Documentação Completa do Software IPJ - Igreja Presbiteriana de Joinville

> O **Software IPJ** foi desenvolvido para auxiliar a administração da **Igreja Presbiteriana de Joinville** no gerenciamento eficiente de seus membros. O sistema oferece funcionalidades como cadastro de membros, controle de listas, edição e deleção de registros, geração de relatórios e gestão de sociedades internas. Ele foi projetado para atender às necessidades do pastor e da administração da igreja, proporcionando uma solução prática e intuitiva.

Funcionalidades Principais:

1. **Cadastro de Membros:**
   - Permite a inserção de novos membros com detalhes como nome, data de nascimento, endereço, entre outros.
2. **Lista de Membros:**
   - Exibição de todos os membros registrados, com opções de busca e filtros personalizados.
3. **Edição de Membros:**
   - Permite a atualização dos dados cadastrais dos membros.
4. **Deleção de Membros:**
   - Função para remover registros de membros que não fazem mais parte da congregação.
5. **Relatórios:**
   - Geração de relatórios detalhados sobre os membros, eventos, e participação nas sociedades internas.

Escopo do Projeto

O Software IPJ tem como objetivo principal gerenciar os membros da Igreja Presbiteriana de Joinville, abrangendo:

- Cadastro, edição e exclusão de membros.
- Gestão de sociedades internas.
- Geração de relatórios.
- Controle de listas e filtragem de membros.

Histórias de Usuário

1. Cadastro de Membros

Como pastor da Igreja Presbiteriana de Joinville,
Eu quero cadastrar novos membros da igreja no sistema,
Para que eu possa ter um registro organizado de todos os membros com informações detalhadas.

Critérios de Aceitação:

- O sistema deve permitir a inserção de informações obrigatórias como nome, data de nascimento, endereço, telefone e e-mail.
- O sistema deve validar campos obrigatórios e exibir uma mensagem de erro se algum campo estiver faltando.
- Após o preenchimento do formulário, o sistema deve exibir uma mensagem de confirmação de cadastro bem-sucedido.
- O membro cadastrado deve aparecer automaticamente na lista de membros.

2. Lista de Membros

Como administrador da igreja,
Eu quero visualizar todos os membros cadastrados e aplicar filtros,
Para que eu possa gerenciar a lista de membros de maneira eficiente.

Critérios de Aceitação:

- O sistema deve exibir uma lista completa de todos os membros cadastrados.
- A lista deve ter uma barra de busca que permita encontrar membros pelo nome.
- O sistema deve permitir o uso de filtros personalizados (ex: membros homens/mulheres).
- A lista deve ser atualizada automaticamente após uma busca ou filtro.

3. Edição de Membros

Como pastor ou administrador da igreja,
Eu quero editar as informações de um membro,
Para que eu possa corrigir ou atualizar os dados quando necessário.

Critérios de Aceitação:

- O sistema deve permitir a busca de um membro específico para edição.
- O formulário de edição deve exibir os dados atuais do membro.
- O sistema deve validar os campos editados.
- Ao salvar, as alterações devem ser refletidas imediatamente na lista de membros.

4. Deleção de Membros

Como pastor ou administrador da igreja,
Eu quero excluir membros que não fazem mais parte da congregação,
Para que eu mantenha a lista de membros sempre atualizada.

Critérios de Aceitação:

- O sistema deve permitir a seleção de um membro para exclusão.
- Ao excluir, uma mensagem de confirmação deve ser exibida para o usuário.
- O membro excluído deve ser removido da lista de membros imediatamente após a confirmação.

5. Geração de Relatórios

Como pastor,
Eu quero gerar relatórios sobre os membros e atividades da igreja,
Para que eu possa obter informações detalhadas de maneira rápida e organizada.

Critérios de Aceitação:

- O sistema deve oferecer diferentes tipos de relatórios (ex: lista de aniversariantes, lista de membros ativos, etc.).
- O sistema deve permitir a exportação dos relatórios em formato PDF.
- O relatório gerado deve estar formatado adequadamente para impressão e consulta.

Tecnologias Utilizadas:

- **Front-end:** Flutter, Dart

- **Back-end:** Node.js
- **Banco de Dados:** MySQL
- **Ferramentas de Integração:** Jira para gestão de tarefas e Confluence para documentação

---

Arquitetura do Sistema:

> O sistema segue o padrão **MVC (Model-View-Controller)**, em que:
>
> - **Model:** O banco de dados MySQL armazena todas as informações de membros e sociedades.
> - **View:** O front-end em Flutter proporciona uma interface amigável para os usuários da igreja.
> - **Controller:** O back-end em Node.js gerencia as lógicas de negócios e integra as informações do banco de dados ao front-end.

Listagem de Tasks concluídas (Jira):

Type	Key	Resumo	Responsável	Prioridade	Status	Atualizado(a)
✓	PAC-124	Criar um cache de armazenamento de imagens na t…	JD Joao Paulo Duarte	High	CONCLUÍDO	5 de dez. de 2024, 22
●	PAC-123	Ao cadastrar um membro com o numero de rol já ca…	JD Joao Paulo Duarte	Medium	CONCLUÍDO	3 de dez. de 2024, 17
●	PAC-122	Ao editar um membro com foto, a foto não continua s…	JD Joao Paulo Duarte	High	CONCLUÍDO	2 de dez. de 2024, 16
✓	PAC-121	Ajustes finais do app	JD Joao Paulo Duarte	High	CONCLUÍDO	27 de nov. de 2024, 1
✓	PAC-117	Adicionar um botão de ordenação na tela de membros	JD Joao Paulo Duarte	Low	CONCLUÍDO	26 de nov. de 2024, 2
✓	PAC-116	Melhorar o campo de busca da tela de membros	JD Joao Paulo Duarte	Low	CONCLUÍDO	26 de nov. de 2024, 2
✓	PAC-115	Migração de banco para o Supabase	K kalebefukudadeol…	Highest	CONCLUÍDO	21 de nov. de 2024, 2
●	PAC-112	Criar um botão para remover foto anexada na tela de…		Medium	CONCLUÍDO	26 de nov. de 2024, 1
●	PAC-111	Imagem nao está sendo salva no banco	JD Joao Paulo Duarte	Medium	CONCLUÍDO	25 de nov. de 2024, 1
✓	PAC-110	Adicionar um tema 'modo do sistema' na listagem de…	JD Joao Paulo Duarte	Lowest	CONCLUÍDO	13 de nov. de 2024, 0
✓	PAC-109	Adicionar uma animação de rolagem no widget de co…	JD Joao Paulo Duarte	Lowest	CONCLUÍDO	14 de nov. de 2024, 1
✓	PAC-108	Adicionar obrigatoriedade em alguns campos do for…	JD Joao Paulo Duarte	High	CONCLUÍDO	22 de nov. de 2024, 1
✓	PAC-107	Criar APK	JD Joao Paulo Duarte	High	CONCLUÍDO	28 de nov. de 2024, 1

101 itens — Sincronizado agora

Sprints atual:

Cada Sprint tem duração de 15 dias, onde no gráfico, é possível visualizar a Sprint atual do projeto:

https://joaoduartte.atlassian.net/jira/software/projects/PAC/boards/2/timeline

Definição de Pronto (Definition of Done - DoD)

A funcionalidade ou tarefa será considerada `PRONTA` quando cumprir todos os seguintes critérios:

1. **Implementação Completa:**
   - A funcionalidade foi desenvolvida conforme descrito na história de usuário.
   - Todas as funcionalidades solicitadas foram implementadas sem falhas.
2. **Testes Automatizados Criados e Executados:**
   - Testes unitários foram escritos para cobrir as principais funções e métodos.
   - Testes de integração foram executados para garantir a correta comunicação entre o front-end e o back-end.
   - Todos os testes passam com sucesso e não há falhas identificadas.
3. **Sem Bugs Críticos ou Impedimentos:**
   - A funcionalidade foi testada e não apresenta bugs críticos.
   - Caso existam bugs não críticos, eles foram registrados com a devida prioridade no backlog.
4. **Revisão de Código Concluída:**
   - O código passou por uma revisão de um colega ou líder técnico.
   - Todas as sugestões e ajustes feitos durante a revisão foram implementados.
   - O código segue os padrões de codificação adotados pela equipe.
5. **Integração e Deploy:**
   - O código foi integrado à branch principal sem conflitos.
   - A funcionalidade foi corretamente implantada no ambiente de teste ou produção, dependendo da necessidade.
6. **Documentação Atualizada:**
   - A documentação técnica foi atualizada para incluir quaisquer mudanças no sistema.
   - A documentação do usuário final, se aplicável, foi atualizada para refletir a nova funcionalidade.
7. **Critérios de Aceitação Atendidos:**
   - Todos os critérios de aceitação definidos na história de usuário foram cumpridos.
   - A funcionalidade foi validada pelo Product Owner e está de acordo com as expectativas.
8. **Demonstração Realizada:**
   - A funcionalidade foi demonstrada à equipe e ao Product Owner.
   - O feedback foi coletado e, se necessário, ajustes foram feitos para atender ao feedback.

Definição de Pronto (DoD) - Checklist:

- [ ] A implementação foi realizada conforme descrito na história de usuário.
- [ ] Testes unitários foram criados e passaram.
- [ ] Testes de integração foram criados e passaram.
- [ ] O código foi revisado e aprovado por um colega.
- [ ] Não há bugs críticos conhecidos.
- [ ] A funcionalidade foi integrada ao branch principal sem conflitos.
- [ ] A funcionalidade foi implantada no ambiente de testes/produção.
- [ ] A documentação técnica e de usuário foi atualizada.
- [ ] O Product Owner aprovou a funcionalidade.

Status de Reuniões:

- **Weekly Scrum:** Realizado semanalmente com duração de 15 minutos para atualizar o status de cada tarefa.
- **Sprint Review:** Realizado ao final de cada Sprint para revisar o progresso das funcionalidades implementadas.

- **Sprint Retrospective:** Reunião para identificar pontos de melhoria no processo da equipe.

Plano de Testes:

O sistema está sendo testado de forma rigorosa para garantir sua estabilidade. Foram realizados os seguintes tipos de testes:

- **Testes Unitários:** Garantem que cada parte do código funciona isoladamente.
- **Testes de Integração:** Garantem que diferentes partes do sistema funcionam bem juntas.

Planejamento Futuro:

Atualmente, o **Software IPJ** está completo com todas as funcionalidades necessárias para atender às necessidades de gerenciamento de membros da Igreja Presbiteriana de Joinville. O foco está em manter a estabilidade, usabilidade e segurança do sistema, garantindo que todas as funcionalidades existentes continuem a operar de forma eficiente.

Objetivos a Longo Prazo:

O objetivo principal é garantir que o **Software IPJ** continue a funcionar de maneira confiável, atendendo plenamente às necessidades da igreja sem a necessidade de adições futuras. A prioridade é assegurar a longevidade do software através de:

- **Suporte Técnico:** Disponibilidade de suporte técnico para resolver quaisquer problemas que possam surgir durante o uso do software.
- **Monitoramento de Performance:** Monitoramento contínuo do desempenho do software para identificar e corrigir quaisquer problemas antes que impactem o usuário final.

Evolução Esperada do Sistema:

Embora não haja planos para a implementação de novos recursos, o software será revisado periodicamente para garantir que continue a atender às necessidades da Igreja Presbiteriana de Joinville. O sistema está projetado para ser robusto e adaptável, permitindo ajustes menores e manutenção contínua conforme necessário.

Guia de Usuário:

Este guia fornece instruções detalhadas para ajudar os usuários finais a navegar e utilizar o **Software IPJ** (Igreja Presbiteriana de Joinville) de forma eficaz. Para informações mais detalhadas sobre cada funcionalidade do sistema, consulte o manual de uso completo.

Manual de Uso Completo:

- Acesse o manual de uso do software para obter instruções detalhadas sobre cada funcionalidade da plataforma: Manual de Uso do Software IPJ - Plataforma WEB .

Exemplos de Funcionalidades:

- **Cadastro de Membros:**
  - Navegue até a seção **"Cadastro"**, preencha o formulário com as informações do membro, e clique em **"Salvar"**.

Para mais detalhes sobre como utilizar todas as funcionalidades do sistema, não deixe de consultar o manual completo no link acima.

Perguntas Frequentes (FAQ):

Esta seção fornece respostas a perguntas comuns que os usuários possam ter ao utilizar o Software IPJ. Para acessar o **FAQ** completo e consultar todas as perguntas e respostas disponíveis, clique no link abaixo:

- Perguntas Frequentes (FAQ) do Software IPJ

Conclusão:

> Com a implementação do **Software IPJ**, a administração da Igreja Presbiteriana de Joinville agora possui uma ferramenta robusta e eficiente para o gerenciamento de seus membros e sociedades internas. A flexibilidade e a usabilidade do sistema garantem que ele possa crescer com as necessidades da igreja, sendo adaptável e evolutivo conforme a comunidade se desenvolve.