Protótipo de aplicação web para plataforma de cursos online

Evandro José da Silva Mariano[1]

[1]Universidade Federal do Pará (UFPA) – Catanhal, PA – Brasil

{Evandro}@evandromariano49@gmail.com

Abstract. *With the growing demand for distance learning in recent years, it has become essential to adopt efficient technological solutions for managing online courses. This work presents a prototype of a web platform designed for offering free courses, utilizing modern frameworks such as Next.js for both back-end and front-end development, and MongoDB Atlas for data persistence. The platform provides features such as user registration, lesson viewing, assessments, and certificate issuance, ensuring a responsive and accessible design.*

Resumo. *Com o crescimento da demanda por ensino a distância nos últimos anos, tornou-se essencial adotar soluções tecnológicas eficientes para a gestão de cursos online. O presente trabalho apresenta um protótipo de uma plataforma web voltada para a disponibilização de cursos gratuitos, utilizando frameworks modernos como Next.js para back-end e front-end e MongoDB Atlas para a persistência de dados. A plataforma oferece funcionalidades como cadastro de usuários, visualização de aulas, realização de avaliações e emissão de certificados, garantindo um design responsivo e acessível.*

1. Introdução

Os avanços tecnológicos impulsionaram significativamente a expansão do ensino a distância (EAD) no Brasil e no mundo, tornando-o uma alternativa cada vez mais consolidada no cenário educacional. Esse crescimento foi amplificado pela pandemia de 2019, que, segundo [Pereira 2023], acelerou a adoção do ensino remoto e redefiniu as dinâmicas da educação no país. Nas universidades particulares, o modelo online não apenas se consolidou, mas também superou o ensino presencial.

A educação a distância tem se consolidado como uma alternativa viável e eficiente para a democratização do ensino no Brasil. Especialmente nos últimos anos, a modalidade de ensino tem apresentado um crescimento expressivo. Um fator de grande relevância que serviu para impulsionar a nova modalidade de ensino foi a adaptação ao contexto da pandemia de COVID-19 de acordo com [Junior and de Meira 2023] o avanço da Educação a Distância (EaD) não se limitou a uma simples resposta à crise pandêmica, ele também marcou uma transformação significativa na estrutura e no modelo de ensino.

Conforme apontado por [Pereira 2023], entre 2011 e 2021, a participação de ingressantes no ensino a distância aumentou de 18,4% para 62,8%, evidenciando uma grande transformação na forma como a educação superior é ofertada e consumida. Esse cenário reflete não apenas uma resposta às restrições impostas pela pandemia, mas também a crescente aceitação do EAD como uma alternativa eficaz e acessível. A

evolução desse modelo reforça sua importância no contexto educacional brasileiro, demonstrando seu potencial para ampliar o acesso ao ensino superior e adaptar-se às novas demandas da sociedade. De acordo com dados do Censo da Educação Superior 2021, entre 2011 e 2021, o número de ingressantes em cursos superiores na modalidade de educação a distância (EaD) aumentou em 474%, enquanto o número de ingressantes em cursos presenciais diminuiu em 23,4% [INEP 2022]. Estes dados demonstram de forma clara que cada vez mais os alunos estão optando pelo modelo de ensino a distância, refletindo uma mudança significativa nas preferências educacionais.

Diante desse cenário, este trabalho justifica-se pela necessidade de compreender e aprimorar as plataformas de ensino a distância, garantindo maior acessibilidade, eficiência e engajamento dos estudantes. A adoção de tecnologias responsivas e funcionalidades intuitivas, como suporte para diferentes dispositivos e modos de exibição personalizáveis, é essencial para proporcionar uma experiência de aprendizado mais inclusiva e eficiente.

O principal objetivo deste trabalho é projetar e implementar uma plataforma de ensino online intuitiva e responsiva, que possibilite aos usuários o acesso a cursos, a visualização de aulas e a realização de avaliações, com a obtenção de certificados, promovendo assim um ambiente de aprendizado acessível, inclusivo e eficiente. Para alcançar esse objetivo busca-se criar um ambiente de aprendizado centralizado, de fácil acesso em qualquer dispositivo por meio da plataforma web, que seja capaz de atender às necessidades dos mais diversos perfis de alunos, assim pretende-se com este protótipo:

- Desenvolver um sistema de login e cadastro para que os usuários possam criar contas e acessar funcionalidades exclusivas da plataforma.
- Implementar páginas que permitam a visualização e inscrição em cursos disponíveis, apresentando os conteúdos de forma organizada e atrativa.
- Criar um sistema de avaliação por meio de questionários vinculados aos cursos, permitindo aos usuários acompanhar seu progresso e obter certificados.
- Garantir que todas as funcionalidades sejam responsivas, adaptando-se a diferentes dispositivos e resoluções de tela, incluindo suporte ao modo escuro.
- Incorporar mecanismos de acessibilidade e usabilidade para atender às demandas de diferentes perfis de usuários.
- Implementar um perfil de usuário que registre o histórico de cursos, progressos nas avaliações e certificados obtidos, proporcionando uma visão personalizada do desempenho do aluno.
- Criar um painel de administração para os administradores da plataforma, permitindo assimo controle sobre a criação de cursos, inscrições, avaliação de usuários e relatórios de progresso de maneira facilitada.

2. Metodologia

2.1. Front-end

O desenvolvimento do front-end da aplicação foi realizado utilizando o framework Next.js [Vercel 2024], que se destaca por oferecer uma grande variedade de recursos para a criação de aplicações web modernas. Além disso, o Next.js também foi empregado no desenvolvimento do back-end, o que possibilitou uma abordagem full-stack integrada e centralizada. Essa escolha foi motivada pela flexibilidade do framework, que suporta tanto a renderização no lado do servidor (Server-Side Rendering, SSR) quanto a geração de páginas estáticas (Static Site Generation, SSG). Estes recursos são particularmente vantajosos para o desenvolvimento de aplicações web do tipo Single Page Application (SPA), ao usar o (SSR) o conteúdo é renderizado no lado do servidor e enviado para o cliente como um HTML pronto, isso faz com que a renderização das páginas torne-se mais rápida além permitr que mecanismos de busca indexem o site mais rapidademente. Com o static site generation, as páginas são geradas durante a construção do site e transmitidadas diretamente para o cliente. A combinação de ambas soluções permite usar a abordagem mais adequada para partes específicas da página.

Esses recursos são particularmente vantajosos para o desenvolvimento de aplicações web do tipo Single Page Application (SPA). Com o Server-Side Rendering (SSR), o conteúdo é renderizado no servidor e enviado ao cliente como um HTML completo. Essa abordagem torna a renderização das páginas mais rápida e facilita a indexação do site por mecanismos de busca como o Google levando em consideração que de acordo com [Clarke 2016], a otimização para motores de busca é essencial para garantir a visibilidade de sites e melhorar o tráfego orgânico. Em contrapartida, com o Static Site Generation (SSG), as páginas são pré-geradas durante o processo de build do site no servidor e transmitidas diretamente ao cliente, isso proporciona um tempo de carregamento extremamente rápido. Essa combinação possibilita selecionar a estratégia mais adequada para cada parte específica da aplicação em questão, otimizando desempenho da aplicação.

2.2. Tailwind CSS

Foi utilizado o Tailwind CSS [Labs 2024] para a estilização das páginas. Essa escolha se deveu à sua abordagem utilitária, que oferece classes pré-definidas para personalização de estilos diretamente no HTML, além de já ser integrado no framework NextJS. Uma das principais vantagens do uso do Tailwind CSS é a semântica clara e objetiva das suas classes utilitárias, o que elimina a necessidade de criar nomes específicos para classes e variáveis CSS personalizadas. Essa abordagem contribui significativamente para a padronização da escrita de código em toda a equipe de desenvolvimento, e reduz divergências o que aumenta a consistência no projeto.

Ademais, esse framework oferece uma grande flexibilidade que o torna diferente de outras bibliotecas ou frameworks como por exemplo o Bootstrap. Enquanto o este fornece classes predefinidas e estilizadas e menos personalizáveis, o Tailwind possiblita não apenas o uso de suas classes utilitárias nativas, mas também a criação de classes personalizadas que atendem a requisitos específicos do projeto, utilizando ferramentas como @apply. Por exemplo, uma classe como flex pode ser aplicada diretamente a qualquer elemento HTML que precise de um comportamento flexível, sem a necessidade de criar novas classes CSS específicas para cada instância.

Tal abordagem reduz o acoplamento ao CSS tradicional e promove um fluxo de trabalho mais ágil. Além disso, o Tailwind incentiva o uso consistente de suas classes, o que facilita o entendimento do código por outros membros da equipe, mesmo aqueles que estão se juntando ao projeto em estágios mais avançados.

2.3. Back-end

O back-end da aplicação também foi desenvolvido utilizando o framework Next.js, dessa maneira foi possível centralizar tanto as funcionalidades do front-end quanto do back-end em um único ambiente de desenvolvimento. Essa versatilidade e capacidade de fornecer uma solução full-stack eficiente, facilitou o processo de renderização tanto no lado do cliente quanto no servidor. Essa centralização permite um fluxo mais ágil de desenvolvimento e facilita a manutenção do código, ao reunir recursos e ferramentas necessários para ambos os lados em um único framework. No Next.js, a criação de APIs REST pode ser realizada utilizando a funcionalidade de **API Routes**. Essas rotas de API são armazenadas dentro da pasta `pages/api`, onde cada arquivo TypeScript se torna automaticamente uma rota de API.

Ademais, com o intuito de implementar os requisitos de navegação e roteamento da aplicação, foi adotada a biblioteca NextRoutes [X. 2024]. Com essa ferramenta, foi possível criar rotas dinâmicas de maneira eficiente, o que resultou em uma estrutura de navegação mais flexível e organizada, alinhada ao conceito de Single Page Application (SPA). O uso do NextRoutes foi crucial para atender aos requisitos da aplicação, pois assegurou que as diversas funcionalidades e páginas da plataforma fossem acessíveis de forma ágil e sem comprometer o desempenho, evitando a sobrecarga da arquitetura com rotas desnecessárias.

2.4. Typescript

O TypeScript [TypeScript Team 2024] foi a principal linguagem utilizada no desenvolvimento deste projeto, considerando que o framework Next.js a adota nativamente. Tanto o front-end quanto o back-end foram implementados com essa linguagem. Uma das principais vantagens do TypeScript, por ser orientado a objetos, é a possibilidade de criar interfaces para facilitar a comunicação com o banco de dados. Essa abordagem promove uma aplicação mais legível, com estruturas bem definidas, o que resulta em maior facilidade para manutenção e escalabilidade. De acordo com [Cherny 2019], o TypeScript combina a flexibilidade do JavaScript com a segurança proporcionada pela tipagem estática, permitindo aos desenvolvedores criar sistemas robustos e menos propensos a erros.

O TypeScript desempenhou um papel crucial no desenvolvimento, especialmente quando se considera a interação com o banco de dados e a validação de dados. Como um superset do JavaScript, o TypeScript introduz tipagem estática, o que permite detectar erros mais cedo e garantir que o código seja mais robusto e confiável. Quando combinado com ferramentas como ORMs (Object-Relational Mapping), o TypeScript proporciona um nível adicional de segurança, organização e clareza no código, facilitando a comunicação entre a aplicação e o banco de dados. Essa tipagem rigorosa contribui para a criação de interfaces mais precisas e seguras, além de otimizar a manutenção e evolução da aplicação ao longo do tempo.

Ao utilizar o Prisma na aplicação para a integração com o banco de dados, a biblioteca já fornece o mapeamento das interfaces de forma automática. Isso significa que, ao

definir o modelo de dados no Prisma, ele gera as interfaces correspondentes, facilitando a comunicação entre a aplicação e o banco de dados. Essa abordagem não só agiliza o processo de desenvolvimento, mas também assegura que as interações com o banco de dados sejam seguras e consistentes, já que as interfaces geradas são tipadas e validadas em tempo de compilação pelo TypeScript. O que resulta em um código mais organizado e menos propenso a erros, permitindo uma maior eficiência no desenvolvimento e manutenção da aplicação.

2.5. Banco de Dados

O MongoDB foi escolhido como banco de dados para gerenciar as informações dos cursos e dos usuários, abrangendo cadastro, progresso e conclusão de cursos. Por ser um banco não relacional orientado a documentos, ele utiliza um formato baseado em JSON, o que facilita a manipulação e integração com linguagens de programação como JavaScript e TypeScript, garantindo adequação às necessidades de aplicações web modernas. Além disso, sua flexibilidade e escalabilidade foram fatores determinantes para o projeto. Outra vantagem significativa foi a possibilidade de utilizar o plano gratuito do MongoDB Atlas para hospedar a base de dados na web sem custos iniciais de infraestrutura, permitindo a implementação da aplicação de forma econômica [MongoDB Inc. 2024].

A integração com o framework Next.js foi realizada por meio do Prisma, que é uma biblioteca que oferece uma abordagem eficiente para gerenciar operações no banco de dados em aplicações JavaScript e TypeScript. Esta biblioteca simplifica tarefas como consultas, inserções e atualizações, reduzindo a complexidade do back-end e otimizando o desempenho. Além disso, ele oferece suporte à migração entre diferentes bancos de dados, aumentando a flexibilidade e a adaptabilidade da solução [Prisma 2024].

A base de dados foi projetada com diversas collections para estruturar as informações necessárias para a aplicação. Ela inclui modelos que representam cursos, categorias, tags, instrutores, módulos, aulas, questões de avaliação, todos interrelacionados entre si de forma a refletir a complexidade e os fluxos do sistema. Cada modelo está vinculado a outros por meio de relações específicas, garantindo a integridade e a organização dos dados.

O MongoDB usa a estrutura de coleções para armazenar os dados, estas que são equivalentes às tabelas em bancos de dados relacionais. No projeto, foram criadas coleções específicas para armazenar diferentes tipos de informações essenciais à aplicação. Cada coleção é projetada para uma parte do sistema, e os modelos de dados foram definidos de forma a garantir a flexibilidade e escalabilidade. As principais coleções são descritas abaixo:

- **Categories (Categorias):** Contém informações sobre as categorias de cursos, como o nome da categoria. Cada categoria pode ter vários cursos relacionados.
- **Course (Cursos):** Contém informações sobre os cursos disponíveis na plataforma, como título, descrição, duração, categoria e instrutores. Relaciona-se com outras coleções, como *Instructors* (instrutores) e *Modules* (módulos).
- **Instructor (Instrutores):** Armazena informações sobre os instrutores, como nome, formação e perfil, e está relacionado com a coleção de *Courses*, pois cada curso possui um ou mais instrutores.

- **Module (Módulos):** Cada curso pode ser dividido em vários módulos. Esta coleção contém informações sobre os módulos, como título, número do módulo e a lista de *Classes* (aulas) associadas.
- **Lesson (Aulas):** Representa as aulas dentro de um módulo. Ela contém detalhes como o título da aula, número da aula, vídeo associado e outros recursos de conteúdo.
- **Questions (Questões de Avaliação):** Armazena as questões de avaliação de cada curso, incluindo as opções de resposta.
- **QuestionOptions (Opções de Resposta):** Relacionada com as questões de avaliação, contém as diferentes opções de resposta, incluindo a informação sobre qual delas é correta.
- **Users (Usuários):** Contém informações sobre os usuários da plataforma, como nome e cursos aos quais estão inscritos.
- **Subscription (inscrições):** Armazena os cursos aos quais os usuários estão inscritos, bem como informações sobre seu progresso (como as aulas completadas e os módulos em andamento).
- **Tags (Etiquetas):** Armazena as etiquetas associadas aos cursos. Cada curso pode ter várias etiquetas, ajudando na categorização e filtragem.

Cada model do Prisma faz uma referência a uma colletion do mongoDB. Ainda que o MongoDB não seja um banco de dados relacional, a modelagem busca adotar boas práticas de organização e estruturação dos dados afim de garantir consistência e facilidade de consulta. Cada coleção foi projetada para armazenar documentos contendo informações relevantes e de maneira a permitir consultas eficientes, reduzindo a necessidade de operações complexas em tempo de execução.

A escolha do Prisma como ORM (Object-Relational Mapping) auxilia nesse processo e oferece uma abstração que facilita tanto a interação com o banco quanto a manutenção e expansão do modelo. Tal abordagem facilita a evolução do banco de dados ao permitir a adição de novos relacionamentos, a atualização de campos existentes e até mesmo a migração para outro banco de dados, caso necessário. Além de proporcionar recursos avançados, como o versionamento do esquema do banco de dados, que oferece maior controle e rastreabilidade durante o desenvolvimento e manutenção da aplicação. O Prisma além de abstrair a interação com o banco de dados, também simplifica a manutenção e a expansão do modelo ao integrar práticas modernas de desenvolvimento, como geração automática de código, validação de esquemas e permitindo o código que gera o modelo do banco está centralizado na aplicação principal.

Na Figura 1, observa-se como as coleções do MongoDB foram organizadas para atender às necessidades específicas do projeto, considerando os vínculos entre cursos, usuários, categorias, módulos e outras entidades relacionadas.

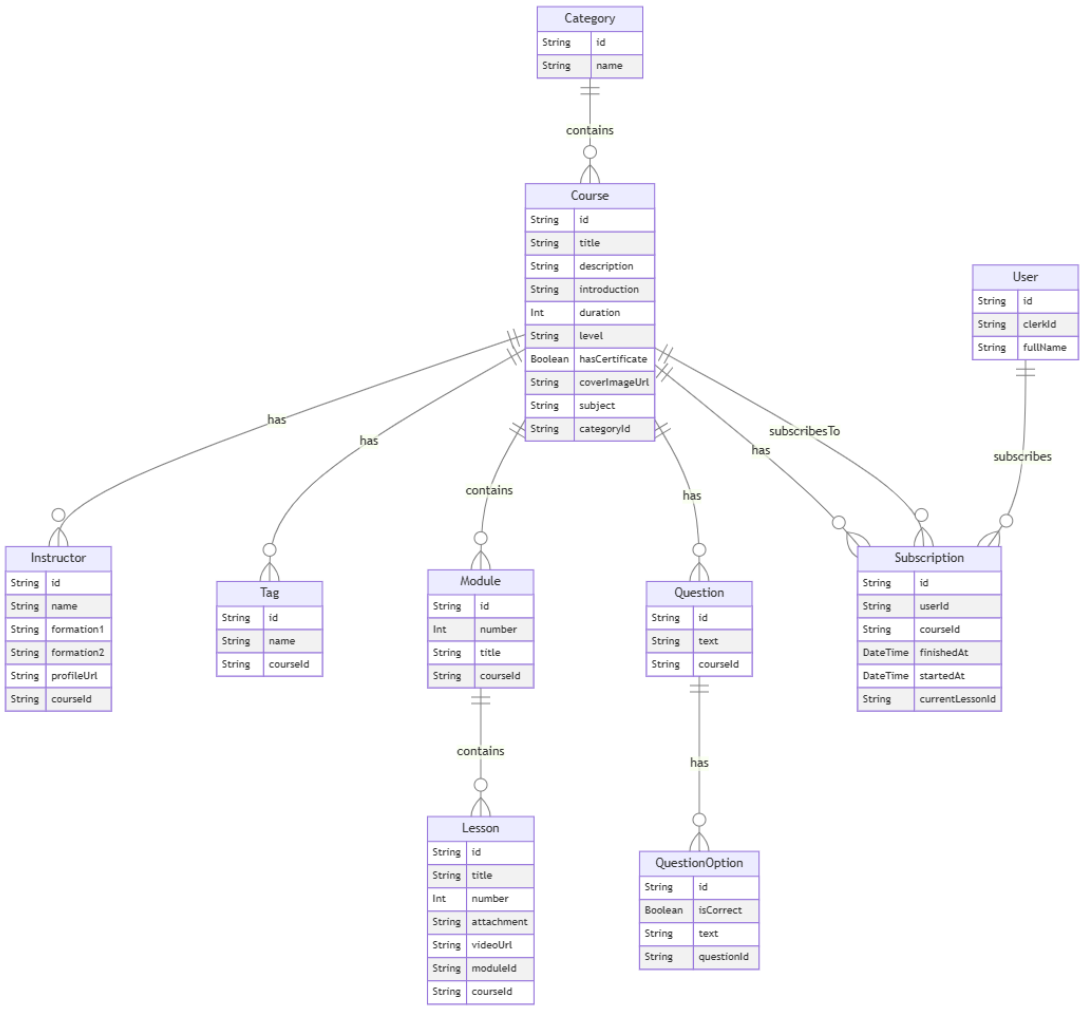

Figura 1. Modelo do Banco de Dados
Fonte: O autor.

2.6. Autenticação de Usuários

Para garantir um controle eficiente dos usuários na aplicação e proporcionar uma experiência mais fluida e intuitiva, foi implementada uma integração com bibliotecas de terceiros que simplificam o processo de autenticação e criação de contas. Especificamente, utilizou-se a biblioteca Clerk [Clerk 2024], uma solução robusta em JavaScript para gerenciamento de autenticação e identidade de usuários.

Essa biblioteca elimina a complexidade envolvida na criação e manutenção de sistemas de autenticação, oferecendo recursos avançados como login com provedores sociais, autenticação multifator e integração com tokens JWT. Isso amplia as opções de autenticação para o usuário, permitindo que ele utilize contas já existentes em provedores como Google ou GitHub para acessar a plataforma. Essa abordagem não apenas simplifica o desenvolvimento e reduz esforços no back-end, mas também aumenta a segurança e melhora significativamente a experiência do usuário ao proporcionar flexibilidade e conveniência no processo de login.

2.7. Requisitos Funcionais - Gestão de Cursos

Os requisitos funcionais e não funcionais são dois pilares fundamentais na definição de sistemas de software. Eles ajudam a estabelecer o que o sistema deve fazer e como ele deve funcionar para atender às necessidades dos usuários e dos stakeholders. A maior parte dos requisitos funcionais foi implementada na plataforma. A Tabela 1 e a Tabela 2 mostram os requisitos funcionais e a Tabela 3 mostra os requistos não funcionais da plataforma.

Tabela 1. Requisitos Funcionais - Gestão de Cursos

N°	Descrição
RF 01	O sistema deve listar os cursos disponíveis.
RF 02	O sistema deve fornecer a página para descrição de cada curso disponível.
RF 03	O sistema deve fornecer página para assistir às aulas de cada curso.
RF 04	O sistema deve permitir fornecer componente para avançar e navegar nas aulas dos cursos.
RF 05	O sistema deve salvar a última aula vista pelo usuário.
RF 06	O sistema deve fornecer o questionário de cada curso.
RF 07	O sistema deve gerar certificado para os concluintes dos cursos.
RF 08	O sistema deve permitir apenas usuários que acertem 70% do questionário concluir o curso.
RF 09	O sistema deve permitir a criação de curso.
RF 10	O sistema deve permitir a edição dos cursos criados.
RF 11	O sistema deve permitir o upload de capas do curso.
RF 12	O sistema deve permitir o usuário inscrever-se em um curso.
RF 13	O sistema deve permitir o usuário acessar os certificados dos cursos já concluídos.
RF 14	O sistema deve fornecer mecanismo de pesquisa dos cursos disponíveis na plataforma.

Tabela 2. Requisitos Funcionais

N°	Descrição
RF 15	O sistema deve permitir o login do usuário de diferentes formas.
RF 16	O sistema deve permitir o usuário recuperar a conta.
RF 17	O sistema deve permitir que apenas usuários no modo professor possam criar cursos.
RF 18	O sistema deve permitir que administradores do sistema gerenciem permissões de usuários, como atribuição de papéis de administrador, moderador ou usuário comum.
RF 19	O sistema deve permitir que os usuários editem as informações do seu perfil, como nome, foto de perfil, e preferências.

2.8. Requisitos não funcionais

Tabela 3. Requisitos não funcionais - Interface e Usabilidade

N°	Descrição
RF 20	O sistema deve permitir alternar entre modo escuro e modo claro.
RF 21	O sistema deve fornecer feedback visual claro para todas as interações do usuário, como confirmações de ações, erros ou sucesso em formulários, e mudanças de estado.
RF 22	O sistema deve ser responsivo, ou seja, deve se ajustar automaticamente a diferentes tamanhos de tela, garantindo uma boa experiência de uso tanto em dispositivos móveis quanto em desktops..

3. Diagrama de casos de uso

Os casos de uso refletem os principais requisitos levantados durante a análise de necessidades da aplicação. Eles atendem a diversas demandas do sistema. O diagrama representado na Figura 2 apresenta três atores principais, cada um representando papéis distintos no sistema que são os seguintes:

1. **Usuário**: Representa os estudantes ou participantes que utilizam a plataforma para acessar os cursos. Suas ações incluem realizar login/cadastro, visualizar cursos disponíveis, inscrever-se em cursos, pesquisar cursos, alternar entre os modos claro e escuro, recuperar a conta, responder questionários e receber feedback visual, isto é, quando cadastram em algum curso ou realizam alguma outra ação.
    - **Login/Cadastro:** Permite aos usuários acessarem a plataforma ou criarem contas para utilizá-la.
    - **Visualizar Cursos:** Apresenta os cursos disponíveis na plataforma, permitindo ao usuário selecionar o curso de interesse.
    - **Inscrever-se nos Cursos:** Habilita o usuário a participar de cursos desejados.
    - **Recuperar Conta:** Proporciona ao usuário uma forma de recuperar acesso à sua conta caso perca as credenciais.
    - **Responder Questionário:** Permite avaliar o aprendizado adquirido por meio de avaliações vinculadas aos cursos.
    - **Alternar entre Modo Claro e Escuro:** Adiciona uma funcionalidade de acessibilidade, adaptando o layout ao gosto do usuário.
    - **Pesquisar Cursos:** Facilita a navegação na plataforma, permitindo que os usuários encontrem cursos específicos.
2. **Professor**: Representa os responsáveis pela criação e edição dos cursos. Atua principalmente na elaboração de conteúdos e materiais educativos, além de configurar questionários e acompanhar o progresso dos usuários.
    - **Criação e Edição de Cursos:** Permite ao professor adicionar novos cursos, editar materiais já existentes e gerenciar os módulos, aulas e avaliações vinculados ao curso.
    - **Visualizar Cursos:** Apresenta os cursos disponíveis na plataforma, permitindo ao usuário selecionar o curso de interesse.

3. **Sistema**: Representa as funcionalidades automatizadas e os processos internos da plataforma. É responsável por ações como gerenciar permissões, gerar certificados, validar aprovações em cursos, listar cursos disponíveis e fornecer feedback visual para os usuários.
    - **Gerenciar Permissões**: Controla o acesso a funcionalidades específicas de acordo com o papel do usuário (usuário comum, professor, administrador).
    - **Gerar Certificado**: Automatiza a emissão de certificados para usuários aprovados nos cursos, oferecendo reconhecimento ao término dos estudos.
    - **Feedback Visual para os Usuários**: Fornece mensagens ou indicadores visuais sobre progresso, resultados e ações realizadas na plataforma.
    - **Listar Cursos**: Apresenta ao usuário todos os cursos disponíveis, organizados por categorias, tags ou relevância.
    - **Validar Aprovação em Curso**: Avalia as respostas de questionários ou testes realizados para determinar se o usuário cumpriu os requisitos para aprovação.

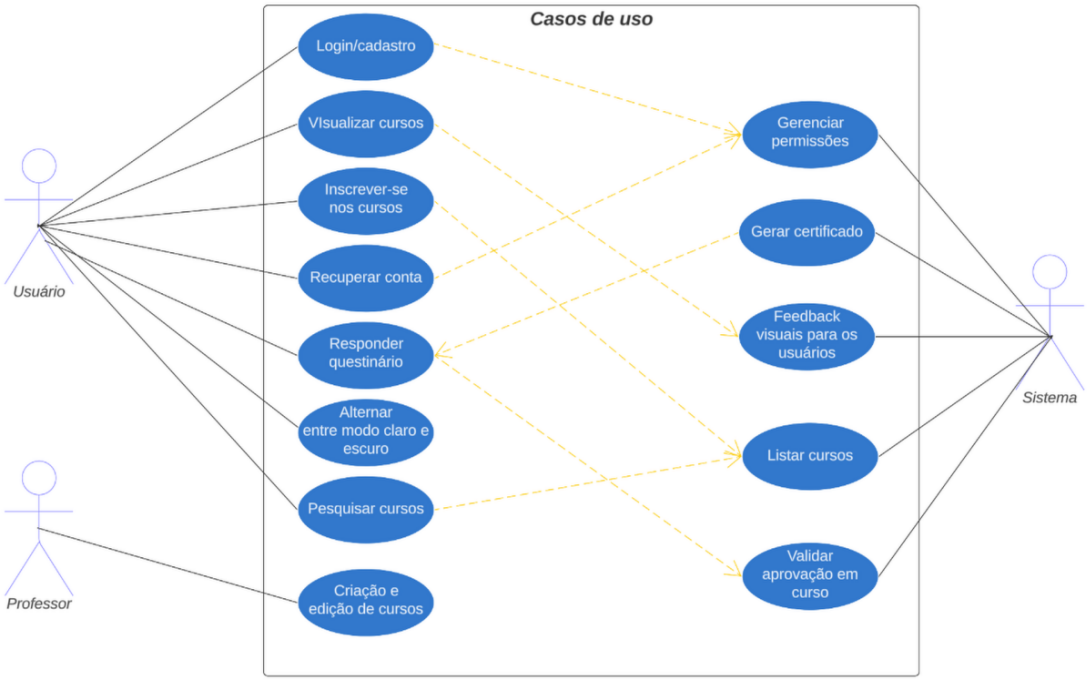

Figura 2. Diagrama de casos de usos
Fonte: O autor.

4. Resultados e discussões

Com a utilização do Next.js, foi possível desenvolver uma aplicação web responsiva, fluida e multiplataforma, alinhada ao principal objetivo do projeto. Considerando que a maior parte do público acessa a internet por meio de dispositivos móveis, a responsividade da plataforma torna-se um fator essencial para a experiência do usuário. A análise dos requisitos demonstrou que a aplicação atendeu aos principais critérios para um ambiente de ensino online, com ênfase na acessibilidade, flexibilidade e suporte ao aprendizado.

Cada caso de uso foi projetado com foco na funcionalidade e na usabilidade, refletindo boas práticas de design centrado no usuário.Dentre os recursos implementados, destaca-se o suporte ao modo escuro, que aprimora a experiência do usuário ao permitir a personalização da interface. Essa funcionalidade transforma completamente a aparência da plataforma, proporcionando maior conforto visual, especialmente em ambientes com baixa luminosidade. Além disso, ao melhorar a acessibilidade e a usabilidade, o modo escuro contribui para uma experiência mais agradável e adaptável, independentemente do dispositivo utilizado.

Uma das principais funcionalidades da plataforma é ilustrada na Figura 3. Essa página exibe a listagem de cursos disponíveis, permitindo que os usuários naveguem pelo catálogo mesmo sem estarem autenticados. No entanto, para acessar o conteúdo de cada curso, é necessário criar uma conta ou efetuar login. Essas funcionalidades foram desenvolvidas utilizando a biblioteca Clerk.js, que oferece recursos avançados de autenticação e gerenciamento de usuários.

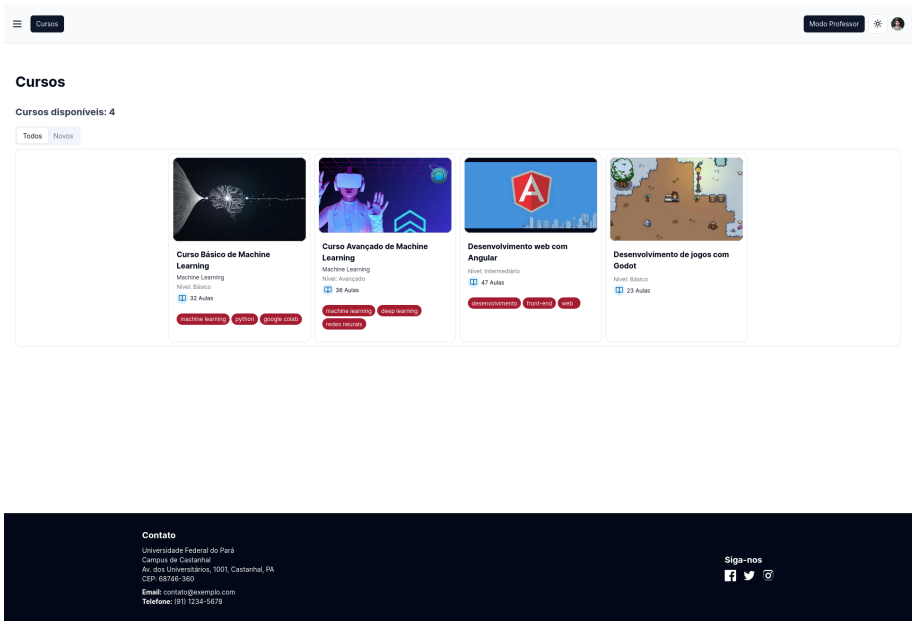

Figura 3. Página de login da listagem dos cursos disponíveis
Fonte: O autor.

A principal página da plataforma é a página onde é possível acessar o conteúdo de um determinado curso que pode ser vista na Figura 4. Nela, o usuário pode assistir às aulas do curso de forma clara e sequencial, promovendo um aprendizado estruturado. Também é possível realizar os questionários avaliativos, essenciais para medir o progresso e garantir a emissão do certificado de conclusão desde que o curso possua certificação.

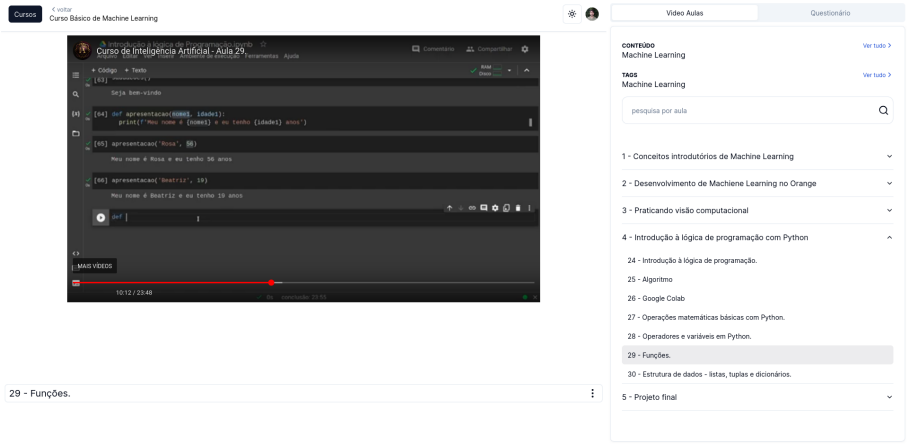

Figura 4. Página de visualização do curso
Fonte: O autor.

A Figura 5 destaca o design responsivo da página de curso, um padrão adotado em toda a plataforma para garantir plena funcionalidade em dispositivos móveis. Essa abordagem tem o objetivo de assegurar uma experiência de navegação fluida, consistente e adaptável a diferentes tamanhos de tela, proporcionando maior acessibilidade e usabilidade para o usuário.

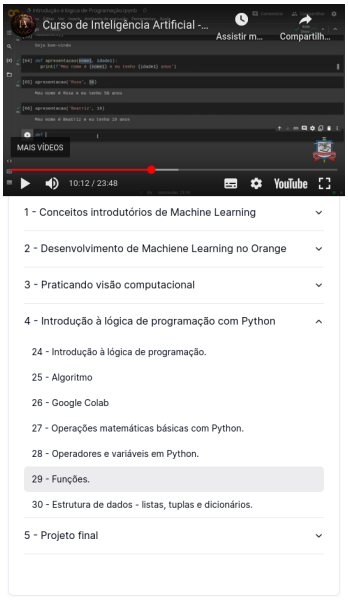

Figura 5. Página de visualização do curso no modo mobile.
Fonte: O autor.

5. Conclusão

O avanço das tecnologias educacionais e a crescente demanda pelo ensino à distância impulsionaram o desenvolvimento de plataformas digitais mais acessíveis e eficientes, especialmente após a pandemia ocorrida 2020, que acelerou ainda mais essa transformação. Neste contexto, o presente trabalho apresentou o desenvolvimento de uma plataforma online para cursos, destacando suas principais funcionalidades, como o sistema de login e cadastro, a visualização e inscrição em cursos, a realização de questionários avaliativos e a emissão de certificados.

Esta ideia surgiu com o objetivo de ampliar o acesso à educação e fortalecer o papel da universidade na disseminação do conhecimento, permitindo que um maior número de pessoas tenha a oportunidade de aprimorar suas habilidades e expandir seu aprendizado de forma acessível e inclusiva. A adoção de um design responsivo reforça o compromisso com a acessibilidade da plataforma em diversos dispositivos, garantindo uma experiência otimizada para todos os usuários. Considerando que a maioria das pessoas acessa a internet por meio de dispositivos móveis, essa abordagem buscou proporcionar uma navegação fluida acessível de qualquer dispositivo. Além disso, a integração de mecanismos de avaliação possibilita o acompanhamento do progresso dos estudantes, promovendo uma aprendizagem mais estruturada e eficiente.

A plataforma desenvolvida se apresenta como uma solução viável para ampliar o acesso ao conhecimento de forma acessível e eficiente. Durante sua implementação, foram adotadas tecnologias gratuitas para viabilizar sua disponibilização, como o YouTube para o upload das aulas, o Vercel para a hospedagem do front-end e back-end utilizando Next.js, e o MongoDB Atlas para a persistência de dados. Essa escolha estratégica permitiu reduzir custos sem comprometer a qualidade e a escalabilidade da plataforma.

Os resultados obtidos demonstram que a integração de recursos tecnológicos com boas práticas de usabilidade pode impactar positivamente a eficácia do ensino online. Para trabalhos futuros, recomenda-se a expansão das funcionalidades da plataforma, incluindo ferramentas interativas para a comunicação entre alunos e professores, a implementação de mecânicas de gamificação e o uso de inteligência artificial para personalizar a experiência de aprendizado, podendo ser usada nos questionários para fornecer feedback quando um aluno erra uma questão. Além disso, sugere-se o desenvolvimento de um fórum integrado para promover a troca de conhecimento entre os alunos por meio de comentários, bem como melhorias nos processos de criação e edição de cursos, tornando a plataforma ainda mais intuitiva e eficiente.

Referências

Cherny, B. (2019). *Programming TypeScript: making your JavaScript applications scale*. O'Reilly Media, Sebastopol, 1ª edição edition.

Clarke, A. (2016). *SEO 2016 Learn Search Engine Optimization With Smart Internet Marketing Strategies*. Guru Publisher.

Clerk, I. (2024). Clerk documentation. Accessed: December 02, 2024.

INEP (2022). Ensino a distância cresce 474% em uma década. Disponível em:. Acesso em: 20 feb. 2025.

Junior, N. T. and de Meira, A. F. (2023). Educação a distância no brasil e suas tendências para os próximos anos. *Cadernos Acadêmicos Unina*, 3(1).

Labs, T. (2024). Tailwind css documentation. Acesso em: 11 dez. 2024.

MongoDB Inc. (2024). *MongoDB Atlas Documentation*. Acesso em: 04 dez. 2024.

Pereira, W. F. A. (2023). Transformação educacional: O ascendente ensino a distância no brasil. *Revista Ibero-Americana de Humanidades, Ciências e Educação*, 9(12):315–328.

Prisma (2024). *Prisma Documentation*. Acesso em: 04 dez. 2024.

TypeScript Team (2024). *TypeScript Documentation*. Microsoft. Acessado em: 5 dez. 2024.

Vercel (2024). *Next.js Documentation*. Accessed: 2024-12-01.

X., S. D. (2024). Nextroutes - a next.js routing library. Acesso em: 3 dez. 2024.