

**UNIVERSIDADE FEDERAL DO MARANHÃO**

BACHARELADO INTERDISCIPLINAR EM CIÊNCIA E TECNOLOGIA

COORDENAÇÃO DO CURSO DE ENGENHARIA DA COMPUTAÇÃO - CCEC PROCESSO E DESENVOLVIMENTO DE SOFTWARE

PROF. DR. THALES LEVI AZEVEDO VALENTE

BRUNO KAUAN RODRIGUES SILVA (2022030340)

ELLEN CRISTINA DE SOUSA CASTRO (2022030206)

MANOEL LUCAS PACHECO JUNIOR (2021052808)

PAULO EDUARDO LIMA RABELO (2021018056)

MATEUS DUTRA VALE (2021018495)

**MANUAL DE INSTALAÇÃO**

SISTEMA DE VISTORIA DE IMÓVEIS

**SÃO LUIS – MA**

**2025**

**1 Introdução**

O Sistema CIVIS foi desenvolvido para facilitar a gestão de inspeções em imóveis, permitindo o agendamento, acompanhamento e registro de vistorias de forma intuitiva. Este manual tem como objetivo orientar os usuários na instalação e configuração da aplicação contida no diretório Sistema de Vistoria. Ele contém instruções detalhadas para garantir a correta execução do software em ambiente local.

**2 Instalação**

**2.1 Requisitos do Sistema**

A aplicação pode ser executada em máquinas com os seguintes sistemas operacionais:

* Windows 10 ou superior
* Linux (distribuições modernas)

**Dependências necessárias:**

* Node.js versão 18 ou superior
* Git
* Conta ativa no Supabase (PostgreSQL)
* Editor de código (VSCode recomendado)

**2.2 Estrutura do Projeto**

Sistema-de-Vistoria/

│

├── backend/ →Backend da aplicação (Node.js + Express)

│ ├── assets/ → Recursos auxiliares (imagens, documentos, etc.)

│ ├── controllers/ →Lógica de controle das rotas

│ ├── models/ →Modelos das entidades do banco de dados

│ ├── relatorios/ → Geração e manipulação de relatórios

│ ├── routes/ → Definição das rotas da API

│ ├── uploads/ → Pasta para arquivos enviados

│ ├── .env → Variáveis de ambiente

│ ├── app.js → Configuração principal do Express

│ ├── db.js → Conexão com o banco de dados

│ ├── enviarEmail.js → Lógica para envio de e-mails

│ └── server.js → Ponto de entrada do backend

│

├── docs-backend/ → Documentação do backend (pasta opcional)

│

├── node\_modules/ →Dependências do Node.js (gerado automaticamente)

│

├── public/ → Arquivos estáticos do frontend

│

├── src/ → Código-fonte do frontend (React.js)

│ ├── pages/ → Páginas divididas por tipo de usuário

│ │ ├── Cadastro/ → Tela de cadastro

│ │ ├── HomeAdm/ → Página inicial do administrador

│ │ ├── HomeCliente/ → Página inicial do cliente

│ │ └── HomeVistoriador/ → Página inicial do vistoriador

│ │ ├── CriarRelatorio/ → Etapa de criação de relatório

│ │ ├── IniciarVistoria/ → Etapa de início da vistoria

│ │ ├── ReagendarVistoria/ → Etapa para reagendamento

│ │ └── RealizarVistoria/ → Etapa para realizar vistoria

│ ├── VistoriaData/ → Dados relacionados às vistorias

│ ├── Inicial/ → Página inicial antes do login

│ ├── Login/ → Página de login

│ ├── utils/ → Funções utilitárias

│ ├── App.jsx → Componente principal da aplicação React

│ ├── index.jsx → Ponto de entrada do React

│ ├── main.jsx → Arquivo de renderização

│ ├── Home.jsx → Página inicial padrão

│ ├── Home.css → Estilos globais da home

│ ├── supabaseClient.js → Conexão com Supabase

│

├── .gitignore → Arquivos/pastas ignorados pelo Git

├── eslint.config.js → Configurações do ESLint

├── index.html → HTML base da aplicação React

├── novo script.sql → Script SQL extra (opcional)

├── package.json → Configurações e dependências do projeto

├── package-lock.json → Versões exatas das dependências

├── README.md →Arquivo de instruções e documentação

└── script.sql → Script para criação do banco de dados

**2.3 Clonando o Repositório**

1. Abra o terminal ou CMD.
2. Execute os comandos abaixo:

Git clone <https://github.com/bkauan099/G2_CONSTRUTORA.git>

**2.4 Instalando Dependências**

Execute no terminal:

npm install

Isso instalará todas as bibliotecas necessárias para execução do frontend e backend.

**2.5 Configurando o Banco de Dados (Supabase)**

1. Acesse [https://supabase.com](https://supabase.com/) e crie um projeto PostgreSQL.
2. No menu lateral, acesse **SQL Editor**.
3. Copie o conteúdo do arquivo “**novo script.sql**”e execute para criar as tabelas.

### **2.6 Configurando a Conexão com o Supabase**

Crie um arquivo .env na raiz do projeto com as seguintes informações:

**SUPABASE\_URL**=postgresql://postgres:IfejFxhbTkme91pI@db.sictbgrpkhacrukvpopz.supabase.co:5432/postgres

**SUPABASE\_SERVICE\_ROLE\_KEY**=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6InNpY3RiZ3Jwa2hhY3J1a3Zwb3B6Iiwicm9sZSI6InNlcnZpY2Vfcm9sZSIsImlhdCI6MTc1MDM0MzgwNSwiZXhwIjoyMDY1OTE5ODA1fQ.fzOJ0fpwQimSH7mwnIpBrOrXlenx4t5zbCZ\_CZgo3zA

Estas informações estão disponíveis no painel do Supabase, na aba **Project Settings → API**.

**2.8 - No código .env coloque a varivel que você pegou e coloque na key**

**Como Criar uma Chave de API da OpenAI**

**Requisitos:**

* Ter uma conta na [OpenAI](https://platform.openai.com/signup" \t "_new) (se ainda não tem, é só criar gratuitamente).
* Um e-mail válido e, em alguns casos, um número de telefone para verificação.

**Passo a Passo**

**1. Acesse a plataforma da OpenAI**

Vá para o site oficial da plataforma:  
 [https://platform.openai.com](https://platform.openai.com" \t "_new)

**2. Faça login ou crie uma conta**

* Se já tiver uma conta, clique em **"Log in"**
* Se ainda não tiver, clique em **"Sign up"** e siga as instruções para criar uma.

**3. Vá até a seção de chaves de API**

* Após logar, vá para este link direto:  
   [https://platform.openai.com/api-keys](https://platform.openai.com/api-keys" \t "_new)

Ou, no menu superior, clique em:

* Seu nome no canto superior direito
* Depois em **"View API Keys"**

**4. Crie uma nova chave**

* Clique no botão **"Create new secret key"**
* Dê um nome para sua chave (opcional, mas recomendado)
* A chave será gerada **uma única vez** – copie e salve imediatamente.

**Importante:** Você **não poderá ver essa chave de novo**, então **guarde bem** em um local seguro como um gerenciador de senhas ou bloco de notas seguro.

**5. Use a chave na sua aplicação**

No seu código, você pode usar essa chave como:

const openai = new OpenAI({

apiKey: 'SUA\_CHAVE\_AQUI',

});

### **2.7 Executando o Sistema**

1. Inicie o frontend:

No terminal, execute:

**npm run dev**

A aplicação será iniciada localmente. Acesse pelo navegador através do link:

<http://localhost:5432>

1. Inicie o backend:

No terminal, navegue até a pasta do backend:

**cd backend**

Em seguida, execute:

**node server.js**

## **3 Login e Acessos**

O sistema possui três perfis de usuários:

* **Administrador:** gerencia usuários, empreendimentos e acessos.
* **Cliente:** visualiza imóveis e solicita vistorias.
* **Vistoriador:** executa e encerra vistorias.

O controle de acesso é feito via autenticação JWT fornecida pelo Supabase Auth.

## **4 Documentação Complementar**

* Manual do Usuário disponível em readm/manual\_usuario.pdf
* Diagramas UML e de Casos de Uso em diagramas/

## **5 Repositório Oficial**

<https://github.com/bkauan099/G2_CONSTRUTORA>

**6 Direitos Autorais e Licença**

**@autor: [Bruno Kauan Rodrigues Silva, Ellen Cristina De Sousa Castro, Manoel Lucas Pacheco Junior, Mateus Dutra Vale, Paulo Eduardo Lima Rabelo]**

**@contato: [**[bruno.kauan@discente.ufma.br](mailto:bruno.kauan@discente.ufma.br)**,** [ellen.castro@discente.ufma.br](mailto:ellen.castro@discente.ufma.br)**,** [manoel.lucas@discente.ufma.br](mailto:manoel.lucas@discente.ufma.br)**,** [rabelo.paulo@discente.ufma.br](mailto:rabelo.paulo@discente.ufma.br)**,** [mateus.dv@discente.ufma.br](mailto:mateus.dv@discente.ufma.br)**]**

**@data última versão: [30/06/2025]**

**@versão: 1.0**

**@outros repositórios: [**<https://github.com/bkauan099>**;** <https://github.com/Mateus-dutravale>**;** <https://github.com/Ellen6185>**]**

**@Agradecimentos: Universidade Federal do Maranhão (UFMA), Professor Doutor Thales Levi Azevedo Valente, e colegas de curso.**

**Copyright/License**

**Este material é resultado de um trabalho acadêmico para a disciplina PROJETO E DESENVOLVIMENTO DE SOFTWARE, sob a orientação do professor Dr. THALES LEVI AZEVEDO VALENTE, semestre letivo 2025.1, curso Engenharia da Computação, na Universidade Federal do Maranhão (UFMA). Todo o material sob esta licença é software livre: pode ser usado para fins acadêmicos e comerciais sem nenhum custo. Não há papelada, nem royalties, nem restrições de "copyleft" do tipo GNU. Ele é licenciado sob os termos da Licença MIT, conforme descrito abaixo, e, portanto, é compatível com a GPL e também se qualifica como software de código aberto. É de domínio público. Os detalhes legais estão abaixo. O espírito desta licença é que você é livre para usar este material para qualquer finalidade, sem nenhum custo. O único requisito é que, se você usá-los, nos dê crédito.**

**Licenciado sob a Licença MIT. Permissão é concedida, gratuitamente, a qualquer pessoa que obtenha uma cópia deste software e dos arquivos de documentação associados (o "Software"), para lidar no Software sem restrição, incluindo sem limitação os direitos de usar, copiar, modificar, mesclar, publicar, distribuir, sublicenciar e/ou vender cópias do Software, e permitir pessoas a quem o Software é fornecido a fazê-lo, sujeito às seguintes condições:**

**Este aviso de direitos autorais e este aviso de permissão devem ser incluídos em todas as cópias ou partes substanciais do Software.**

**O SOFTWARE É FORNECIDO "COMO ESTÁ", SEM GARANTIA DE QUALQUER TIPO, EXPRESSA OU IMPLÍCITA, INCLUINDO MAS NÃO SE LIMITANDO ÀS GARANTIAS DE COMERCIALIZAÇÃO, ADEQUAÇÃO A UM DETERMINADO FIM E NÃO INFRINGÊNCIA. EM NENHUM CASO OS AUTORES OU DETENTORES DE DIREITOS AUTORAIS SERÃO RESPONSÁVEIS POR QUALQUER RECLAMAÇÃO, DANOS OU OUTRA RESPONSABILIDADE, SEJA EM AÇÃO DE CONTRATO, TORT OU OUTRA FORMA, DECORRENTE DE, FORA DE OU EM CONEXÃO COM O SOFTWARE OU O USO OU OUTRAS NEGOCIAÇÕES NO SOFTWARE.**

**Para mais informações sobre a Licença MIT:** <https://opensource.org/licenses/MIT>