Universidade catolica de mocambique

Faculdade de Gestao De Turismo e Informatica

Tema: Aplicacao Web para ensino de Inglês

Nome: minina da silvia casimiro Gomes Liasse

70623019220

Docente: Ismael Chande Junior

1.Introdução

Esta documentação descreve a aplicação web para o ensino de ingles, um sistema desenvolvido para facilitar o ensino de idiomas online. A aplicação permite o cadastro de alunos e professores, autenticação, pagamento simulado, interação em tempo real via chat com um bot (limitado a 3 perguntas em inglês), e visualização de relatórios de progresso e utilização. Este documento cobre as funcionalidades, estrutura, tecnologias, instruções de uso, manutenção, e outros aspectos relevantes para desenvolvedores e usuários.

2. Informações Gerais

- **Nome da aplicação**: Ensino Online
- **Objetivo principal**: Proporcionar uma plataforma para ensino de idiomas, com cadastro de alunos e professores, autenticação, pagamento simulado, chat interativo com bot, e relatórios de desempenho.
- **Público-alvo**:
    - **Alunos**: Pessoas interessadas em aprender idiomas (foco em inglês), com diferentes níveis (iniciante, intermediário, avançado) e objetivos (profissional, acadêmico, viagens).
    - **Professores**: Profissionais que oferecem aulas de idiomas, com especialidades como conversação, gramática, negócios ou exames.
- **Tecnologias utilizadas**:
    - **Frontend**:
        - HTML5: Estrutura das páginas.
        - CSS3: Estilização, com estilo.css personalizado.
        - Bootstrap 5.3.3: Framework CSS para responsividade e componentes (navbar, modais, formulários).
        - JavaScript (ES6): Lógica client-side, com fetch para chamadas API e Socket.IO para chat em tempo real.
    - **Backend**:

        - Node.js: Ambiente de execução server-side.
        - Express 4.21.1: Framework para servidor HTTP e APIs REST.
        - Socket.IO 4.8.0: Comunicação em tempo real para o chat.
        - CORS 2.8.5: Suporte a requisições cross-origin.
    - **Outros**:
        - npm: Gerenciador de pacotes.
        - JSON: Formato para configuração (package.json) e respostas API.

3. Instalação e Configuração

Esta seção descreve como configurar e executar a aplicação localmente em um ambiente Windows.

- **Pré-requisitos**:
    - Node.js (versão 16 ou superior):
    - npm (incluído com Node.js).
    - Navegador web (Chrome, Firefox, Edge, etc.).
    - Git para clonar repositórios.
- **Comandos de instalação**:

cd C:\Users\o nome\Desktop\projecto

Instar as dependências

cd server

npm install

Como iniciar o projeto:

npm start

4. Estrutura do Projeto

A aplicação está organizada em duas pastas principais:

public/ (frontend):

- index.html: Página para cadastro de alunos.
- professores.html: Página para cadastro de professores.
- login.html: Página de autenticação.
- pagamento.html: Página para pagamento simulado.
- chat.html: Página para chat com bot e professores.
- reports.html: Página para relatórios de progresso e utilização.
- estilo.css: Estilos personalizados para a interface.

server/ (backend):

- app.js: Configura o servidor Express, Socket.IO, e serve arquivos estáticos. Gerencia conexões WebSocket para o chat.
- routes.js: Define rotas API para cadastro, login, pagamento, e relatórios.
- package.json: Lista dependências e scripts npm.
- node_modules/: Dependências instaladas (não versionadas).

README.md: Documentação inicial com instruções básicas.

. Funcionalidades

A aplicação implementa as seguintes funcionalidades:

- **Página de Cadastro de Alunos (index.html)**:
  - Permite que alunos criem contas com nome, email, senha, nível de inglês (iniciante, intermediário, avançado), objetivo (profissional, acadêmico, viagens), e horário preferido.
  - Validação client-side (campos obrigatórios) e server-side (verificação de campos).
  - Após sucesso, redireciona para login.html em 2 segundos.
  - Integração com API /api/cadastro (POST).
- **Página de Cadastro de Professores (professores.html)**:
  - Permite que professores criem contas com nome, idiomas ensinados, especialidade (conversação, gramática, negócios, exames), e senha.
  - Validação client-side e server-side.
  - Redireciona para login.html após sucesso.

    - Integração com API /api/professores (POST).
- **Página de Login (login.html)**:
    - Autentica alunos e professores com email e senha.
    - Exibe mensagens de erro ou sucesso em um modal Bootstrap.
    - Redireciona para pagamento.html após login bem-sucedido.
    - Integração com API /api/login (POST).
- **Página de Pagamento (pagamento.html)**:
    - Simula pagamento com email, número de cartão (16 dígitos), validade (MM/AA), e CVV (3 dígitos).
    - Validação client-side (campos obrigatórios) e server-side (formato de cartão e CVV).
    - Redireciona para chat.html após sucesso.
    - Integração com API /api/pagamento (POST).
- **Página de Chat (chat.html)**:
    - Permite interação em tempo real com um bot (EnglishBot) que responde a 3 perguntas específicas em inglês: "What is your name?", "How are you?", "What time is it?".
    - Após 3 perguntas, o bot é desativado, e o usuário pode interagir com professores.
    - Usa Socket.IO para comunicação WebSocket.
    - Interface com área de mensagens rolável e formulário para envio.
- **Página de Relatórios (reports.html)**:
    - Exibe relatórios de progresso de alunos (nome, nível, progresso em %) e utilização de professores (nome, número de aulas).
    - Dados mock são retornados pelas APIs /api/progress-report e /api/teacher-report (GET).
    - Usa Bootstrap cards para exibição.
- **Responsividade**:
    - Todas as páginas usam Bootstrap 5.3.3 com grid system (row, col-md-*) para layouts responsivos.
    - Testado em dispositivos móveis e desktops, com navbar que se adapta a telas maiores ou menores e formulários adaptáveis.
- **Integração com APIs**:

- APIs REST em /api/cadastro, /api/professores, /api/login, /api/pagamento (POST).
- APIs de relatórios em /api/progress-report, /api/teacher-report (GET).
- WebSocket via Socket.IO para o chat (chatMessage e message events).

- **Filtros e Pesquisa**:
  - Não implementados na versão atual.

6. Documentação de Código

- **Comentários no código**:
  - **JavaScript** (app.js, routes.js): Cada linha relevante tem comentários explicando sua função
  - **HTML**: Comentários descrevem seções como navbar, formulários, e modais
  - **CSS** (estilo.css): Comentários indicam o propósito de cada regra

7. Testes

- **Tipos de testes**:
  - **Manuais**: Testes realizados durante o desenvolvimento para verificar:
    - Cadastro: Envio de formulários com dados válidos/inválidos.
    - Login: Autenticação com credenciais corretas/incorretas.
    - Pagamento: Validação de formato de cartão
    - Chat: Envio de perguntas ao bot e interação após 3 perguntas.
    - Relatórios: Exibição correta de dados mock.
  - **End-to-end**: Testes manuais no navegador simulando fluxo completo (cadastro → login → pagamento → chat → relatórios).
- **Ferramentas utilizadas para testes**:
  - **Jest**: Para testes unitários de funções em routes.js (ex.: validar entrada de dados).
  - **postman**
- **Plano futuro**:
  - Melhorar o site e hospedar para o uso publico

8. Hospedagem

- A aplicacao encontra-se hospedada no GitHub e Render

9. Conclusão

- **Resumo**: A aplicação **Ensino Online** é um sistema funcional para ensino de idiomas, com cadastro, autenticação, pagamento simulado, chat interativo, e relatórios. Construída com tecnologias modernas (Express, Socket.IO, Bootstrap), é simples, responsiva, e extensível. O fluxo cadastro → login → pagamento → chat → relatórios foi implementado com sucesso, usando armazenamento em memória para prototipagem rápida.
- **Lições aprendidas**:
    - **Integração WebSocket**: Configurar Socket.IO com CORS e gerenciar conexões em tempo real foi um desafio mais ou menos superado.
    - **Responsividade**: Bootstrap facilitou layouts adaptáveis, mas ajustes em estilo.css foram necessários para consistência.
    - **Validação**: Combinar validações client-side e server-side melhorou a robustez.
    - **Documentação**: Comentar o código linha por linha aumentou a clareza para manutenção futura.