Sistema Educacional Interativo para o Ensino de Banco de Dados Voltado à formação de Iniciantes na Área de Tecnologia da Informação

Luís Felipe Colósimo

RESUMO

Este trabalho apresenta o desenvolvimento de um sistema educacional voltado ao ensino prático de SQL, composto por um banco de dados relacional MySQL, um backend em Node.js/NestJS e um aplicativo móvel em React Native. A construção iniciou-se pela modelagem conceitual, que evoluiu para um modelo relacional abrangendo usuários, perguntas, alternativas, temas, blocos de resposta, comentários, histórico e score. A modelagem priorizou normalização, integridade referencial e escalabilidade. O backend foi implementado utilizando TypeORM para mapear as entidades e expor rotas REST que permitem autenticação, listagem de conteúdos, registro de respostas e consulta de desempenho. O aplicativo móvel consome essas rotas, fornecendo uma experiência interativa onde o aluno responde perguntas de múltipla escolha ou monta comandos SQL por blocos. Também foram desenvolvidas telas de resultado, perfil e score, garantindo acompanhamento da evolução do usuário. O sistema final integra de forma coerente banco de dados, backend e front-end, demonstrando a importância do modelo relacional como núcleo funcional de aplicações educacionais.

Palavras-chave

Banco de Dados Relacional, Aprendizagem de SQL, Aplicativo Educacional

I. INTRODUÇÃO

1. Contexto

O crescimento acelerado da área de Tecnologia da Informação tem gerado demanda constante por profissionais com conhecimento sólido em banco de dados, tecnologia fundamental para praticamente todos os sistemas corporativos e acadêmicos. No entanto, muitos iniciantes encontram dificuldades em aprender conceitos essenciais, como modelagem, normalização, consultas e segurança. Nesse cenário, ferramentas educacionais interativas podem contribuir para reduzir a barreira de entrada e facilitar o aprendizado. Este trabalho propõe o desenvolvimento de uma aplicação mobile voltada ao ensino de conceitos básicos de banco de dados, de modo a unir teoria e prática em um ambiente acessível e interativo.

2. Objetivo

O objetivo deste trabalho é Projetar, implementar e otimizar um banco de dados relacional em MySQL para suportar uma aplicação educacional interativa desenvolvida em React Native, com backend em Node.js.

Para isso, separei meu objetivo em diversos objetivos específicos, sendo eles:

- Objewtivos de banco:
  - Levantar requisitos funcionais e não funcionais relacionados ao armazenamento e manipulação de dados;
  - Elaborar o Modelo Entidade-Relacionamento (MER) e o Diagrama Entidade-Relacionamento (DER);
  - Implementar o banco de dados no MySQL, aplicando normalização, integridade referencial e segurança;
  - Criar índices e otimizar consultas para melhorar a performance;

- Objetivos de desenvolvimento:
  - Criação de um protótipo não funcional para validar a proposta visual e de navegação;
  - Implementação do front-end em React Native a partir do protótipo, inicialmente sem integração com o backend;
  - Construção do backend em Node.js + Express, responsável por processar as requisições e interagir diretamente com o banco de dados MySQL;
  - Integração do front-end com o backend, permitindo a comunicação em tempo real com o banco de dados e validando o funcionamento da solução como um todo;

3. Motivação

A motivação para a realização deste trabalho veio a partir da minha visão de que a ausência de uma base sólida em banco de dados pode vir a comprometer a formação e o desempenho de iniciantes na área de TI, impactando negativamente sua inserção no mercado de trabalho.

Um banco de dados bem projetado e otimizado não apenas garante o funcionamento eficiente de sistemas, mas também serve como exemplo prático para quem está aprendendo. Ao criar uma aplicação educacional, busco oferecer uma solução realista e funcional, que possa ser utilizada como recurso didático e como estudo de caso sobre boas práticas de modelagem e administração de bancos de dados.

4. Materiais

Este trabalho foi realizado utilizando os seguintes recursos tecnológicos, modelos ou práticas:

- Banco de dados MYSQL
- Gerenciador de Banco de Dados workbench;
- Ferramentas de programação Visual Studio Code;
- framework nodejs, baseada em typecript;

- framework react native, baseada em typecript;
- GitHub para armazenamento e versionamento de codigo.

5. Metodologia do Trabalho

Este trabalho foi realizado nas seguintes etapas :

- Levantamento de requisitos junto ao escopo da aplicação educacional;
- Modelagem conceitual por meio do Modelo Entidade-Relacionamento (MER);
- Implementação física do banco de dados no MySQL;
- Configuração de índices e otimização de consultas;
- Desenvolvimento da plataforma mobile:
  - Criação de um protótipo não funcional;
  - Criação do front-end a partir do prototipo(sem ligação com o back-end);
  - Criação do back-end integrando com o banco desenvolvido;
  - Integração do front-end com o back-end;
- Testes e validação, medindo desempenho, tempo de resposta das consultas, consistência dos dados e experiência do usuário no aplicativo, de forma a avaliar a adequação do banco de dados e a escalabilidade da arquitetura proposta.

II. Revisão de Tecnologias e Práticas

1. NodeJS

A tecnologia de Node.js para desenvolvimento de aplicações backend tem sido amplamente utilizada em projetos que demandam alto desempenho e escalabilidade. Por adotar um modelo event-driven e I/O não bloqueante, o Node.js permite lidar com múltiplas conexões simultâneas com baixo consumo de recursos.

No contexto deste trabalho, o Node.js é empregado na camada de servidor, responsável por processar as requisições do aplicativo mobile e interagir com o banco de dados MySQL. O framework Express.js é utilizado para estruturar rotas, controlar respostas HTTP e organizar o código em módulos reutilizáveis. Essa abordagem garante eficiência, segurança e facilidade de manutenção no backend do sistema.

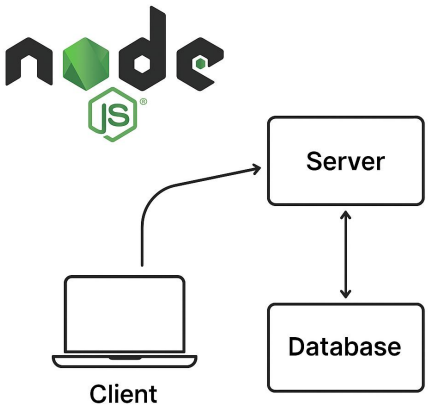


Figura 1. NodeJS

2. React Native

A tecnologia de React Native para desenvolvimento mobile multiplataforma tem sido utilizada com grande aceitação por empresas e desenvolvedores devido à sua capacidade de gerar aplicativos nativos para Android e iOS a partir de um único código JavaScript. Ela oferece desempenho próximo ao nativo, além de permitir a criação de interfaces modernas e responsivas.

Neste trabalho, o React Native é utilizado para construir a interface gráfica do usuário (front-end), exibindo conteúdos educacionais e recebendo respostas de exercícios. O aplicativo consome dados provenientes da API em Node.js, que, por sua vez, se comunica com o banco de dados MySQL. Essa arquitetura garante integração em tempo real e facilita a validação prática do modelo de dados desenvolvido.

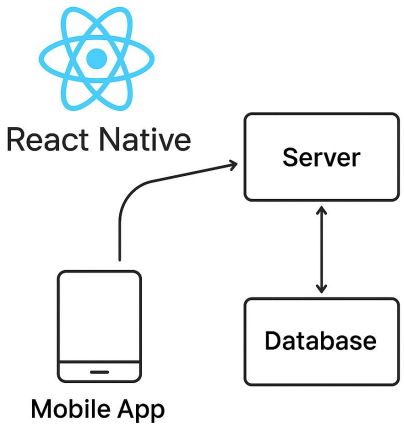


Figura 2. React Native

3. MySQL

A tecnologia de MySQL para gerenciamento de dados relacionais tem sido amplamente utilizada em ambientes corporativos e acadêmicos pela sua confiabilidade, simplicidade e compatibilidade com diferentes linguagens e plataformas. Baseado em SQL (Structured Query Language), o MySQL possibilita a criação de estruturas normalizadas, aplicação de índices, restrições de integridade e execução de consultas otimizadas.

No presente trabalho, o MySQL constitui o núcleo central da aplicação, responsável pelo armazenamento e gerenciamento das informações sobre usuários, conteúdos, exercícios e progresso dos alunos. O banco foi projetado com base em boas práticas de modelagem relacional, visando eficiência, integridade e segurança.

4. GitHub

A tecnologia de GitHub para controle de versão e colaboração tem sido amplamente utilizada em projetos de software, possibilitando o gerenciamento distribuído do código-fonte e o rastreamento de mudanças ao longo do ciclo de desenvolvimento. Através do sistema de versionamento Git, é possível registrar cada alteração, revisar históricos de commits e integrar equipes de forma segura e organizada.

Neste trabalho, o GitHub é utilizado para armazenar e versionar o código do backend (Node.js), frontend (React Native) e scripts do banco de dados (MySQL). Essa prática assegura a rastreabilidade das versões e facilita o acompanhamento da evolução do projeto, além de permitir integração contínua e backup remoto do código.

III. DESENVOLVIMENTO

1. Modelagem do banco

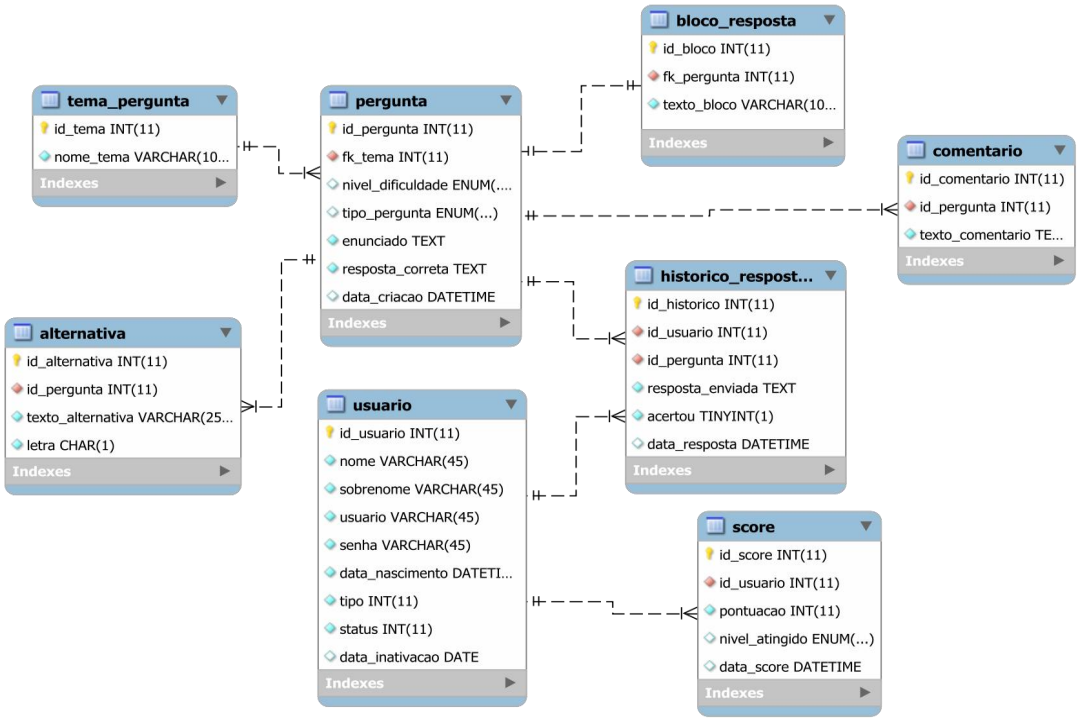
O desenvolvimento do projeto teve início com a criação do modelo MER (Modelo Entidade-Relacionamento). Inicialmente, identificou-se a necessidade de duas entidades principais: Usuário e Pergunta. Com as tabelas centrais definidas, procedeu-se à elaboração das tabelas auxiliares, responsáveis por complementar as funcionalidades do sistema.

Durante o processo de modelagem da tabela de perguntas, verificou-se a necessidade de representar as alternativas de respostas para as questões de múltipla escolha. Para atender a esse requisito, foi criada a tabela “alternativa”, responsável por armazenar as opções disponíveis para cada pergunta.

Em seguida, observou-se a conveniência de classificar as perguntas por temas, o que levou à criação da tabela “tema_perguntas”, permitindo a organização e o agrupamento das questões por área de conhecimento.

Na sequência, foram consideradas duas novas funcionalidades pedagógicas: a primeira consiste em fornecer dicas ao usuário em caso de erro, e a segunda em permitir que o aluno monte respostas manualmente, selecionando blocos de texto que formam um comando SQL. Para viabilizar essas funcionalidades, foram criadas, respectivamente, as tabelas “comentario” (destinada às dicas e observações sobre as questões) e “bloco_resposta” (responsável pelos blocos de texto que o usuário combina para compor a resposta).

Por fim, foram planejadas duas tabelas adicionais: “historico_resposta”, que registra todas as respostas submetidas pelos usuários, independentemente de acerto ou erro; e “score”, destinada a armazenar a pontuação obtida com base no desempenho e progresso individual. Ressalta-se que a implementação da tabela de pontuação ainda está em avaliação quanto à sua permanência no escopo final do projeto.



2. Criação do banco

Com o modelo MER finalizado, iniciou-se a implementação física do banco de dados no MySQL, adotando o nome de esquema “seipebd”. Essa etapa teve como objetivo transformar o modelo conceitual em estruturas relacionais concretas, garantindo a integridade referencial e o adequado relacionamento entre as entidades.

O primeiro passo consistiu na criação da tabela “usuario”, responsável por armazenar os dados dos alunos cadastrados no sistema. Essa tabela inclui informações como nome, sobrenome, login, senha e data de nascimento. Além disso, foram adicionados os campos “tipo” e “status”, que permitem distinguir diferentes perfis de usuários (por exemplo, aluno e administrador) e controlar a ativação ou desativação de contas. A definição de uma chave primária autoincremental (id_usuario) assegura a identificação única de cada registro.

Em seguida, foi criada a tabela “tema_pergunta”, destinada a organizar as questões por categorias temáticas, facilitando a navegação e filtragem no aplicativo. Cada tema é identificado por um id_tema, também definido como chave primária.

A tabela “pergunta” foi concebida como o núcleo do sistema de avaliação, contendo o enunciado, o nível de dificuldade e o tipo de pergunta — podendo ser de múltipla escolha ou de blocos de montagem de resposta. A relação entre perguntas e temas é estabelecida por meio da chave estrangeira fk_tema, garantindo consistência entre as tabelas. O campo resposta_correta armazena a solução esperada, sendo utilizado posteriormente para validação automática das respostas dos usuários.

Para as perguntas de múltipla escolha, foi implementada a tabela “alternativa”, que armazena as opções possíveis de resposta. Essa tabela é associada à entidade “pergunta” através da chave estrangeira id_pergunta, com a política ON DELETE CASCADE, permitindo a exclusão automática das alternativas vinculadas a uma pergunta removida.

No caso das perguntas interativas, em que o usuário deve montar comandos SQL por meio de blocos, foi criada a tabela “bloco_resposta”, que contém palavras ou expressões reutilizáveis, como “SELECT”, “FROM”, “WHERE”, entre outras. Esses blocos podem ser associados a perguntas específicas ou utilizados de forma genérica, dependendo do contexto do exercício.

Além das estruturas de perguntas e respostas, a modelagem contempla a tabela “comentario”, destinada a armazenar mensagens de feedback ou dicas fornecidas ao aluno após respostas incorretas. Essa funcionalidade visa reforçar o aspecto educativo da aplicação, promovendo aprendizado mesmo em casos de erro.

Para acompanhar o desempenho dos usuários, foi implementada a tabela “historico_respostas”, responsável por registrar todas as tentativas de resposta, incluindo o conteúdo enviado, o resultado (acerto ou erro) e a data de submissão. Essa tabela possui relacionamentos diretos com “usuario” e “pergunta”, possibilitando análises posteriores sobre evolução de desempenho e comportamento dos alunos.

Por fim, foi criada a tabela “score”, cuja função é armazenar a pontuação acumulada pelos usuários com base nas respostas corretas. Essa tabela mantém vínculo com o usuário e registra, além da pontuação total, o nível de dificuldade atingido, de forma a refletir o progresso educacional do aluno dentro da plataforma.

A estrutura final do banco de dados, portanto, apresenta um modelo relacional completo e coerente, permitindo a integração direta com o backend desenvolvido em Node.js e o aplicativo móvel em React Native. A combinação de integridade referencial, normalização e modularidade das tabelas garante flexibilidade e escalabilidade para futuras expansões do sistema.

3. Otimização do Banco de Dados

Com o modelo MER finalizado, iniciou-se a implementação física do banco de dados no MySQL, adotando o nome de esquema “seipebd”. Essa etapa teve como objetivo transformar o modelo conceitual em estruturas relacionais concretas, garantindo a integridade referencial e o adequado relacionamento entre as entidades.

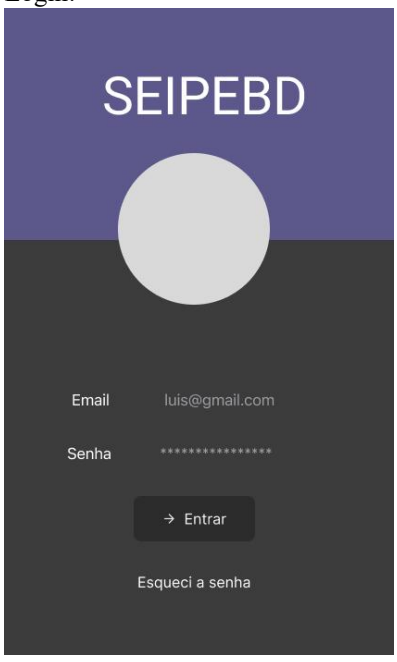
O primeiro passo consistiu na criação da tabela “usuario”, responsável por armazenar os dados dos alunos cadastrados no sistema. Essa tabela inclui informações como nome, sobrenome, login, senha e data de nascimento. Além disso, foram adicionados os campos “tipo” e “status”, que permitem distinguir diferentes perfis de usuários (por exemplo, aluno e administrador) e controlar a ativação ou desativação de contas. A definição de uma chave primária autoincremental (id_usuario) assegura a identificação única de cada registro.

Em seguida, foi criada a tabela “tema_pergunta”, destinada a organizar as questões por categorias temáticas, facilitando a navegação e filtragem no aplicativo. Cada tema é identificado por um id_tema, também definido como chave primária.

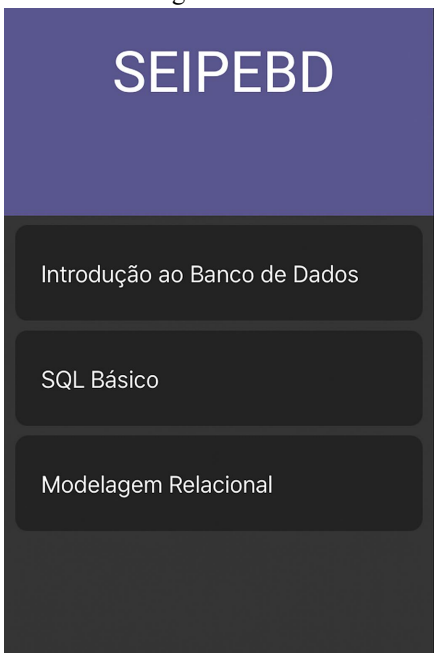
4. protótipo não funcional

Para os prototipo, foi ultilizado um designer simples, pois o ideal desse prototipo é entender como sera o funcionamento do portal. Não foi feito prototipo de todas as telas, somente das consideradas principais no projeto.

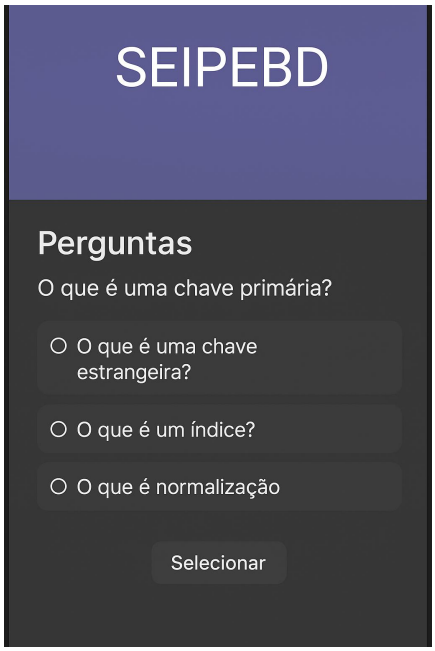
Login:



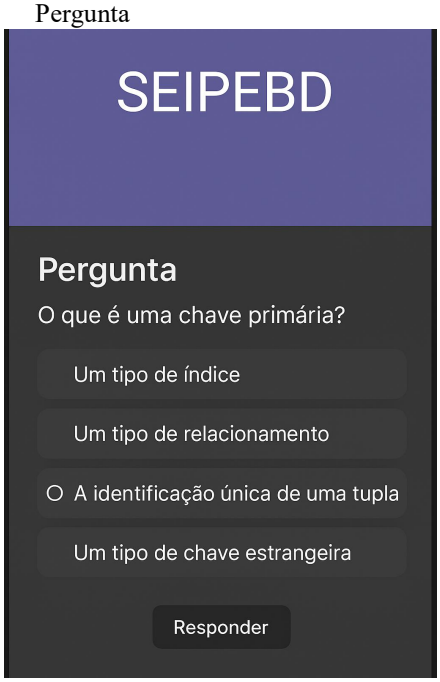
Escolha de categoria



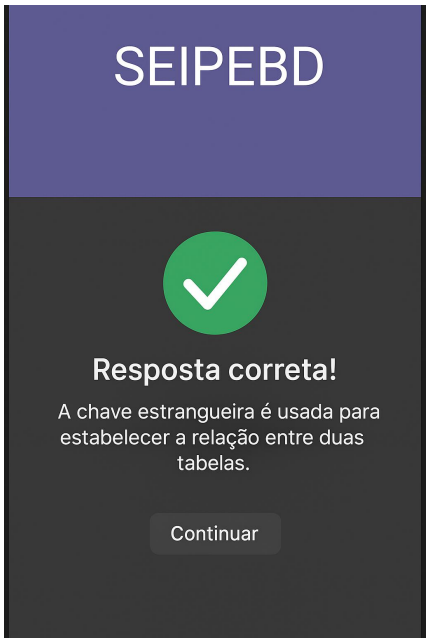
Escolha de perguntas:



Pergunta



Resultado:

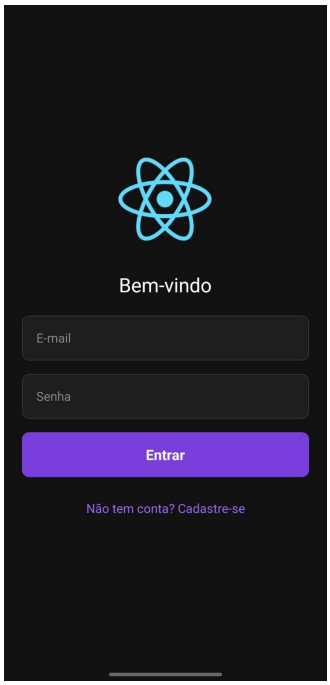


5. Criação do front-end

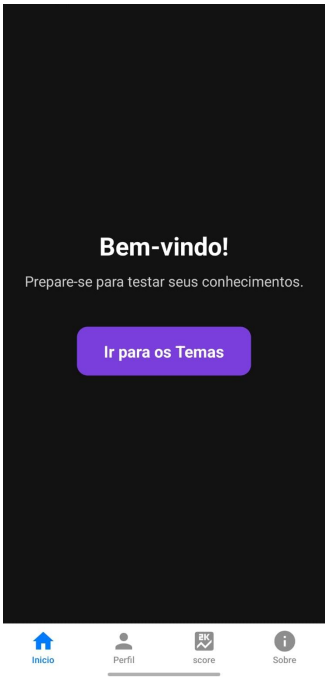
Na etapa de desenvolvimento do front-end teve como foco a construção da interface do usuário utilizando React Native aliado ao ecossistema Expo, permitindo uma experiência multiplataforma sem a necessidade de configurações complexas. O objetivo principal foi criar um aplicativo intuitivo, visualmente coeso e funcional, capaz de interagir de forma eficiente com o backend e oferecer ao aluno uma navegação fluida durante a resolução das questões.

Inicialmente, foram definidas as telas essenciais do aplicativo: login, cadastro, seleção de temas, lista de perguntas, questões de múltipla escolha, questões por blocos, resultado, histórico de respostas, perfil do usuário e pontuação (score). Na criação do front-end, foi feito modificações no leyout.

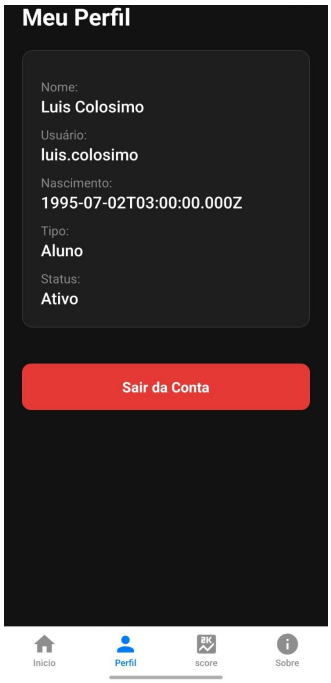
Tela de login:



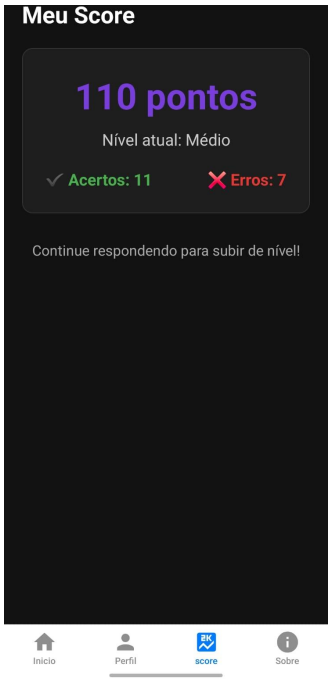
Inico:



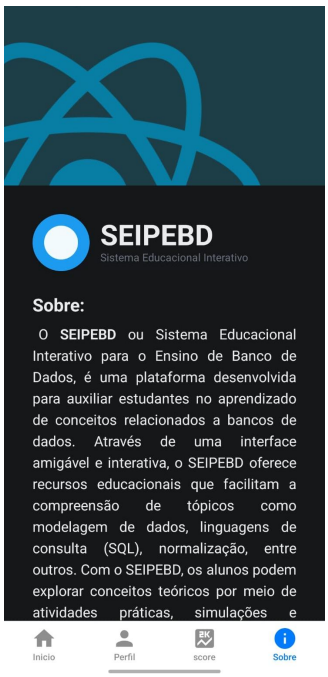
Perfil:



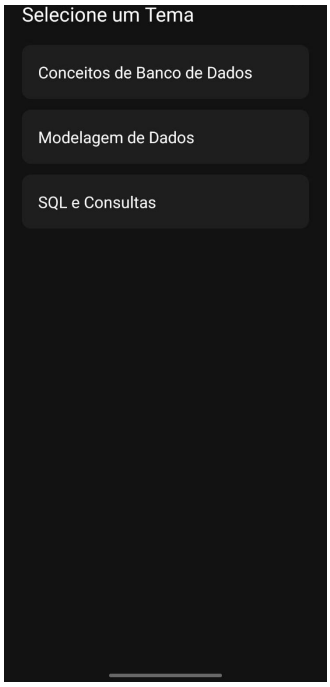
Score:



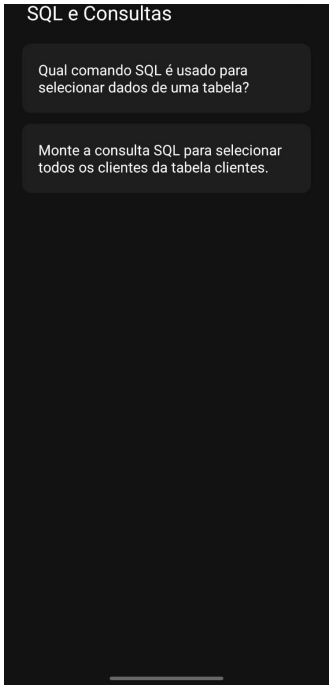
Sobre:



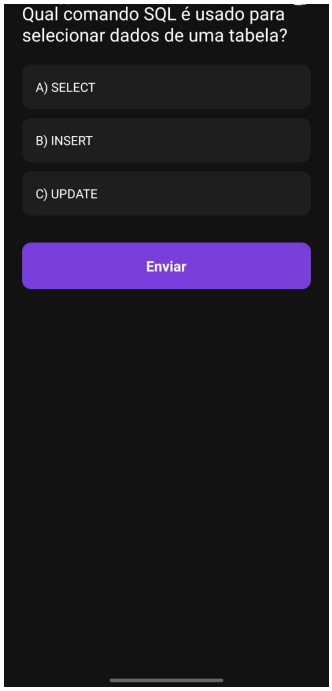
Escolha de temas:



Selecionar pergunta:

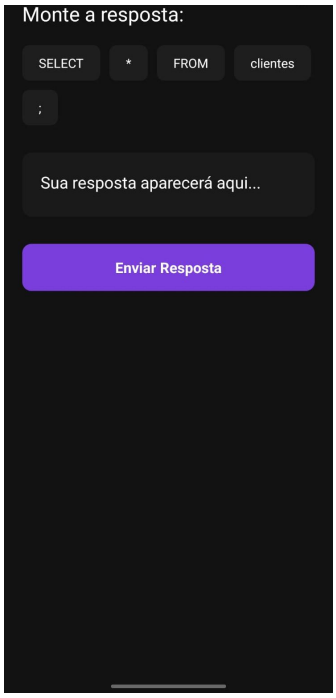


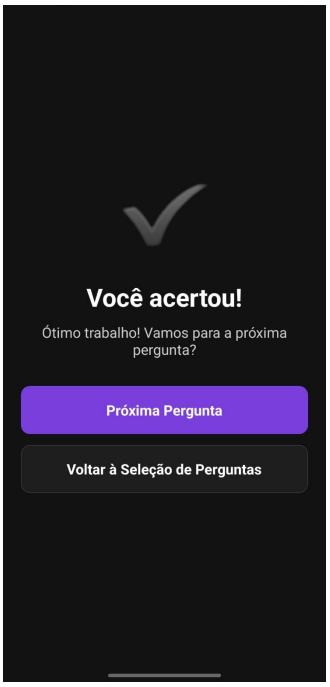
Resposta com Escolhas:



Resposta com blocos:

Resposta certa:





Resposta Errada:

6. Criação do back-end

O backend foi implementado utilizando o framework NestJS, que possibilita uma arquitetura modular, escalável e bem estruturada. A aplicação adota o banco de dados MySQL, utilizando a biblioteca TypeORM como ORM para mapear as entidades do sistema para tabelas e gerenciar automaticamente operações como inserções, consultas, atualizações e exclusões.

A estrutura de desenvolvimento foi organizada em módulos, sendo o principal deles o módulo User, que concentra rotas, controllers, services e repositories. A partir das entidades definidas no modelo MER, foram criadas classes representando as tabelas do banco.

IV. Resultados e Análise

1. Funcionamento Geral do Sistema

O sistema desenvolvido se apoia fortemente na estrutura relacional criada no MySQL. Todas as funcionalidades essenciais — desde o login do usuário até o registro das respostas, cálculo de score e filtragem de temas — dependem diretamente das consultas e relacionamentos definidos no banco..

Durante os testes, verificou-se que:

- A modelagem atendeu perfeitamente às necessidades do aplicativo.
- A integridade referencial evitou registros inconsistentes.
- As consultas SQL responderam rapidamente, mesmo com múltiplos joins.
- A normalização das tabelas reduziu redundância e diminuiu o tamanho da base.

A correta construção do banco foi essencial para garantir a confiabilidade dos dados e o bom desempenho do sistema como um todo.

2. Qualidade da Modelagem Relacional

A modelagem proposta inicialmente evoluiu ao longo do desenvolvimento e demonstrou ser adequada às demandas reais. Abaixo estão seus principais destaques:

2.1 Separação eficiente das entidades

As tabelas criadas (usuário, tema, pergunta, alternativa, bloco_resposta, comentário, histórico_respostas e score) seguem os princípios de 1FN, 2FN e 3FN, evitando duplicação desnecessária.

2.2 Uso correto de chaves estrangeiras

- pergunta → tema
- alternativa → pergunta
- bloco_resposta → pergunta
- historico_respostas → usuario e pergunta
- score → usuario

Esse encadeamento garante consistência lógica entre registros. Ao excluir uma pergunta, por exemplo, suas alternativas são removidas automaticamente graças ao ON DELETE CASCADE.

2.3 Adequação às regras de negócio

- A estrutura do banco conseguiu representar:
- Perguntas de múltipla escolha
- Perguntas de montagem SQL
- Feedback pedagógico (comentário)
- Temas agrupadores
- Registro histórico de tentativas
- Pontuação progressiva

O modelo, portanto, não só atende às necessidades atuais, como também é flexível para extensões futuras.

3. Desempenho e Otimização das Consultas SQL

Com o banco implementado, realizou-se uma análise de desempenho durante o uso integrado com o aplicativo.

3.1 Consultas essenciais

As consultas mais frequentes foram:

- Listagem de temas: SELECT * FROM tema_pergunta
- Listagem de perguntas por tema: SELECT * FROM pergunta WHERE fk_tema = ?
- Alternativas de pergunta: SELECT * FROM alternativa WHERE id_pergunta = ?
- Blocos de montagem SQL: SELECT * FROM bloco_resposta WHERE id_pergunta = ?
- Histórico de respostas: Join entre usuario, pergunta e historico_respostas

3.2 Desempenho observado

- Todas as consultas apresentaram baixa latência.
- Mesmo consultas com joins múltiplos retornaram em tempo reduzido.
- O banco se mostrou estável, sem gargalos perceptíveis.

3.3 Benefícios da modelagem

A performance elevada foi diretamente influenciada por:

- Tabelas normalizadas.
- Uso correto de chaves primárias indexadas.
- Volumes pequenos, mas estruturados de forma eficiente.
- Operações CRUD otimizadas pelo TypeORM.

V. CONCLUSÃO

O trabalho realizado permitiu avaliar de forma prática todas as etapas do desenvolvimento de um sistema educacional baseado no uso intensivo de banco de dados, desde a modelagem conceitual até a implementação final e integração com um aplicativo móvel. O projeto demonstrou que uma modelagem bem estruturada, aliada a boas práticas de normalização e ao uso adequado de relacionamentos, é capaz de sustentar uma aplicação completa, responsiva e funcional, garantindo consistência dos dados e suporte adequado à lógica pedagógica proposta.

A criação das tabelas, a definição das chaves primárias e estrangeiras, e o planejamento das relações mostraram-se fundamentais para possibilitar as funcionalidades desejadas: autenticação de usuários, registro de histórico, feedback pedagógico, classificação por temas, armazenamento de alternativas e blocos de código, além da atribuição de pontuações. Verificou-se que um banco de dados relacional, quando bem desenhado, é capaz de desempenhar não apenas o papel de repositório, mas também de núcleo operacional de um sistema de aprendizagem.

Além disso, a integração do backend em Node.js/NestJS com o banco de dados MySQL e o aplicativo em React Native comprovou a robustez da solução, evidenciando como as consultas SQL e as relações entre tabelas sustentam diretamente a experiência do usuário. O resultado final cumpre o objetivo pedagógico de auxiliar no ensino de SQL e reforça a importância de um banco de dados bem planejado em sistemas educacionais interativos.

1. Dificuldade Encontradas

Durante o desenvolvimento deste trabalho, diversas dificuldades surgiram, tanto técnicas quanto conceituais. Entre elas, destacam-se:

1.1 Modelagem do Banco de Dados

A definição exata de quais entidades seriam necessárias e quais funcionalidades demandariam novas tabelas exigiu revisões e refinamentos constantes. Particularmente:

- A separação entre perguntas de múltipla escolha e perguntas por blocos exigiu ajustes no MER.
- A inclusão de tabelas como comentários e blocos de resposta surgiu apenas após a identificação de necessidades pedagógicas específicas.
- A tabela de score passou por várias versões até que ficasse aderente ao restante do modelo.

1.2 Definição da Integridade Referencial

Outra dificuldade esteve na escolha adequada das relações e definições de ON DELETE CASCADE ou RESTRICT.
Em alguns cenários, remover perguntas acarretava a remoção de alternativas e blocos, sendo necessário cuidar para não remover registros essenciais ao histórico do aluno.

1.3 Integração Backend ↔ Banco

Embora o TypeORM facilite o mapeamento das tabelas, surgiram desafios como:

- Configuração múltipla de entidades.
- Ajustes nos relacionamentos importados.
- Conversão correta dos retornos para os modelos usados no front-end.
- Erros de status code (404, 400) devido a chamadas incorretas ou rotas mal definidas.

1.4 Integração Front-end ↔ Backend

Do lado do aplicativo, dificuldades incluíram:

- Controle da navegação usando Expo Router.
- Cache e carregamento assíncrono dos dados.
- Múltiplas requisições simultâneas por erros de uso do useEffect.
- Rotações incorretas ao enviar parâmetros entre telas.

1.5 Sincronização de Estados

O fluxo de perguntas (seleção → resposta → resultado → próxima pergunta) exigiu tratamento cuidadoso do estado global, especialmente ao considerar temas, tipos de perguntas e tentativas.

Mesmo diante desses desafios, todas as etapas foram superadas com aprendizado significativo, resultando em uma aplicação funcional e coerente com o objetivo proposto.

2. Aplicabilidade do Trabalho

O sistema desenvolvido possui alta aplicabilidade em diferentes contextos educacionais e profissionais, especialmente por se tratar de uma plataforma de aprendizagem fundamentada em banco de dados. Entre suas principais possibilidades de aplicação, destacam-se:

2.1 Instituições de Ensino

O aplicativo pode ser utilizado como:

- Ferramenta complementar em disciplinas de Banco de Dados.
- Plataforma de exercícios práticos de SQL.
- Ambiente de avaliação automática e personalizada.
- Recurso de reforço e revisão para provas.

2.2 Ambientes Corporativos

Empresas que trabalham com:

- Análise de dados,
- Engenharia de dados,
- Administração de banco de dados,
- BI e ETL,

podem utilizar a plataforma para treinamento interno de novos colaboradores, avaliação de conhecimento ou nivelamento técnico.

2.3 Plataformas de Ensino Online

O sistema pode ser integrado a LMS (Learning Management Systems) como:

- Moodle
- Canvas
- Blackboard

funcionando como módulo de exercícios práticos com banco de dados.

2.4 Processos Seletivos Técnicos

A plataforma pode ser usada para:

- Avaliar habilidades SQL em pré-seleções.
- Aplicar testes automatizados com correção imediata.
- Registrar desempenho de candidatos de forma estruturada.

2.5 Treinamento Personalizado

Com o histórico de respostas, abre-se espaço para:

- Recomendação de temas com mais dificuldade.
- Geração automática de trilhas de estudo.
- Avaliação contínua com base no progresso real do aluno.

3. Link para o projeto

https://github.com/luisfelipecolosimo/TCC-Unicamp_SEIPEBD

BIBLIOGRAFIA

AGUIAR, Adilson. *Modelagem de Dados: conceitos, melhores práticas e técnicas*. São Paulo: Novatec, 2020.

BEIGHLEY, Lynn; MORRISON, Michael. *SQL para Leigos*. 3. ed. Rio de Janeiro: Alta Books, 2019.

ELMASRI, Ramez; NAVATHE, Shamkant. *Sistemas de Banco de Dados*. 7. ed. São Paulo: Pearson, 2019.

HEUSER, Carlos Alberto. *Projeto de Banco de Dados*. 6. ed. Porto Alegre: Bookman, 2022.

MySQL. *MySQL 8.0 Reference Manual*. Oracle, 2024. Disponível em: https://dev.mysql.com/doc/. Acesso em: 10 jan. 2025.

DATE, C. J. *Introdução a Sistemas de Banco de Dados*. 8. ed. Rio de Janeiro: Elsevier, 2004.

SILBERSCHATZ, Abraham; KORTH, Henry; SUDARSHAN, S. *Sistema de Banco de Dados*. 7. ed. São Paulo: McGraw-Hill, 2020.

HUNT, Andrew. *Clean Code em SQL*. Nova Iorque: Prentice Hall, 2021.

NESTJS. *NestJS Documentation*. 2024. Disponível em: https://docs.nestjs.com/. Acesso em: 10 jan. 2025.

NODE.JS FOUNDATION. *Node.js Documentation*. 2024. Disponível em: https://nodejs.org/en/docs. Acesso em: 10 jan. 2025.

REACT NATIVE. *React Native Official Documentation*. Meta, 2024. Disponível em: https://reactnative.dev/docs. Acesso em: 10 jan. 2025.

EXPO. *Expo Documentation*. 2024. Disponível em: https://docs.expo.dev/. Acesso em: 10 jan. 2025.

AXIOS. *Axios GitHub Repository*. 2024. Disponível em: https://github.com/axios/axios. Acesso em: 10 jan. 2025.

FOWLER, Martin. *Patterns of Enterprise Application Architecture*. Boston: Addison-Wesley, 2002.

PRESSMAN, Roger; MAXIM, Bruce. *Engenharia de Software: Uma Abordagem Profissional*. 9. ed. Porto Alegre: AMGH, 2020.

WAZLAWICK, Raul Sidnei. *Análise e Projeto de Sistemas de Informação Orientados a Objetos*. Rio de Janeiro: Elsevier, 2014.

TANENBAUM, Andrew. *Arquitetura de Sistemas Operacionais*. 3. ed. São Paulo: Pearson, 2020.