

**FACULDADE E ESCOLA TÉCNICA ALCIDES MAYA**

**CURSO TÉCNICO EM INFORMÁTICA**

LEONARDO MEDEIROS LEIMAN

**Cinema App – Uma aplicação para o gerenciamento de filmes e ingressos de cinema.**

Projeto de Pesquisa apresentado como requisito parcial para obtenção do título de Técnico em Informática, pelo Curso de Técnico de Informática.

Orientador: Prof. Me. João Padilha Moreira[[1]](#footnote-2)

Porto Alegre

2024

**Cinema App – Uma aplicação para o gerenciamento de filmes e ingressos de cinema.**

**RESUMO**

Este projeto apresenta um sistema que busca facilitar o contato entre clientes e uma empresa que oferta serviços de cinema permitindo a compra de ingressos pela rede de Internet com a finalidade de reduzir filas e incômodos relacionados de forma fácil e intuitiva dentro de um ambiente configurável e controlado pelos funcionários da empresa.

O sistema foi construído com a livraria .NET e sua expansão ASP, o que permite uma fácil integração com bancos de dados já existentes no mercado e força o código a sempre estar receptivo para novas atualizações.

Em seguida será explicado em detalhes sobre como essas tecnologias foram aplicadas e como executar o aplicativo em um ambiente de servidor.

O projeto não foi avaliado e não foi executado em um servidor de “produção” e portanto não possui avaliações de usuários.

**Palavras chaves**: cinema, ASP.NET, .NET, C#, Web, Blazor, tecnologia, sistema, educação, WEB

**LISTA DE FIGURAS**

[**FIGURA 1** Tela home 21](#_heading=h.147n2zr)

[**FIGURA 2** Tela de login 23](#_heading=h.23ckvvd)

[**FIGURA 3** LocalStorage de usuário 23](#_heading=h.32hioqz)

[**FIGURA 4** LocalStorage de dificuldade 24](#_heading=h.41mghml)

[**FIGURA 5** LocalStorage de tipo 24](#_heading=h.2grqrue)

[**FIGURA 6** Componente de questionário 25](#_heading=h.3fwokq0)

**LISTA DE SIGLAS**

|  |  |
| --- | --- |
| ABNT | Associação Brasileira de Normas Técnicas |
| NBR | Normas Brasileiras de Regulação |
| SWBES | Semantic Web-Based Educational Systems |
| TIC | Tecnologias da Informação e Comunicação |

SUMÁRIO

[1](#_heading=h.4d34og8) INTRODUÇÃO 6

[1.1](#_heading=h.17dp8vu) Definições do Tema ou Problema 7

[1.2](#_heading=h.3rdcrjn) Delimitações do Tema 7

[1.3](#_heading=h.lnxbz9) Objetivos 8

[1.3.1](#_heading=h.35nkun2) Objetivo Geral 8

[1.3.2](#_heading=h.1ksv4uv) Objetivos Específicos 8

[1.4](#_heading=h.44sinio) Justificativa 9

[2](#_heading=h.2jxsxqh) REVISÃO BIBLIOGRÁFICA 10

[2.1](#_heading=h.1y810tw) Tecnologia e educação 10

[2.2](#_heading=h.4i7ojhp) Usabilidade de sistemas Web para a pratica de língua inglesa 11

[2.3](#_heading=h.2xcytpi) Tecnologias usadas na construção do sistema 12

[2.3.1](#_heading=h.1ci93xb) JavaScript 12

[2.3.2](#_heading=h.3whwml4) React JS 13

[2.3.3](#_heading=h.2bn6wsx) Node JS e Express 14

[2.3.4](#_heading=h.qsh70q) Bootstrap e JQuery 15

[2.3.5](#_heading=h.3as4poj) Banco de dados MongoDB 16

[3](#_heading=h.1pxezwc) METODOLOGIA 18

[4](#_heading=h.49x2ik5) DESCRIÇÃO DA SOLUÇÃO 20

[4.1](#_heading=h.2p2csry) Construção do sistema 21

[4.1.1](#_heading=h.3o7alnk) Entrar e cadastrar 22

[4.1.2](#_heading=h.1hmsyys) Configuração do questionário 24

[4.1.3](#_heading=h.vx1227) Questionário 25

[4.1.4](#_heading=h.1v1yuxt) A estrutura das questões 26

[4.1.5](#_heading=h.4f1mdlm) Construção da API 28

[4.2](#_heading=h.2u6wntf) Pontuação e recordes 29

[5](#_heading=h.19c6y18) VALIDAÇÃO 30

[5.1](#_heading=h.3tbugp1) Análise das tecnologias utilizadas 30

[5.2](#_heading=h.28h4qwu) Análise geral 31

[6](#_heading=h.nmf14n) CONCLUSÃO 32

[7](#_heading=h.37m2jsg) CRONOGRAMA 34

[REFERÊNCIAS BIBLIOGRÁFICA 35](#_heading=h.1mrcu09)

# INTRODUÇÃO

A expansão dos serviços de internet no mercado mundial mostrou para todos que para uma empresa se manter relevante ela precisa centralizar seus dados conforme ela cresce no mercado, não fazer impedirá o crescimento futuro da empresa por não possuir dados em comum de seus clientes para descobrir para onde expandir, além de perder oportunidades de atendimento para competidores que fornecem um serviço pela internet.

Um serviço web permite os clientes fazerem pedidos independente de um atendente humano, cortando a barreira de tempo que todo atendimento presencial possui além do requisito de estar presente para fazer a demanda e permitir padronizar a experiência de cada usuário a um preço extremamente baixo.

Para oferecer um serviço web é necessário registrar um domínio web e possuir maquinário para processar os pedidos dos clientes, normalmente estes são custos que apenas grandes empresas conseguiriam sustentar, porém com serviços de processamento em nuvem como AWS,Firebase,Azure entre outros é possível ter um serviço acessado globalmente pela fração do preço de um maquinário local.

A separação do processamento em um servidor dedicado também permite a criação de aplicativos que se interconectam e permitem ofertar o serviço para diferentes situações, como telões na frente do cinema ou a página web na internet.

## 1.1 – Delimitações do Tema

O cinemaProject é um aplicativo web desenvolvido para permitir a interação de clientes pela internet com uma empresa ofertora de serviços relacionados a sessões de cinemas. O aplicativo em si não se responsabiliza pela organização da empresa em si e apenas se conecta e expande um sistema já existente ao mesmo tempo que permite ter dados locais se mantendo como uma parte independente.

O aplicativo apenas se responsabiliza em mostrar sessões compráveis para um usuário que serão disponibilizadas pela interface gráfica administrativa por padrão ou por um terceiro aplicativo que se conecta diretamente com o banco de dados. As compras no aplicativo não são reais e apenas são marcadas como o relacionamento entre clientes e ingressos e por padrão o cliente apenas será redirecionado para a tela inicial com uma mensagem de sucesso.

## Objetivos

Mostrar as capacidades de aplicações web para prestação de serviços pela internet a fim de criar atendimentos flexíveis para os usuários.

### **Objetivo Geral**

O objetivo deste trabalho é mostrar a capacidade da linguagem C# com Blazor ASP.NET para a criação de páginas web ao mesmo tempo que se é verbalizado as maneiras como essas tecnologias facilitam o contato entre empresas e clientes com ênfase a um sistema relacionado a cinemas.

### **Objetivos Específicos**

* Identificar o uso de sistemas web para venda de ingressos;
* Verificar nosso sistema web e como ele funcionará;
* Estudar a união da esfera de comércio e serviços com tecnologia.
* Descrever a criação de um sistema para a venda de ingressos.

## Justificativa

A aplicação também permite a centralização dos múltiplos processos administrativos como uma base de dados unificada, expansão para aplicativos de telão e a possibilidade de criar salas VIP com descontos pessoais para cada usuário conforme utilizam o sistema.

A criação de aplicativos web tornam empresas competitivas a um nível local ou potencialmente global permite o contato entre a empresa e um usuário de forma flexível independente de horário por preços que são possivelmente ignoráveis considerando os benefícios da tecnologia.

## Tecnologia e educação

A tecnologia relacionada a computadores avançou de forma exponencial conforme os anos passaram, atigindo todas as empresas da área com um exemplo famoso sendo a evolução dos produtos da Apple: em 1990 o Apple 1, um dos primeiros computadores de mesa pessoal lançados, possuía cerca de 4 kilobytes de memória RAM e cerca de 250 Bytes de memória, em 20 anos a Apple lançou o seu primeiro MacBook em 2009 onde além de ser portátil a ponto de caber confortavelmente dentro de uma mochila também possuía 8 gigabytes de memória RAM e cerca de 256Gb de armazenamento em disco.

O avanço exponencial dos computadores abriu espaço para o processamento em massa de dados de usuários ou da lógica de negócio empresarial ao mesmo tempo que a internet permitia facilmente se conectar a qualquer servidor aberto á ela. A demanda por servidores mais próximos dos clientes para reduzir a latência da conexão e permitir processar mais dados a cada segundo resultou na explosão da indústria de hospedagem em nuvem com a Amazon Web Services assumindo o volante por ser uma das primeiras com um serviço modular e por ser parte da já estabelecida no mercado Amazom.

As páginas web se moldaram aos serviços em nuvem, permitindo que as páginas se separassem não apenas em blogs ou repositórios de informações mas também como uma plataforma de serviço graças ao fácil acesso à hospedagem e processamento de dados dos serviços de nuvem.

## Usabilidade de sistemas Web para satisfazer demandas do mercado

A usabilidade de um sistema web refere-se aqui à qualidade da interação entre o usuário com o sistema hospedado na internet muitas vezes dependendo de muitos aspectos como a latência da conexão, facilidade de entendimento do layout da página, eficacia para transmissão da mensagem principal e eficiência do produto web.

Um serviço web normalmente para ser considerado satisfatório no mercado precisa satisfazer uma demanda popular. O Youtube satisfaz a necessidade de uma plataforma para a transmissão de informações em formato de vídeo ao mesmo tempo que a Wikipédia satisfaz a necessidade de termos documentos atualizados sobre o conhecimento humano e de quais documentos as informações foram retiradas com ambas podendo também trabalhar em conjunto.

A internet possibilita maneiras criativas para se conquistar cada nicho do mercado, mesmo se o CinemaApp tornar-se um aplicativo popular ainda assim poderia satisfazer uma porção maior de forma mais barata hospedando uma cópia digital no banco de dados e transmitindo para os usuários assistirem em seus computadores pessoais, a maneira como atender estas demandas é o que verdadeiramente torna e mantêm um website popular.

## Tecnologias usadas na construção do sistema

De forma geral, o sistema é inteiramente construído em C# com o framework .NET com sua expansão ASP. A linguagem web Javascript sendo substituída em grande parte pela camada de abstração Blazor que permite escrever código C# que é compilado para WebAssembly e executado pelo navegador do usuário. Para o backend por padrão é utilizado o banco de dados SQLite para armazenar os dados localmente.

Para a estilação da página será utilizada a livraria Bootstrap junto com Popper e Jquery para lidar com as interações básicas na página.

### **C#**

C# é uma linguagem de programação de alto nivel, multiparadigma, orientada a objetos com tipagem forte e desenvolvida pela Microsoft para ser utilizada em conjunto com o framework .NET.

A linguagem surgiu em Abril de 2000 junto com a primeira versão do framework .NET, inicialmente sendo mantido e produzido apenas pela Microsoft porém sendo transformada em um projeto de código aberto a fim de reduzir custos e aumentar o contato com a comunidade.

O pacote .NET é um framework que busca conectar múltiplas livrarias em um único pacote de fácil uso do desenvolvedor, inicialmente integrando apenas as APIs do sistema Windows para aplicativos desktop, porém expandindo para aplicações web com a livraria ASP e eventualmente para múltiplos sistemas operacionais com projetos como Mono runtime.

A C# é uma linguagem de multiúso com uma linha de aprendizado relativamente alta, sendo utilizada para criação desde aplicativos padrões para Desktop como uma calculadora ou editor de texto se conectando automaticamente com as livrarias para renderização de janelas no sistema operacional, Videogames utilizando plataformas como Unity que se conecta com livrarias de vídeo e a criação de executáveis para múltiplos sistemas operacionais, livrarias para auxiliar com a criação de modelos de inteligência artificial e para a criação de páginas web utilizando ASP.

Junto ao C# veio a linguagem F# em 2005 com seu maior diferencial sendo seu foco em programação funcional ao invés de ser dedicada a orientação objeto.

### **Asp.Net**

Asp se refere á tecnologia chamada Active Server Pages lançada pela microsoft em 1990 como a maneira oficial oferecida para criar páginas dinâmicas utilizando a linguagem VB script, atualmente substituída pela C# e F#.

O ASP.NET nasceu em 2002 com integração para C#, F# e visual Basic porem lentamente dando preferencia para C#/F# com atualizações futuras.

O ASP lida com diversos problemas relacionados ao desenvolvimento do lado servidor tipicamente utilizando o modelo MVC para controlar como os usuários acessam os dados do servidor, Muitas funcionalidades básicas já vem por padrão com o ASP, porém existem livrarias que expandem ainda mais como o entity framework para a criação de bancos de dados com código C# ou o Identity framework que tras consigo um sistema completo de login compatível com padrões modernos como e-mail de confirmação, autenticação de dois fatores e uma página pessoal de usuário.

### **Blazor**

Blazor é um framework frontend para páginas web que permite substituir a linguagem Javascript pela C#. O diferencial do Blazor é que a tecnologia pode se integrar com as páginas Razor de um servidor criando API automáticas permitindo ter páginas de cliente similar a uma aplicação de página única com componentes renderizados pelo servidor e automaticamente enviados ao cliente.

Normalmente servidores web são divididos entre duas versões: um servidor API que conversa diretamente com uma aplicação Frontend separando a lógica empresarial da lógica visual de apresentação dos dados ou um servidor com páginas web renderizadas pelo servidor.

Razor é uma ferramenta similar ao Blazor, permitindo criar páginas HTML utilizando a linguagem C#, porem no lado do servidor.

### **Bootstrap, Jquery e Popper**

Bootstrap é uma livraria de estilos css originalmente criada e desenvolvida no Twitter com o nome de Twitter Blueprint, A livraria foi aberta ao publico e tornada open-source em 2011 e vem sendo atualizada desde então buscando seguir padrões modernos da web como flex-boxes e Sass.

O objetivo do Bootstrap é acelerar a criação de páginas dando um estilo padrão para os elementos e componentes prontos para serem utilizados, permitindo os desenvolvedores focarem apenas na lógica da aplicação sem se preocupar em criar um design responsivo para a página. A maneira que isto é feito é utilizando comandos javascript e css já inclusos na livraria.

A partir da versão 5 do bootstrap o pacote Jquery não é mais utilizado com um aplicativo Blazor utilizando por padrão a versão 5.10 (em 2024).

O JQuery é uma biblioteca de funções Javascript que interagem com o HTML, criada em 2006 focando em padronizar o antigo javascript com comandos customizados para modificar o DOM, criada em um periodo onde nem todos os navegadores suportavam todas as funcionalidades da javascript ou do css.

No desenvolvimento de páginas web é importante utilizar estas livrarias para padronizar o código a fim de consistência de interface e manter suporte com navegadores diferentes com funcionalidades diferentes.

### **Banco de dados SQLite**

MongoDB é um software livre para banco de dados não relacional orientado a documentos. Foi criado por Eliot Horowitz, Kevin P. Ryan, Dwight Merriman e lançado oficialmente em 2009 (SILVA, 2014)

O MongoDB é um banco de dados orientado a documentos no formato JSON. Diferente de um banco de dados relacional, ele não possui restrições de estrutura, isto é, não é necessário a criação de tabelas e colunas criadas previamente, permitindo que um documento represente toda a informação necessária. Não é necessário um padrão de informações, pode-se salvar informações distintas para cada documento salvo, isto é, você pode salvar a conta de um usuário no MongoDB contendo as informações nome, e-mail e cep, porém caso você queira salvar outro usuário sem a informação “e-mail”, contendo somente nome e cep isto é possível, pois diferente de um banco de dados relacional, o MongoDB não possui uma estrutura padrão para salvar os dados.

Os documentos são agrupados em collections. E um conjunto de collections forma um database (banco de dados). O MongoDB permite que seu database seja replicado para outros servidores, aumentando assim a disponibilidade de suas informações, sendo esse recurso conhecido por replica set. Dessa forma, cada servidor terá uma cópia dos dados.

O MongoDB é rápido, ele possui uma excelente performance, além de ser um banco com alta disponibilidade, e muita flexibilidade, com suporte a um alto volume de dados, e ainda conta com um forte suporte de queries para consultas, dessa forma ele traz uma série de possibilidades para sua utilização, principalmente para atender sistemas com dados dinâmicos e metamodelos, como um sistema para gerenciar catálogos de produtos, os chamados PIM (*product information management*).

# METODOLOGIA

Para a construção do sistema proposto neste trabalho, o MyEnglish, primeiramente foi feito a seleção das bibliotecas e tecnologias, havendo alteração nas mesmas, isto é, conforme o sistema foi sendo desenvolvido foram surgindo novas necessidades de bibliotecas que foram implementadas com o passar do tempo.

Neste sentido, as principais pesquisas utilizadas na programação do sistema estão relacionadas a bibliotecas. A programação do sistema teve muito auxilio de bibliotecas prontas, que estão disponíveis na Internet. Porém, para selecionar estas bibliotecas foram feitos vários testes de compatibilidade com o React JS, para evitar bugs e erros desnecessários. Outras pesquisas foram feitas para auxiliar na resolução de erros que o sistema gerou, fazendo pesquisas principalmente em fóruns de programação na Internet como Stack Overflow e GUJ.

Em relação as questões e métodos de ensino e prática, foram feitas pesquisas que estava fora do escopo da programação, e atingindo mais a área de educação linguística. Foi pesquisado sobre os métodos de fixação de conteúdo e como se poderia mesclar essas técnicas em um sistema. Para isso, já existem atualmente algumas plataformas de ensino de línguas, que utilizam estas técnicas, o que facilitou a pesquisa, pois assim, foi encontrado mais facilmente materiais acadêmicos de ensino de língua com tecnologias. Porém, o MyEnglish teve um diferencial importante, excluindo a parte teórica do ensino e substituindo-a pela prática, tendo assim um sistema que se utiliza de técnicas de fixação de conteúdo focadas na prática de conhecimento teóricos de língua inglesa.

Após a seleção das tecnologias inicias, foi feita uma pesquisa, de como seria a tela principal do sistema, o foco era algo minimalista, porém prático, fácil e o mais atrativo possível. Em suma, nos primeiros momentos da construção do sistema não houve programação lógica e sim a criação do design das telas. As principais bibliotecas que auxiliaram neste processo foram Bootstrap e Styled-componentes.

A tela Home foi o primeiro componente criado juntamente com seus componentes filhos (cabeçalho, conteúdo e rodapé), esta tela também seguiu o foco de minimalismo e não contém um conteúdo muito extenso. O cabeçalho possui apenas o nome do sistema com o ícone e um botão para acessar as ações da conta que está logada. Já o rodapé conta algumas informações básicas, como ano de criação do site e o nome dos desenvolvedores.

Os demais componentes foram criados na sequência, no caso, os componentes de login e register (entrar e cadastro), contendo em seu corpo apenas um formulário para efetuar a criação ou login da conta. Os componentes de configuração do questionário também seguiram a ideia de design minimalista, apenas contendo as opções de configurações em formato de imagens, o que foi feito utilizando o sistema de GRID e FLEXBOX do CSS3.

Em seguida, o design do principal componente da aplicação foi criado, o componente do questionário, contendo apenas ícones de ação do lado esquerdo (sair, mudar tema e report) e a pontuação do lado direito, além de um espaço para os botões de resposta.

Após a criação do design das telas, o próximo objetivo foi deixar o sistema dinâmico, saindo da parte de estilo e entrando na parte de programação lógica, fazendo o sistema funcionar como deveria, programando cada função e efeito de cada componente, corrigindo bugs e fazendo pesquisas de bibliotecas, de resolução de erros, etc.

# DESCRIÇÃO DA SOLUÇÃO

O MyEnglish surgiu inicialmente com uma ideia de criação de um sistema que auxiliava viajantes a aprender inglês, porém com o passar do tempo, esta ideia foi adaptada, tendo então como objetivo não somente auxiliar viajantes, mas sim qualquer pessoa brasileira que estivesse disposta a praticar seu conhecimento na língua inglesa, focando apenas na prática e não em conteúdos teóricos. Como a ideia do sistema foi alterada, e mudou de um público alvo específico (viajantes) para um mais abrangente (qualquer pessoa que queira praticar seu inglês), o sistema sofreu mudanças de design, contendo um estilo mais amigável para os usuários (user friendly), sobre técnicas de design amigáveis Mark afirma que hoje em dia existem várias informações sobre técnicas de design de interface e padrões que você pode usar quando elabora suas interfaces de usuário, soluções para problemas comuns e recomendações gerais de usabilidade (NUNES, 2017), além disso o sistema contou com alterações em sua programação, tendo escolha de níveis de conhecimento.

O sistema também sofreu mudanças nas escolhas de tecnologias. Inicialmente o framework principal do frontend, ou seja, o framework que seria responsável pela construção das telas foi mudado de Vue JS para React JS. O Vue JS também trabalha com uma tecnologia moderna, o sistema de componentes, porém, foi trocado pelo React JS, pois o Vue não possui boas bibliotecas auxiliares que contribuíssem para a construção do sistema, como bibliotecas de toast (biblioteca de avisos na tela), reconhecimento de voz e reprodução de áudios, enquanto o React JS possui inúmeras bibliotecas eficazes como estas disponíveis para download na Internet, o que facilitou e muito o desenvolvimento do sistema.

Já no backend, temos a construção de uma API feita com Node JS, pois esta tecnologia é uma das mais utilizadas para a construção de API em sistemas web, sendo mais eficaz e compatível em comunicar-se com o React JS. Para a construção da API foram utilizadas outras bibliotecas adicionais que ajudaram e facilitaram o desenvolvimento da mesma, como por exemplo: Express, body-parser, consign, nodemon e mongodb (biblioteca responsável por comunicar o Node JS com o banco de dados MongoDB).

## Construção do sistema

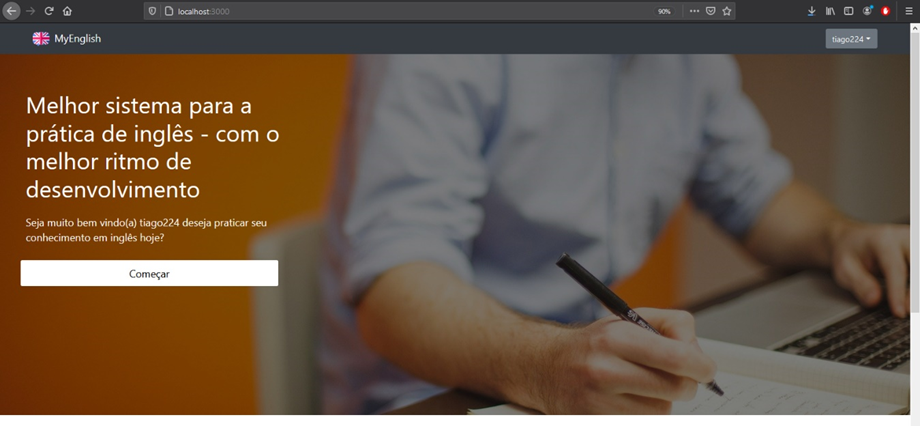
O MyEnglish inicializou-se com a construção da tela Home, ou seja, a tela inicial do sistema. A tela Home é um componente que possui dentro de si mais 3 componentes, são eles, cabeçalho, corpo (conteúdo), e rodapé. Todos estes componentes foram programados utilizando a biblioteca React-hooks, uma forma

moderna de criar telas.

Para a Home foi atribuída a rota padrão, ou seja a rota “/”, isto significa que caso o usuário esteja acessando a URL [https://MyEnglish.com](https://myenglish.com/) ou [https://MyEnglish.com/](https://myenglish.com/) aparecerá a tela Home do sistema. Parar fazer isso foi utilizado a biblioteca react-router-dom, que é uma biblioteca responsável por gerenciar todas as rotas do sistema.

A principal função da tela Home, Figura 1, é exibir outras telas, como a tela de login, cadastro, nível etc. Caso o usuário não tenha uma conta no sistema será exibido uma tela contendo os botões de login e cadastro, pois só será permitido o acesso ao sistema caso o usuário tenha uma conta.

**FIGURA 1** - Tela home



**Fonte:** Captura de tela feita pelo autor.

Conforme as telas forem exibidas a rota vai mudando, ou seja se a tela de login estiver aberta, a URL ficará [https://MyEnglish.com/login](https://myenglish.com/login) por exemplo.

Algumas telas do sistema, incluindo a Home e as que aparecem na home possuem tema claro e escuro, que pode ser configurado no cabeçalho, somente o cabeçalho e rodapé que não possuem cores fixa, esta funcionalidade foi adicionada utilizando sistemas da biblioteca styled-componentes, biblioteca responsável por incluir estilos componentes com estilos “dinâmicos” através do JavaScript, ou seja é possível estilizar componentes e elementos através de JavaScript, porém o CSS não foi descartado, muito de suas funcionalidades foram utilizadas na construção deste sistema, cada componente possui seu arquivo de estilo CSS3 e alguns possuem também o arquivo JS de estilo utilizando styled-components para estilos dinâmicos. Com isso, o sistema armazena no localStorage (local de armazenamento do navegador) do usuário qual o tema selecionado pelo mesmo.

Além disso, a home conta com outro componente importante, o cabeçalho que armazena o nome do sistema com um ícone no lado esquerdo, e do lado direito contém um dropdown (caixa de seleção) que terá o nome de usuário e que, ao clicar, exibe algumas opções como, sair, mudar tema e exibir tela de perfil do usuário. O cabeçalho possui um design minimalista que foi feito com base em um componente de menus do Bootstrap.

O rodapé também possui um design minimalista, apenas apresentando informações básicas em formato de texto.

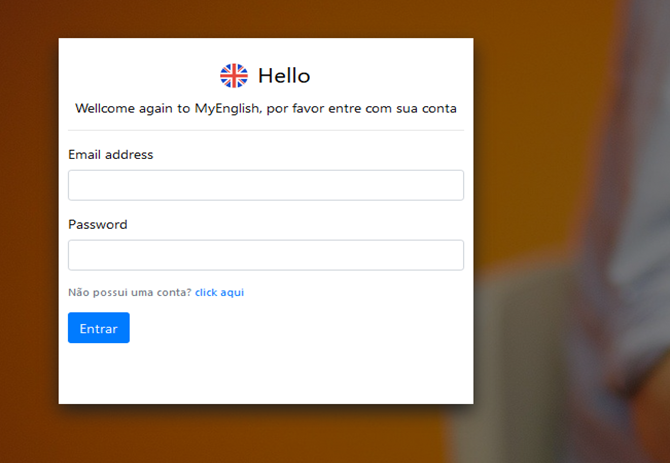
### **Entrar e cadastrar**

Caso o usuário acesse o sistema e o mesmo não tenha uma conta logada, será exibido uma mensagem para o mesmo efetuar o login com botões de login ou cadastro, caso clique em algum desses botões a tela correspondente aparecerá.

Ao efetuar login o usuário será direcionado para a tela home novamente, dessa vez com uma mensagem de boas-vindas com seu nome e com o botão “começar” habilitado.

Para entrar com sua conta na tela de Login, conforme ilustra na Figura 2, o sistema coleta as informações do campo de texto e-mail e senha e lança estas informações em uma função que é responsável por se comunicar com as rotas da API, sendo assim esta função transmite as informações para dentro da API e a mesma retornou para a função se existia um usuário com este e-mail e senha no banco dados, caso exista, o sistema recupera algumas informações do usuário e armazena no localStorage do mesmo, é assim que o sistema reconhece se há um usuário logado, isto é, se possuir informações de login no localStorage, significa que há usuário logado, caso contrário, não. Como pode ser visto na FIGURA 2.

**FIGURA 2** Tela de login

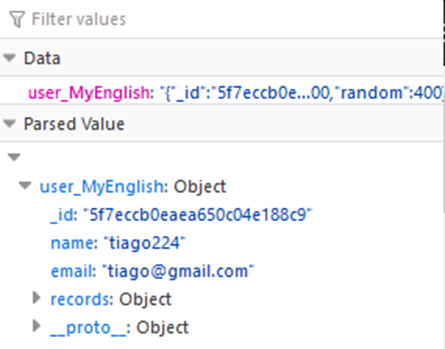


**Fonte:** Captura de tela feita pelo autor

O processo de comunicação do frontend com a API (backend) foi feito com auxílio da biblioteca Axios, que fez com que informações fossem passadas para as funções da API responsável pela rota “/login”.

Para efetuar o cadastro de uma conta é necessário ter um e-mail que já não foi utilizado no sistema, e não pode ter uma senha muito fraca com menos de 6 caracteres, a senha deve conter letras e números, após efetuar o cadastro deverá fazer em seguida o login (entrar).

**FIGURA 3** LocalStorage de usuário



**Fonte**: Captura de tela efetuada pelo autor.

O processo de cadastro é semelhante ao de entrar, é feito uma requisição na API com o uso do Axios, porém desta vez a rota da API será “/register”, esta rota terá uma função que fará uma validação, caso as informações de e-mail não estejam presentes no banco dados, as informações serão salvas, ou seja, a conta será criada, caso contrário, já existe uma conta com este e-mail e as informações não serão salvas, retornando um aviso.

### **Configuração do questionário**

As telas de configurações são as telas onde o usuário escolherá qual o nível do questionário, podendo escolher entre básico, intermediário, avançado e aleatório, porém caso for a primeira vez do usuário no sistema, a opção de “aleatório” será substituída por “não sei”, caso o usuário não saiba qual nível escolher.

Após escolher o nível do questionário o usuário escolherá qual o tipo de questões que aparecerão, poderá escolher entre, diálogos, tradução, ouvir e traduzir, imagens e áudios, fala e pronuncia ou aleatório (todos), após isso o usuário será direcionado para a tela do questionário.

A programação destes componentes de configuração é bem simples, apenas armazenando as escolhas do usuário no localStorage, ou seja, conforme a Figura 4 e a Figura 5, no localStorage será armazenado o nível de dificuldade e o tipo das questões conforme.

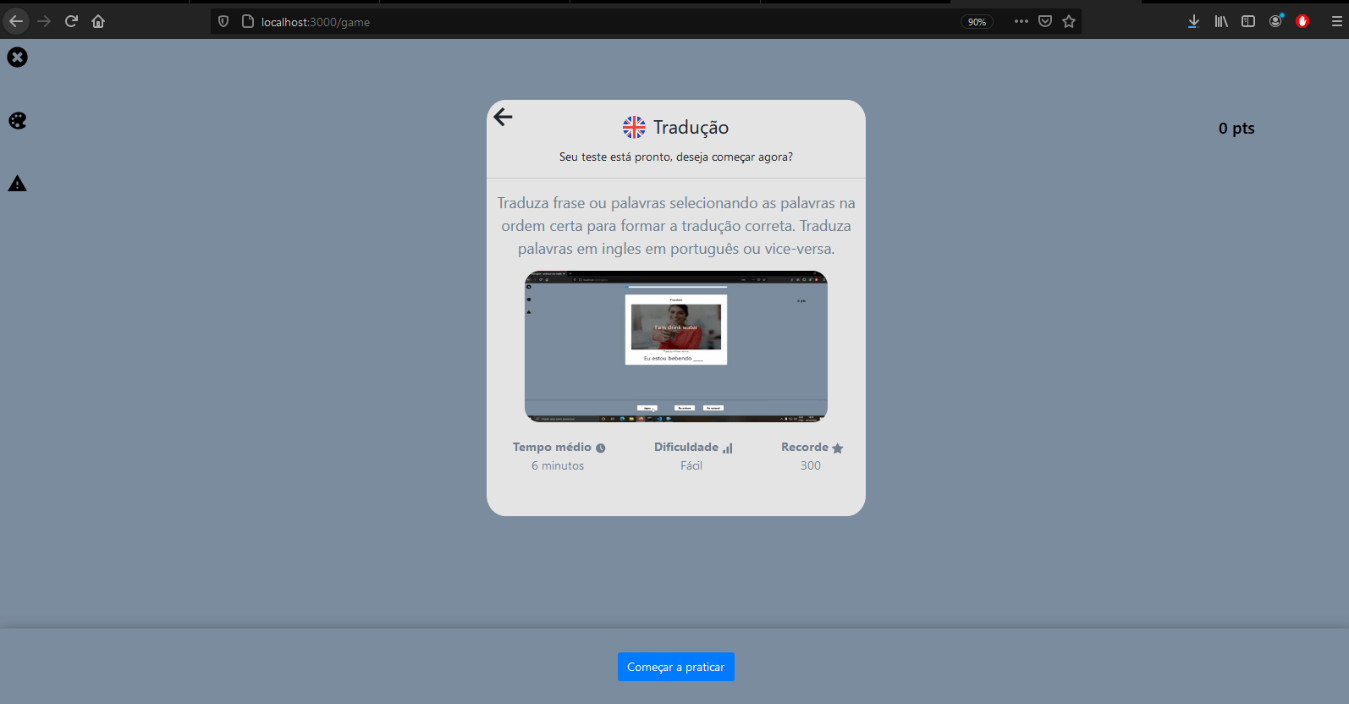
|  |  |
| --- | --- |
| **FIGURA 4** LocalStorage de dificuldade    **Fonte**: Captura de tela efetuada pelo autor. | **FIGURA 5** LocalStorage de tipo    **Fonte**: Captura de tela efetuada pelo autor. |

Estas informações são armazenadas no localStorage para que a tela de questionário as consuma e saiba qual arquivo de questões utilizar, ou seja, se o usuário escolher o nível fácil e questões do tipo diálogo, a tela de questionário entenderá que deve usar o arquivo que contém perguntas do tipo diálogo de nível fácil. Para as telas de configuração do questionário foi atribuído as rotas “/levels” e “/types”.

### **Questionário**

A tela de questionário, Figura 6, inicialmente é responsável por apresentar um breve resumo de como funcionará as questões escolhidas pelo usuário, além de um gif demonstrativo, isto é, se o usuário escolheu questões de tipo diálogo, terá um resumo explicando como funcionam este tipo de questão juntamente com um gif para auxiliar na compreensão do funcionamento das questões, além de outras informações como, dificuldade, recorde, tempo médio de conclusão do questionário, isso é feito para que o componente seja carregado utilizando uma função especial do react-hooks que dispara assim que o componente for carregado.

**FIGURA 6** Componente de questionário



**Fonte**: Captura de tela efetuada pelo autor.

Do lado esquerdo da tela existem três ícones, o primeiro (de cima para baixo) fecha o questionário, porém apresenta primeiro uma tela de confirmação, informando o usuário, que caso ele saia seu progresso será perdido. O próximo ícone muda o tema do sistema com o click do mouse, o ultimo exibe a tela de report de bugs e erros, onde o usuário escreverá resumidamente sobre o erro ou bug que experimente.

Quando o usuário apertar no botão “começar a praticar”, uma barra de progresso aparecerá~~,~~ e o componente será encarregado de exibir as questões uma de cada vez em ordem aleatória, aplicando a estrutura de cada questão. Isto é, se a questão for do tipo diálogo, logo a estrutura que o componente aplicará nesta questão será a estrutura de diálogo~~,~~ e assim por diante. Quando não houver mais questões para mostrar, ou seja, quando o usuário já tiver respondido todas as questões, será exibido um componente de resultados, mostrando a pontuação obtida, o número de acertos, o número de erros e demais informações do exercício.

### **A estrutura das questões**

Cada tipo de questão possui uma estrutura única, pois cada tipo de questão tem funcionalidades diferentes, a seguir veremos os tipos de questões e como funcionam.

**Diálogos:** Este tipo de questão simula diálogos entre duas pessoas, contendo lacunas entre as frases, onde o usuário deverá preencher com as palavras corretas no contexto correto.

Utilizando o JQuery, assim que o componente desta questão é carregado, todas as frases do diálogo são exibidas na tela. O componente, quando carregado, recebe as informações de quais serão as respostas corretas na ordem correta, então, quando o usuário seleciona uma palavra, é feita uma verificação. Se a palavra selecionada for igual a palavra correta, significa que o usuário acertou, então, em seguida a palavra é adicionada na lacuna, quando todas as lacunas forem preenchidas corretamente a questão é acertada. Porém, caso o usuário erre alguma palavra a questão é perdida imediatamente.

**Tradução**: Este tipo de questão tem como objetivo apresentar uma palavra ou frase para o usuário traduzir, seja do inglês para o português ou vice-versa. A tradução é feita selecionando as palavras na ordem correta.

Quando o componente é carregado, ele faz uma contagem de palavras, para saber quantas palavras é necessário que o usuário escolha, ou seja, caso a frase para traduzir seja “hello world”, o sistema entenderá que o usuário deverá selecionar apenas duas palavras na hora de responder (Olá mundo), assim o componente se aplicará de forma correta para qualquer frase ou palavra, grande ou pequena.

Após isso, o componente armazena a resposta e exibe a frase que deverá ser traduzida na tela e, quando respondida, faz a verificação com a resposta armazenada. Ou seja, se a frase que o usuário formulou selecionando as palavras for igual a frase de resposta da questão, significa que ele acertou, caso contrário, errou.

**Ouvir e traduzir**: Este tipo de questão possui o objetivo de treinar a capacidade de audição do usuário, exibindo um áudio juntamente com uma frase na qual haverá, lacunas a serem preenchidas. Com o auxílio do áudio e com o contexto da frase, o usuário deverá preencher as lacunas corretamente.

O componente, primeiramente exibe o áudio na tela, então, é armazenado na questão o caminho relativo do áudio (o sistema entenderá onde o áudio está armazenado), assim o sistema sempre colocará o áudio correto para o usuário ouvir, após isso é exibido a frase juntamente com as lacunas. Quando a frase é respondida pelo usuário é feita a verificação com a resposta, se essa verificação de igualdade for verdadeira, o usuário acertou, caso contrário, errou

**Imagens e áudios:** Este tipo de questão é um pouco diferente, tem como foco treinar a audição e tradução do usuário. É exibido para o usuário uma palavra em inglês, e 4 imagens. O usuário deve então selecionar a imagem que corresponde a palavra, isto é, se a palavra for “black hole” (buraco negro), significa que o usuário deverá clicar na imagem que possui um buraco negro. Quando clicar na imagem correta, a palavra será dita por uma voz automática, ao carregar a tela o componente recebe a informação de qual imagem é a correta, em seguida ele exibe a palavra na tela caso o usuário clique na imagem correta a palavra será repetida por uma voz do sistema, caso contrário, a palavra não será mencionada e a questão será perdida.

**Fala e pronúncia:** Este tipo de questão visa praticar a fala do usuário, contendo palavras, frases ou áudios para o usuário reproduzir e repetir. Este tipo de questão só funciona em navegadores compatíveis com a biblioteca de reconhecimento de voz SpeechRecognition da Mozilla, uma das bibliotecas de reconhecimento de voz mais populares da web. Os navegadores compatíveis são Chrome, Edge, Samsung Internet e Android webView. Caso o usuário não esteja usando um navegador compatível as questões de fala serão substituídas por outras questões compatíveis.

A implementação deste tipo de questão foi bem simples, somente com o uso das funções da biblioteca, códigos que executam determinadas tarefas, foi implementado o reconhecimento de voz. Por padrão, a biblioteca vem configurada com a língua inglesa, isto é, quando o usuário falar no microfone a biblioteca estará esperando palavras e frases em inglês, o que facilitará na compreensão do que é dito pelo usuário. Após a pronúncia a fala é transformada em texto, caso a frase formado, for igual a frase de resposta, o usuário acertou a questão, caso contrário, errou.

Caso o sistema não entenda o que foi dito, ou seja, não foi capaz de transformar em palavras em inglês o que foi falado no microfone, o botão mudará de “estou te ouvindo” para “clique para falar”, caso isso aconteça o usuário deverá clicar novamente no botão para que o sistema comece a ouvi-lo e repetir novamente a frase ou palavra.

### **Construção da API**

A API foi feita em um servidor local (localhost) e foi construída seguindo o padrão de projetos MVC, Nas palavras de Metsker (2004) "Um padrão é uma maneira de fazer algo, ou de buscar um objetivo. Tal ideia se aplica a cozinhar fazer fogos de artifício, desenvolver softwares e qualquer outro ofício.".

A API conta com a rota de “/login”, rota responsável por retornar se o e-mail do usuário e a senha são válidos no banco de dados. Caso seja válido, a função retorna as informações da conta do usuário em formato JSON para o frontend (React com axios) e em seguida as informações são armazenadas no localStorage do usuário, como já foi visto anteriormente.

Já a rota “/register”, faz uma procura no banco de dados em busca do e-mail fornecido pelo usuário. Caso este e-mail exista no banco de dados não será possível criar a conta, caso não exista, outra função será disparada, cuja tarefa será responsável por adicionar as informações de e-mail, senha e nome no banco de dados, porém, no caso da senha, por questão de segurança, ela será criptografada antes de ser armazenada, com o uso das funções da biblioteca cripto, com uma criptografia de 32 caracteres.

A rota “/update” é responsável por atualizar principalmente as informações de recorde do usuário, isto é, sempre que o usuário ultrapassar o seu recorde em determinado questionário, será armazenado este novo valor e em seguida, será retornado para o frontend as novas informações atualizadas da conta do usuário, para que o localStorage também seja atualizado com os novos recordes.

Vale mencionar, que as rotas da API são diferentes das rotas do frontend, por mais que a API e o React JS tenham rotas iguais (“/login” e “/register”) elas cumprem funções diferentes, pois estão em URL diferentes uma administrada pelo react-router-dom do React JS e a outra pela API.

## Pontuação e recordes

O sistema de pontuação do MyEnglish é bem simples, cada questão fácil (independentemente do tipo da questão) vale 100 pontos, questões médias valem 200, e questões difíceis valem 300 pontos. Caso erre a questão, o usuário não ganha os pontos.

Este sistema de pontuação é necessário principalmente quando o usuário não sabe qual seu nível de conhecimento e deseja fazer o teste de nivelamento, para que o sistema lhe recomende um nível compatível com o seu nível de conhecimento é necessária a pontuação, para que o sistema possa saber qual foi sua taxa de acertos e erros e, com base nisso, recomendar de forma mais precisa possível um nível adequado para o usuário.

Os recordes são definidos por tipo de questão e por dificuldade, ou seja, um usuário pode ter um recorde em questões do tipo tradução de nível fácil, e também um recorde diferente em questões do mesmo tipo, porém de nível médio. Ou seja, os recordes variam de acordo com o tipo e nível da questão.

Caso um novo recorde seja obtido será exibida uma notificação do lado esquerdo da tela. Em seguida, é feita uma requisição na API na rota “/update” para efetuar a atualização do recorde no banco de dados.

# VALIDAÇÃO

A análise deste projeto foi feita do ponto de vista de desenvolvimento, isto é, foram feitas avaliações por parte dos desenvolvedores em relação a criação do sistema e objetivos propostos.

Todas as funcionalidades propostas foram implementadas no sistema, com o uso e auxílio das as tecnologias mencionadas nos capítulos anteriores. As tecnologias usadas tiveram um bom desempenho para a construção do MyEnglish, cumprindo corretamente com as funções pretendidas e gerando poucos bugs e erros para serem concertados ao longo do desenvolvimento. O que proporcionou maior fluidez no processo de construção do sistema.

## Análise das tecnologias utilizadas

Em questão de tecnologias, destaca-se o uso do React JS, com ele a aplicação teve uma grande variedade, de bibliotecas para auxiliar no desenvolvimento, além de ser o framework responsável por toda a criação do frontend, cumprindo com as principais tarefas de exibição de telas (componentes), se comunicar e consumir dados da API.

No backend, foi analisada a eficácia do Node JS com Express para a construção da API, e assim, geraram ótimos resultados tanto na parte de construção quanto na parte de produção (termo que se refere quando o sistema está no “ar”, ou seja, produzindo aquilo que foi programado). A maneira e a velocidade com que o Node JS trata as requisições foi algo extremamente positivo para a aplicação, que se mostrou capaz de lidar de forma excelente com atualizações, buscas e remoções no banco de dados (MongoDB) e de forma adequada enviar dados e informações para o frontend.

As tecnologias auxiliares, ou seja, as tecnologias como Bootstrap e JQuery por exemplo, foram usadas apenas para tarefas menores e responsáveis por alguns aspectos de responsividade (Bootsrap) e dinâmica (JQuery) do sistema. Ainda assim a escolha de tecnologias para tarefas menores, é algo importante em qualquer projeto de programação, pois concedem agilidade ao desenvolvimento e economizando esforços de tempo. Um grande exemplo disso são funcionalidades de grid e flexbox do Bootstrap, que é implementado no atributo “classe” dos elementos, fazendo com que não seja necessário criar a responsividade de certas telas com códigos extensos e complexos em CSS3, que hoje é o principal mecanismo de estilo para sistemas web.

## Análise geral

Foram observados inúmeros pontos positivos no sistema construído, que foi capaz de alcançar todos os objetivos propostos de desenvolvimento e de forma que todas as funcionalidades se encaixaram perfeitamente com a proposta de um sistema para a prática de inglês. Além de ter superado as nossas expectativas, foi analisado que o sistema foi finalizado com grandes possibilidades de mudanças e atualizações futuras, aprimorando-o cada vez mais.

No que toca as atualizações do sistema, consideramos também a forma com que o sistema foi construído, gerando uma maior facilidade na busca e correção de erros e bugs. Esta facilidade está ligada com as funcionalidades de componentização de React JS, discutidas no subcapítulo 2.3.2, isto significa que mesmo concluído, o sistema tem uma boa eficiência quando se trata de reparos, fazendo com que possam ser realizados com maior rapidez caso surjam bugs e erros.

Além de quesitos técnicos de programação, fez-se uma análise com foco nas questões, com uma visão mais “educativa”, isto é, foste verificado a eficácia das questões quando se trata de praticar o inglês. Esta análise foi limitada pela falta da presença de usuários dispostos a testar o sistema e a sua eficiência na prática do idioma estrangeiro. Porém com base nas revisões de metodologias de ensino e prática de língua verificou-se que cada questão atende os aspectos principais para auxiliar na fixação de conteúdo seguindo a metodologia Sociointeracionista, de forma possível concluir que dentro da proposta do trabalho, todas possuem uma boa performance e cumprem com seus objetivos de prática e fixação de conhecimento teórico.

# CONCLUSÃO

O MyEnglish atingiu todos os objetivos propostos no projeto em questão de programação e desenvolvimento. Ao longo do projeto foi estudado e pesquisado como ocorre a mesclagem de duas áreas, neste caso, a educação e a tecnologia. O que se fez a partir de estudos focados em prática de conhecimento, que foram aplicados no sistema a partir da construção de questões, cumprindo com um dos principais objetivo deste projeto, que era estudar a união da educação com a tecnologia.

Foi concluído que as tecnologias usadas apresentaram um comportamento adequado, cada uma fazendo seu papel dentro do sistema. Tanto o design quanto a programação superaram expectativas em relação às propostas iniciais, contando com, uma programação repleta de bibliotecas auxiliares, com inúmeras funcionalidades implementadas, tanto complexas quanto simples, em especial no que toca o design, desde o início foi proposto um design minimalista, com poucos detalhes, porém atrativo, além de algumas funcionalidades de design como modo claro e escuro. Características que foram implementadas no sistema com o uso de bibliotecas auxiliares disponibilizadas por comunidades de programação na Internet.

O uso de bibliotecas auxiliares, principalmente as utilizadas com React JS, foram de suma importância no projeto, o uso de bibliotecas foi muito grande, algo inesperado para os desenvolvedores, porém positivo. O uso destas bibliotecas fez com que inúmeras funcionalidades fossem implementadas de maneira rápida e fácil desde tarefas simples até tarefas mais complexas, o que facilitou o desenvolvimento da aplicação.

O sistema é completamente dinâmico, responsivo e assíncrono, isto é, as funcionalidades de comunicação com a API não necessitam de atualizar a página. Todo o processo de desenvolvimento foi pensando na experiência do usuário no sistema MyEnglish. O usuário poderá ter qualquer nível de conhecimento em inglês, até mesmo aqueles que não sabem qual é seu nível, isto cumpre um dos objetivos iniciais do sistema de atender o maior número possível de pessoas dispostas a aprimorar e praticar seu inglês, aumentando suas habilidades nesta língua estrangeira gerando uma espécie de, “efeito colateral” positivo que poderá fazer com que os usuários possam ter melhores chances no mercado de trabalho ou em experiências no exterior.

Para implementações futuras do projeto, a principal ideia é uma testagem mais ampla do sistema, disponibilizando um formulário com fins de reconhecer e analisar as opiniões e experiências dos usuários com o sistema a fim de que possa ser aprimorado futuramente. Este formulário será simples e objetivo, com perguntas descritivas para que o usuário possa relatar sua experiência de forma resumida com o objetivo de verificar a qualidade e efetividade das questões proporcionadas pelo sistema em relação a prática de conhecimento em inglês do usuário. Com o resultado adquirido nestas testagens irão ser feitas as melhorias sugeridas, sempre com foco de agregar mais para o sistema.

# CRONOGRAMA

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Atividades** | **Projeto Final** | | | | | | | |
| **Projeto 1** | | | | **Projeto 2** | | | |
| 1ª sem | 2ª sem | 3ª sem | 4ª sem | 1ª sem | 2ª sem | 3ª sem | 4ª sem |
| Escolha do assunto do projeto | x |  |  |  |  |  |  |  |
| Elaboração da estrutura do projeto | x | x |  |  |  |  |  |  |
| Seleção e leitura das obras para elaboração do projeto |  | x | x |  | x | x | x |  |
| Elaboração dos objetivos, delimitação do tema, definição do problema, etc. |  | x | x |  | x |  | x |  |
| Elaboração da pesquisa bibliográfica e documental do projeto |  |  | x | x |  | x | x |  |
| Coleta de dados |  | x | x |  |  | x | x |  |
| Tratamento dos dados |  |  |  | x |  |  | x |  |
| Revisão final do texto e elaboração da introdução e conclusão |  |  |  | x |  |  | x | x |
| Data limite de entrega do Projeto de Estágio |  |  |  | x |  |  |  | x |

# REFERÊNCIAS BIBLIOGRÀFICA

BELLONI, Maria Luíza. **O que é mídia-educação**. 2. ed. Campinas, SP: Autores associados, 2005

BROWN, H. D. **English Language Teaching in the "Post-Method" Era: Towards Better Diagnosis, Treatment, and Assessment**. In: Richards, J. C.; Renandya, W. A. Methodology in Language Teaching: an Anthology of Current Practice. New York: Cambridge, 2002, p. 9-18.

KUMARAVADIVELU, B. Beyond. **Methods: Macrostrategies for language teaching. New Haven**, CT: Yale University Press, 2003

LEFFA, V. J. **Metodologia do ensino de línguas**. In: BOHN, H. I.; VANDRESEN, P. Tópicos em linguística aplicada: O ensino de línguas estrangeiras. Florianópolis: EDUFSC, 1988, p. 211-236.

LÉVY, Pierre. **A máquina universo: criação, cognição e cultura informática**. Lisboa: Instituto Piaget, 1995.

\_\_\_\_\_\_. **Cibercultura**. São Paulo: Editora 34, 1999.

MENEZES, Vera. **English Language teaching and learning in the Age of Technology** (2012) - palestra no III Congresso Internacional da ABRAPUI. Disponível em: <http://www.veramenezes.com/abrapui2012.pdf>

PRABHU, N. S. **There is no best method-Why? TESOL Quarterly**. v. 24, n. 2, 1990.

WARSCHAUER, Mark. **A Developmental Perspective on Technology in Language Education**. TESOL QUARTERLY, v. 36, n. 3, 2020

1. Professor Curso Técnico de Informática – Faculdade e Escola Técnica Alcides Maya - joao\_moreira@alcidesmaya.edu.br [↑](#footnote-ref-2)