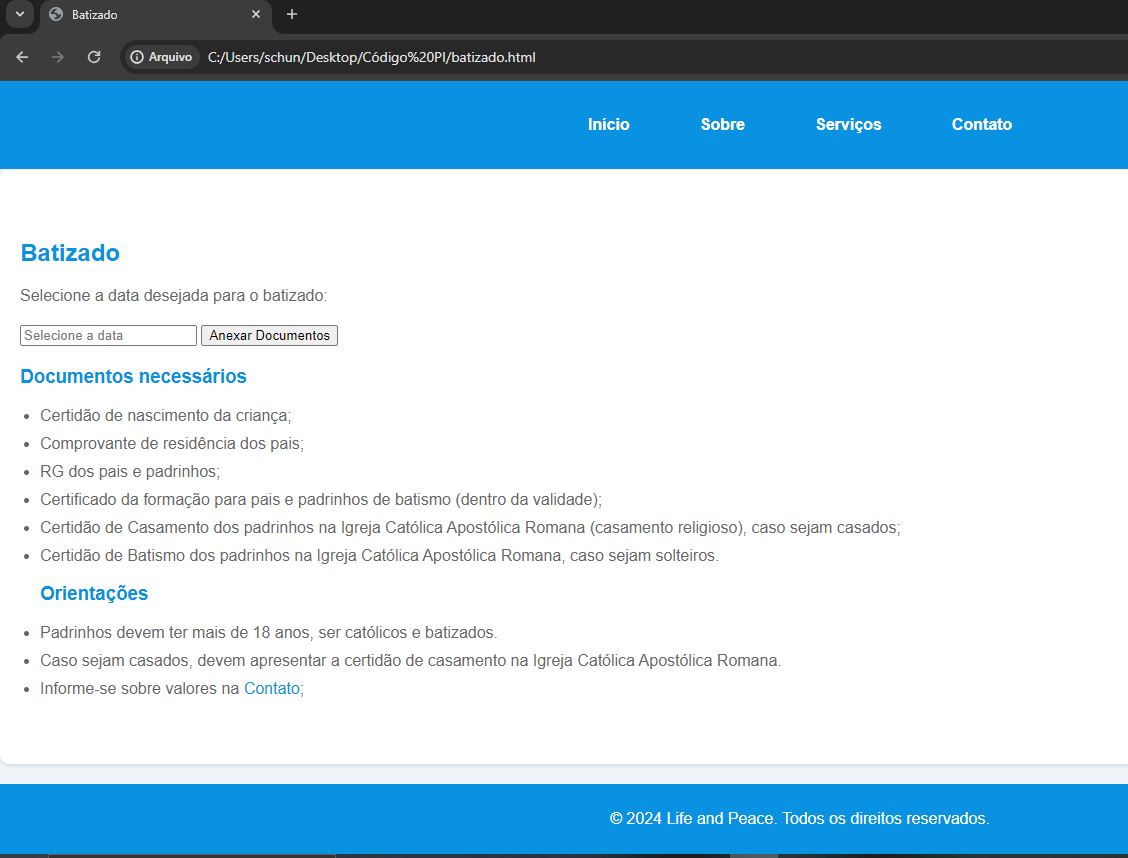


Fonte: Projeto Life and Peace

Figura 5

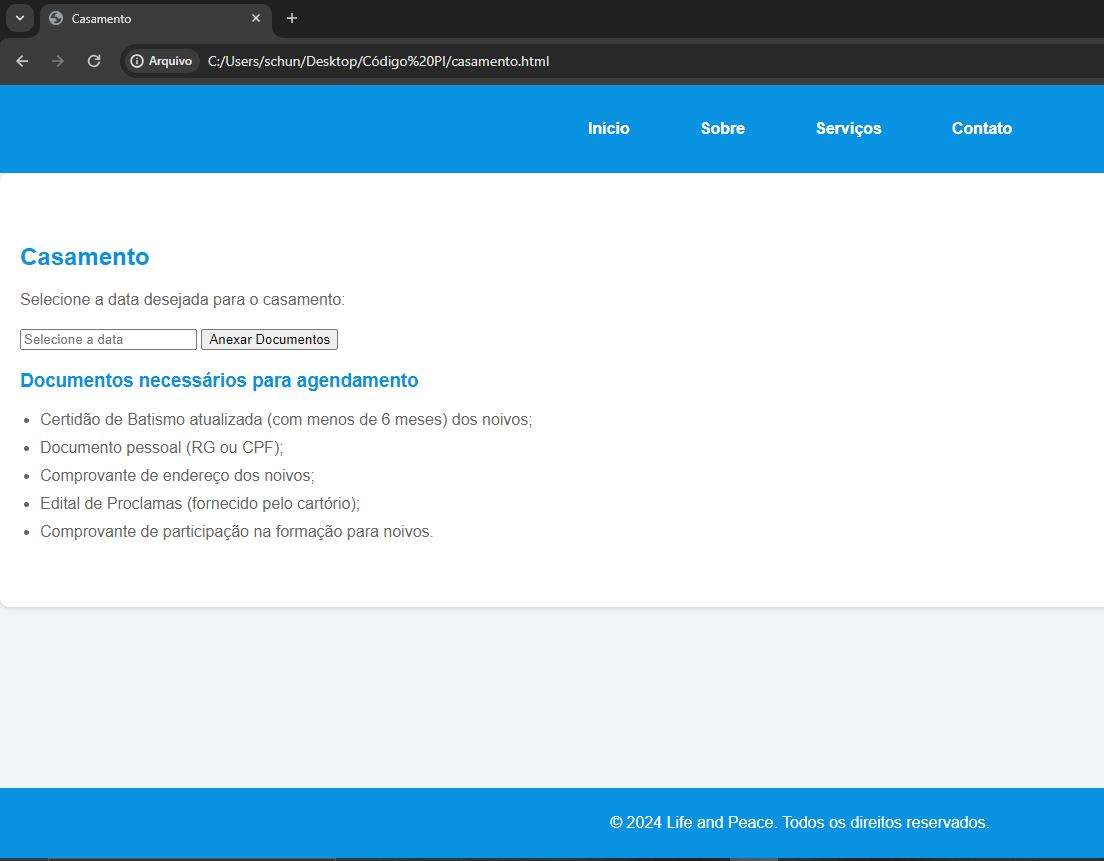
A tela de sobre foi desenvolvida a fim do usuário ter acesso as informações da igreja e o desenvolvimento de tal, este é um modelo protótipo com um texto de exemplo a fim de mostrar a funcionalidade desta janela.

Figura 6



Fonte: Projeto Life and Peace

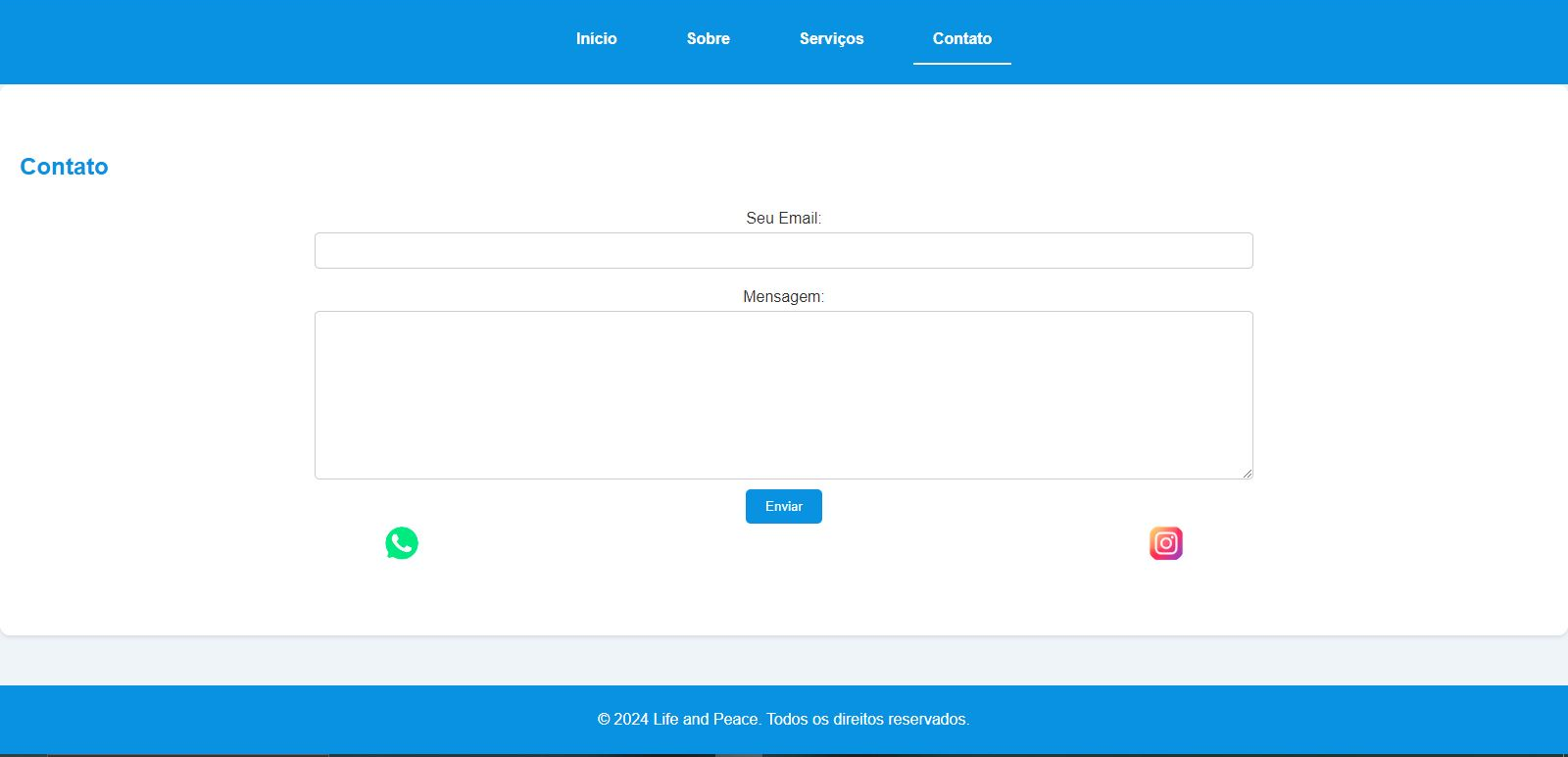
Figura 7



Fonte: Projeto Life and Peace

Acima temos as janelas de casamento e batizado, semelhantes por serem funcionalidades que exigem basicamente dos mesmos critérios, com as orientações , calendário para seleção de data e a opção de anexar os documentos.

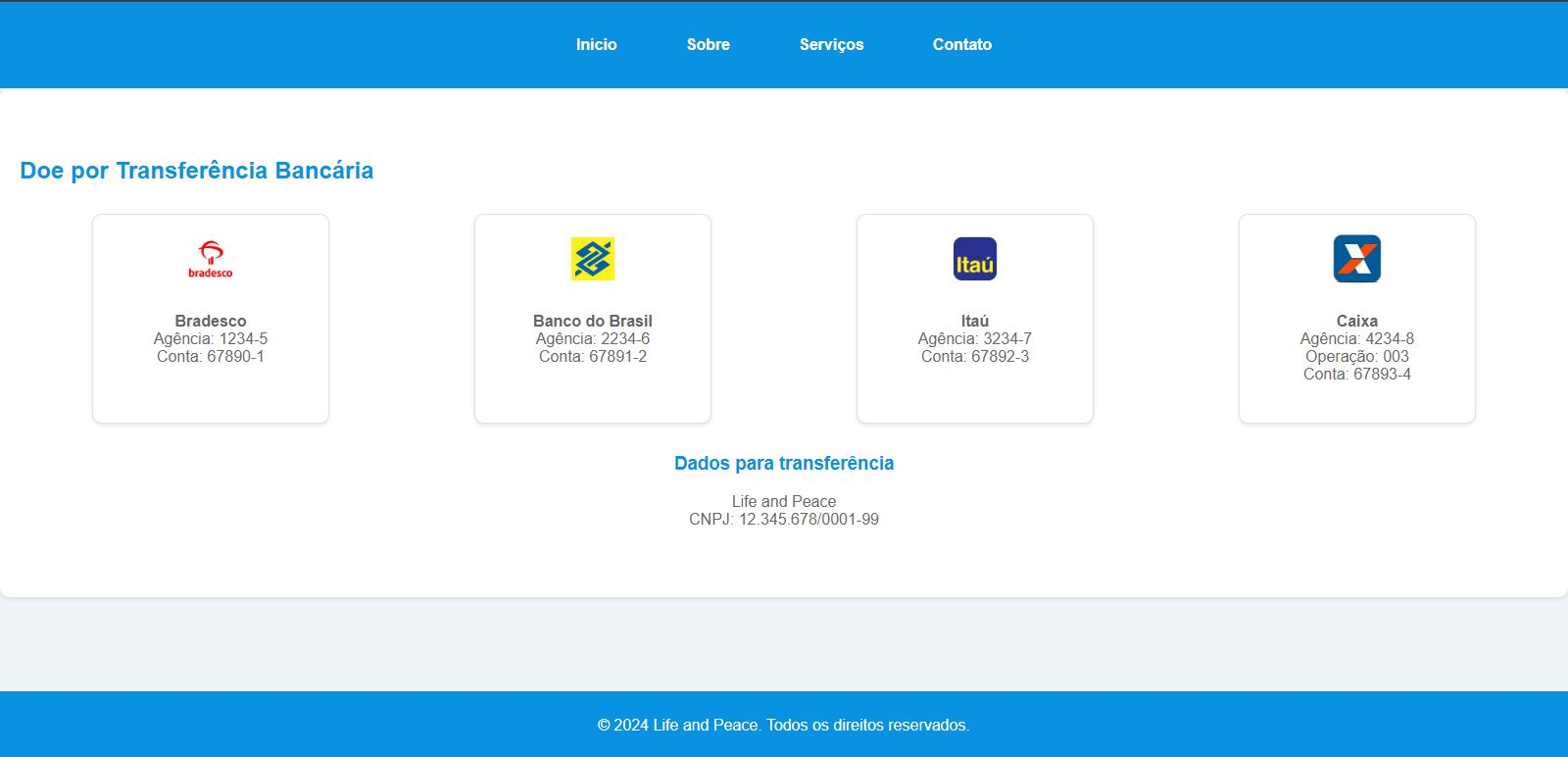
Figura 8



Fonte: Projeto Life and Peace

Este é o exemplo da tela de contato, onde será possivel encaminhar um ticket através de um email, e tambem a comunicação via whatsapp e instagram da igreja.

Figura 9



Fonte: Projeto Life and Peace

Por fim a tela de Doações Online, onde terão os dados para que o usuário faça doações para a igreja através de transferência bancaria ou pix, dados no protótipo são fictícios a fim apenas de demonstrar a tela.

**8.1. Mapa de Empatia**

* **O que o usuário pensa e sente:** Os usuários desejam um sistema fácil de usar que os ajude a se conectar com a igreja de forma rápida e eficiente. Querem sentir-se seguros ao inserir informações pessoais e realizar transações online.
* **O que o usuário vê:** Eles veem a necessidade de um sistema que ofereça informações claras sobre eventos, avisos e serviços da igreja. Esperam uma interface limpa e organizada.
* **O que o usuário ouve:** Os usuários ouvem falar da importância da segurança dos dados online e da facilidade de uso de sistemas similares em outras instituições religiosas.
* **O que o usuário diz e faz:** Eles expressam o desejo de realizar agendamentos online, receber avisos importantes e acessar informações atualizadas sobre a igreja. Eles agem buscando um sistema conveniente e confiável.

**8.2.** [**PERSONAS / Protopersonas**](#_3as4poj)

1. **Ana**, a Administradora da Igreja:

* **Idade**: 45 anos
* **Profissão**: Administradora da Igreja
* **Necessidades**: Aprovar agendamentos, gerenciar usuários, atualizar informações da igreja.
* **Comportamento**: Acessa o sistema regularmente para realizar tarefas administrativas.

1. **João**, o Membro Ativo da Congregação:

* **Idade**: 30 anos
* **Profissão**: Advogado
* **Necessidades**: Agendar eventos, receber avisos, acessar informações sobre a igreja.
* **Comportamento**: Utiliza o sistema para se manter atualizado sobre eventos e participar das atividades da igreja.

1. **Maria**, a Voluntária Engajada:

* **Idade**: 60 anos
* **Profissão**: Aposentada
* **Necessidades**: Ajudar na organização de eventos, acessar informações sobre voluntariado.
* **Comportamento**: Interage com o sistema para colaborar com a igreja em suas atividades voluntárias.

Essas personas representam diferentes perfis de usuários e suas necessidades específicas ao utilizar o sistema da igreja. Isso ajuda a direcionar o design e as funcionalidades do sistema de acordo com as expectativas e comportamentos dos usuários reais.

**9. MODELO DE CASOS DE USO**

Para compreender melhor as necessidades da igreja e o funcionamento de sistemas semelhantes, foram realizados estudos de casos de outras instituições religiosas que utilizam sistemas de gestão. Isso permitiu identificar boas práticas e funcionalidades relevantes a serem incorporadas ao projeto.

**9.1 IDENTIFICAÇÃO DOS ATORES E SUAS RESPONSABILIDADES**

**Membro**:

* Responsabilidades:
  + Efetuar o cadastro com suas informações pessoais.
  + Solicitar agendamentos de eventos da igreja.
  + Acessar informações gerais disponibilizadas pela igreja.

**Membro Administrativo**:

* Responsabilidades:
  + Gerenciar os agendamentos de eventos da igreja, incluindo adição, atualização e aprovação.
  + Gerenciar os avisos da igreja, incluindo adição, atualização e remoção.

**Sistema**:

* Responsabilidades:
  + Armazenar e validar as informações dos membros da igreja.
  + Armazenar e validar os agendamentos de eventos da igreja.
  + Armazenar e validar os avisos da igreja.
  + Exibir informações relevantes para os membros.
  + Permitir a interação entre membros e membros administrativos.

**9.2 DEFINIÇÃO DE PRIORIDADE DE DESENVOLVIMENTO DOS CASOS DE USO**

Abaixo estão informados os casos de uso identificados e a prioridades para o desenvolvimento de cada um deles:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Número*** | ***Nome do caso de uso*** | ***Prioridade*** | ***Justificativa*** | ***Aluno Responsável*** |
| *UC001* | *Manter Membro* | *Alta* | *Gerenciar informações dos membros* | *Enrico, Gabriel* |
| *UC002* | *Manter Agendamento* | *Alta* | *Gerenciar agendamentos* | *Enrico, Gabriel* |
| *UC003* | *Manter Avisos* | *Alta* | *Gerenciar Avisos* | *Italo, Kaio* |

* 1. **DESCRIÇÃO DETALHADA DOS CASOS DE USO:**

**UC001 – Manter Membro**

* Breve Descrição: Permite aos membros administrativos gerenciar as informações dos membros da igreja, incluindo aprovação, atualização e exclusão de registros.
* Atores Principais:
  + **Membro:** Efetua o cadastro com suas informações pessoais.
* Pré-condições: O membro administrativo está autenticado no sistema.
* Pós-condição: As informações dos membros estão atualizadas ou membros foram removidos do sistema.
* Fluxo de Eventos:
  + **Fluxo Básico:**
    - O membro efetua seu cadastro com suas informações.
    - O sistema valida e armazena as informações no banco de dados.
    - O sistema valida e atualiza as informações.
    - O administrador remove membros da igreja, confirma a exclusão e o sistema remove o membro.
  + **Fluxos Alternativos:**
    - **Dados de cadastro incompletos:** Se o membro deixar campos obrigatórios em branco, o sistema exibe uma mensagem de erro e solicita que esses campos sejam preenchidos.
    - **Cancelar o cadastro:** Em qualquer momento antes de confirmar o cadastro, o membro pode optar por cancelar a operação. O sistema interrompe o processo de cadastro e retorna ao menu principal.

**UC002 – Manter Agendamento**

* **Breve Descrição:** Permite aos membros administrativos gerenciar os agendamentos de eventos da igreja, incluindo atualização e aprovação de agendamentos feitos pelos membros, além de incluir a assinatura digital dos documentos necessários para o agendamento..
* Atores Principais:
  + **Membro:** Efetua a solicitação.
  + **Membro Administrativo:** Interage com o sistema para adicionar, atualizar ou aprovar agendamentos feitos pelos membros.
* Pré-condições: O membro administrativo está autenticado no sistema.
* **Pós-condição:** Horários solicitados devem ficar com status de indisponível. Documentos necessários devem ser assinados digitalmente..
* Fluxo de Eventos:
  + **Fluxo Básico:**
  + O membro acessa a página de agendamento de eventos.
  + O membro preenche os detalhes do evento, como data, hora e tipo de evento.
  + O membro anexa os documentos necessários para o evento (por exemplo, termos de responsabilidade, contratos, etc.).
  + O sistema verifica se todos os campos obrigatórios estão preenchidos e se os documentos necessários foram anexados.
  + O sistema verifica a validade dos documentos.
  + O sistema confirma o recebimento dos documentos assinados e envia a solicitação de agendamento para o administrador.
  + O sistema exibe os detalhes dos agendamentos e os documentos anexados.
  + O administrador aprova ou rejeita os agendamentos:
  + Se rejeitado, o sistema notifica o membro e permite a re-submissão dos documentos, se necessário.
  + Caso de uso encerrado.
* **Fluxos Alternativos:**
  + **Dados de solicitação incompletos:**

Se o membro deixar campos obrigatórios em branco, o sistema exibe uma mensagem de erro e solicita que esses campos sejam preenchidos. O caso de uso retorna ao passo do formulário para correção.

* + **Documentos não anexados ou assinatura digital incompleta:**

Se o membro não anexar os documentos necessários ou não completar a assinatura digital, o sistema exibe uma mensagem de erro e solicita que esses passos sejam concluídos antes de prosseguir.

* + **Cancelar solicitação:**

Em qualquer momento antes de confirmar a solicitação, o membro pode optar por cancelar a operação. O sistema interrompe o processo de cadastro e retorna ao menu principal.

**UC003 – Manter Avisos**

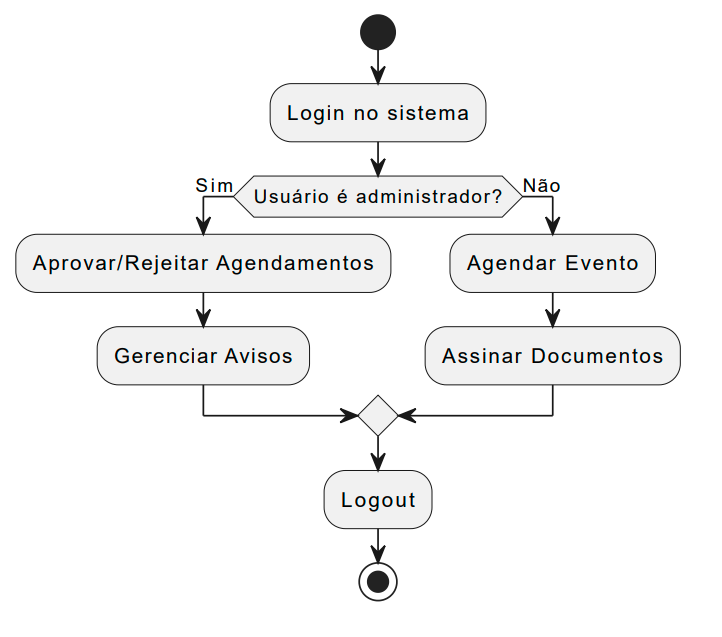
* Breve Descrição: Permite aos membros administrativos gerenciar os avisos da igreja, incluindo adição, atualização e remoção de avisos.
* Atores Principais:
  + **Membro Administrativo:** Interage com o sistema para adicionar, atualizar ou excluir avisos da igreja.
* Pré-condições: O membro administrativo está autenticado no sistema.
* Pós-condição: Os avisos devem ser atualizados para visualização imediatamente.
* Fluxo de Eventos:
  + **Fluxo Básico:**
    - O administrador acessa a funcionalidade de gerenciamento de avisos.
    - O administrador cadastra novos avisos com título e conteúdo.
    - O sistema valida e armazena os avisos.
    - O administrador atualiza títulos ou conteúdos dos avisos existentes.
    - O sistema valida e atualiza os avisos.
    - O administrador remove avisos, confirma a exclusão e o sistema remove o aviso.
  + **Fluxos Alternativos:**
    - **Dados de avisos incompletos:** Se o administrador deixar campos obrigatórios em branco, o sistema exibe uma mensagem de erro e solicita que esses campos sejam preenchidos.
    - **Cancelar o cadastro:** Em qualquer momento antes de confirmar o cadastro, o administrador pode optar por cancelar a operação. O sistema interrompe o processo de cadastro e retorna ao menu principal.

**Diagrama de Atividade:**

Representa o fluxo de atividades dentro do sistema, desde o login até a gestão de eventos e avisos.

Destaca as decisões e ações tomadas pelos diferentes atores.

Figura 10



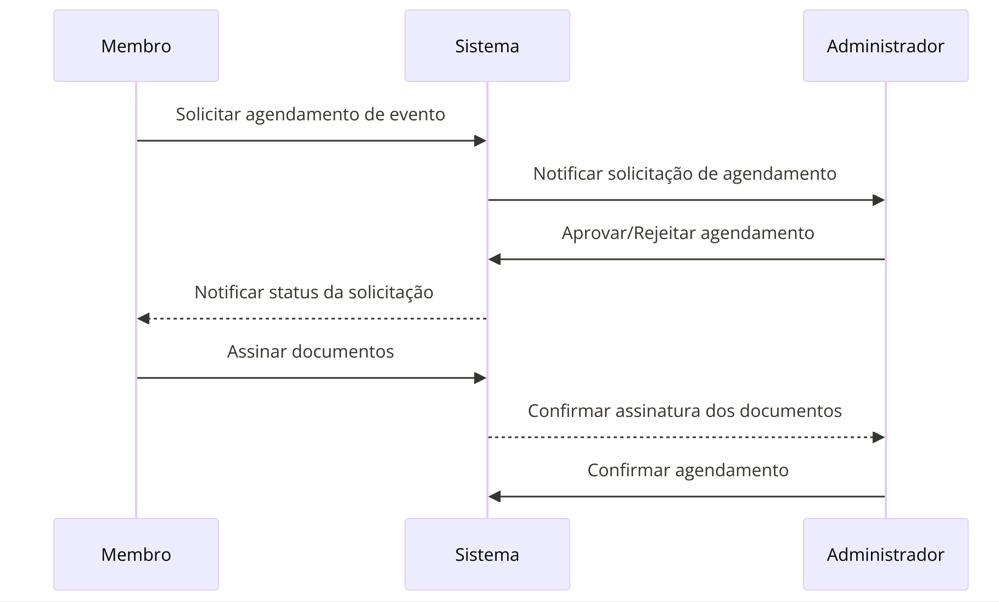
Fonte: www.lucidchart.com

**Diagrama de Sequência:**

Ilustra a interação entre os membros, sistema e administradores durante o processo de agendamento de eventos e assinatura de documentos.

Mostra a ordem das mensagens trocadas para completar uma ação.

Figura 11



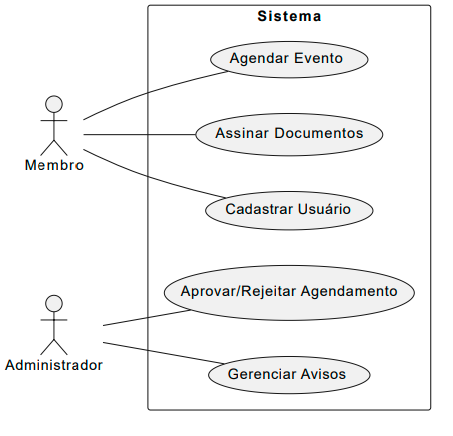
Fonte: www.lucidchart.com

**Diagrama de Caso de Uso:**

Identifica os principais casos de uso do sistema, como Cadastrar Usuário, Agendar Evento, Aprovar/Rejeitar Agendamento, Gerenciar Avisos, Assinar Documentos.

Define as interações entre os atores (Membro, Administrador, Sistema) e os casos de uso.

Figura 12



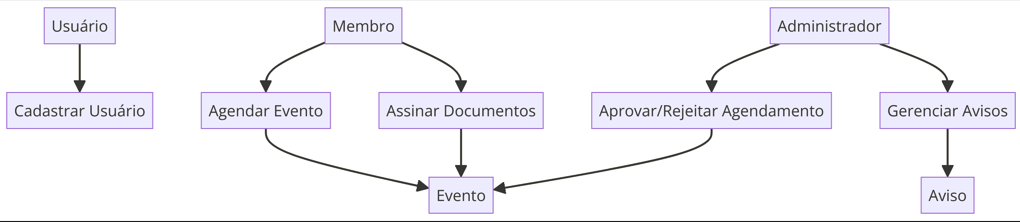
Fonte: www.lucidchart.com

**Diagrama de Processo de Negócio:**

Descreve o fluxo de atividades no processo de agendamento de eventos, assinatura de documentos e gestão de avisos.

Mostra as atividades realizadas pelos membros e administradores para concluir um processo.

Figura 13



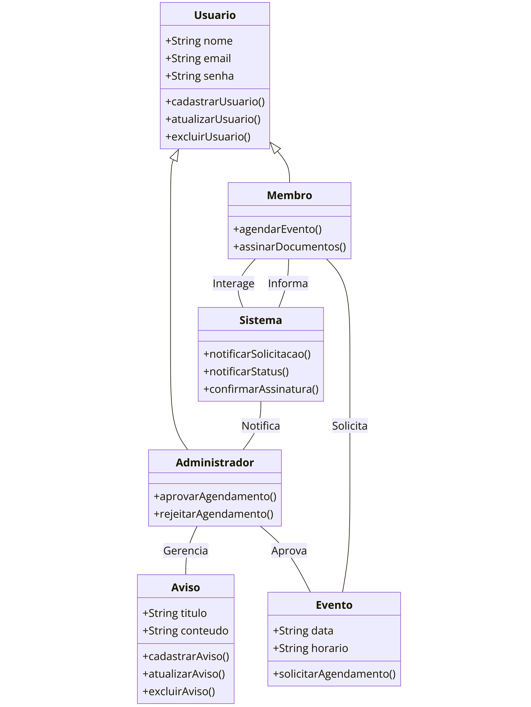
Fonte: www.lucidchart.com

**Diagrama UML:**

Combina diversos aspectos do sistema em um único diagrama, oferecendo uma visão holística das classes, interações e processos.

Inclui todos os elementos discutidos nos diagramas anteriores, proporcionando uma visão integrada do sistema.

Figura 14



Fonte: www.lucidchart.com

**Metodologia:**

**Análise de Requisitos** Realizamos uma análise detalhada dos requisitos funcionais e não funcionais, levantando as necessidades dos usuários e administradores da igreja. Entrevistamos stakeholders para garantir que todas as funcionalidades necessárias fossem contempladas.

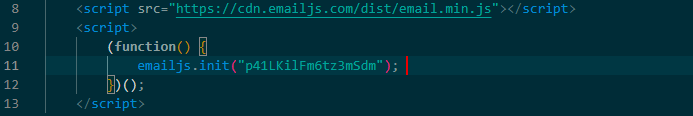
**Desenvolvimento em Sprints** Utilizamos a metodologia ágil, dividindo o desenvolvimento em sprints curtas para garantir entregas contínuas e frequentes. Cada sprint foi planejada para abordar um conjunto específico de funcionalidades, permitindo ajustes rápidos com base no feedback dos usuários.

**Arquitetura em Camadas** Optamos por uma arquitetura em camadas para separar claramente as responsabilidades dentro do sistema, facilitando a manutenção e escalabilidade. Cada camada tem uma função específica, desde a interface do usuário até a persistência dos dados.

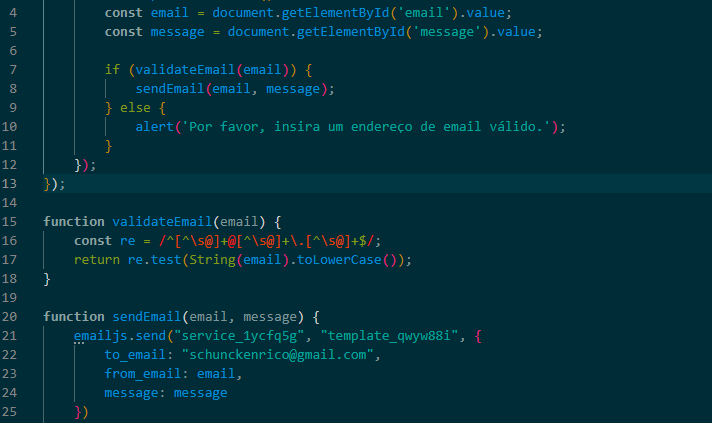
**Descrição das APIs Utilizadas**: Essas APIs são usadas para facilitar a comunicação entre os membros da igreja e os administradores, bem como para melhorar a usabilidade do site através de widgets interativos e integração com redes sociais.

* **EmailJS API** Funcionalidade: O EmailJS é utilizado para enviar emails diretamente do JavaScript sem necessidade de configurar um servidor de email. No sistema da igreja, ele é usado para enviar formulários de contato preenchidos pelos usuários diretamente para um endereço de email especificado. Trecho Utilizado:

Figura 15



Fonte: VsCode

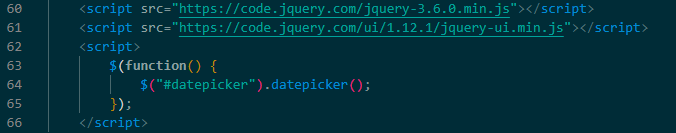


Fonte: VsCode

Figura 16

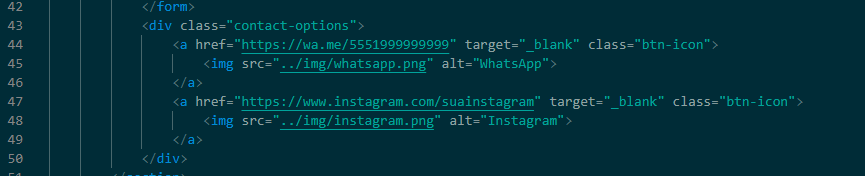
* **jQuery UI Datepicker** Funcionalidade: O jQuery UI Datepicker é utilizado para fornecer um widget de seleção de data interativo. No sistema da igreja, ele permite que os usuários selecionem datas para batizados e casamentos de forma visual e intuitiva. Trecho Utilizado:

Figura 17



Fonte: VsCode

* **API de Comunicação com Redes Sociais** Funcionalidade: A comunicação com redes sociais é realizada através de links diretos para serviços de WhatsApp e Instagram, facilitando a interação dos usuários com as plataformas sociais. Trechos Utilizados:



Fonte:VsCode

Figura 18

**Projetos Futuros:**

Como projetos futuros, pretende-se implementar funcionalidades adicionais ao sistema, como integração com sistemas de pagamento online para doações e eventos, criação de um aplicativo móvel para facilitar o acesso dos membros e melhorias na interface de usuário, dando continuidade nos projetos desenvolvidos anteriormente.

# Desenvolvimento do Trabalho do Grupo

Este projeto foi desenvolvido por Enrico Andrade, Gabriel Jesus, Italo Daniel e Kaio Paulo, sob orientação do professor Calyton Mendonça Feliciano. A seguir, detalhamos as contribuições de cada membro do grupo:  
  
Enrico Andrade  
 Responsável pelo desenvolvimento do back-end, Enrico enfrentou desafios como a integração com bancos de dados e a implementação de segurança dos dados. Ele utilizou tecnologias como MySQL e criptografia de senhas para garantir a integridade e segurança do sistema.  
  
Gabriel Jesus  
 Gabriel foi responsável pela documentação do projeto e pelo desenvolvimento do back-end. Ele trabalhou na definição dos requisitos, na elaboração dos diagramas UML e na implementação do controle de acesso dos usuários.  
  
Italo Daniel  
 Italo focou no desenvolvimento do front-end, criando interfaces amigáveis e intuitivas. Utilizou HTML, CSS e JavaScript para desenvolver as páginas web, garantindo a responsividade e acessibilidade do sistema.  
  
Kaio Paulo  
 Kaio colaborou no desenvolvimento do front-end e na documentação. Ele foi responsável pela criação do módulo de avisos e pela integração com APIs externas, como o EmailJS e o jQuery UI Datepicker.

**Conclusão:**

O desenvolvimento do sistema de gestão para a igreja Life And Peace demonstrou ser uma solução abrangente e eficaz para otimizar processos administrativos e melhorar a comunicação interna. Este projeto abrangeu tanto o desenvolvimento do código quanto a documentação detalhada, proporcionando uma base sólida e escalável para as futuras necessidades da igreja.

**Desenvolvimento do Código** O código foi desenvolvido utilizando uma abordagem modular e escalável, baseada na arquitetura em camadas. Isso permitiu uma separação clara de responsabilidades, facilitando a manutenção e expansão do sistema. As principais etapas do desenvolvimento incluíram:

1. Configuração do Ambiente de Desenvolvimento: Estabelecemos um ambiente de desenvolvimento robusto com as ferramentas necessárias para suportar o ciclo de vida do projeto.
2. Implementação das Funcionalidades: Desenvolvemos funcionalidades essenciais, como cadastro de usuários, controle de acesso, gestão de avisos e agendamento de eventos. Cada funcionalidade foi implementada com foco na usabilidade e segurança.
3. Integração com APIs: Integramos APIs como EmailJS para envio de emails e jQuery UI Datepicker para seleção de datas, melhorando a interatividade e eficiência do sistema.
4. Segurança de Dados: Implementamos criptografia nos campos sensíveis, como senhas, para garantir a segurança e proteção dos dados dos usuários.
5. Digitalização: Adicionamos a funcionalidade de digitalizar anexos para validar legalmente os documentos necessários para agendamentos de eventos.

**Desenvolvimento da Documentação** A documentação foi elaborada com o objetivo de fornecer uma visão completa e clara do sistema, facilitando a compreensão e manutenção futura. As principais seções da documentação incluem:

1. Análise de Requisitos: Detalhamos os requisitos funcionais e não funcionais do sistema, baseados nas necessidades identificadas durante a fase de levantamento de requisitos.
2. Modelo de Casos de Uso: Identificamos e descrevemos detalhadamente os principais casos de uso, definindo os atores envolvidos e suas responsabilidades.
3. Arquitetura de Software: Documentamos a arquitetura em camadas adotada, explicando como cada camada contribui para a funcionalidade geral do sistema.
4. Metodologia de Desenvolvimento: Utilizamos a metodologia ágil, dividindo o desenvolvimento em sprints, permitindo entregas contínuas e frequentes, e possibilitando ajustes rápidos com base no feedback dos usuários.
5. Descrições de API: Incluímos descrições acadêmicas das APIs utilizadas, explicando suas funcionalidades e como foram integradas no sistema.

**Benefícios e Impacto** O sistema contribui significativamente para a organização e eficiência das atividades da igreja, proporcionando uma experiência mais positiva para membros e líderes religiosos. As principais melhorias incluem:

1. Eficiência Administrativa: Automatização de processos administrativos, como cadastro de usuários, agendamentos e gestão de avisos, reduzindo a carga de trabalho manual.
2. Melhoria na Comunicação: Centralização das informações e avisos, facilitando a comunicação entre a administração e os membros da igreja.
3. Acessibilidade e Inclusão: Implementação de uma interface amigável e intuitiva, com funcionalidades de acessibilidade, garantindo que todos os membros, independentemente de suas habilidades, possam utilizar o sistema eficazmente.
4. Segurança de Dados: Garantia da proteção dos dados dos usuários através de criptografia e controle de acesso, aumentando a confiança no sistema.

Em resumo, o desenvolvimento do sistema de gestão para a igreja Life And Peace foi um sucesso, atendendo às necessidades identificadas e proporcionando uma plataforma robusta para a administração e comunicação da igreja. A documentação completa assegura que o sistema pode ser mantido e expandido no futuro, continuando a beneficiar a comunidade da igreja.

**Referencias Bibliograficas:**

ABNT NBR ISO/IEC 25010:2011. "Sistema e software de engenharia - Requisitos de qualidade e avaliação de software (SQuaRE) - Modelos de qualidade de produto e de qualidade em uso". Associação Brasileira de Normas Técnicas, 2011.

PRESSMAN, Roger S. "Engenharia de Software: uma abordagem profissional". 8. ed. Porto Alegre: AMGH, 2016.

SOMMERVILLE, Ian. "Engenharia de Software". 10. ed. São Paulo: Pearson, 2011.

BEZERRA, Eduardo. "Qualidade de Software: Aprenda as Metodologias e Ferramentas Mais Utilizadas no Mercado". 2. ed. Rio de Janeiro: Elsevier, 2014.

FIELDING, Roy Thomas. "Architectural Styles and the Design of Network-based Software Architectures". University of California, Irvine, 2000.

BROOKS, Frederick P. "The Mythical Man-Month: Essays on Software Engineering". Anniversary Edition. Addison-Wesley Professional, 1995.

GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. "Design Patterns: Elements of Reusable Object-Oriented Software". Addison-Wesley, 1994.

SCHWABER, Ken; SUTHERLAND, Jeff. "The Scrum Guide". Scrum.org, 2017. Disponível em: <https://scrumguides.org/scrum-guide.html>. Acesso em: 2 jun. 2024.

REACTJS. "React – A JavaScript library for building user interfaces". Disponível em: <https://reactjs.org/>. Acesso em: 2 jun. 2024.

JQUERY. "jQuery UI". Disponível em: <https://jqueryui.com/>. Acesso em: 2 jun. 2024.

EMAILJS. "EmailJS – Send email directly from client-side JavaScript code". Disponível em: <https://www.emailjs.com/>. Acesso em: 2 jun. 2024.

GOOGLE. "Material Design". Disponível em: <https://material.io/design>. Acesso em: 2 jun. 2024.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. "NBR 6023:2018 - Informação e documentação - Referências - Elaboração". Rio de Janeiro: ABNT, 2018.