README GABINETE

gabinete/

│

├── public/ # Arquivos públicos (acessíveis diretamente pelo navegador).

│ ├── css/

│ │ └── styles.css # Seu arquivo de estilos principal.

│ ├── js/

│ │ └── main.js # Seu arquivo de scripts JavaScript.

│ ├── images/

│ │ └── logo.png # Logotipos e outras imagens.

│ └── uploads/

│ └── documentos/ # Onde os ofícios, requerimentos, etc., serão salvos.

│

├── src/ # O "coração" da sua aplicação (código PHP).

│ ├── config/

│ │ └── config.php # Constantes e configurações (banco de dados, URL base).

│ │

│ ├── controllers/ # Classes que controlam a lógica e o fluxo.

│ │ ├── AdminController.php

│ │ ├── EleitorController.php

│ │ └── UsuarioController.php

│ │

│ ├── helpers/ # Funções e classes auxiliares.

│ │ └── AuthHelper.php

│ │

│ ├── models/ # Classes que interagem com o banco de dados.

│ │ ├── Documento.php

│ │ ├── Eleitor.php

│ │ └── Usuario.php

│ │

│ └── views/ # Arquivos de HTML e PHP para a apresentação.

│ ├── admin/

│ │ └── dashboard.php

│ ├── eleitores/

│ │ └── lista.php

│ ├── partials/

│ │ ├── footer.php

│ │ └── header.php

│ └── usuarios/

│ └── login.php

│

├── sql/ # Scripts para a criação do banco de dados.

│ └── database.sql

│

├── .htaccess # Regras de reescrita de URL para o servidor Apache.

│

└── index.php # Ponto de entrada ÚNICO da aplicação (Roteador).

**Resumo das Responsabilidades:**

* **public/**: Tudo que o usuário pode ver ou baixar diretamente (CSS, JS, Imagens, Documentos).
* **src/**: Toda a sua lógica interna. O acesso a estes arquivos é controlado pelo index.php, o que torna o sistema mais seguro.
* **sql/**: O histórico do seu banco de dados. Ajuda a configurar o ambiente em novas máquinas.
* **.htaccess e index.php**: Trabalham juntos para criar URLs amigáveis e centralizar todas as requisições, sendo a base do padrão MVC.

Com certeza! Entender o papel de cada arquivo é o passo mais importante para se sentir confortável com a estrutura e começar a desenvolver.

Vamos detalhar a responsabilidade de cada arquivo dentro da estrutura do seu projeto gabinete.

**Pasta Raiz: gabinete/**

Esta é a pasta principal do seu projeto.

* **Arquivo:** .htaccess
  + **Pasta:** gabinete/
  + **Objetivo:** Configurar o servidor web (Apache).
  + **Detalhes:** Sua principal função aqui é redirecionar todas as requisições para o arquivo index.php. Isso permite que usemos URLs amigáveis (como /eleitores/listar em vez de /eleitores.php?acao=listar), o que é fundamental para o funcionamento de um sistema MVC moderno. Ele atua como um "porteiro" invisível.
* **Arquivo:** index.php
  + **Pasta:** gabinete/
  + **Objetivo:** Ponto de Entrada Único e Roteador da Aplicação.
  + **Detalhes:** Este é o arquivo mais importante da estrutura. Todo o tráfego do sistema passa por ele primeiro. Ele lê a URL que o usuário acessou (graças ao .htaccess), descobre qual Controller e qual método devem ser chamados para responder àquela requisição, e então os aciona. Ele é o "cérebro" que distribui as tarefas.

**Pasta: public/**

Contém todos os arquivos que podem e devem ser acessados diretamente pelo navegador. São os recursos visuais e de interação da sua aplicação.

* **Arquivo:** css/styles.css
  + **Pasta:** public/css/
  + **Objetivo:** Estilização Visual.
  + **Detalhes:** Aqui você escreverá todo o código CSS para definir as cores, fontes, layout, espaçamentos e a aparência geral do seu sistema, deixando-o com um visual profissional e agradável.
* **Arquivo:** js/main.js
  + **Pasta:** public/js/
  + **Objetivo:** Interatividade no Navegador.
  + **Detalhes:** Contém o código JavaScript. Ele será usado para criar interações dinâmicas, como validações de formulário em tempo real, máscaras para campos (CPF, telefone), mostrar/esconder elementos, e enviar dados sem precisar recarregar a página (AJAX).
* **Arquivo:** images/logo.png
  + **Pasta:** public/images/
  + **Objetivo:** Armazenar imagens estáticas.
  + **Detalhes:** Um local central para guardar o logotipo do gabinete, fotos de perfil padrão, ícones e qualquer outra imagem utilizada na interface do sistema.
* **Pasta:** uploads/documentos/
  + **Pasta:** public/uploads/
  + **Objetivo:** Armazenar os arquivos enviados pelos usuários.
  + **Detalhes:** Quando um usuário fizer o upload de um ofício, projeto ou prestação de contas, o arquivo físico (o PDF, DOCX, etc.) será salvo nesta pasta pelo seu script PHP. O acesso a esses arquivos será feito através de links gerados pelo sistema.

**Pasta: src/ (Source/Fonte)**

O coração da sua aplicação. Nenhum arquivo aqui dentro deve ser acessado diretamente pelo navegador. O acesso é sempre controlado pelo index.php.

* **Arquivo:** config/config.php
  + **Pasta:** src/config/
  + **Objetivo:** Centralizar as configurações.
  + **Detalhes:** Armazena informações que podem mudar dependendo do ambiente (desenvolvimento, produção), como as credenciais do banco de dados (host, usuário, senha, nome do banco) e a URL base do seu sistema. Ter isso em um único arquivo facilita muito a manutenção.
* **Arquivos na pasta controllers/**
  + **Objetivo Geral:** São os "Gerentes de Projeto". Eles recebem as ordens do roteador (index.php), coordenam as ações, pedem dados aos Models e escolhem qual View será exibida ao usuário.
  + AdminController.php: Gerencia as ações da área administrativa, como exibir o painel principal (dashboard) e processar o upload de novos documentos.
  + EleitorController.php: Gerencia todas as ações relacionadas aos eleitores: listar, mostrar formulário de cadastro, salvar um novo eleitor, editar, excluir, etc.
  + UsuarioController.php: Responsável pelas ações de autenticação: exibir a tela de login, verificar as credenciais (auth) e fazer o logout.
* **Arquivo:** helpers/AuthHelper.php
  + **Pasta:** src/helpers/
  + **Objetivo:** Fornecer funções auxiliares reutilizáveis.
  + **Detalhes:** AuthHelper é um exemplo. Sua única função é verificar se o usuário está logado. Ao chamar AuthHelper::check() no início de um método do controller, você protege aquela página, evitando a repetição do mesmo código de verificação de $\_SESSION várias vezes.
* **Arquivos na pasta models/**
  + **Objetivo Geral:** São os "Especialistas em Banco de Dados". A única responsabilidade deles é se comunicar com o banco de dados. Eles contêm todas as queries SQL (INSERT, SELECT, UPDATE, DELETE) e não sabem nada sobre HTML ou a aparência do site.
  + Documento.php: Contém todos os métodos para interagir com a tabela documentos (criar registro de documento, buscar por categoria, etc.).
  + Eleitor.php: Contém todos os métodos para interagir com a tabela eleitores.
  + Usuario.php: Contém todos os métodos para interagir com a tabela usuarios (buscar por e-mail, verificar senha, etc.).
* **Arquivos na pasta views/**
  + **Objetivo Geral:** São os "Designers de Interface". Representam a camada de apresentação, ou seja, o que o usuário vê. São arquivos majoritariamente HTML, com um pouco de PHP para exibir os dados que foram preparados pelo Controller.
  + admin/dashboard.php: A tela do painel administrativo, contendo o formulário de upload e a lista/busca de documentos.
  + eleitores/lista.php: A página que exibe a tabela com a lista de todos os eleitores cadastrados.
  + partials/header.php e footer.php: Pedaços de código reutilizáveis. header.php contém o topo do site (incluindo o menu de navegação), e footer.php contém o rodapé. Eles são incluídos em quase todas as outras views para evitar repetição de código.
  + usuarios/login.php: A tela com o formulário de login.

**Pasta: sql/**

* **Arquivo:** database.sql
  + **Pasta:** sql/
  + **Objetivo:** A "Planta Baixa" do seu banco de dados.
  + **Detalhes:** Contém os comandos SQL CREATE TABLE para todas as tabelas do seu sistema. É essencial para configurar o projeto rapidamente em uma nova máquina ou para restaurar a estrutura do banco de dados, se necessário.

Com este mapa em mãos, você tem uma visão clara do propósito de cada peça do quebra-cabeça

### **1. O código está responsivo? Como testar?**

A responsividade é a capacidade do site se adaptar a diferentes tamanhos de ecrã (computadores, tablets, telemóveis).

* **Análise do seu código:** O código que criámos, especialmente o do calendário e dos formulários (como o que está no Canvas), tem uma **responsividade básica**. Por exemplo, o formulário modal tem max-width: 500px e width: 80%, o que significa que ele irá ocupar 80% da largura do ecrã em dispositivos pequenos, mas nunca passará de 500px em ecrãs grandes. O calendário também tem um max-width. Isto já ajuda muito! A estrutura geral do seu site (o header.php, etc.) também parece usar uma abordagem que se adapta.
* **Como testar localmente (o melhor método):** Você não precisa de um telemóvel para testar! Pode simular qualquer dispositivo diretamente no seu navegador.
  1. Abra o seu sistema no Google Chrome (ou Firefox).
  2. Clique com o botão direito do rato em qualquer parte da página e selecione **"Inspecionar"** (ou "Inspect"). Pode também usar a tecla **F12**.
  3. Isto abrirá as Ferramentas de Desenvolvedor. No canto superior esquerdo dessa nova janela, procure por um ícone que parece um telemóvel e um tablet juntos. Clique nele.
  4. A sua página irá mudar para uma "vista de dispositivo". Agora, no topo do seu site, aparecerá uma barra onde pode escolher diferentes modelos de telemóveis (iPhone, Samsung Galaxy, etc.) ou simplesmente arrastar as bordas para ver como o site se comporta em qualquer tamanho.

Use esta ferramenta para navegar no seu sistema e ver se todos os elementos (menus, tabelas, formulários) se ajustam bem e se são fáceis de usar em ecrãs pequenos.

**2. Alterações para hospedar num servidor cPanel**

Ótima pergunta. Mover um projeto de um ambiente local (MAMP) para um servidor real (cPanel) envolve algumas alterações de configuração cruciais. Não precisa de mudar a lógica do seu código PHP, apenas as configurações de ambiente.

Aqui estão as alterações que terá de fazer, sem fornecer os códigos:

1. **Configuração do Banco de Dados (config.php):**
   * No cPanel, você terá de criar um novo banco de dados MySQL e um novo utilizador para esse banco.
   * O seu ficheiro src/config/config.php terá de ser editado. Os valores de DB\_NAME, DB\_USER e DB\_PASS terão de ser atualizados com as novas credenciais que acabou de criar no cPanel. O DB\_HOST geralmente continua a ser localhost.
2. **URL Base do Projeto (config.php):**
   * Esta é uma alteração muito importante. A constante BASE\_URL no seu config.php terá de ser mudada de http://localhost:8888/gabinete para o seu domínio real, por exemplo, https://www.seusite.com.br. Se não fizer isto, todos os links e redirecionamentos do site irão falhar.
3. **Importação do Banco de Dados:**
   * Você terá de exportar o seu banco de dados local (gabinete\_bd) a partir do phpMyAdmin no MAMP. Isto gera um ficheiro .sql.
   * Depois, no cPanel, aceda ao phpMyAdmin e importe esse ficheiro .sql para o novo banco de dados que criou. Isto irá transferir todas as suas tabelas e dados.
4. **Configuração do Servidor para URLs Amigáveis (.htaccess):**
   * O seu sistema usa URLs amigáveis (ex: /eleitores/cadastrar). Para que isto funcione num servidor real, precisa de um ficheiro chamado .htaccess na pasta raiz do seu projeto (gabinete/). Este ficheiro diz ao servidor para enviar todas as requisições para o seu index.php. O conteúdo deste ficheiro é padrão para a maioria dos frameworks e roteadores como o seu.
5. **Versão do PHP:**
   * No cPanel, existe uma opção para selecionar a versão do PHP. Certifique-se de que está a usar uma versão compatível com o seu código (PHP 8.0 ou superior).

Seguindo estes passos, a sua aplicação deverá funcionar no servidor cPanel da mesma forma que funciona localmente.