Escola Pregma

Relatório

Nome: Daniel de Oliveira - RA: 172214678

UniCuritiba - Ciência da Computação

Introdução

Inicialmente com a proposta da criação de um modelo escolar, surgiu a Escola Pregma, uma companhia fictícia para a aplicação escolar com meios de aprendizado, constituindo principalmente de funções básicas como leitura de dados com único registro e conjunto de registros, atualização de registros, criação de novos registros e a deleção de registros, proposta inicialmente com a construção de um web site com a tela do cliente e com a parte do servidor.

Modelos do Projeto

Partindo inicialmente pelo banco de dados, foi restringida a utilização do Banco PostgreSQL, foi criado um modelo com nove tabelas, sendo duas distintas da regra de negócio.

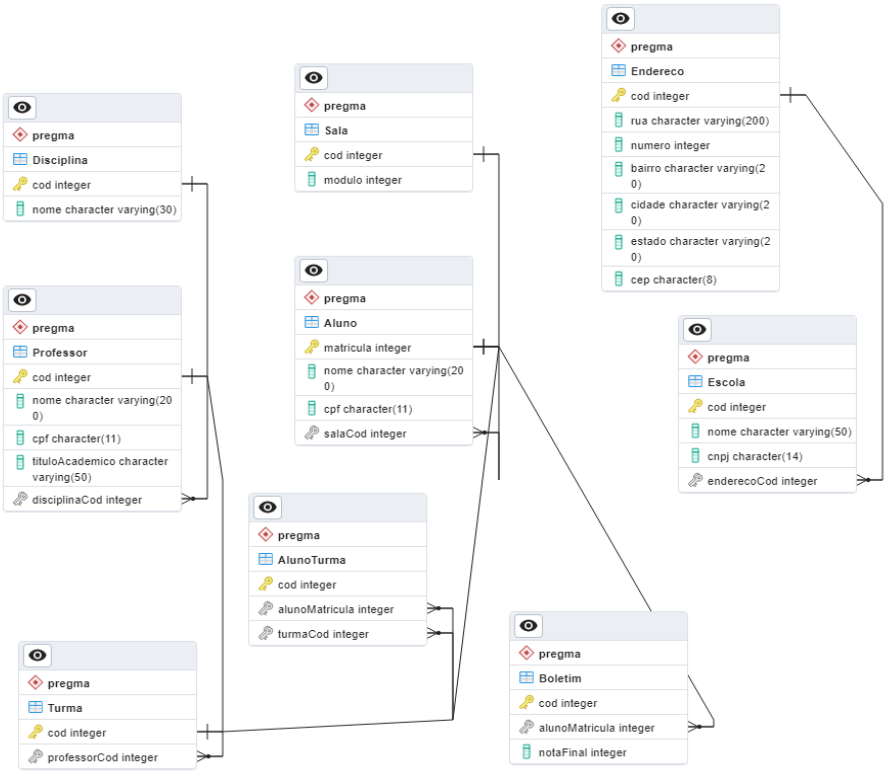

Figura 1: Modelo do Banco de Dados

Como dito anteriormente, podemos diferir os registros de Endereço e Escola, do resto, com a proposta de serem dados fixos, sem alteração ou inserção externa.

Já na regra de negócio, podemos seguir o seguinte fluxo, contemos um Aluno com seu Boletim fixamente, cada Aluno tem a referência de uma Sala e cada Aluno pertence a um AlunoTurma (Conjunto de vários Alunos), contendo a Turma, cada Turma necessita de um Professor e cada Professor tem uma Disciplina no qual se especializa.

Cliente

A cerca do desenvolvimento do layout do cliente, foi feito o Wireframe inicial com tabelas e a pagina estática

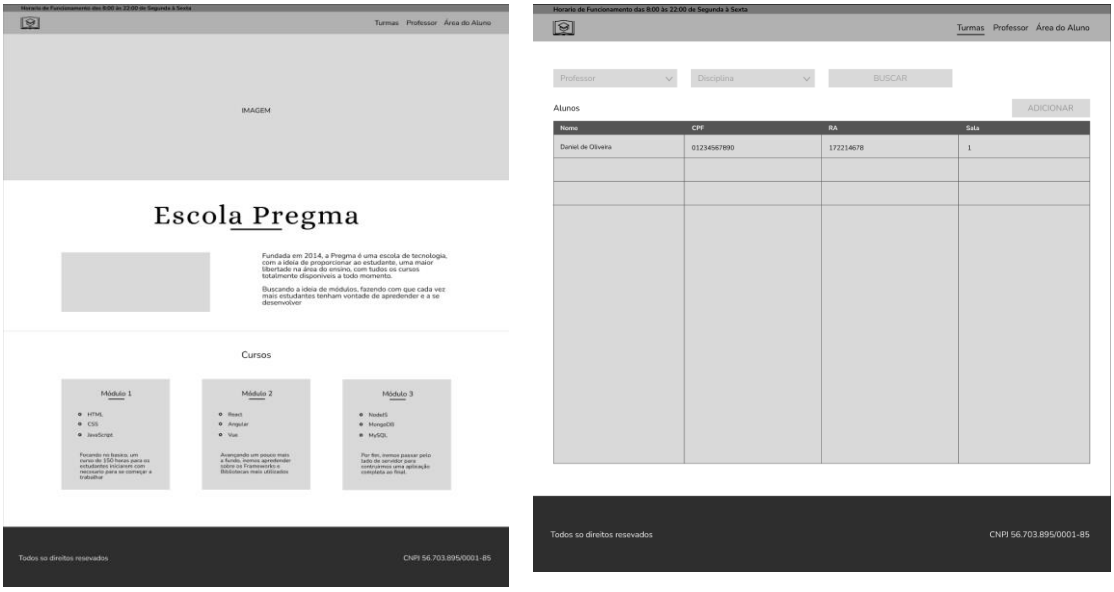

Figura 2: Wireframe

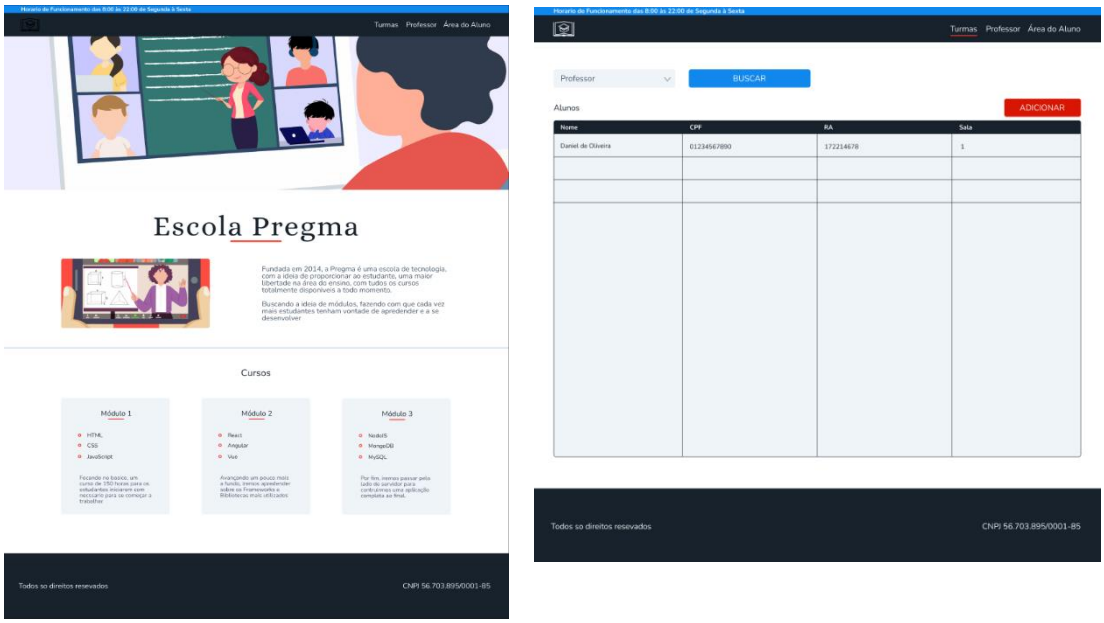

Figura 3: Design

Desenvolvimento com discussão acerca do código fonte

O primeiro passo na realização deste projeto foi analisar o problema descrito, partindo com o modelo do banco de dados já citado anteriormente, e pela exigência da utilização do NodeJS e React, foi construída toda a arquitetura do servidor separado do cliente.

Na parte do Servidor, foi escolhida a biblioteca Prisma para a comunicação com o banco de dados, por ter uma interligação com o PostgreSQL bem única, e por criar e implementar automaticamente um Modelo da nossa Query. A partir disso foi criado a estrutura de rotas, pelos controllers sendo eles: AlunoController, EscolaController, ProfessorController e TurmaController. Para níveis de aprendizado foi criado uma Seed para auto popular as tabelas com inserção de dados.

A respeito da parte do cliente, foi criado quatro telas, a página inicial, turma, professores e a área do aluno, cada um com seus métodos de inserção, como nossa API contem dados de endereço, foi criado um contexto dentro da aplicação, deixando nosso site apenas funcionar caso nosso servidor esteja rodando. Pensando no roteamento das rotas foi feita a utilização do React-Router-Dom implementando todo sistema de paginação e navegação a nossa aplicação.

Resultados

Criando um fluxo, de cada página podemos ter os seguintes resultados, ao acesso, podemos coletar dados do nome da escola, cursos disponíveis e todas as informações sobre como seriam as aulas, seguindo a diante temos a página de turmas onde podemos inicialmente escolher um professor e filtrar os alunos cadastrados em sua sala, logo em seguida podemos adicionar um novo aluno, excluir um aluno ou atualizar sua nota atual.

Pensando no modelo de CRUD aplicamos todos os métodos já nessas duas páginas, mas continuamos com duas páginas relacionadas primeiramente aos professores, onde temos a lista de professores, contendo informações do nome, CPF, titulo academia e a sala no qual está encarregado, juntamente temos as opções de adicionar um novo professor, escolhendo a disciplina que irá cursar.

Seguindo em frente temos a Área do Aluno, onde podemos buscar o aluno pela sua Matricula ou RA, e nos deparamos com informações como nome, CPF e módulo, logo a baixo temos seu boletim, com as principais opções, a sua nota, seu status de aprovado ou reprovado baseado numa média de 6 pontos, e por fim um botão para se matricular caso a nota seja maior que a média ou para fazer a rematrícula caso seja menor, e por fim caso o aluno já tenha concluído todos os módulos disponíveis e seja aprovado, não irá aparecer as duas opções padrões, mas sim um aviso de que o curso já foi concluído.

Conclusões

Por fim concluímos o projeto, do início ao fim, com todos os requisitos completos e com todas as funcionalidades em perfeito estado, foi um projeto extenso por ser um sistema quase completo, faltando a parte de login onde o usuário seria barrado de algumas permissões, mas sim foi um resultado satisfatório dentro do esperado.

Bibliografia

[1] Prisma, www.prisma.io. Acessado em 21/10/22.

[2] PostgreSQL, www.postgresql.org. Acessado em 21/1022.

[3] Express, https://expressjs.com. Acessado em 21/10/22.

[4] NodeJS, https://nodejs.org. Acessado em 21/10/22.

[5] ReactJS, www.reactjs.org. Acessado em 21/10/22.

[6] React-Router-Dom, v5.reactrouter.com. Acessado em 21/10/22.