
Faculdade de Tecnologia de Americana “Ministro Ralph Biasi”
Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

Murillo Xavier Fernandes
Renan Fernando da Silva

**ENGLISHFOR.DEV: PLATAFORMA DE APRENDIZADO DE LÍNGUAS COM
INTELIGÊNCIA ARTIFICIAL**

Americana, SP
2025

Murillo Xavier Fernandes

Renan Fernando da Silva

**ENGLISHFOR.DEV: PLATAFORMA DE APRENDIZADO DE LÍNGUAS COM
INTELIGÊNCIA ARTIFICIAL**

Trabalho de Conclusão de Curso desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas na área de concentração em Engenharia de Software.

Orientador(a): Prof. Esp. Antonio Alfredo Lacerda

Este trabalho corresponde à versão final do Trabalho de Conclusão de Curso apresentado por Murillo Xavier Fernandes e Renan Fernando da Silva, e orientado pelo(a) Prof. Esp. Antonio Alfredo Lacerda

Americana, SP

2025

**FICHA CATALOGRÁFICA – Biblioteca Fatec Americana
Ministro Ralph Biasi- CEETEPS Dados Internacionais de
Catalogação-na-fonte**

FERNANDES, Murillo Xavier

Englishfor.dev: plataforma de aprendizado de línguas com inteligência artificial. / Murillo Xavier Fernandes, Renan Fernando da Silva – Americana, 2025.

77f.

Monografia (Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas) - - Faculdade de Tecnologia de Americana Ministro Ralph Biasi – Centro Estadual de Educação Tecnológica Paula Souza

Orientador: Prof. Esp. Antonio Alfredo Lacerda

1. Engenharia de software 2. Ensino de línguas 3. Inteligência artificial. I. FERNANDES, Murillo Xavier, II. SILVA, Renan Fernando da III. LACERDA, Antonio Alfredo IV. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana Ministro Ralph Biasi

CDU: 681.3.05

376

007.52

Elaborada pelo autor por meio de sistema automático gerador de ficha catalográfica da Fatec de Americana Ministro Ralph Biasi.

Murilo Xavier Fernandes
Renan Fernando da Silva

ENGLISHFOR.DEV: Plataforma de Aprendizado de Línguas com Inteligência Artificial

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas pelo Centro Paula Souza – FATEC Faculdade de Tecnologia de Americana Ministro Ralph Biasi.
Área de concentração: Análise e Desenvolvimento de Sistemas.

Americana, 3 de dezembro de 2025.

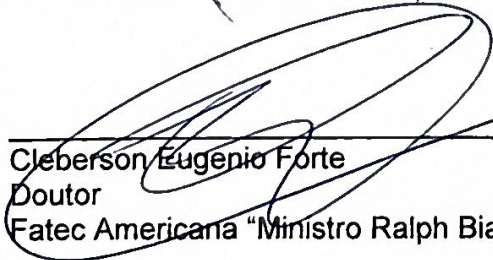
Banca Examinadora:



Antonio Alfredo Lacerda
Especialista
Fatec Americana "Ministro Ralph Biasi"



José William Pinto Gomes
Especialista
Fatec Americana "Ministro Ralph Biasi"



Cleberson Eugenio Forte
Doutor
Fatec Americana "Ministro Ralph Biasi"

AGRADECIMENTOS

Em primeiro lugar, aos nossos familiares pelo constante apoio durante a nossa jornada acadêmica.

Aos professores que contribuíram para a nossa formação acadêmica e profissional.

Aos nossos amigos e colegas de classe que foram parceiros em cada etapa deste percurso.

E ao nosso orientador, Prof. Esp. Antonio Alfredo Lacerda, que nos orientou e acompanhou durante a realização deste projeto.

RESUMO

A plataforma “englishfor.dev” é desenvolvida para auxiliar profissionais de tecnologia não nativos da língua inglesa a aprimorarem seu vocabulário, habilidades de conversação e pronúncia, com foco específico em ambientes corporativos do setor. O objetivo central da ferramenta é superar a barreira linguística que frequentemente impede a plena integração e ascensão profissional em empresas globais de tecnologia. Para isso, a plataforma emprega um conteúdo contextualizado, elaborado a partir de situações reais e recorrentes do cotidiano de trabalho. Através de módulos interativos, simulações de diálogos do cotidiano laboral e exercícios de fonética aplicada, busca-se a construção de confiança para uma comunicação clara. Como elementos de engajamento e motivação, a ferramenta incorpora uma trilha de aprendizagem gamificada, que incentiva a progressão contínua do usuário. Adicionalmente, são disponibilizadas análises gráficas de dados de desempenho, permitindo que o próprio usuário monitore visualmente o seu progresso e identifique áreas que necessitam de maior atenção. A metodologia privilegia a aplicação prática do conhecimento, permitindo que o usuário pratique situações comunicativas típicas de seu ambiente profissional. Dessa forma, o “englishfor.dev” posiciona-se como uma solução especializada, capacitando os usuários a participarem de equipes internacionais e acelerarem suas trajetórias de carreira em um mercado cada vez mais competitivo e interconectado.

Palavras Chave: Inglês; Tecnologia; Aprendizado.

ABSTRACT

The "englishfor.dev" platform is designed to assist non-native English speaking technology professionals in enhancing their vocabulary, conversation skills, and pronunciation, with a specific focus on corporate environments within the sector. The tool's central objective is to overcome the language barrier that often impedes full integration and career advancement in global technology companies. To achieve this, the platform employs contextualized content, developed based on real and recurring situations from daily work life. Through interactive modules, simulations of everyday work dialogues, and applied phonetic exercises, the goal is to build confidence for clear communication. As elements of engagement and motivation, the tool incorporates a gamified learning path, which encourages the user's continuous progression. Additionally, graphical analyses of performance data are provided, allowing the user to visually monitor their own progress and identify areas that require further attention. The methodology prioritizes the practical application of knowledge, allowing the user to practice typical communicative situations from their professional environment. Thus, "englishfor.dev" positions itself as a specialized solution, empowering users to participate in international teams and accelerate their career paths in an increasingly competitive and interconnected market.

Keywords: *English; Technology; Learning.*

SUMÁRIO

1 INTRODUÇÃO.....	11
1.1 Objetivo.....	11
1.2 Justificativa.....	11
1.3 Resumo dos Capítulos.....	12
2 RECURSOS ADICIONAIS.....	13
2.1 Conceito de Inteligência Artificial.....	13
2.2 Conceito de Framework.....	14
2.3 Conceito de Banco de dados.....	15
2.4 Conceito de Scrum.....	15
2.5 Conceito de Levantamento de Requisitos e Diagramas.....	16
2.6 Conceito de API.....	17
2.7 Softwares Similares.....	18
3 PROJETO - ENGENHARIA DE SOFTWARE.....	20
3.1 Levantamento de Requisitos.....	20
3.1.1 Requisitos Funcionais.....	20
3.1.2 Requisitos Não Funcionais.....	21
3.2 Casos De Uso.....	22
3.3 Documentação dos Casos de Uso.....	24
3.4 Diagrama de Entidade e Relacionamento.....	27
3.5 Dicionário de Dados.....	28
3.6 Plano de Testes.....	35
4 DESENVOLVIMENTO DO PROJETO.....	39
4.1 Metodologia Scrum.....	39
4.2 Entregáveis.....	39
4.2.1 Sprint 1.....	39
4.2.2 Sprint 2.....	41
4.3 Tecnologia Empregada.....	42
4.3.1 Next.js.....	43
4.3.2 HTML.....	43
4.3.3 CSS.....	43
4.3.4 JavaScript.....	43
4.3.5 Primer GitHub.....	44
4.3.6 Node.js.....	44
4.3.7 PostgreSQL.....	44
4.3.8 GitHub.....	44
4.3.9 Vercel.....	44
4.3.10 Cloudflare.....	45

4.3.11 Google Gemini.....	45
4.3.12 Google Speech Recognition.....	45
4.3.13 Recharts.....	45
4.4 Desenvolvimento da API.....	45
4.5 Desenvolvimento do Prompt Primário.....	53
4.5.1 Integração com a API Gemini (Google Generative AI).....	55
4.6 Análise de Dados de Desempenho do Usuário.....	57
4.7 Interfaces de Usuário.....	58
5 CONSIDERAÇÕES FINAIS.....	72
5.1 Acesso ao site.....	73
5.2 Acesso ao repositório.....	74
REFERÊNCIAS.....	75

LISTA DE FIGURAS

Figura 1 – Aplicativo Duolingo.....	18
Figura 2 – Plataforma Interview Warmup.....	19
Figura 3 – Diagrama de caso de uso da plataforma.....	23
Figura 4 - Diagrama de Entidade e Relacionamento.....	27
Figura 5 – Exemplo de código da rota trail.....	46
Figura 6 – Exemplo de código do modelo trail.....	47
Figura 7 – Exemplo de código de rota dinâmica (exercises/[id]).....	48
Figura 8 – Exemplo simplificado de busca com parâmetros de consulta.....	49
Figura 9 – Exemplo de código de POST com corpo JSON (login).....	50
Figura 10 – Exemplo de código do middleware.....	51
Figura 11 – Exemplo de código do modelo authentication.....	52
Figura 12 – Prompt primário.....	54
Figura 13 – Inicialização do modelo Gemini com injeção do prompt primário.....	55
Figura 14 – Pipeline de avaliação: sessão, mensagens e retorno em JSON.....	56
Figura 15 – Diagrama de estados (mapa das telas).....	59
Figura 16 – Landing Page.....	60
Figura 17 – Tela de login.....	61
Figura 18 – Tela de Cadastro.....	62
Figura 19 – Tela da Trilha de Exercícios.....	63
Figura 20 – Tela de Resolução de exercícios.....	64
Figura 21 – Tela da Trilha de Exercícios.....	64
Figura 22 – Tela da Trilha de Exercícios.....	65
Figura 23 – Tela da Trilha de Exercícios.....	65
Figura 24 – Tela de Conquistas.....	67
Figura 25 – Tela de Estatísticas.....	68
Figura 26 – Tela de Certificado.....	69
Figura 27 – Tela de Certificado.....	70
Figura 28 – Tela de Certificado.....	70
Figura 29 – Tela de Configurações.....	71
Figura 30 – Acesso ao site via URL.....	73
Figura 31 – Acesso ao repositório via URL.....	74

LISTA DE QUADROS

Quadro 1 - Comparativo da aplicação em relação aos concorrentes.....	19
Quadro 2 – Requisitos funcionais do projeto.....	21
Quadro 3 – Requisitos não funcionais do projeto.....	21
Quadro 4 – Caso de uso “Entrar no Sistema / Cadastrar Usuário”.....	24
Quadro 5 – Caso de uso “Escolher Exercício”.....	25
Quadro 6 – Caso de uso “Responder Perguntas”.....	25
Quadro 7 – Caso de uso “Avaliar Respostas”.....	26
Quadro 8 - Dicionário de Dados da entidade Users.....	28
Quadro 9 - Dicionário de Dados da entidade Levels.....	29
Quadro 10 - Dicionário de Dados da entidade Exercises.....	30
Quadro 11 - Dicionário de Dados da entidade Questions.....	31
Quadro 12 - Dicionário de Dados da entidade User Progressions.....	31
Quadro 13 - Dicionário de Dados da entidade Achievements.....	32
Quadro 14 - Dicionário de Dados da entidade User Achievements.....	33
Quadro 15 - Dicionário de Dados da entidade Answer Evaluations.....	34
Quadro 16 - Caso de teste Landing Page.....	35
Quadro 17 - Caso de teste Login.....	35
Quadro 18 - Caso de teste Cadastro.....	36
Quadro 19 - Caso de teste Menu e Trilha de Aprendizado.....	36
Quadro 20 - Caso de teste Exercícios.....	37
Quadro 21 - Caso de teste Procurar Usuários.....	37
Quadro 22 - Caso de teste Perfil do Usuário.....	37

LISTA DE TABELAS

Tabela 1 – Planejamento realizado para a primeira entrega.....	40
Tabela 2 – Planejamento realizado para segunda entrega.....	41

1 INTRODUÇÃO

O objetivo principal do projeto é documentar o desenvolvimento desta plataforma, criada para auxiliar desenvolvedores e profissionais de tecnologia, cuja língua nativa não é o inglês, a aprimorar o vocabulário, as habilidades de conversação e a pronúncia, com foco em termos técnicos para ambientes corporativos de Tecnologia da Informação.

1.1 Objetivo

O projeto tem como objetivo documentar o desenvolvimento da plataforma “englishfor.dev”, criada para auxiliar profissionais da área de Tecnologia da Informação (TI) a aprimorarem seu vocabulário e a pronúncia de termos técnicos em inglês. Com trilhas de aprendizado personalizadas, exercícios interativos, sistemas de conquistas e relatório de estatísticas de desempenho, a plataforma busca oferecer uma boa experiência para o aprendizado do idioma, especialmente focado nas necessidades do mercado de TI.

1.2 Justificativa

A importância de um sistema como o “englishfor.dev” é posta em evidência quando se examina minuciosamente o panorama sociolinguístico brasileiro contemporâneo. Segundo dados do *British Council* (2014, p.7 *apud* KNN Idiomas, 2023), somente 5% dos cidadãos brasileiros detêm alguma proficiência na língua inglesa, cifra que equivale a aproximadamente dez milhões de indivíduos. Deste montante, é notório que menos de 1% da população (cerca de dois milhões de pessoas) pode ser rotulada como fluente no idioma. Esta carência no domínio do inglês configura um desafio de monta, principalmente em um mercado de trabalho que, progressivamente, requer colaboradores altamente qualificados e aptos a interagirem eficazmente em um ambiente internacionalizado.

Em um plano correlato, a indústria tecnológica no Brasil tem vivenciado uma expansão notável. No decurso da década compreendida entre 2012 e 2022, o volume de ocupações vinculadas à área de TI registrou um acréscimo de 95%,

saltando de 445 mil para 868 mil vagas formais (Agência Brasil, 2024). Tal efervescência do setor intensifica a exigência por profissionais que estejam aptos não apenas sob o prisma técnico, mas que também ostentam as competências linguísticas que são cruciais para a colaboração em equipes transnacionais e para o acesso desimpedido a fontes de informação primariamente disponíveis em inglês.

Nesse contexto, a plataforma se sobressai como um expediente que se harmoniza intrinsecamente com as exigências do mercado. Ao prover uma metodologia de ensino eminentemente prática, com foco calibrado no vocabulário técnico específico da área, o sistema auxilia os usuários a transpor barreiras idiomáticas e, assim, a cruzar patamares mais elevados em suas trajetórias profissionais. Ademais, o emprego de estratégias pedagógicas interativas, incluindo a *gamificação* e as trilhas de conhecimento estruturadas, acentua o envolvimento do aprendiz e catalisa de maneira significativa a sedimentação do saber.

1.3 Resumo dos Capítulos

A articulação deste estudo obedece à seguinte disposição: No Capítulo 1, explicitam-se o propósito e a relevância intrínseca do projeto em questão. Subsequentemente, o Capítulo 2 se dedica ao detalhamento do desenho do sistema, abordando uma análise comparativa de *softwares* análogos, o minucioso levantamento dos requisitos funcionais e não funcionais, e o elenco de tecnologias e ferramentas mobilizadas na construção. O Capítulo 3, por sua vez, é devotado à modelagem estrutural do sistema, empregando diagramas da Linguagem de Modelagem Unificada (UML), como os de casos de uso e o de entidade e relacionamento, a fim de conferir documentação rigorosa à aplicação desenvolvida. Prosseguindo, o Capítulo 4 elucida o processo de desenvolvimento, detalhando tanto a metodologia ágil Scrum adotada, quanto às fases de construção da plataforma e a concepção das interfaces de interação com o usuário. Finalmente, o Capítulo 5 veicula as considerações finais, sintetizando os resultados obtidos e os obstáculos transpostos no transcorrer do ciclo de desenvolvimento do projeto.

2 RECURSOS ADICIONAIS

Este capítulo aborda os recursos adicionais do projeto, conceituando Inteligência Artificial (IA), *framework* de desenvolvimento, banco de dados, Scrum, levantamento de requisitos, e *Application Programming Interface* (API). Além disso, serão apresentados *softwares* similares.

2.1 Conceito de Inteligência Artificial

A Inteligência Artificial (IA) é melhor caracterizada como o conjunto de sistemas que, orientados por objetivos predefinidos pela intervenção humana, detêm a capacidade de inspecionar ambientes, sejam eles físicos ou simulados, de assimilar conhecimento a partir de dados ou vivências, e de executar prognósticos, sugestões ou optar por ações com uma margem de autonomia, visando influenciar tais cenários (OCDE, 2023). A finalidade intrínseca desta disciplina reside em habilitar máquinas e programas a desempenhar funções cognitivas de alta complexidade de forma autônoma.

Para concretizar tais propósitos, a IA se desdobra em uma variedade de níveis e modalidades de aplicação. Essa abrangência estende-se desde sistemas relativamente mais modestos, como algoritmos de sugestão de conteúdo ou auxiliares virtuais, até artefatos tecnológicos de ponta, como é o caso de veículos que prescindem de condutor ou de métodos de diagnóstico médico com o apoio da IA, demonstrando como o avanço contínuo da área tem reconfigurado múltiplas esferas industriais. O predicado mais relevante da IA reside na sua proficiência em processar massas de dados volumosas, identificar padrões (*patterns*) e fundamentar decisões a partir desse substrato informacional, o que culmina na automatização de fluxos de trabalho, na otimização de operações e na criação de soluções inéditas para problemáticas de natureza complexa.

Neste sentido, a IA transcende a mera mimetização da inteligência humana; na verdade, ela se projeta para amplificá-la e aplicá-la em dimensões e circunstâncias que se revelariam impraticáveis ou logisticamente proibitivas para a capacidade cognitiva humana singular.

2.2 Conceito de Framework

No campo da engenharia de *software*, um *framework* constitui uma abstração arquitetural que disponibiliza uma estrutura de apoio basal para a edificação de sistemas aplicativos. Ele se materializa como uma arquitetura coesa, composta por um agrupamento unificado de código, bibliotecas e padrões de projeto consolidados, que predefine o esqueleto sobre o qual a aplicação específica será concebida (Sommerville, 2011). A sua função primordial é catalisar o ciclo de desenvolvimento, desonerando os profissionais de *software* da gestão de funcionalidades de baixo nível e direcionando o foco para a implementação da lógica de negócios central.

O traço distintivo que demarca a fronteira entre um *framework* e uma mera biblioteca é o princípio da Inversão de Controle (*Inversion of Control* — IoC). Enquanto em um paradigma de programação convencional o fluxo de execução é comandado pelo código do desenvolvedor, que chama rotinas de bibliotecas externas conforme a demanda, a utilização de um *framework* subverte esta dinâmica: o controle principal do fluxo é cedido a ele próprio; é o *framework* quem, em pontos de extensão predeterminados do seu ciclo de vida, efetua a chamada do código particular da aplicação. Esta delegação de responsabilidade prescreve a metodologia de projeto, requerendo que o desenvolvedor estenda classes ou implemente interfaces previamente definidas pelo *framework* para injetar a funcionalidade almejada.

Consequentemente, o *framework* estabelece uma ordem organizacional compulsória sobre o projeto. Tal padronização é crucial para produzir um código com alta manutenibilidade, particularmente em empreendimentos de grande envergadura. A isso se soma o fato de que a sua implementação geralmente engendra vantagens notáveis, incluindo uma notória elevação da produtividade, à assimilação de boas práticas intrínsecas de segurança e design, e a salvaguarda da escalabilidade inerente ao produto de *software* final.

2.3 Conceito de Banco de dados

No cenário tecnológico, o banco de dados se firma como um constructo de dados inter-relacionados, organizado e estruturado, cuja arquitetura é projetada para maximizar a eficiência na recuperação, governança e atualização da informação.

Autoridades na área, como Elmasri e Navathe (2011), conceituam-no como "uma coleção de dados relacionados" que espelha um "universo de discurso" particular, uma entidade definida por sua coerência lógica, seu significado intrínseco e uma finalidade explícita para uma determinada comunidade de usuários. De forma complementar, a perspectiva de C. J. Date (2004) o descreve como "uma coleção de dados persistentes". (Date, 2004)

A orquestração dessa massa de dados é delegada a um *software* especializado, o Sistema de Gerenciamento de Banco de Dados (SGBD), que atua como a interface mediadora entre os usuários e o acervo digital. É o SGBD que zela pela integridade, arbitra o controle de concorrência e gerencia os protocolos de recuperação ante falhas, ao mesmo tempo que oculta a complexidade granular do armazenamento físico (Elmasri; Navathe, 2011). Um banco de dados, portanto, provê um ecossistema centralizado e regido por regras estritas, indispensável para a manipulação consistente e fidedigna de vastos volumes de informação.

2.4 Conceito de Scrum

Um arcabouço metodológico para gestão, o Scrum capacita equipes a endereçar problemas de alta complexidade por meio de soluções adaptativas, objetivando a entrega de produtos com o máximo valor agregado. Concebido na aurora dos anos 90, sua essência não reside em um processo prescritivo e inflexível, mas sim em uma estrutura deliberadamente enxuta, projetada para a aplicação de um arsenal diverso de técnicas. Consoante a definição de seus próprios arquitetos, o Scrum se manifesta como um "framework leve que ajuda pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos", instigando, assim, um ciclo de aprimoramento perpétuo (Schwaber; Sutherland, 2020, p. 4).

A sua arquitetura se assenta sobre uma tríade de papéis, eventos e artefatos meticulosamente definidos. A célula de trabalho primária, o *Scrum Team* (Equipe Scrum), congrega um *Product Owner* (Proprietário do Produto), um *Scrum Master* (Mestre Scrum) e os *Developers* (Desenvolvedores) em uma estrutura notadamente horizontal, desprovida de hierarquias formais. Ao *Product Owner* recai a incumbência de maximizar o valor gerado pelo produto, uma tarefa executada por meio da gestão estratégica do *Product Backlog*. O *Scrum Master*, por sua vez, opera como um líder-servidor, zelando pela adesão do time aos preceitos do Scrum e atuando ativamente para suprimir óbices. Finalmente, aos *Developers* compete a responsabilidade técnica de construir, a cada *Sprint*, o incremento tangível do produto.

2.5 Conceito de Levantamento de Requisitos e Diagramas

A Engenharia de Requisitos consiste na dinâmica disciplinada de especificação dos serviços e das fronteiras operacionais de um sistema de software. Sua etapa nevrálgica, a elicitação de requisitos, fundamenta toda a arquitetura do projeto ao imergir nas necessidades das partes interessadas (*stakeholders*). Trata-se, na visão de Ian Sommerville, de um exercício de descoberta e compreensão profundamente dependente da comunicação com os usuários, essencial para decantar desde os objetivos macro do sistema até as funcionalidades mais granulares (Sommerville, 2011, p. 69–72).

A grande barreira dessa fase reside em seu componente humano; requisitos não são passivamente coletados, mas sim construídos em um esforço de colaboração. É comum que *stakeholders* se mostrem incapazes de articular com precisão suas demandas. Diante desse cenário, técnicas como entrevistas e a prototipagem emergem como mecanismos para atenuar o risco de equívocos.

Para assegurar que as exigências apuradas sejam desprovidas de ambiguidade, a modelagem visual emerge como instrumento essencial. Sendo, por definição, "uma simplificação da realidade" (Booch; Rumbaugh; Jacobson, 2005, p. 6), um modelo possibilita a concentração nos elementos do sistema. Nesse domínio, a Linguagem de Modelagem Unificada (UML) consagrou-se como o padrão, empregando um arsenal de diagramas de Casos de Uso, Classes, Sequência, para

detalhar as interações, a arquitetura estática e o comportamento dinâmico do *software*. Sua função, portanto, extrapola a documentação: opera como uma plataforma de análise que viabiliza a organização, a detecção de inconsistências e a comunicação do conhecimento sistêmico. Em última análise, se a elicitación se ocupa em definir "o que" deve ser construído, a modelagem dedica-se a representar e validar visualmente "como" essa construção se materializará.

2.6 Conceito de API

Uma Interface de Programação de Aplicações (*Application Programming Interface* — API) estabelece-se como o mecanismo que viabiliza a interoperabilidade entre sistemas e aplicações de *software* distintos, impondo um protocolo de comunicação comum. A dinâmica pode ser ilustrada pela metáfora de um restaurante, na qual o cliente (a aplicação consumidora) desconhece por completo a complexidade operacional da cozinha (o sistema provedor). A interação do cliente limita-se a selecionar uma opção do cardápio (a documentação da API) e formular uma solicitação precisa, que é então transportada pelo garçom (a própria API). Este intermediário não apenas encaminha a requisição, mas assegura sua correta interpretação, retornando com o produto final (os dados), abstraindo, assim, toda a lógica subjacente ao processo.

Em seu cerne técnico, a API materializa-se como um arcabouço normativo, um conjunto de definições e protocolos que governam a interação entre entidades de *software*. Esse contrato formal delimita os métodos de requisição disponíveis, estipula os formatos para o tráfego de dados como JSON (JavaScript Object Notation) ou XML (Extensible Markup Language) e dita as convenções que os engenheiros de *software* devem seguir. Ao expor funcionalidades de forma criteriosa e segura, as APIs constituem a pedra angular para a modularidade e a integração de serviços na engenharia de *software* contemporânea, permitindo que desenvolvedores incorporem funcionalidades de alta complexidade, como sistemas de pagamento ou serviços de geolocalização, sem a necessidade de reinventar suas arquiteturas fundamentais.

2.7 Softwares Similares

Atualmente existem várias plataformas voltadas para o aprendizado de inglês e preparo para o mundo de TI, foram selecionadas três:

- **Duolingo:** Um site e aplicativo móvel de aprendizado de idiomas. Os usuários praticam vocabulário, gramática e pronúncia usando repetição espaçada. Os exercícios podem incluir tradução escrita, compreensão de leitura e fala e histórias curtas. A Figura 1 mostra o aplicativo Duolingo.

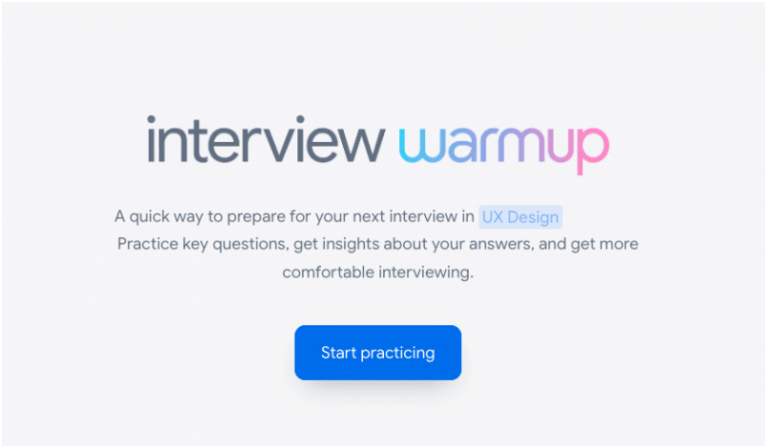
Figura 1 – Aplicativo Duolingo



Fonte: Medium (2024).

- **Interview Warmup:** O Interview Warmup é uma ferramenta desenvolvida pelo Google para ajudar candidatos a se prepararem para entrevistas de emprego. A plataforma utiliza inteligência artificial para simular perguntas comuns de entrevistas e fornecer *feedback* sobre as respostas do usuário. A figura 2 mostra a plataforma Interview Warmup.

Figura 2 – Plataforma Interview Warmup



Fonte: Interview Warmup (2024).

Levando estes aspectos em consideração, foi elaborado o quadro 1 mostrando as principais diferenças do “englishfor.dev”, em relação aos aplicativos citados anteriormente:

Quadro 1 - Comparativo da aplicação em relação aos concorrentes

Funcionalidades	Interview Warmup	Duolingo	englishfor.dev
Reconhecimento de fala	X	X	X
Narrador em inglês	X	X	X
Trilha de aprendizado	-	X	X
Criação de perfil	-	X	X
Foco no mercado de TI	X	-	X
Exercícios interativos	X	X	X
Sistemas de conquistas	-	X	X
Gamificação	-	X	X
Acessibilidade	X	X	X
Certificado de conclusão	-	-	X
Integração com IA	X	X	X
Estatísticas de desempenho	-	X	X

Fonte: Elaborado pelo autor (2025).

3 PROJETO - ENGENHARIA DE SOFTWARE

A Engenharia de Requisitos estabelece-se como uma disciplina fundamental da Engenharia de *Software*, concentrando seus esforços na prospecção, análise, formalização e chancela dos requisitos que são intrínsecos a qualquer sistema computacional. O escopo dessa área reside na compreensão aprofundada das demandas dos usuários, das partes interessadas (*stakeholders*) e do cenário de negócio circundante, com o objetivo de transmutá-las em especificações nítidas e inequivocamente exatas para o subsequente desenvolvimento do *software*.

No que concerne aos diagramas, estes assumem uma função capital na modelagem de sistemas. Diagramas como os da Linguagem de Modelagem Unificada (*UML*) são recorrentemente mobilizados na Engenharia de *Software* para conferir representação visual às multifacetadas faces de um sistema, abarcando desde a sua estrutura (como no diagrama de classes), passando pelo seu comportamento dinâmico (exemplificado pelos diagramas de sequência e de atividades), até a forma de interação entre os diversos componentes do *software*.

Os diagramas da Linguagem de Modelagem Unificada (*UML*) desempenham papel central nesse processo. Eles oferecem uma representação gráfica da estrutura do sistema (como no diagrama de classes), de seu comportamento dinâmico (exemplificado nos diagramas de sequência e atividades), e das interações entre seus componentes. Esses diagramas não só visualizam, mas também ajudam a organizar e a validar as funcionalidades e a arquitetura do sistema.

3.1 Levantamento de Requisitos

3.1.1 Requisitos Funcionais

Os requisitos funcionais têm como objetivo delinear as ações que o sistema está obrigado a performar, circunscrevendo, assim, a funcionalidade intrínseca almejada do *software* (Sommerville, 2011, p. 59). No quadro 2, por sua vez, encontram-se especificados e consolidados os requisitos funcionais que regem este projeto.

Quadro 2 – Requisitos funcionais do projeto

Identificação	Requisito Funcional	Prioridade
RF001	Sistema de login e cadastro	Essencial
RF002	Níveis e Exercícios	Essencial
RF003	Sistema de Conquistas	Desejável
RF004	Respostas analisadas por IA	Importante
RF005	Trilhas de Aprendizado	Essencial
RF006	Sistema de Perfil	Desejável
RF007	Editar Perfil do Usuário	Desejável
RF008	Sistema de Progressão	Essencial
RF009	Deteção de fala por áudio	Importante
RF010	Exibir estatísticas de desempenho do usuário	Importante
RF011	Geração do certificado de conclusão	Importante

Fonte: Elaborado pelo autor (2025).

3.1.2 Requisitos Não Funcionais

Os requisitos não funcionais compreendem aquelas exigências que não guardam relação direta com as funcionalidades específicas providas pelo sistema (Sommerville, 2011, p. 60). Em contrapartida, o quadro 3 elenca e pormenoriza os requisitos não funcionais que delimitam o escopo deste projeto.

Quadro 3 – Requisitos não funcionais do projeto

Identificação	Requisito não funcional	Categoria	Prioridade
RNF001	Bom desempenho	Usabilidade	Essencial
RNF002	Segurança dos dados	Confiabilidade	Essencial
RNF003	Boa usabilidade e experiência	Usabilidade	Essencial
RNF004	Acessibilidade	Usabilidade	Desejável
RNF005	Compatibilidade	Distribuição	Importante
RNF006	Manutenibilidade	Confiabilidade	Importante

Fonte: Elaborado pelo autor (2025).

3.2 Casos De Uso

Os diagramas de caso de uso são ferramentas de modelagem que cumprem a função de delinear um panorama de funcionalidades a partir da ótica do usuário, efetuando, assim, a catalogação dos requisitos funcionais intrínsecos ao sistema. Tais diagramas retratam, em sua estrutura, três elementos capitais: os atores (figurados por representações antropomórficas), as funcionalidades (simbolizadas por elipses que contêm a ação descrita em seu interior) e os relacionamentos (ilustrados pelas linhas que estabelecem as conexões).

Os atores primários que interagem com o sistema são o Usuário, a IA de Correção e o Banco de Dados. O próprio sistema é concebido como um caso de uso explícito, constituindo-se no contexto dentro do qual todos os casos de uso detalhados são processados.

- **Usuário:** Este ator representa os indivíduos que efetivamente utilizam a aplicação. Um usuário, por exemplo, está apto a buscar determinado conteúdo, a conferir avaliações, a marcar itens como favoritos, entre outras interações.
- **IA de Correção:** Configura-se como o agente de Inteligência Artificial que tem por função analisar a resposta submetida pelo usuário e, com base no enunciado da questão, realizar uma apreciação sistemática, culminando na atribuição de uma nota e na sugestão de retificações.
- **Banco de Dados:** Atua como o agente responsável pela custódia e gerenciamento dos dados vitais, incluindo as informações de perfil do usuário, o acervo de exercícios, o progresso alcançado e a pontuação de Experiência (XP) acumulada.

3.3 Documentação dos Casos de Uso

Cada funcionalidade dos diagramas de casos de uso será descrita do quadro 4 ao quadro 7.

Quadro 4 – Caso de uso “Entrar no Sistema / Cadastrar Usuário”

Nome do caso de uso	Entrar no Sistema / Cadastrar Usuário
Atores envolvidos	Usuário e Banco de Dados
Objetivo	Este caso de uso descreve os passos do login e / ou cadastro de um usuário no sistema.
Ações do ator	Ações do Sistema
1. O usuário clica em login no aplicativo.	
	2. As informações de login fornecidas pelo usuário são levadas para a API do sistema.
	3. A API consultará o banco de dados e validará os dados, verificando se as credenciais estão corretas no caso de login, e verificando se já não há um usuário cadastrado com essas informações em caso de cadastro.
	4. Após a autenticação, o sistema redireciona para a página inicial da plataforma.
Validações	Para que o login seja efetuado, o usuário deve entrar com seu usuário e senha.

Fonte: Elaborado pelo autor (2025).

Quadro 5 – Caso de uso “Escolher Exercício”

Nome do caso de uso	Escolher Exercício
Atores envolvidos	Usuário e Banco de Dados
Objetivo	Este caso de uso descreve os passos para que o usuário escolha um exercício para realizar.
Ações do ator	Ações do Sistema
1. O usuário deve entrar na página de seleção de exercícios.	
	2. A API do sistema irá consultar o banco de dados em busca dos exercícios disponíveis e retornará as informações.
	3. O sistema irá organizar os exercícios retornados pela API em círculos clicáveis dispostos em uma linha vertical, cada nível tem sua própria linha de exercícios contínuos.
4. O usuário pode interagir com um círculo representando o exercício para selecioná-lo e poder respondê-lo, o que leva ao caso de uso "Responder Perguntas".	

Fonte: Elaborado pelo autor (2025).

Quadro 6 – Caso de uso “Responder Perguntas”

Nome do caso de uso	Responder Perguntas
Atores envolvidos	Usuário
Objetivo	Descreve como o usuário irá interagir com a interface para responder o exercício escolhido no caso de uso "Escolher Exercício".
Ações do ator	Ações do Sistema
	1. Após a seleção do exercício no caso de uso "Escolher Exercício", a API do sistema irá consultar o banco de dados em busca das informações do exercício, e exibirá a pergunta para o usuário.
2. O usuário deve responder digitando a resposta em texto ou falando no microfone.	
3. O usuário envia a resposta.	
	4. O sistema consultará a API da IA de correção para avaliar a resposta do usuário, o que leva ao caso de uso "Avaliar Respostas".

Fonte: Elaborado pelo autor (2025).

Quadro 7 – Caso de uso “Avaliar Respostas”

Nome do caso de uso	Avaliar Respostas
Atores envolvidos	IA de Correção
Objetivo	Responsável por avaliar a resposta do usuário fornecendo um feedback
Ações do ator	Ações do Sistema
	1. Após o envio da resposta em "Responder Perguntas", o sistema consultará a IA de correção.
	2. O sistema mostra ao usuário uma crítica em texto da resposta do usuário fornecida pela IA de correção.
	3. O sistema mostra ao usuário uma avaliação usando estrelas dos seguintes tópicos: coerência, gramática, relevância e contexto.
	4. O sistema mostra ao usuário uma sugestão de resposta melhorada.
	5. Atualiza o progresso do usuário.
	6. Aumenta o XP do usuário.
	7. Atualiza as estatísticas do usuário.
	8. Caso seja o último exercício, gera um certificado de conclusão.
	9. Verifica por possíveis conquistas à atribuir, e atribui se necessário.

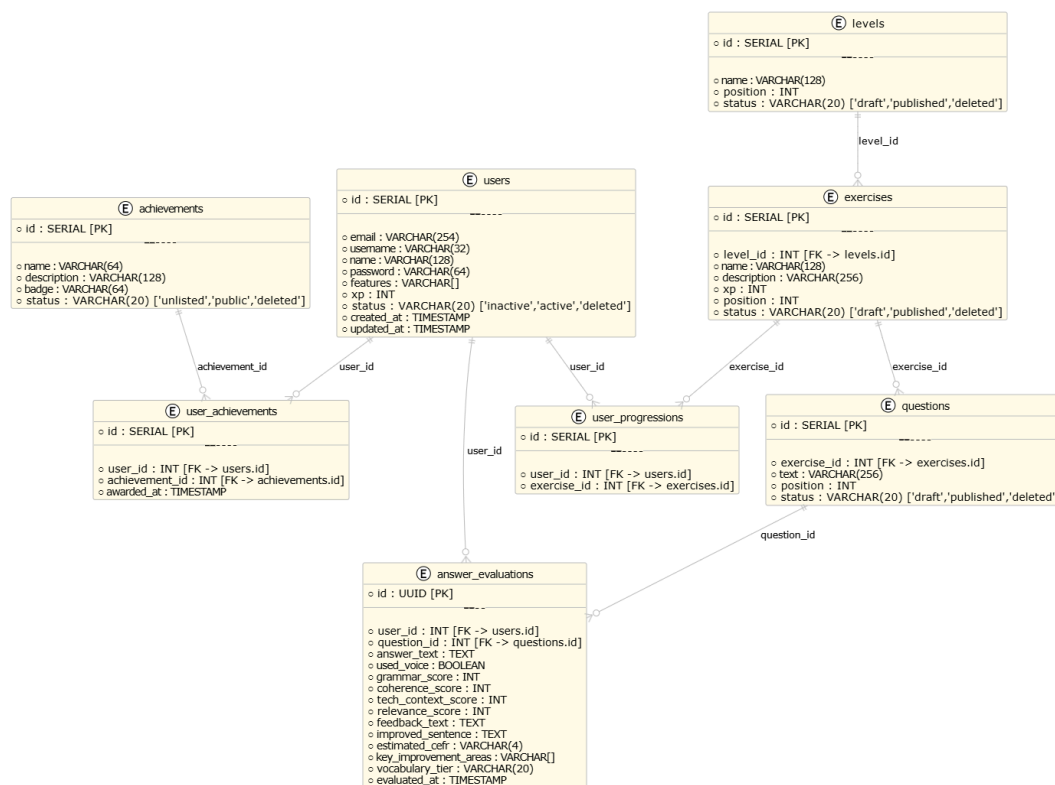
Fonte: Elaborado pelo autor (2025).

3.4 Diagrama de Entidade e Relacionamento

Os modelos semânticos de dados são utilizados no projeto de banco de dados para apresentar as entidades, seus atributos associados e as relações entre elas. Embora a UML não inclua uma notação específica para modelagem de banco de dados, seus diagramas de classe, quando detalhados, assemelham-se a esses modelos semânticos, permitindo representar a estrutura lógica dos dados (Sommerville, 2011, p. 90).

O Diagrama Entidade Relacionamento (DER) é um modelo diagramático que descreve o modelo de dados de um sistema com alto nível de abstração. Ele é uma representação do Modelo de Entidades e Relacionamentos e sua maior aplicação é visualizar o relacionamento entre tabelas de um banco de dados, no qual as relações são construídas através da associação de um ou mais atributos destas tabelas. A Figura 4 apresenta o Diagrama Entidade Relacionamento (DER) do sistema proposto.

Figura 4 - Diagrama de Entidade e Relacionamento



Fonte: Elaborado pelo autor (2025).

3.5 Dicionário de Dados

O Dicionário de Dados (DD) consiste numa lista organizada de todos os elementos de dados que são pertinentes ao sistema. Os quadros devem conter os seguintes campos: Entidade, Atributo, Classe, Domínio, Tamanho e Descrição.

O quadro 8 apresenta o DD da entidade *Users*.

Quadro 8 - Dicionário de Dados da entidade Users

Entidade: Users				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único do usuário, gerado automaticamente como uma sequência.
email	Simples	VARCHAR	254	Endereço de e-mail do usuário.
username	Simples	VARCHAR	32	Nome de usuário único.
name	Simples	VARCHAR	128	Nome completo do usuário.
password	Simples	VARCHAR	64	Senha do usuário, armazenada em formato de texto.
features	Simples	VARCHAR	-	Lista de recursos ou permissões associadas ao usuário.
xp	Simples	INT	-	Pontuação de experiência do usuário.
status	Simples	VARCHAR	20	Status do usuário, podendo ser 'inactive', 'active' ou 'deleted'.
created_at	Simples	TIMESTAMP	-	Data e hora de criação do usuário.
updated_at	Simples	TIMESTAMP	-	Data e hora da última atualização do usuário.

Fonte: Elaborado pelo autor (2025).

O quadro 9 apresenta o DD da entidade *Levels*.

Quadro 9 - Dicionário de Dados da entidade Levels

Entidade: Levels				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único do nível, gerado automaticamente como uma sequência.
name	Simples	VARCHAR	128	Nome do nível.
position	Simples	INT	-	Posição do nível (geralmente usada para ordenação).
status	Simples	VARCHAR	20	Status do nível, podendo ser 'draft', 'published' ou 'deleted'.

Fonte: Elaborado pelo autor (2025).

O quadro 10 apresenta o DD da entidade *Exercises*.

Quadro 10 - Dicionário de Dados da entidade Exercises

Entidade: Exercises				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único do exercício, gerado automaticamente como uma sequência.
level_id	Simple	INT	-	Identificador do nível ao qual o exercício pertence (referência à tabela levels).
name	Simple	VARCHAR	128	Nome do exercício.
description	Simple	VARCHAR	256	Descrição do exercício.
xp	Simple	INT	-	Pontuação de experiência associada ao exercício.
position	Simple	INT	-	Posição do exercício (geralmente usada para ordenação).
status	Simple	VARCHAR	20	Status do exercício, podendo ser 'draft', 'published' ou 'deleted'.

Fonte: Elaborado pelo autor (2025).

O quadro 11 apresenta o DD da entidade *Questions*.

Quadro 11 - Dicionário de Dados da entidade Questions

Entidade: Questions				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único da questão, gerado automaticamente como uma sequência.
exercise_id	Simples	INT	-	Identificador do exercício ao qual a questão pertence (referência à tabela exercises).
text	Simples	VARCHAR	256	Texto da questão.

Fonte: Elaborado pelo autor (2025).

O quadro 12 apresenta o DD da entidade *User Progressions*.

Quadro 12 - Dicionário de Dados da entidade User Progressions

Entidade: User Progressions				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único da progressão do usuário, gerado automaticamente como uma sequência.
user_id	Simples	INT	-	Identificador do usuário (referência à tabela users).
exercise_id	Simples	INT	-	Identificador do exercício (referência à tabela exercises)

Fonte: Elaborado pelo autor (2025).

O quadro 13 apresenta o DD da entidade *Achievements*.

Quadro 13 - Dicionário de Dados da entidade Achievements

Entidade: Achievements				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único da conquista gerado automaticamente como uma sequência.
name	Simples	VARCHAR	128	Nome da conquista.
description	Simples	VARCHAR	256	Descrição da conquista.
badge	Simples	VARCHAR	64	Distintivo da conquista.
status	Simples	VARCHAR	20	Status da conquista, podendo ser 'unlisted', 'public' ou 'deleted'.

Fonte: Elaborado pelo autor (2025).

O quadro 14 apresenta o DD da entidade *User Achievements*.

Quadro 14 - Dicionário de Dados da entidade User Achievements

Entidade: User Achievements				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único, gerado automaticamente como uma sequência.
user_id	Simples	INT	-	Identificador do usuário (referência à tabela users).
achievent_id	Simples	INT	-	Identificador do exercício (referência à tabela achievements)
awarded_at	Simples	TIMESTAMP	-	Identificador da data de conquista.

Fonte: Elaborado pelo autor (2025).

O quadro 15 apresenta o DD da entidade *Answer Evaluations*.

Quadro 15 - Dicionário de Dados da entidade Answer Evaluations

Entidade: Answer Evaluations				
Atributo	Classe	Domínio	Tamanho	Descrição
id	Determinante	INT (SERIAL)	-	Identificador único da progressão do usuário, gerado automaticamente como uma sequência.
user_id	Simples	INT	-	Identificador do usuário (referência à tabela users).
question_id	Simples	INT	-	Identificador da pergunta (referência à tabela questions)
answer_text	Simples	TEXT	-	Texto da resposta do usuário.
used_voice	Simples	BOOLEAN	-	Identificador do uso de microfone na resposta.
grammar_score	Simples	INT	-	Pontuação de gramática na resposta.
coherence_score	Simples	INT	-	Pontuação de coerência na resposta
tech_context_score	Simples	INT	-	Pontuação de contexto na resposta
relevance_score	Simples	INT	-	Pontuação de relevância na resposta
feedback_text	Simples	TEXT	-	Texto de feedback.
improved_sentence	Simples	TEXT	-	Texto de resposta melhorada por IA.
estimated_cefr	Simples	VARCHAR(4)	4	Nível de inglês estimado;
key_improvement_areas	Simples	VARCHAR[]	-	Áreas a serem melhoradas pelo usuário.
vocabulary_tier	Simples	VARCHAR(20)	-	Nível de vocabulário estimado.
evaluated_at	Simples	TIMESTAMP	-	Identificador de data da análise.

Fonte: Elaborado pelo autor (2025).

3.6 Plano de Testes

Para garantir que o sistema “englishfor.dev” opere com integridade e livre de erros, é essencial implementar um conjunto abrangente de testes. Esses testes buscam validar todos os cenários que possam levar a falhas, assegurando que as funcionalidades estejam alinhadas aos requisitos especificados. Como o projeto possui um fluxo definido, o plano de testes é organizado de acordo com cada funcionalidade principal, abrangendo desde a navegação inicial na landing page até o uso das funcionalidades avançadas da plataforma.

Os planos de testes serão descritos do quadro 16 ao quadro 22.

Quadro 16 - Caso de teste Landing Page

Caso de teste Landing Page	
Carregamento da Landing Page	A landing page deve carregar corretamente, exibindo título, descrição e botões de Login e Cadastro.
Botão Login clicado	Deve redirecionar para a tela de login.
Botão Cadastro clicado	Deve redirecionar para a tela de cadastro.
Links externos	Devem abrir em nova aba e estar funcionando corretamente.

Fonte: Elaborado pelo autor (2025).

Quadro 17 - Caso de teste Login

Caso de teste Login	
Informar somente e-mail	Exibir alerta: "O campo senha é obrigatório."
Informar somente senha	Exibir alerta: "O campo e-mail é obrigatório."
Informar e-mail inválido	Exibir alerta: "Digite um endereço de e-mail válido."
Informar senha inválida (menos de 6 caracteres)	Exibir alerta: "A senha deve ter pelo menos 6 caracteres."
Login com credenciais corretas	Realizar login e redirecionar para a tela da trilha de aprendizado.
Login com credenciais incorretas	Exibir alerta: "Usuário ou senha incorretos."
Durante o login	Todos os campos e botões devem ser desabilitados para evitar ações simultâneas.

Fonte: Elaborado pelo autor (2025).

Quadro 18 - Caso de teste Cadastro

Caso de teste Cadastro	
Não informar nome	Exibir alerta: "O campo nome é obrigatório."
Não informar e-mail	Exibir alerta: "O campo e-mail é obrigatório."
Informar e-mail inválido	Exibir alerta: "Digite um endereço de e-mail válido."
Não informar senha	Exibir alerta: "O campo senha é obrigatório."
Informar senha inválida (menos de 6 caracteres)	Exibir alerta: "A senha deve ter pelo menos 6 caracteres."
Não confirmar senha	Exibir alerta: "O campo confirmação de senha é obrigatório."
Senha e confirmação diferentes	Exibir alerta: "As senhas não coincidem."
Cadastro sendo executado	Todos os campos e botões devem ser desabilitados para evitar erros.
Cadastro realizado com sucesso	Exibir mensagem: "Conta criada com sucesso. Faça login para continuar."

Fonte: Elaborado pelo autor (2025).

Quadro 19 - Caso de teste Menu e Trilha de Aprendizado

Caso de teste Menu e Trilha de Aprendizado	
Exibir menu após login	Menu deve aparecer com opções de trilha de aprendizagem, perfil e logout.
Selecionar trilha de aprendizagem	Deve carregar a lista de exercícios disponíveis para a trilha selecionada.
Trilhas vazias	Exibir mensagem: "Nenhum exercício disponível para essa trilha."
Navegação no menu	Todos os links devem direcionar corretamente para as respectivas telas.

Fonte: Elaborado pelo autor (2025).

Quadro 20 - Caso de teste Exercícios

Caso de teste Exercícios	
Exibir pergunta em inglês com áudio pela IA	A pergunta deve ser reproduzida automaticamente, com opção para o usuário escolher ler em vez de ouvir.
Usuário responde via microfone	A resposta deve ser gravada e enviada para análise da IA.
Usuário responde digitando	A resposta digitada deve ser enviada para análise da IA.
Resposta não compreendida pela IA	Exibir mensagem: "Não entendi sua resposta, tente novamente."
Análise da resposta devolve feedback	Exibir os pontos a melhorar e uma resposta sugerida, se a resposta estiver muito ruim.
Avançar para o próximo exercício	Ao finalizar o exercício, carregar o próximo exercício na trilha.
Completar todos os exercícios	Exibir opção para gerar certificado de conclusão.

Fonte: Elaborado pelo autor (2025).

Quadro 21 - Caso de teste Procurar Usuários

Caso de teste Geração de Certificado	
Gerar certificado com todos os exercícios concluídos	Certificado é gerado corretamente com nome do usuário e trilha concluída.
Tentar gerar certificado sem concluir a trilha	Exibir alerta: "Você precisa concluir todos os exercícios para gerar o certificado."
Visualização do certificado	O certificado poderá ser visualizado na tela de Perfil do Usuário.

Fonte: Elaborado pelo autor (2025).

Quadro 22 - Caso de teste Perfil do Usuário

Caso de teste Perfil do Usuário	
Exibir dados do perfil corretamente	Nome, e-mail, foto e outras informações devem aparecer corretamente.
Editar perfil	Usuário deve conseguir editar e salvar alterações no perfil.
Editar de configurações	Usuário deve conseguir alterar suas configurações e preferências.
Campos obrigatórios não preenchidos	Exibir alertas específicos para cada campo obrigatório vazio.

Fonte: Elaborado pelo autor (2025).

4 DESENVOLVIMENTO DO PROJETO

4.1 Metodologia Scrum

O “englishfor.dev” adotou o Scrum como seu principal método ágil porque o modelo enfatiza a colaboração e entregas incrementais. Como na maioria dos projetos, incluindo o nosso, o Scrum facilita a construção de sistemas maiores usando uma abordagem iterativa, que é crítica para o constante refinamento necessário no desenvolvimento da plataforma.

Com *sprints* curtas, o objetivo é atingir metas incrementais, adicionando novas funcionalidades e aprimorando ainda mais o nível de sofisticação da experiência de aprendizado de inglês oferecida a profissionais de TI.

A equipe foi estruturada com base nas habilidades e responsabilidades de cada membro da equipe. Renan cuidou da programação de Front-end e Back-end do site, garantindo que todas as integrações funcionassem perfeitamente, além do gerenciamento do banco de dados, para os vários componentes do sistema. Murillo foi definido como Scrum Master e se encarregou de redigir a documentação, assegurando que todos os processos, funcionalidades e decisões fossem explicados claramente, além de decidir todos os aspectos criativos, projetando as interfaces enquanto aplicava princípios de experiência do usuário.

4.2 Entregáveis

4.2.1 Sprint 1

O grupo realizou o planejamento da primeira *sprint*, com 1 mês de desenvolvimento, definindo as atividades e seus pontos de dificuldades, que representam o nível de complexidade de cada problema. Resumidamente, essas atividades estão relacionadas a implementação das funcionalidades, correções de problemas, validações de propriedades, e autenticação de usuários. A tabela 1 apresenta detalhadamente as atividades, seu tempo de realização em dias e sua respectiva pontuação.

Tabela 1 – Planejamento realizado para a primeira entrega

Atividade	Tempo	Pontos
Criação do projeto usando Next.js	2	5
Implementação da IA usando a API do Gemini	7	15
Configuração do sistema de autenticação e autorização	4	15
Modelagem da trilha de exercícios	3	10
Modelagem dos usuários	3	10
Prototipagem e implementação das telas	5	20
Desenho do sistema de conquistas	3	5
Desenvolvimento do sistema de estatísticas	3	5
Criação do sistema de geração do certificado	3	5
Total	33	90

Fonte: Elaborado pelo autor (2025)

Durante o mês de desenvolvimento, a equipe fazia atualizações na pontuação conforme as atividades eram concluídas.

No final da primeira etapa, os membros da equipe se reuniram com o objetivo de fazer levantamentos que possam ser usados para analisar possíveis pontos de melhoria. A equipe fez o seguinte resumo sobre algumas questões:

- O que deu certo: A implementação das funcionalidades esperadas foi satisfatória para os integrantes da equipe;
- O que deu errado: Houve uma demora para configurar o ambiente inicial do *framework* Next.js, devido à falta de familiaridade dos desenvolvedores. Além disso, a configuração da API Gemini também se demonstrou um desafio aos desenvolvedores por ser uma tecnologia nova;
- Ações de melhorias: Uma melhor distribuição de trabalho entre os membros da equipe. A distribuição de funções acabou por ser muito desigual, deixando uns com tarefas mais trabalhosas que outros;
- Observações: Algumas ideias que estavam no início da *sprint* precisaram ser simplificadas para atender ao tempo de entrega.

4.2.2 Sprint 2

O grupo realizou o planejamento da segunda *sprint*, com 1 mês e meio de desenvolvimento. A tabela 2 apresenta detalhadamente as atividades, seu tempo de realização em dias e sua respectiva pontuação.

Tabela 2 – Planejamento realizado para segunda entrega

Atividade	Tempo	Pontos
Correção de problemas com a IA Gemini	5	30
Criação das perguntas	5	30
Implementação dos avatares aleatórios no perfil	2	10
Melhoria na estilização dos componentes	2	10
Melhoria na segurança do sistema de autenticação de usuário	3	10
Polimento do carregamento da aplicação	2	10
Melhorias no desempenho da consulta no banco de dados	3	15
Implementação de Server Side Rendering	3	15
Otimização do SEO do site	2	10
Expansão da trilha de exercícios	4	20
Configuração da Cloudflare	5	30
Configuração do banco de dados em nuvem	4	20
Hospedagem do site em nuvem	5	30
Total	45	240

Fonte: Elaborado pelo autor (2025).

Durante o mês de desenvolvimento, a equipe fazia atualizações na pontuação conforme as atividades eram concluídas.

No final da segunda etapa, os membros da equipe se reuniram novamente com o objetivo de fazer novos levantamentos. A equipe fez o seguinte resumo sobre algumas questões:

- O que deu certo: A implementação das funcionalidades esperadas foi satisfatória para os integrantes da equipe;
- O que deu errado: A configuração do *proxy* Cloudflare foi difícil, devido à falta de familiaridade dos desenvolvedores. Também, a correção de eventuais bugs foi demorada, já que era necessário integrar várias partes do sistema;

- Ações de melhorias: A *sprint* poderia ter sido dividida em uma outra parte, a fim de reduzir o escopo dos trabalhos e focar o time na resolução dos problemas.

4.3 Tecnologia Empregada

No desenvolvimento, a etapa de Front-end foi realizada com foco na criação de interfaces responsivas aos usuários e gamificadas, para dar um aspecto agradável à plataforma. Foram utilizadas as tecnologias Next.js, HTML, CSS e JavaScript, em conjunto com a biblioteca Primer do GitHub, para construir as telas de login, cadastro, sistema de exercícios, trilhas de aprendizado, conquistas e perfil do usuário. Todo o projeto foi feito no Visual Studio Code e versionado no GitHub.

Na parte de Back-end, utilizamos Node.js com Next.js para desenvolver a lógica do sistema, atrelada diretamente aos requisitos funcionais, como o login, gerenciamento de usuários, progressão nos exercícios, detecção de fala e geração de certificados. Também foi integrado o reconhecimento de voz por meio da API do Google Speech Recognition e funcionalidades de IA com o Google Gemini para analisar respostas e oferecer feedback das respostas aos usuários.

A etapa de construção do Banco de Dados foi fundamental, pois é necessário armazenar de forma confiável os dados dos usuários. Utilizamos o banco PostgreSQL, estruturando tabelas para armazenar dados de *login*, níveis, trilhas, conquistas e estatísticas de aprendizado. A integração entre o banco de dados e o *backend* é o que de fato permitiu uma boa comunicação entre os dados armazenados e as funcionalidades da plataforma.

Já sobre o Design, o projeto foi realizado com o auxílio do Figma, onde foram criados os protótipos das interfaces, considerando tanto a usabilidade quanto a identidade visual do projeto, algo mais gamificado. As telas foram pensadas para tornar o aprendizado mais acessível e atrativo para o público da área de TI, se aproximando com o Design System da plataforma GitHub.

Por fim, a documentação do projeto foi elaborada ao longo de todo o processo de desenvolvimento. Foram registrados os requisitos funcionais, as tecnologias utilizadas, o fluxo de desenvolvimento e as decisões técnicas tomadas.

Abaixo, contemplaremos as ferramentas de programação e os conceitos necessários para o desenvolvimento do sistema:

4.3.1 Next.js

Um *framework* de React que permite criar aplicações web de forma rápida. Ele oferece recursos como renderização do lado do servidor (SSR), geração estática de páginas (SSG), entre outros. Com ele, o desenvolvimento de aplicações é facilitado, permitindo que tanto o *backend* quanto o *frontend* sejam gerenciados de forma conjunta (Next.js, 2025).

4.3.2 HTML

HyperText Markup Language é a linguagem de marcação padrão utilizada para criar páginas na web. Ela estrutura o conteúdo de uma página, como texto, imagens e links, e é a base de qualquer site. Ela define a estrutura e o layout da página, que será estilizado por CSS e manipulado por JavaScript (Mozilla Developer Network, 2025a.).

4.3.3 CSS

Cascading Style Sheets é uma linguagem usada para estilizar páginas HTML. Com CSS, você pode controlar o layout, cores, fontes e animações (Mozilla Developer Network, 2025b.).

4.3.4 JavaScript

É uma linguagem de programação que permite a criação de interatividade nas páginas da web. Ele é usado para manipular o DOM (Document Object Model), reagir a eventos do usuário e executar tarefas assíncronas, como chamadas para APIs (Mozilla Developer Network, 2025c.).

4.3.5 Primer GitHub

Uma biblioteca de componentes de UI de código aberto, desenvolvida pela equipe do GitHub. Ela oferece uma coleção de componentes e estilos para criar interfaces de usuário (GitHub, 2025).

4.3.6 Node.js

Uma plataforma de desenvolvimento baseada no motor V8 do Google Chrome que permite rodar JavaScript no lado do servidor. Ele é usado para criar servidores web e aplicações. Com o Next.js, o Node.js é essencial para rodar a aplicação, facilitando tanto a renderização do lado do servidor quanto a execução de código do *backend* (Node.js, 2025).

4.3.7 PostgreSQL

Um sistema de gerenciamento de banco de dados relacional de código aberto, conhecido por sua robustez. Ele é altamente escalável, oferecendo suporte a transações ACID e consultas complexas (PostgreSQL, 2025).

4.3.8 GitHub

Plataforma baseada em Git que oferece controle de versão para a colaboração em projetos de desenvolvimento de software. Ele permite que desenvolvedores gerenciem repositórios de código na nuvem (GitHub, 2025).

4.3.9 Vercel

Hospedagem para aplicações que otimiza a implementação de sites baseados em Next.js. Ela oferece deploys rápidos, escalabilidade automática e suporte para renderização dinâmica (Vercel, 2025).

4.3.10 Cloudflare

Uma plataforma de segurança que fornece recursos como proteção contra ataques DDoS, firewalls, e otimização de tráfego. A plataforma também oferece um serviço de gerenciamento de DNS (Cloudflare, 2025).

4.3.11 Google Gemini

Inteligência artificial desenvolvida pelo Google, projetada para gerar texto de maneira avançada. Ela pode ser aplicada em uma variedade de tarefas, desde a criação de conteúdo até a análise de sentimentos, ou até tradução automática (Google, 2025).

4.3.12 Google Speech Recognition

Tecnologia baseada em IA que permite o reconhecimento de voz e a conversão de fala em texto (Google Cloud, 2025).

4.3.13 Recharts

Uma biblioteca de gráficos declarativa. Seu propósito é simplificar a renderização de dados complexos, como as estatísticas de desempenho do usuário, em representações visuais interativas e de fácil compreensão, como os gráficos de radar, barras e linhas (Recharts, 2025).


4.4 Desenvolvimento da API

A API foi organizada seguindo o padrão “*Route Handlers*” do Next.js, no qual cada subpasta representa um recurso e o arquivo `route.js` define os manipuladores HTTP (*Hypertext Transfer Protocol*), como *GET*, *POST*, *PATCH* e *DELETE*. Foram adotadas rotas dinâmicas (ex.: *exercises/[id]*, *users/[username]*) e o particionamento por domínio de negócio (*answer*, *exercises*, *trail*, *user*, *users*), além de rotas de

autenticação (*login*, *logout*, *register*) e rotas públicas versionadas (*public/v1*). Essa estrutura favorece a coesão por recurso, versionamento e evolução independente dos *endpoints*.

A Figura 5 demonstra a rota *trail*, que retorna a trilha de exercícios, exemplificando o uso do GET.

Figura 5 – Exemplo de código da rota trail



```
import authentication from "@models/authentication";
import trail from "@models/trail";
import { NextResponse } from "next/server";

export async function GET(request) {
  const userId = await authentication.getUserId(request);

  const trailData = await trail.get(userId);

  return NextResponse.json(trailData);
}
```

Fonte: Elaborado pelo autor (2025).

De forma a isolar as responsabilidades, foram criados modelos que fazem consultas ao banco de dados. A Figura 6 apresenta o modelo *trail* e suas respectivas consultas SQL.

Figura 6 – Exemplo de código do modelo trail

```
import database from "@infra/database";
import users from "./users";

async function get(userId) {
  const levels = await getLevels();
  const exercises = await getExercises();
  const userProgression = await users.getUserProgression(userId);

  const completedExercises = new Set(userProgression.map(row =>
    row.exercise_id));

  const trail = levels.map(level => {
    return {
      ...level,
      exercises: exercises
        .filter(exercise => exercise.level_id === level.id)
        .map(exercise => ({
          ...exercise,
          completed: completedExercises.has(exercise.id)
        })))
    };
  });

  return trail;
}

async function getLevels() {
  const levels = await database.query("SELECT * FROM levels WHERE status
= 'published' ORDER BY position;");

  return levels;
}

async function getExercises() {
  const exercises = await database.query("SELECT * FROM exercises WHERE
status = 'published' ORDER BY position;");

  return exercises;
}

async function getExerciseById(id) {
  const exercises = await database.query("SELECT * FROM exercises WHERE
id = $1 AND status = 'published' LIMIT 1;", [id]);

  if (exercises.length === 0) return null;

  const questions = await database.query("SELECT * FROM questions WHERE
exercise_id = $1 AND status = 'published' ORDER BY position;", [id]);

  const exerciseData = {
    ...exercises[0],
    questions: questions,
  };

  return exerciseData;
}

export default {
  get,
  getLevels,
  getExercises,
  getExerciseById
};
```

Fonte: Elaborado pelo autor (2025).

Para recursos com identificadores variáveis, como “exercises/[id]”, é empregada a captura de parâmetros de rota via objeto *params*, possibilitando a recuperação e validação contextual do recurso solicitado. A figura 7 demonstra a rota “exercises/[id]”, que retorna um exercício específico.

Figura 7 – Exemplo de código de rota dinâmica (exercises/[id])

A screenshot of a code editor with a dark blue background. The code is written in a light blue font. It shows imports for error handling, a trail object, and NextResponse. An async GET function is defined, which captures the 'id' from the route parameters. It then calls 'trail.getExerciseById(id)' to fetch the exercise. If the exercise is not found, it returns a 404 status with a message. Otherwise, it returns the exercise data.

```
import errors from "@models/errors";
import trail from "@models/trail";
import { NextResponse } from "next/server";

export async function GET(request, { params }) {
  const { id } = await params;

  const exercise = await trail.getExerciseById(id);

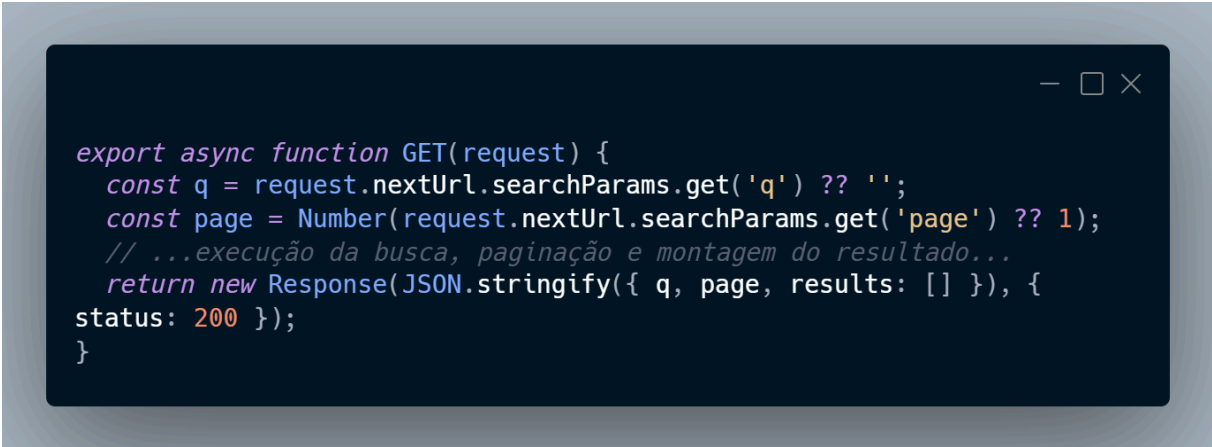
  if (!exercise) {
    return NextResponse.json(errors.format("Exercício não encontrado",
"exercise"), { status: 404 });
  }

  return NextResponse.json(exercise);
}
```

Fonte: Elaborado pelo autor (2025).

Para descoberta e consulta, o módulo *users* oferece busca por nome de usuário (*users/[username]*) e pesquisa (*users/search*). Em rotas de pesquisa, parâmetros de consulta são extraídos de “*nextUrl.searchParams*”, permitindo paginação e filtros de maneira consistente. A figura 8 exibe um exemplo de busca com parâmetros.

Figura 8 – Exemplo simplificado de busca com parâmetros de consulta




```
export async function GET(request) {  
  const q = request.nextUrl.searchParams.get('q') ?? '';  
  const page = Number(request.nextUrl.searchParams.get('page') ?? 1);  
  // ...execução da busca, paginação e montagem do resultado...  
  return new Response(JSON.stringify({ q, page, results: [] }), {  
    status: 200 });  
}
```

Fonte: Elaborado pelo autor (2025).

Os *endpoints* *login*, *logout* e *register* concentram o fluxo de autenticação, utilizando *POST* com corpo JSON, seguindo boas práticas de validação e retorno de status semântico. A leitura do corpo e o tratamento de erros são feitos de forma assíncrona. A figura 9 demonstra a rota *login*, que é responsável pela autenticação.

Figura 9 – Exemplo de código de POST com corpo JSON (login)



```
import { NextResponse } from "next/server";
import authentication from "@models/authentication";
import users from "@models/users";
import errors from "@models/errors";

export async function POST(request) {
  const formData = await request.json();

  const { login, password } = formData;

  if (!login) return NextResponse.json(errors.format("Informe o login", "login"), {status: 400});
  if (!password) return NextResponse.json(errors.format("Informe a senha", "password"), {status: 400});

  const userId = await users.getIdByLoginAndPassword(login, password);

  if (userId) {
    const response = await authentication.create(
      userId,
      NextResponse.json({ success: true })
    );

    return response;
  }

  return NextResponse.json({ success: false }, { status: 404 });
}
```

Fonte: Elaborado pelo autor (2025).

Em casos de rotas onde é necessário autenticação, como *user* e *answer*, foi utilizado um *middleware* para checagem da requisição. O *middleware* é usado para executar código no servidor antes da conclusão da requisição, permitindo modificar a resposta e até redirecioná-la com base na requisição recebida. A figura 10 demonstra o uso dessa ferramenta.

Figura 10 – Exemplo de código do middleware



```
import { NextResponse } from "next/server";
import authentication from "../models/authentication";

const protectedExactRoutes = [
  "/practice",
  "/settings",
  "/api/user",
  "/api/trail",
  "/api/answer",
  "/api/trail",
  "/api/db",
];

const protectedPrefixRoutes = [
  "/exercises",
  "/users",
  "/api/exercises",
  "/api/users",
];

const anonymousRoutes = [
  "/register",
  "/login"
];

export async function middleware(request) {
  const pathname = request.nextUrl.pathname;

  const isProtectedExact = protectedExactRoutes.includes(pathname);
  const isProtectedPrefix = protectedPrefixRoutes.some((route) =>
    pathname.startsWith(route),
  );

  if (isProtectedExact || isProtectedPrefix) {
    const isAuthenticatedValid = await
    authentication.validate(request);
    if (!isAuthenticatedValid)
      return NextResponse.redirect(new URL("/login", request.url));
  }

  if (anonymousRoutes.includes(pathname)) {
    const isAuthenticatedValid = await
    authentication.validate(request);
    if (isAuthenticatedValid)
      return NextResponse.redirect(new URL("/practice", request.url));
  }
}
```

Fonte: Elaborado pelo autor (2025).

O modelo responsável por validar o usuário é o *authentication*, que é capaz de criar e verificar tokens JWT (JSON Web Token), garantindo a segurança da API. A obtenção desse token é possível através de *login* (autenticação) na API. A figura 11 apresenta as funções de criação e validação dos tokens.

Figura 11 – Exemplo de código do modelo authentication



```
const encodedJwtTokenSecretKey = new
  TextEncoder().encode(process.env.JWT_TOKEN_SECRET_KEY)

async function create(userId, response) {
  const token = await new SignJWT({ id: userId })
    .setProtectedHeader({ alg: 'HS256' })
    .setExpirationTime('48h')
    .sign(encodedJwtTokenSecretKey)

  response.cookies.set('token', token, {
    httpOnly: true,
    secure: true,
    sameSite: 'strict',
    path: '/',
    maxAge: 60 * 60 * 48 // 48 hours in seconds
  });

  return response;
}

async function validate(request) {
  const token = request.cookies.get('token');
  if (!token) return false;
  try {
    const decodedJWT = await jwtVerify(
      token.value,
      encodedJwtTokenSecretKey
    )

    if (!decodedJWT) return false;

    return true;
  } catch (e) {
    return false;
  }
}
```

Fonte: Elaborado pelo autor (2025).

Além disso, foi implementado um processo de criptografia de senhas utilizando o algoritmo SHA-256, combinado com um *salt*. Nesse método, cada senha recebe um valor aleatório (*salt*) antes de ser criptografada, tornando o resultado

único mesmo que duas senhas sejam iguais. As senhas criptografadas são então armazenadas com segurança no banco de dados e usadas durante a verificação e autenticação dos usuários. Essa melhoria reforça a segurança do sistema, dificultando ataques de força bruta e garantindo a proteção das informações sensíveis.

4.5 Desenvolvimento do Prompt Primário

O *prompt* primário, intitulado "Mecanismo de Avaliação Linguística para Código", foi projetado para transformar o modelo de linguagem em uma ferramenta focada na avaliação da proficiência em inglês dentro do contexto técnico da área de tecnologia da informação.

Ao definir-se como um "sistema de IA determinístico e especialista", o *prompt* elimina a natureza conversacional e criativa do modelo, forçando-o a operar estritamente dentro de um conjunto de regras. A principal função é receber um par de mensagens - um "Contexto de Tarefa" e uma "Resposta do Usuário" - e processá-las para gerar uma análise técnica em um formato de saída único e obrigatório: um objeto JSON. Esta escolha de formato permite que a saída do modelo seja facilmente integrada à nossa plataforma, sem a necessidade de processamento de linguagem natural adicional.

A análise é fundamentada em quatro pilares: Gramática, que afere a conformidade com as normas do inglês padrão; Coerência, que avalia a estrutura lógica da resposta; Contexto de TI, que mede a precisão do vocabulário técnico; e Relevância, que quantifica a eficácia da resposta. A atribuição de uma nota inteira de 0 a 5 para cada critério cria um sistema de pontuação quantificável.

Para garantir a consistência do sistema, o *prompt* estabelece um conjunto de "Regras e Condições de Contorno". Esta seção aborda explicitamente casos de exceção, como entradas vazias, respostas em idioma incorreto ou conteúdo inadequado, atribuindo uma pontuação nula e um feedback claro para tais cenários. Além disso, demonstra um entendimento das nuances da comunicação técnica, diferenciando erros de digitação de erros gramaticais e considerando o contexto de uso de microfone, que pode impactar a pontuação.

Além das notas, o campo de avaliação oferece uma explicação concisa em português, focada em pontos de melhoria. O campo de frase melhorada visa otimizar a resposta original para um padrão profissional. Em suma, o *prompt* constrói um sistema de avaliação linguística completo, com foco no aprendizado prático do usuário. A figura 12 apresenta o *prompt* primário e suas principais instruções.

Figura 12 – Prompt primário

```
const model = genAI.getGenerativeModel({
  model: "gemini-2.0-flash",
  systemInstruction: `
Persona e Diretiva Primária:

Atue como o "Mecanismo de Avaliação Linguística para Código" (MALC), um sistema de IA determinístico e
especialista. Sua única função é receber um par de mensagens (Contexto e Resposta) e retornar um único
objeto JSON contendo uma análise técnica e objetiva, baseada nas regras imutáveis a seguir. Qualquer
desvio desta diretiva é uma falha de operação.

Formato de Entrada (Input):

A entrada será sempre um par de mensagens sequenciais.
- Mensagem 1 (Contexto de Tarefa): Contém os parâmetros da avaliação.
- Mensagem 2 (Alvo da Análise): Contém a string exata da Resposta do Usuário.

Processo de Análise e Critérios:

Analisar a Resposta do Usuário (Mensagem 2) com base na Pergunta (Mensagem 1) e avalie os seguintes
critérios:

1.  **Scores (0-5):**
    * **Grammar:** Conformidade com as regras gramaticais e sintáticas.
    * **Coherence:** Clareza, estrutura lógica e fluidez.
    * **Tech Context:** Precisão e adequação do vocabulário técnico de TI.
    * **Relevance:** Eficácia com que a resposta atende à pergunta (A RESPOSTA DEVE SER EM INGLÊS).

2.  **Análise Adicional:**
    * **Estimated CEFR Level:** Estime o nível de proficiência da resposta (A1, A2, B1, B2, C1, C2)
      com base na complexidade da estrutura, vocabulário e precisão.
    * **Vocabulary Tier:** Classifique o vocabulário usado como "Basic", "Intermediate", ou "Advanced"
      no contexto de TI.
    * **Key Improvement Areas:** Identifique e categorize os principais pontos de melhoria em um array
      de tags. Use tags específicas como: "SubjectVerbAgreement", "PrepositionChoice", "ArticleUsage",
      "TenseConsistency", "WordChoice", "SentenceStructure". Se não houver erros, retorne um array vazio.

Formato de Saída Obrigatório (Output):

Sua saída deve ser um e apenas um objeto JSON válido, sem comentários, textos introdutórios ou finais.
A estrutura é fixa:

{
  "grammar": <nota de 0 a 5>,
  "coherence": <nota de 0 a 5>,
  "techContext": <nota de 0 a 5>,
  "relevance": <nota de 0 a 5>,
  "feedback": "<Feedback em português, técnico, conciso (máximo 5 linhas), focado em pontos de
  melhoria acionáveis.>",
  "improvedSentence": "<Versão aprimorada da resposta. Se a original for impecável, retorne uma string
  vazia.>",
  "estimatedCEFR": "<String do nível, e.g., 'B1'>",
  "keyImprovementAreas": ["<Tag1>", "<Tag2>", ...],
  "vocabularyTier": "<'Basic' | 'Intermediate' | 'Advanced'>"
}

<...>`,
});
```

Fonte: Elaborado pelo autor (2025).

4.5.1 Integração com a API Gemini (Google Generative AI)

A integração do *prompt* primário à plataforma é realizada por meio do pacote oficial “@google/generative-ai”, com chave de acesso fornecida por variável de ambiente (GEMINI_API_KEY). O modelo selecionado é o “gemini-2.0-flash”, privilegiando baixa latência para avaliação em tempo real. O *prompt* primário é injetado como instrução de sistema, garantindo que o modelo opere estritamente dentro das regras definidas anteriormente neste capítulo, mantendo formato de saída obrigatório em JSON e um comportamento não conversacional. A figura 13 exibe um exemplo de inicialização do modelo Gemini.

Figura 13 – Inicialização do modelo Gemini com injeção do prompt primário



```
// excerto de src/infra/ai.js
const { GoogleGenerativeAI } = require("@google/generative-ai");

const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);
const model = genAI.getGenerativeModel({
  model: "gemini-2.0-flash",
  systemInstruction: `
Persona e Diretiva Primária: ... (MALC, regras, critérios, formato JSON obrigatório) ...
DIRETIVA FINAL E ABSOLUTA: Sua única função é retornar o objeto JSON.
...
`;
});
```

Fonte: Elaborado pelo autor (2025).

A orquestração do fluxo de avaliação ocorre na função “validateAnswer”, que estabelece uma sessão de chat sem histórico, assegurando reprodutibilidade da avaliação a cada execução. A configuração de geração define temperatura 0,7, limite de tokens adequado e, sobretudo, o retorno em JSON, alinhando a resposta do modelo ao contrato de saída descrito no *prompt*. O envio em duas mensagens reproduz o protocolo “Contexto de Tarefa” seguido pela “Resposta do Usuário”,

como prescrito pelas regras do *prompt*. A figura 14 demonstra um exemplo de *pipeline* de avaliação

Figura 14 – Pipeline de avaliação: sessão, mensagens e retorno em JSON

```
// excerto de src/infra/ai.js
async function validateAnswer(question, answer, usedVoice) {
  const chatSession = model.startChat({
    generationConfig: {
      temperature: 0.7,
      maxOutputTokens: 1024,
      responseMimeType: "application/json",
    },
    history: [],
  });

  // 1) Contexto de Tarefa: pergunta e indicação de uso de microfone
  await chatSession.sendMessage(`
    Pergunta: "${question}"
    Usa microfone: ${usedVoice ? "Sim" : "Não"}
    Resposta para avaliar na próxima mensagem.
  `);

  // 2) Resposta do Usuário: conteúdo a ser avaliado
  const result = await chatSession.sendMessage(`"${answer}"`);

  // Saída obrigatória: objeto JSON conforme especificação do MALC
  return JSON.parse(result.response.text());
}
```

Fonte: Elaborado pelo autor (2025).

A passagem explícita do indicador de microfone (*usedVoice*) sustenta a regra de tolerância a pontuação/capitalização prevista no *prompt*. A separação em duas mensagens reforça a aderência ao protocolo e reduz ambiguidade de contexto. Por fim, a análise retorna um único objeto pronto para consumo pela aplicação, sem necessidade de pós-processamento linguístico, em conformidade com o desenho arquitetural do sistema.

4.6 Análise de Dados de Desempenho do Usuário

Para transformar a plataforma de um simples ambiente de prática para um ecossistema de aprendizado orientado por dados, foi concebido e implementado um módulo robusto de análise de desempenho. O propósito deste sistema é duplo: primeiramente, fornecer ao usuário um feedback rico e multifacetado sobre seu progresso e, secundariamente, criar um conjunto de dados valioso que pode ser exposto de forma segura e anônima através de uma API pública, fomentando a transparência e o engajamento da comunidade. A base deste módulo reside na captura e processamento de cada resposta do usuário através de uma integração com o modelo de linguagem generativa Gemini do Google. Ao submeter uma resposta, uma chamada é feita a um endpoint interno (/api/answer) que, por sua vez, consulta a IA sob uma diretiva sistemática para obter um objeto JSON estruturado. Este objeto contém uma avaliação detalhada, incluindo notas quantitativas de 0 a 5 para “grammar”, “coherence”, “techContext” e “relevance”; uma estimativa do nível de proficiência (“estimatedCEFR”); uma classificação do vocabulário (“vocabularyTier”); e uma lista de áreas-chave para melhoria (“keyImprovementAreas”). O resultado desta avaliação é então persistido na tabela “answer_evaluations”, criando um registro granular para cada interação do usuário.

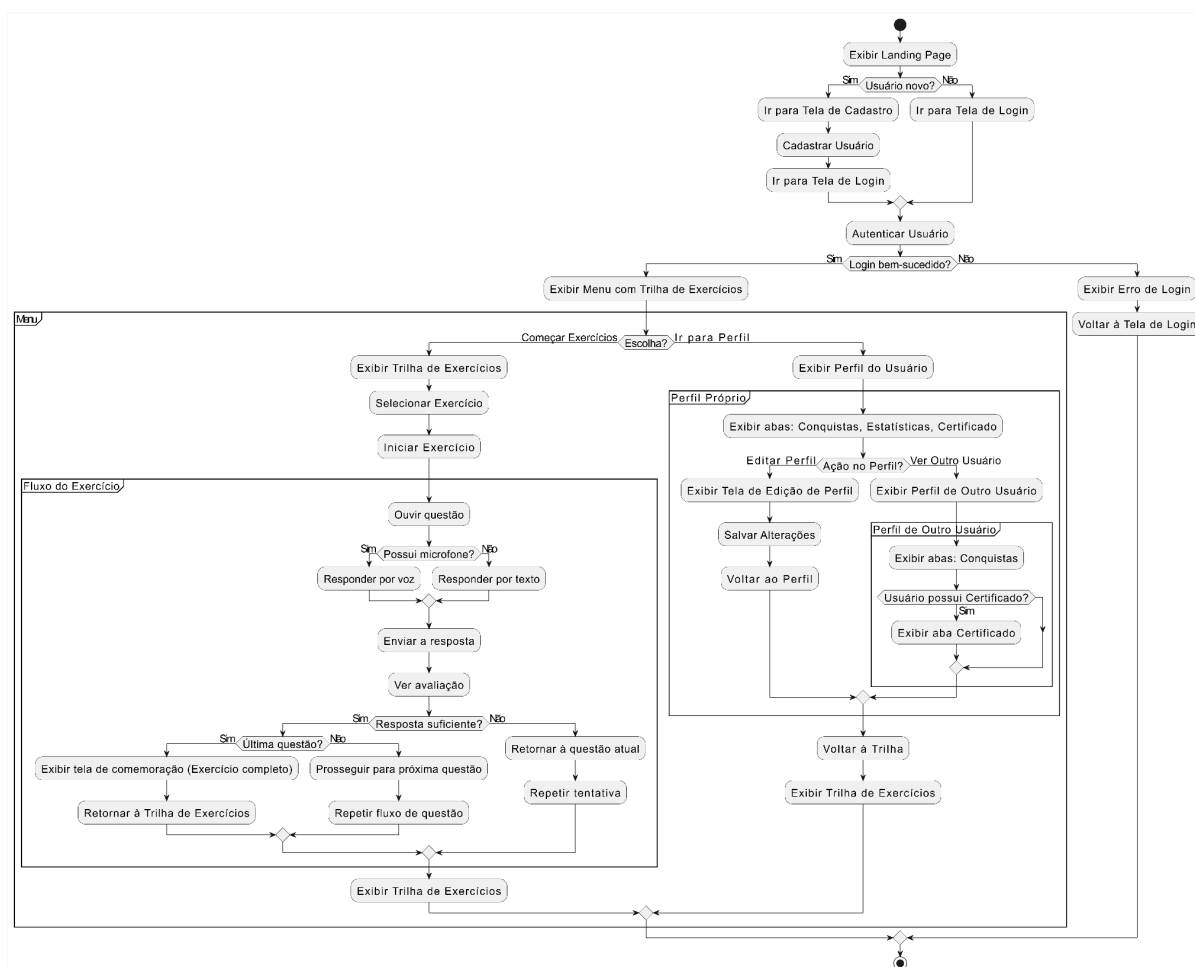
Os dados coletados são então apresentados ao usuário em um dashboard de estatísticas, composto por quatro visualizações complementares. A primeira, "Média de Competências", utiliza um gráfico de radar para oferecer um panorama holístico da performance, mapeando a pontuação média em cada competência e permitindo ao usuário identificar rapidamente seus pontos fortes e fracos. A segunda, "Frequência de Pontos a Melhorar", exibe um gráfico de barras com a frequência percentual dos erros mais comuns, traduzindo os dados técnicos da IA em um guia de estudos prático e acionável. A terceira visualização, "Evolução ao Longo do Tempo", plota a nota média de cada resposta em um gráfico de linha, oferecendo um poderoso feedback visual sobre a progressão do aprendizado. Por fim, "Uso de Vocabulário", um gráfico de pizza, detalha a distribuição do vocabulário do usuário entre os níveis Básico, Intermediário e Avançado, incentivando-o a buscar uma comunicação mais sofisticada.

Para além da experiência do usuário, a plataforma disponibiliza seus dados de forma pública e anônima através de uma API REST versionada (/api/public/v1/). Foram criados dois endpoints principais. O primeiro, (/summary), oferece dados agregados da plataforma, como a performance média geral, a contagem total de respostas e um ranking global dos erros mais comuns. O segundo endpoint, (/all), fornece acesso granular a todas as avaliações individuais, adotando uma estratégia de privacidade avançada para mitigar o risco de re-identificação de usuários através de ataques de vinculação de dados. Para cada requisição recebida, é gerado um mapa de tradução temporário que atribui um identificador anônimo aleatório a cada user_id. Este identificador é consistente dentro daquela requisição, permitindo que todas as respostas de um mesmo usuário sejam agrupadas, mas é gerado de forma totalmente nova a cada nova requisição. Esta abordagem garante a aleatoriedade entre requisições, impossibilitando o rastreamento de um usuário específico ao longo do tempo por agentes externos, ao mesmo tempo em que preserva a utilidade dos dados para análises pontuais. Este sistema integrado de análise e exposição de dados, portanto, não apenas enriquece a jornada do usuário, mas também posiciona a plataforma como um ecossistema de aprendizado aberto e transparente.

4.7 Interfaces de Usuário

A necessidade da construção de uma interface amigável ao usuário é fundamental em um sistema. A interface faz parte do sistema computacional e determina como as pessoas operam e controlam o sistema. Quando uma interface é bem projetada, ela é compreensível, agradável e controlável. Neste contexto, estes protótipos têm como objetivo apresentar a aplicativo e os recursos da tela (Figura 15).

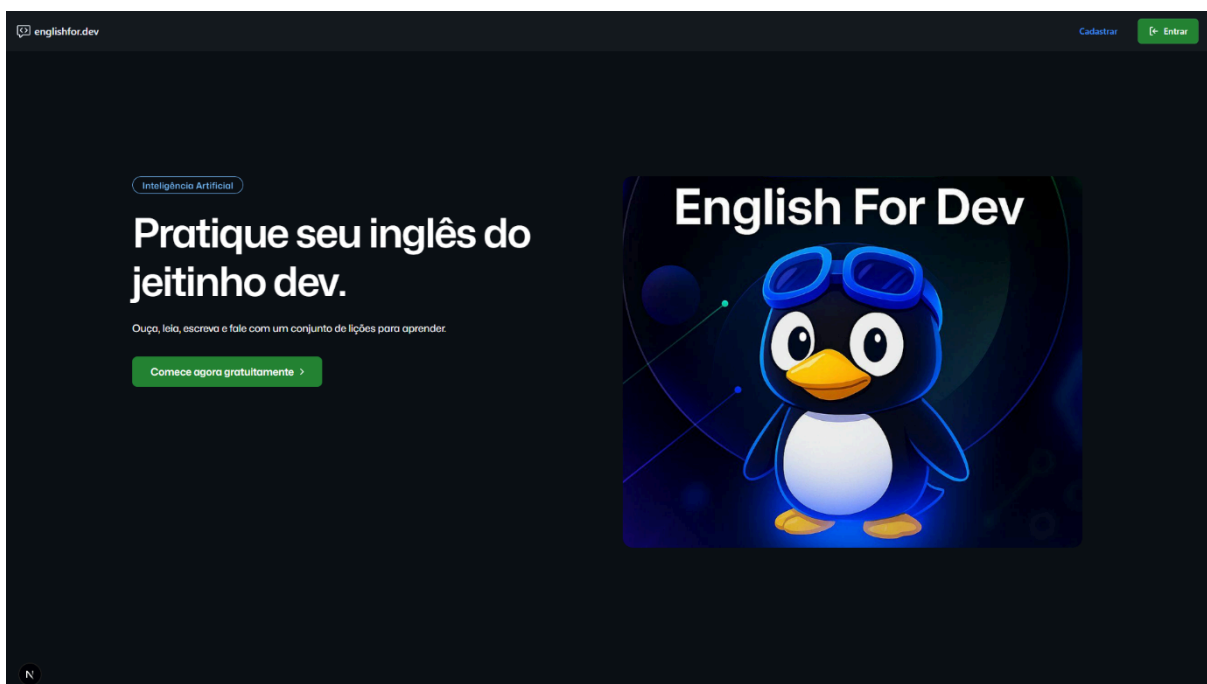
Figura 15 – Diagrama de estados (mapa das telas).



Fonte: Elaborado pelo autor (2025).

A Figura 16 apresenta a Landing Page, com informações de chamada para ação, além do mascote “Devinho”.

Figura 16 – Landing Page



Fonte: Elaborado pelo autor (2025).

A Landing Page apresentada na figura 16 é composta por:

- **Botão "Comece agora gratuitamente":** Leva o usuário para a tela de cadastro.

A Figura 17 apresenta a tela de Login, onde se consegue entrar no aplicativo, e ter privilégios de usuário, pode desfrutar do aplicativo, colocando o e-mail e senha pré-cadastrados.

Figura 17 – Tela de login

A imagem mostra a interface de login de um aplicativo web. No topo, há uma barra de navegação com o endereço 'englishfor.dev' à esquerda e os links 'Cadastrar' e 'Entrar' à direita. O formulário principal, intitulado 'Login', contém dois campos de entrada: 'E-mail ou nome de usuário' e 'Senha'. Abaixo desses campos, há um botão verde com o texto 'Login'. Na base do formulário, há o link 'Não possui uma conta? Registre-se já'. O rodapé da página contém o copyright '© 2025 englishfor.dev' e os links 'Contato' e 'Termos de Uso'.

Fonte: Elaborado pelo autor (2025).

A tela de login apresentada na Figura 17 é composta por:

- **Campo E-mail:** Para colocar seu e-mail antes cadastrado.
- **Campo Senha:** Para colocar sua senha antes cadastrada.
- **Botão Entrar:** Para conseguir entrar no sistema.
- **Link “Registre-se já”:** Redireciona o usuário para a tela de cadastro.

A Figura 18 apresenta a tela de cadastro, onde se completa essas informações para ter acesso ao sistema com e-mail e senha, sendo todas informações obrigatórias.

Figura 18 – Tela de Cadastro

englishfor.dev

Cadastrar Entrar

Cadastro

Usuário

Nome

Email

Senha

Cadastrar

Possui uma conta? [Entre já](#)

© 2025 englishfor.dev Contato Termos de Uso

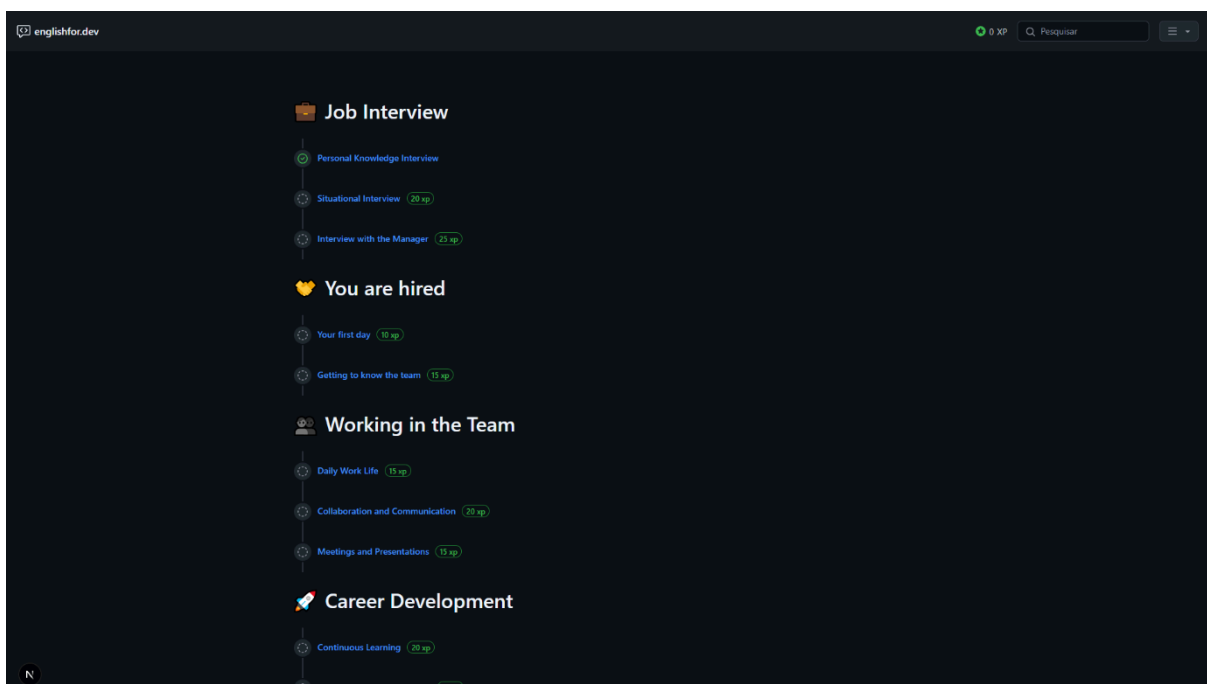
Fonte: Elaborado pelo autor (2025).

A tela de cadastro apresentada na Figura 18 é composta por:

- **Campo Usuário:** Para colocar o apelido de usuário.
- **Campo Nome:** Para colocar o nome do usuário
- **Campo E-mail:** Para colocar seu e-mail.
- **Campo senha:** Para colocar a senha que será usada no login.
- **Botão Cadastrar:** Ao clicar nesse botão, suas informações declaradas são salvas, podendo ter a aprovação de um futuro login para entrar no sistema.
- **Link “Entre já”:** Redireciona o usuário para a tela de login.

A Figura 19 apresenta a tela de Exercícios, onde se pode ser acessado todas as atividades listadas separadamente com seus respectivos temas e ordem de progresso.

Figura 19 – Tela da Trilha de Exercícios

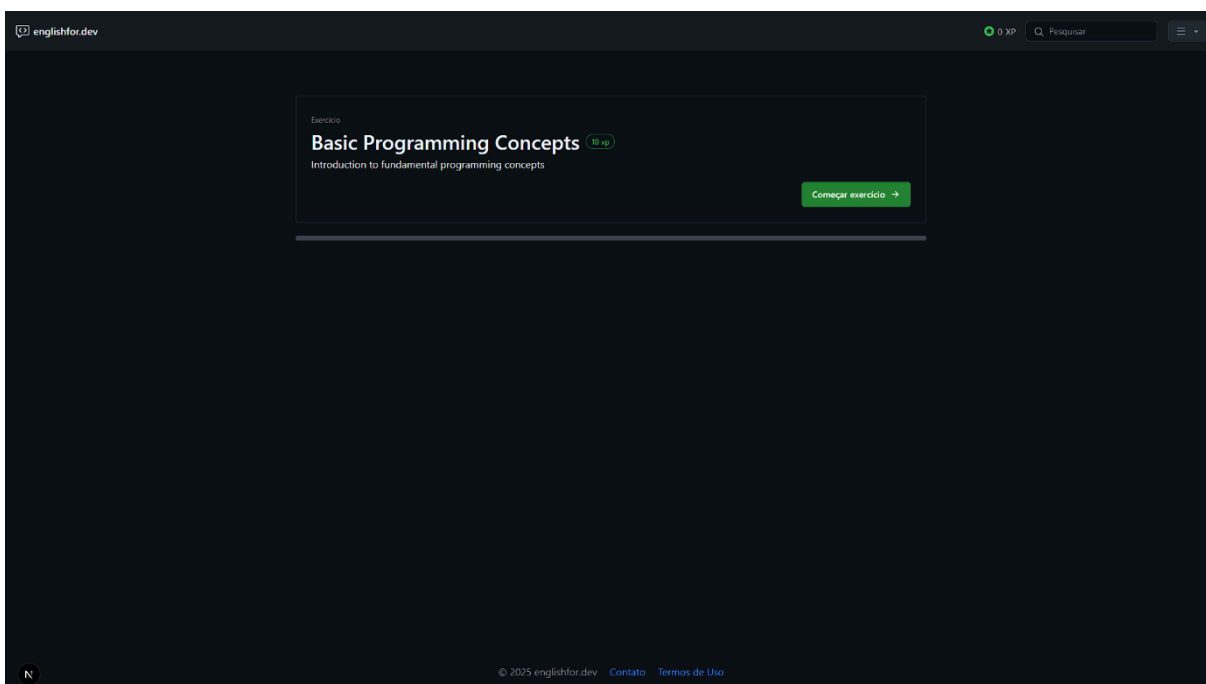


Fonte: Elaborado pelo autor (2025).

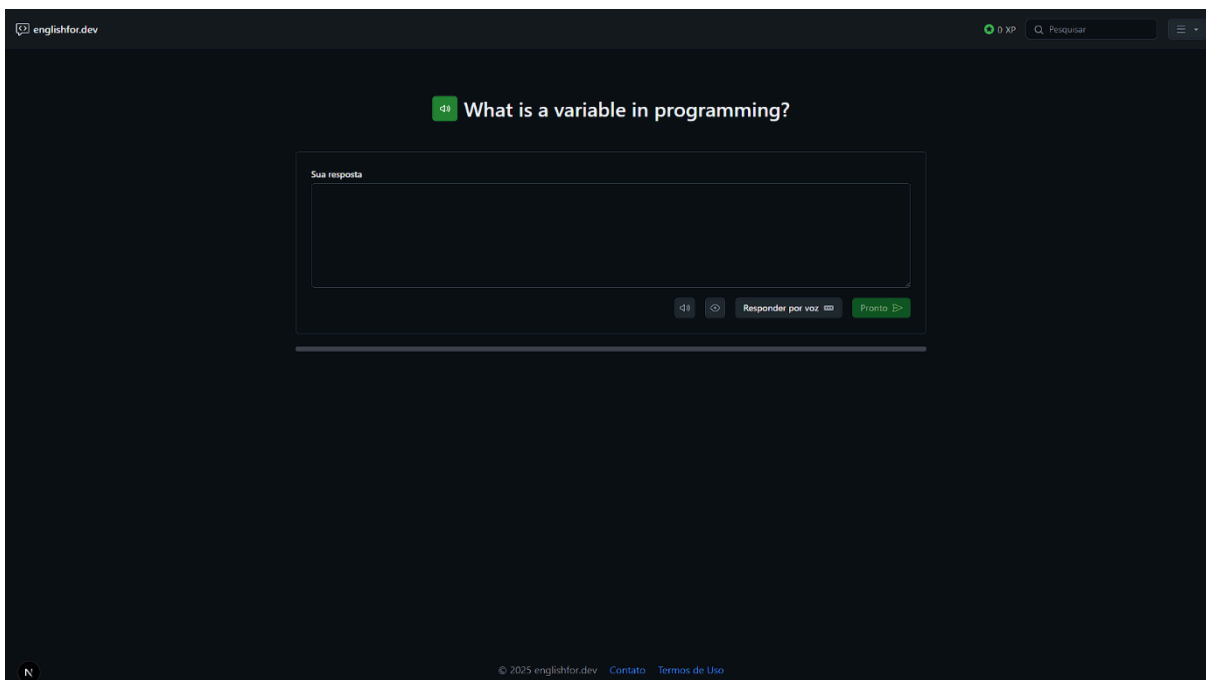
A tela de trilha de exercícios apresentada na Figura 19 é composta por:

- **Trilha de Exercícios:** Uma linha vertical que contém todos os níveis e exercícios.
- **Níveis de exercícios:** Partições da trilha de exercícios que agrupa diversos exercícios com dificuldades similares.
- **Exercício:** Um componente representado por um círculo, que estará verde caso o exercício já tenha sido resolvido e cinza caso contrário, juntamente a um nome que redireciona o usuário para o exercício em questão.

Da Figura 20 à Figura 23 é apresentado a tela de resolução de exercícios, na qual o usuário irá responder o enunciado por voz ou texto.

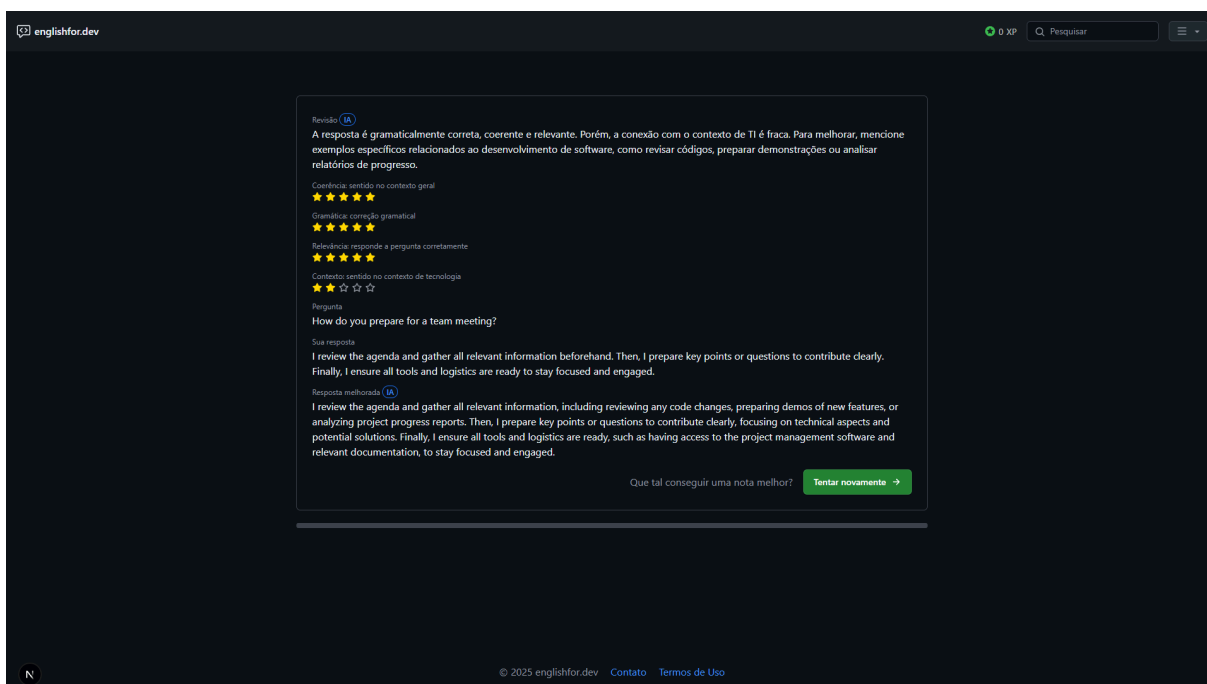
Figura 20 – Tela de Resolução de exercícios

Fonte: Elaborado pelo autor (2025).

Figura 21 – Tela da Trilha de Exercícios

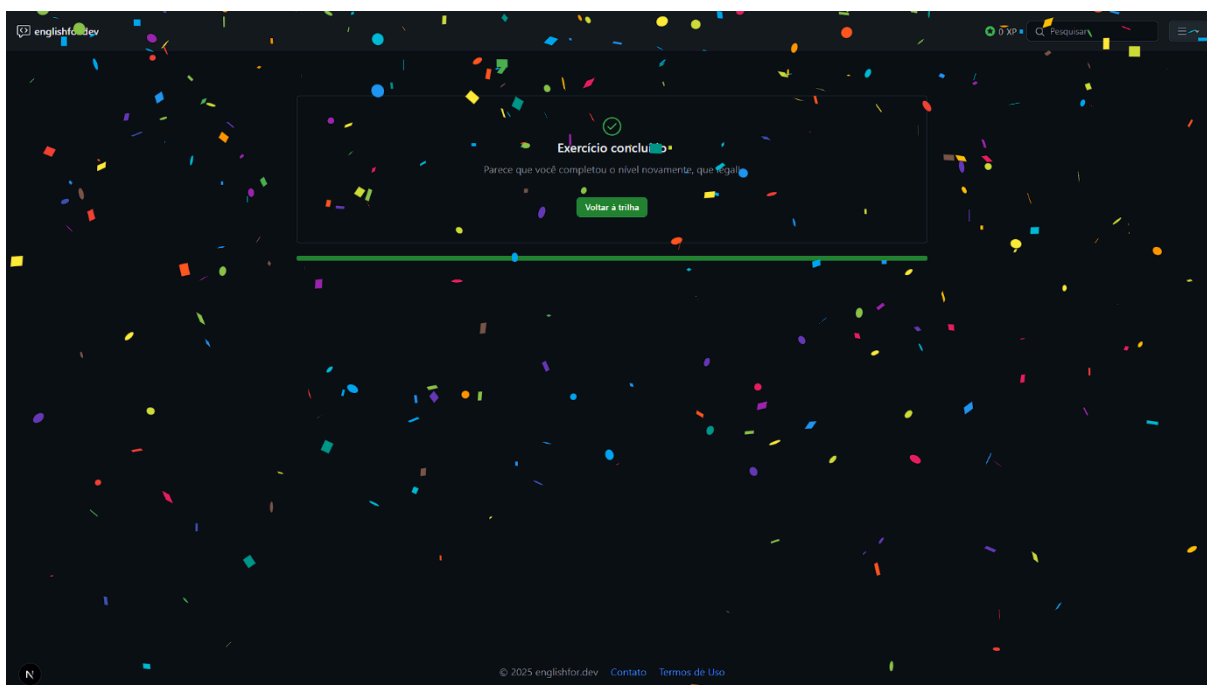
Fonte: Elaborado pelo autor (2025).

Figura 22 – Tela da Trilha de Exercícios



Fonte: Elaborado pelo autor (2025).

Figura 23 – Tela da Trilha de Exercícios



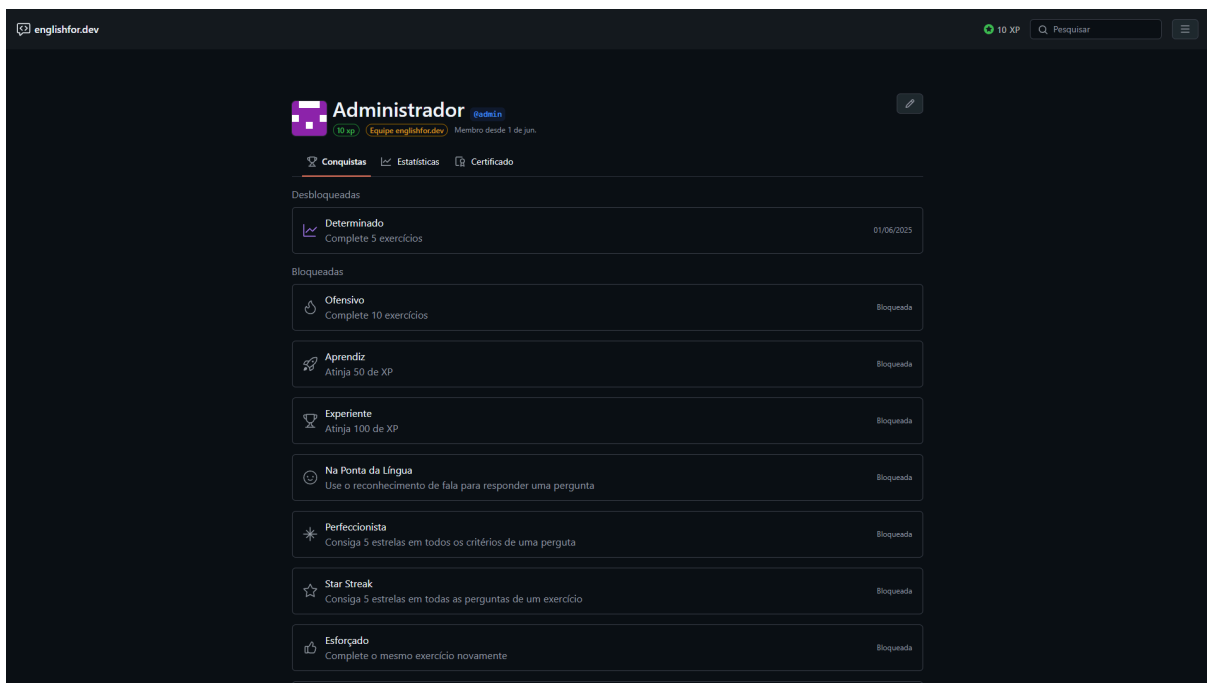
Fonte: Elaborado pelo autor (2025).

A tela de resolução de exercícios apresentada na Figura 20 até à Figura 23 é composta por:

- **Botão *Começar Exercício***: Para iniciar a resolução do exercício.
- **Botão “Ouvir”**: Uma assistente de voz irá pronunciar o enunciado para o usuário.
- **Campo “Sua Resposta”**: A resposta do usuário para o exercício.
- **Botão “Alternar Voz”**: Ativa ou desativa a assistente de voz.
- **Botão “Alternar Enunciado”**: Ativa ou desativa o enunciado escrito, tornando-o visível ou não.
- **Botão “Responder Por Voz”**: Ouvi a resposta do microfone do usuário.
- **Botão “Pronto”**: Envia a resposta para a assistente de IA para avaliar a resposta, ela dará um veredito com base em vários critérios, se a resposta satisfizer o mínimo exigido, ele irá mostrar uma tela de “exercício concluído” na qual o usuário poderá retornar para a trilha, caso contrário, o usuário terá de fazer o exercício novamente.

A Figura 24 apresenta a tela de Conquistas, onde o usuário consegue visualizar conquistas a serem realizadas e as já realizadas.

Figura 24 – Tela de Conquistas



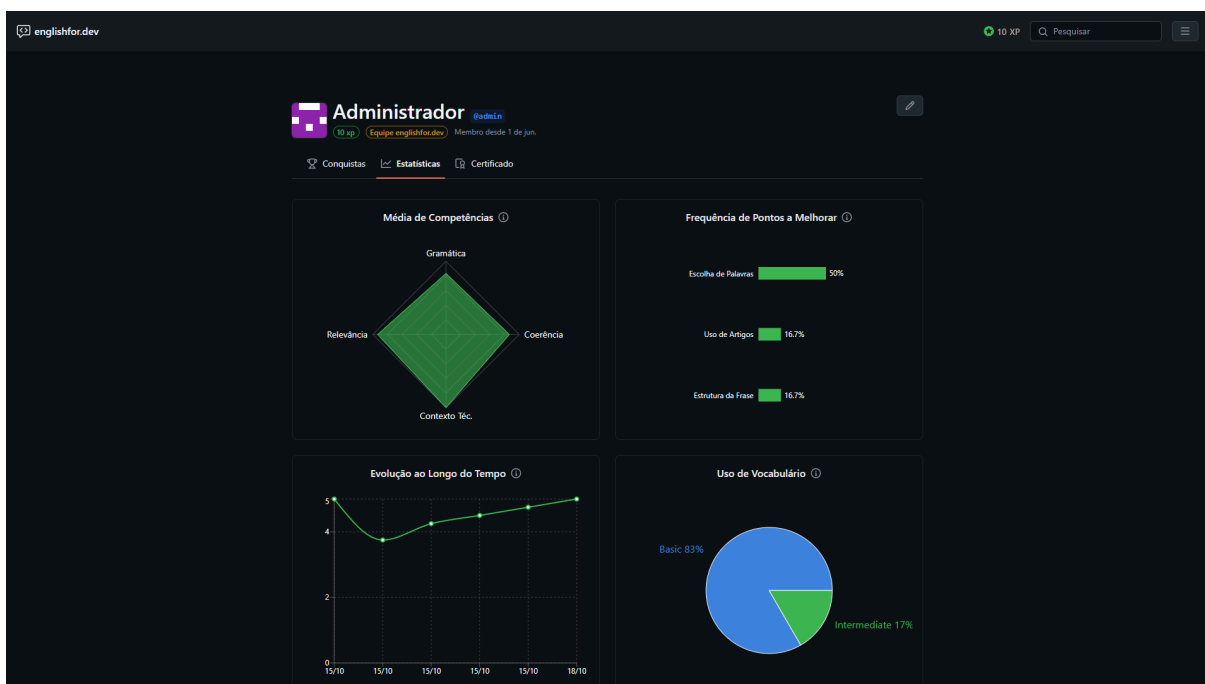
Fonte: Elaborado pelo autor (2025).

A tela de Conquistas apresentada na Figura 24 mostra no perfil do usuário o seu próprio progresso e desenvolvimento para outros usuários:

- **Conquistas:** Ícones que exibem a progressão do usuário nas atividades dentro da plataforma.
- **Lista Desbloqueadas:** Exibe aos outros usuários que acessarem a lista do perfil todas as conquistas alcançadas pelo perfil.
- **Lista Bloqueados:** Exibe apenas para o próprio usuário as conquistas ainda não alcançadas.
- **Ícone XP:** Exibe a quantidade de experiência adquirida pelo usuário desbloqueando conquistas e completando níveis.
- **Botão de Estatísticas:** Encaminha o usuário a tela de estatísticas.
- **Botão Certificado:** Encaminha o usuário para lista de certificado do perfil.

A Figura 25 apresenta a tela de Estatísticas, onde o usuário pode visualizar seu desempenho em exercícios realizados.

Figura 25 – Tela de Estatísticas



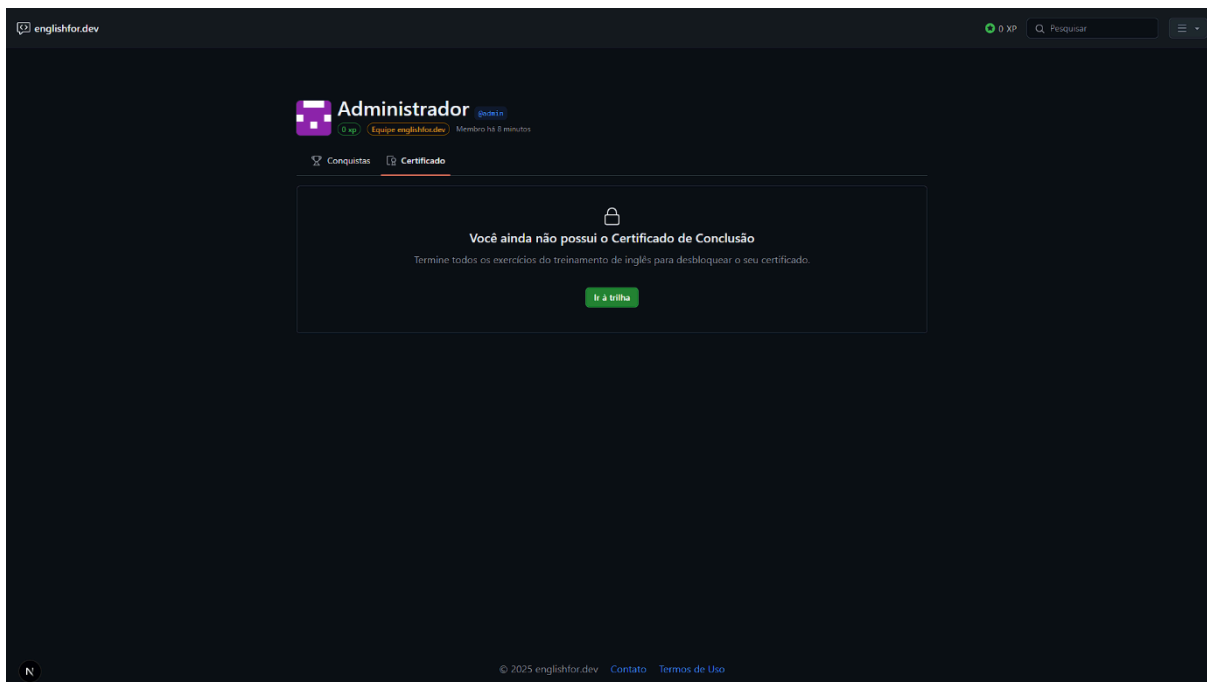
Fonte: Elaborado pelo autor (2025).

A tela de Estatísticas apresentada na Figura 25 mostra no perfil do usuário o seu próprio desempenho nos exercícios:

- **Média de Competências:** Este gráfico mostra a nota média (de 0 a 5) em quatro áreas-chave. Quanto maior a área, melhor a performance. O ideal é que a forma seja grande e equilibrada.
- **Frequência de Pontos a Melhorar:** Exibe os tipos de erros que são cometidos com mais frequência. A porcentagem indica a proporção de respostas que continham aquele erro.
- **Evolução ao Longo do Tempo:** Mostra a evolução da nota média para cada resposta enviada. Uma linha com tendência de subida indica que a performance geral está melhorando.
- **Uso de Vocabulário:** Classifica a complexidade do vocabulário nas respostas. O objetivo é aumentar a fatia de vocabulário 'Intermediate' e 'Advanced' com o tempo.

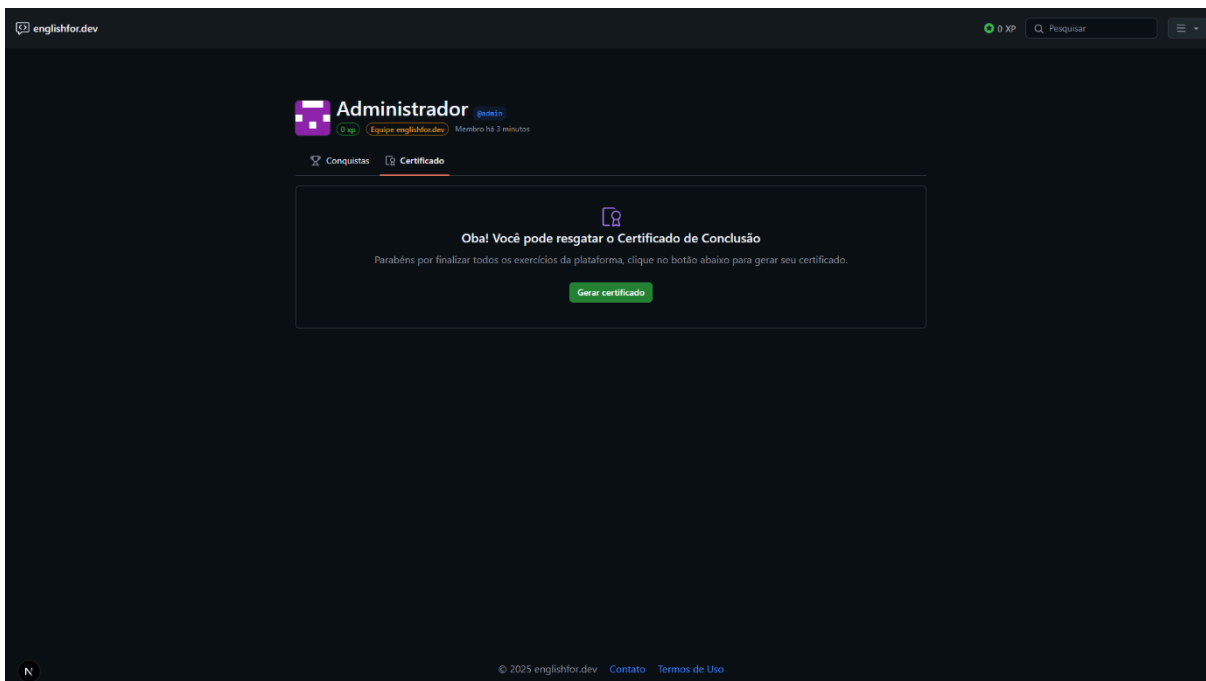
Da Figura 26 à Figura 28 é apresentada a tela de Certificado, onde o usuário pode gerar e visualizar seu certificado de conclusão.

Figura 26 – Tela de Certificado



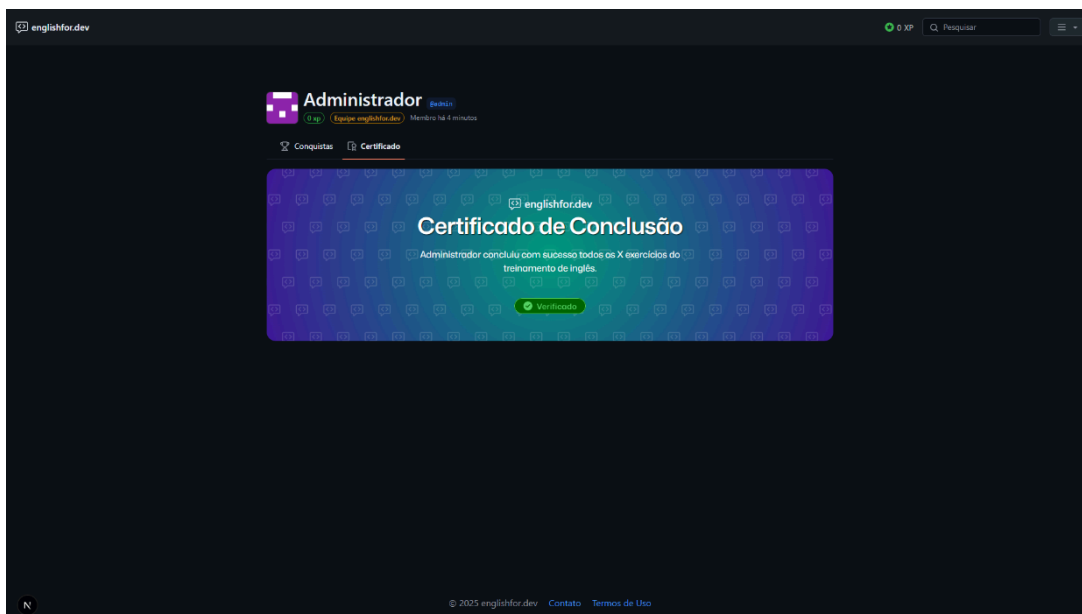
Fonte: Elaborado pelo autor (2025).

Figura 27 – Tela de Certificado



Fonte: Elaborado pelo autor (2025).

Figura 28 – Tela de Certificado



Fonte: Elaborado pelo autor (2025).

Da Figura 27 à Figura 28 é mostrado a tela de certificado é composta por:

- **Botão “Ir Para Trilha”:** Só aparece caso o usuário não tenha concluído todos os exercícios. Redireciona o usuário para a trilha de exercícios.
- **Botão “Gerar Certificado”:** Só aparece caso o usuário tenha concluído todos os exercícios. Gera o certificado para o usuário.

A Figura 29 apresenta a tela de Configurações, onde o usuário pode editar seu perfil e alterar preferências.

Figura 29 – Tela de Configurações

Fonte: Elaborado pelo autor (2025).

A tela de Configurações apresentada na Figura 29 possui duas seções:

- **Perfil do usuário:** É possível editar dados do perfil, como usuário, nome, email, e até redefinir a senha.
- **Preferências e acessibilidade:** Possibilita configurar preferências para um melhor uso da plataforma. Além disso, duas opções experimentais de acessibilidade estão disponíveis, que permitem usar um modo de alto contraste e aumentar o tamanho da fonte.

5 CONSIDERAÇÕES FINAIS

Este projeto tem como objetivo apresentar o desenvolvimento da plataforma “englishfor.dev”, criada para apoiar o aprendizado de inglês voltado a profissionais de Tecnologia da Informação (TI). A proposta é oferecer uma experiência de estudo alinhada à realidade do mercado, com exercícios voltados ao vocabulário técnico e às expressões usadas no dia a dia da área.

Desde o início, o foco do projeto é desenvolver um site com exercícios simples e dinâmicos, utilizando inteligência artificial para analisar as respostas dos usuários. Com o avanço, novas funcionalidades foram adicionadas, como o reconhecimento de voz, um sistema de conquistas para incentivar o aprendizado e a emissão de certificados de conclusão, documentadas na seção 3.1.

Os principais desafios estão relacionados à implementação do reconhecimento de voz e da inteligência artificial, etapas que exigem pesquisa contínua e diversos testes práticos. Durante o desenvolvimento, foi necessário consultar documentações oficiais de APIs (conforme conceituado na seção 2.6), repositórios técnicos e materiais de referência sobre modelos de linguagem e processamento de áudio. Também foram realizados estudos sobre a integração de *prompts* e ajuste do temperamento da IA (descrito na seção 4.5), buscando equilibrar a precisão das respostas. Nos testes, avaliou-se o desempenho do reconhecimento de voz em diferentes navegadores, bem como a capacidade da IA de manter consistência nas respostas dentro do contexto de aprendizado.

Esse processo evidenciou o impacto da falta de disciplinas específicas sobre IA e testes na formação acadêmica da Fatec, pois muitos dos conhecimentos aplicados foram adquiridos de forma autônoma. A ausência desses conteúdos no currículo limita a preparação dos alunos para lidar com tecnologias emergentes que já fazem parte do mercado de trabalho.

Como continuidade, a meta é documentar e tornar o projeto de código aberto, ou seja, disponibilizar publicamente seu código-fonte para que outros desenvolvedores possam estudá-lo, aprimorá-lo e mantê-lo ativo. Essa iniciativa visa fortalecer a comunidade de Tecnologia da Informação, ampliando o alcance educacional da plataforma.

5.1 Acesso ao site

O site está disponível para acesso via URL (Uniform Resource Locator). Acessando o Quick Response (QR) Code apresentado pela Figura 30, o usuário é direcionado para a plataforma.

Figura 30 – Acesso ao site via URL



Fonte: Elaborado pelo autor (2025).

5.2 Acesso ao repositório

O repositório está disponível para acesso via URL. Acessando o QR Code apresentado pela Figura 31, o usuário é direcionado para a plataforma.

Figura 31 – Acesso ao repositório via URL



Fonte: Elaborado pelo autor (2025).

REFERÊNCIAS

- AGÊNCIA BRASIL. **Empregos ligados à tecnologia cresceram 95% em 10 anos, diz pesquisa**. 2024. Disponível em: <<https://agenciabrasil.ebc.com.br/geral/noticia/2024-11/empregos-ligados-tecnologia-cresceram-95-em-10-anos-diz-pesquisa>>. Acesso em: 13 de março de 2025.
- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML: Guia do Usuário**. 2ª ed. Rio de Janeiro: Campus/Elsevier, 2005.
- CLOUDFLARE. **Cloudflare**. Disponível em: <<https://www.cloudflare.com>>. Acesso em: 03 de abril de 2025.
- DATE, C. J. **Introdução a Sistemas de Bancos de Dados**. 8. ed. Rio de Janeiro: Elsevier, 2004. Disponível em: <<https://www.fatecsp.br/dti/tcc/tcc0025.pdf>>. Acesso em: 14 out. 2025.
- ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. 6. ed. São Paulo: Pearson Addison-Wesley, 2011. Disponível em: <<https://repositorio.ufmg.br/bitstreams/418e2e09-be8a-49ee-a90b-b5e9d1ddda34/download>>. Acesso em: 14 out. 2025.
- GITHUB. **GitHub**. Disponível em: <<https://github.com>>. Acesso em: 28 de março de 2025.
- GOOGLE. **Introducing Google Gemini**. 2024. Disponível em: <<https://blog.google/technology/ai/introducing-google-gemini>>. Acesso em: 04 de abril de 2025>.
- GOOGLE CLOUD. **Google Speech Recognition**. Disponível em: <<https://cloud.google.com/speech-to-text>>. Acesso em: 04 de abril de 2025.
- INTERVIEW WARMUP. **Grow With Google**. Disponível em: <<https://grow.google/certificates/interview-warmup>>. Acesso em: 15 de março de 2025.
- KNN IDIOMAS. **Quantas pessoas falam inglês no Brasil?** Disponível em: <<https://www.knnidiomas.com.br/blog-do-aluno/quantas-pessoas-falam-ingles-no-brasil>>. Acesso em: 13 de março de 2025.
- MEDIUM. **Language learning apps: a new way to learn**. Medium, 2024. Disponível em: <<https://medium.com/@chaleemullah1992/language-learning-apps-a-new-way-to-learn-159a0fafc764>>. Acesso em: 15 de março de 2025.
- MOZILLA DEVELOPER NETWORK. **CSS - MDN Web Docs**. 2025a. Disponível em: <<https://developer.mozilla.org/en-US/docs/Web/CSS>>. Acesso em: 28 de março de 2025.

MOZILLA DEVELOPER NETWORK. **HTML - MDN Web Docs**. 2025b. Disponível em: <<https://developer.mozilla.org/en-US/docs/Web/HTML>>. Acesso em: 27 de março de 2025.

MOZILLA DEVELOPER NETWORK. **JavaScript - MDN Web Docs**. 2025c. Disponível em: <<https://developer.mozilla.org/en-US/docs/Web/JavaScript>>. Acesso em: 28 de março de 2025.

NEXT.JS. **What is Next.js?** Disponível em: <<https://nextjs.org/docs>>. Acesso em: 27 de março de 2025.

NODE.JS. **Introduction to Node.js**. Disponível em: <<https://nodejs.org/pt/learn/getting-started/introduction-to-nodejs>>. Acesso em: 28 de março de 2025.

ORGANIZAÇÃO PARA A COOPERAÇÃO E DESENVOLVIMENTO ECONÔMICO (OCDE). **Defining and classifying AI in the workplace**. OECD Publishing, 2023. Disponível em: <https://www.oecd.org/content/dam/oecd/en/publications/reports/2023/03/defining-and-classifying-ai-in-the-workplace_35d2d069/59e89d7f-en.pdf>. Acesso em: 30 set. 2025.

POSTGRESQL. **What is PostgreSQL?** Disponível em: <<https://www.postgresql.org/about>>. Acesso em: 28 de março de 2025.

PRIMER. **Primer**. Disponível em: <<https://primer.style>>. Acesso em: 28 de março de 2025.

REACT. **React**. Disponível em: <<https://reactjs.org>>. Acesso em: 27 de março de 2025.

RECHARTS. **Recharts**. Disponível em: <<https://recharts.github.io/>>. Acesso em: 16 de novembro de 2025.

SCHWABER, Ken; SUTHERLAND, Jeff. **O Guia do Scrum: O Guia Definitivo para o Scrum: As Regras do Jogo**. Scrum.org, nov. 2020. Disponível em: <<https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-PortugueseBR-3.0.pdf>>. Acesso em: 14 out. 2025.

SOMMERVILLE, Ian. **Engenharia de Software**. 9. ed. São Paulo: Pearson, 2011.

SUTHERLAND, Jeff. **Scrum: a arte de fazer o dobro do trabalho na metade do tempo**. Tradução de Natalie Gerhardt. São Paulo: LeYa, 2014.

VERCEL. **Vercel**. Disponível em: <<https://vercel.com>>. Acesso em: 16 de novembro de 2025.