**(DEVE CONTER NO MÍNIMO 15 PÁGINAS E NO MÁXIMO 20 PÁGINAS)**

**RESUMO**

**Objetivo:** Desenvolver um site para auxiliar a inserção de estrangeiros e brasileiros no mercado de trabalho do Brasil. **Método:** Na criação do website, a primeira decisão foi definir o tipo de arquitetura cliente e servidor, em seguida as linguagens de programação utilizadas, sendo elas o framework React com a linguagem Typescript no front-end, uma API REST Django e a linguagem Python no back-end na parte do servidor junto com as regras de negócio e o MySQL como sistema de Gerenciamento de Banco de Dados, sendo todos eles gerenciados por um repositório privado no GitHub. A criação do protótipo do design foi feita com a ferramenta Figma, pensando na melhor experiência para o usuário e o website inteiro foi feito em cima dessas decisões. **Resultados:** O site de vagas de emprego Brazil Career Hub tem uma estrutura completa com todas as informações necessárias para a inserção no mercado de trabalho brasileiro para qualquer pessoa interessada, mas com foco no público estrangeiro, trazendo páginas com explicações sobre legislações e normas específicas do país, promovendo a imigração de mão de obra qualificada no país. **Conclusão:** Após revisão de dados obtidos em buscas de sites oficiais do governo brasileiro, desenvolveu-se o website Brazil Career Hub para contribuição no preenchimento com mão de obra qualificada de vagas de emprego no Brasil, independente se o candidato à vaga reside ou não no país.

**Palavras-chave:** Imigrantes; Website; Emprego; Refugiados; Mão de obra qualificada.

**INTRODUÇÃO**

**OBJETIVO GERAL**

O presente trabalho é sobre busca e preenchimento de vagas de emprego, tem como objetivo a criação de um website para as pessoas que estão procurando emprego no território brasileiro possam identificar vagas, leis e normas relacionadas a documentações, independente de já residirem no país ou não, tendo um foco maior para imigrantes, desta forma o website terá como principal funcionalidade a listagem de vagas, a diversidade de dados auxiliará na redução de desemprego, resultando em geração de oportunidade para todos e preenchendo vagas com mão de obra qualificada. Com a crescente globalização, muitas pessoas estão buscando oportunidades de trabalho em países estrangeiros, e a falta de informação e barreiras culturais podem dificultar esse processo. Além disso, muitas vezes empresas têm dificuldades em encontrar talentos com habilidades específicas.

**OBJETIVO ESPECÍFICO**

O objetivo específico é a criação do website com a arquitetura cliente e servidor, usando o framework React no front-end juntamente com a linguagem de programação typescript, MySQL como banco de dados, API REST Django e a linguagem de programação Python no back-end. A metodologia de pesquisa utilizada foi bibliográfica, a análise de dados sobre imigrantes foi feita em cima de dados oficiais obtidos de documentos do governo brasileiro.

**JUSTIFICATIVA**

Existem alguns sites que focam em empregos para imigrantes, porém poucos citam vagas sobre o Brasil. Segundo o artigo de Freitas (2023), atualmente nos Estados Unidos possuem aproximadamente 28 milhões de imigrantes, o que é um valor muito grande comparado ao número no Brasil representando apenas 181.385 imigrantes registrados em 2020 (TST, 2023). Com um website específico para imigração para o Brasil, poderíamos facilitar a imigração trazendo múltiplos benefícios, como por exemplo a diminuição do desemprego que, segundo o IBGE (2022), a taxa se encontra em 7,9% no 4° trimestre de 2022. Essa taxa abaixaria, pois o nosso site seria mais uma fonte de empregos.

É difícil achar informações sobre processos do nosso dia a dia, como por exemplo a declaração do imposto de renda, em 2022 oitenta mil goianos ainda não havia declarado seu imposto de renda com apenas 1 dia para o fim do prazo (Alexandria, 2022). Sendo assim, acreditamos que o site poderá ajudar pessoas a nível nacional e internacional, diminuindo o desemprego e combatendo a desinformação.

**DESENVOLVIMENTO**

**GITHUB**

Para mantermos um controle de versão e suportar o desenvolvimento simultâneo do projeto utilizamos o GitHub como o principal local para armazenar e organizar o código. O GitHub é um serviço baseado em nuvem que hospeda um sistema de controle de versão (VCS) chamado Git, o qual é utilizado para controlar o histórico de alterações de arquivos e principalmente de projetos, permitindo mais flexibilidade no fluxo de trabalho, segurança e desempenho. De todas as funcionalidades que o serviço disponibiliza, as principais utilizadas foram: workflows, branches, issues, pull requests, projects e controle de dependência automatizada.

Um workflow é um conjunto de processos e ferramentas automatizadas que auxiliam no controle de qualidade do código, o projeto utiliza três principais workflows. Um para a avaliação do ambiente de frontend, outro para a avaliação do código Python escrito e outro para a avaliação do ambiente de backend.

Tela de computador

Descrição gerada automaticamente

O workflow para avaliação do ambiente de frontend consiste em compilar o código utilizando a própria biblioteca do Node. Já o fluxo para avaliação do código Python consiste na utilização de uma biblioteca de padrão de código chamado Pylint, o qual irá validar diversos pontos e informar uma “nota” para o código, caso essa nota seja menor que 9, o workflow irá falhar. Por fim, o workflow para validação do ambiente de backend consiste na atualização da estrutura do banco de dados, coleta de arquivos estáticos, avaliação do código e execução de casos de testes.

As branches são ramificações do projeto que possuem o objetivo de isolar desenvolvimentos e alterações para impedir modificações, quebras e problemas em outros ambientes. O nosso projeto possui duas principais ramificações, uma para desenvolvimento e outra para produção. A ramificação de desenvolvimento é destinada para o uso cotidiano, onde serão enviados os códigos finalizados, todas as ramificações pessoais (utilizado para o desenvolvimento ativo) são baseadas nessa branch, permitindo um controle ainda maior nas alterações. Já a ramificação de produção tem o objetivo de centralizar o ambiente e o código que estará publicamente disponível para o usuário final, isolando o ambiente e evitando que modificações de desenvolvimento quebrem o ambiente.

Os issues (do Inglês, problemas) são “tickets” abertos para controle de tarefas e problemas que devem ser corrigidos, neles são descritos os passos para reprodução, qual ambiente é afetado por esse problema, qual a criticidade do problema, quem será o responsável da resolução desse problema entre outros. Esses issues são os principais agentes para o controle de tarefas e pendências do projeto, permitindo que novas implementações e correções possam ser realizada futuramente sem correr o risco de “cair no esquecimento”.

Tela de computador com jogo

Descrição gerada automaticamente

Conforme comentado anteriormente, projeto possui algumas ramificações para permitir o isolamento do desenvolvimento. Quando o desenvolvimento é finalizado, as alterações dessas ramificações devem ser mescladas com as alterações existentes da ramificação pai (no caso, a de desenvolvimento), para realizar essa mesclagem é necessária a abertura de um pull request, essa requisição é um método para permitir que outros desenvolvedores revisem o novo código antes de ser oficializado numa branch geral, no pull request também são executados os pipelines, os quais validam a qualidade do código de forma automatizada para que a revisão por outros desenvolvedores possa ser mais rápida e fácil.

Tela de celular com aplicativo aberto

Descrição gerada automaticamente

A funcionalidade de projects é algo que foi recentemente adicionado ao GitHub, permitindo uma exibição, ordenação e gerenciamento melhor, mais otimizado e persistente dos issues, conforme comentado anteriormente, os issues são constantemente utilizados para manter um controle referente aos problemas e pontos pendentes do projeto, o projects pode ser utilizado para criar boards, roadmaps entre outros recursos visuais e automatizados para facilitar esse controle. O nosso projeto utiliza um board geral, dois específicos e um roadmap, permitindo que o desenvolvedor tenha uma melhor visão de suas tarefas em andamento, e pendentes.

Tela de computador com jogo

Descrição gerada automaticamente

Por fim, mas não menos importante, temos o controle de dependência automatizada. Comumente conhecido como Dependabot, o autômato criar pull requests ao detectar uma nova versão de alguma biblioteca ou dependência que o projeto utiliza, qualquer que seja o ambiente. Além disso, também aponta falhas de segurança e outros problemas relacionados a dependências e pacotes externos utilizados, mantendo uma maior qualidade, atualidade e segurança no projeto.

**FRONT-END**

No desenvolvimento do nosso projeto do TCC, um dos problemas que tivemos que pensar é a interface do sistema pois é uma das partes mais importantes porque com o nosso propósito de construir um site de empregos a nível mundial focado para vagas do Brasil, precisaríamos de uma interface simples de se usar, fácil de se aprender e sem muita burocracia para que os candidatos não desistam facilmente de criar sua conta e aplicar-se para uma vaga criada no nosso site.

Então para planejarmos sobre, criamos alguns esboços das telas utilizando o software Figma, com ele pudemos decidir as cores principais do site, alguns layouts de tela para as funcionalidades que o usuário irá interagir conforme figura (x.x)

A screenshot of a computer

Description automatically generated

Para o desenvolvimento da interface decidimos usar o React, pela sua facilidade de criação de telas e muitas quantidades de biblioteca que podem ser integradas, com o intuito de facilitar ainda mais a criação de pequenos componentes que são usados nas telas e já com padronizações criadas por outros desenvolvedores. O React é uma biblioteca criada pela empresa Meta com a ajuda da comunidade, com data de lançamento inicial em 29 de maio de 2013, sendo open source onde qualquer pessoa pode contribuir para o projeto, o React está sendo muito utilizado em diversas áreas e empresas, portando temos muita facilidade de encontrar conteúdo sobre a maioria das coisas que procuramos desenvolver, e problemas que identificamos no meio do caminho.

Sobre as bibliotecas que utilizamos com o React, a primeira delas é a “styled-components”, uma biblioteca que permite utilizarmos estilização para componentes em específico sem que seja utilizada uma estilização global importando arquivos CSS, como é feito por padrão no React sem qualquer biblioteca.

Também utilizamos react-router-dom, essa biblioteca nos permite mapear rotas do navegador com componentes do React, por exemplo, quando o usuário acessar o nosso domínio com o caminho /home, então podemos mapear para que algum componente que criamos possa ser renderizado nesse momento.

A biblioteca mais importante utilizada no nosso projeto foi a Material UI, ela nos fornece diversos componentes prontos, no qual precisamos apenas passar algumas propriedades conforme a necessidade de cada desenvolvimento, nosso projeto teve muito mais agilidade devido a isso, pois não precisamos ficar criando componentes do zero, ela também facilita a criação de sites responsivos, não foi o foco do nosso projeto, mas caso quiséssemos seria muito fácil fazer uma adaptação.

**Estrutura do projeto**

A estrutura do projeto front-end foi pensada visando separar cada responsabilidade do código fonte, para que assim seja fácil escalar e implementar novas funcionalidades caso seja necessário.

**Diretorio de api**

Na pasta de api temos todos as funções que foram implementadas para realizar requisições para o back-end, cada diretório contém chamadas de acordo com sua entidade, então a pasta “job” por exemplo contém todas as chamadas de API que tem relação com os trabalhos.

A screenshot of a computer

Description automatically generated

**Pasta de assets**

A black rectangular object with white text

Description automatically generated

A pasta de assets do projeto contém todos os arquivos de midia que irão ser utilizadas no projeto, como imagens, aúdios, vídeos, gif, figurinhas etc.

**Pasta de components**

A black background with white text

Description automatically generated

Sobre a pasta de components, ela contém todos os componentes que podemos criar utilizando React, componentes nada mais são que trechos de código HTML, CSS e Javascript que são criados e podem ser reutilizados em diversas partes do nosso sistema, sendo extremamente poderoso e simples de ser feito, também podemos receber propriedades nos componentes tornando-o personalizável para cada parte do código.

**Pasta de contexts**

A black rectangular object with a white border

Description automatically generated with medium confidence

A pasta de contexts possuí contextos que podem ser utilizados em qualquer parte da aplicação, com ela podemos salvar dados e funções que serão reutilizados, ela possuí uma vantagem muito grande comparado com passar dados via componentes, pois se fossemos fazer pelo componentes teríamos que ter uma hierarquia de propriedades enorme, o que tornaria a manutenção da mesma muito complicada, também tornaria o código frágil fazendo com que caso um componente no meio da arvore quebre, façam com que todos os outros que estão abaixo dele também pare de funcionar, no nosso caso utilizamos muito para tratar a parte de autenticação na interface, fornecendo todos os dados do usuário ou empresa logados, métodos para deslogar, tipo de usuário logado etc.

**Pasta de hooks**

A black rectangular object with a white stripe

Description automatically generated

Sobre a pasta de hooks, ela contém funcionalidades que podem ser reutilizadas sobre os componentes, hooks nada mais são que uma função facilitada que podem ser utilizados para diversos princípios, como buscar dados, configurar variáveis, executar funcionalidade em qualquer ciclo de um componente, etc. No nosso projeto criamos um para que certas páginas do nosso projeto só fossem acessadas após o usuário ou empresa terem feitos login.

**Pasta de models**

A screenshot of a computer

Description automatically generated

A pasta de models contém todas as estruturas de dados que foram mapeados durante o desenvolvimento da aplicação, ela contém tipos criados com Typescript, na qual contém todos os dados de uma estrutura, também possuí algumas funcionalidades sobre a própria estrutura. Essa parte do código fonte está relacionadas com diversas outras partes do sistema, como por exemplo páginas que simbolizam essas estruturas, chamadas de API que retornam estruturas mapeadas aqui, criação de módelos no back-end etc.

**Pasta de styles**

A black rectangular object with white text

Description automatically generated

A pasta de styles contém tudo que seja relacionado a estilização global da aplicação já que estilizações locais de componentes podem ficar dentro da pasta deles ou no seu próprio arquivo, dentro do arquivo global.css podem conter códigos CSS que irão ser utilizados no projeto inteiro, então se quisermos alterar a estilização de uma tag span por exemplo, colocaríamos a forma da estilização dela nesse arquivo, já o arquivo theme.ts contém a estilização global relacionada ao framework Material UI, o mesmo além de nós fornece diversos componentes prontos, ele também nos dá a possibilidade de estilizarmos o mesmo, como alterarmos cores principais, tamanhos de fontes, as fontes etc.

**Pasta de utilities**

A screenshot of a computer

Description automatically generated

Essa pasta possuí todos os arquivos que contém funcionalidades uteis que podem ser usadas ao longo do projeto como por exemplo, formatação de datas, formatação de campos, utilidades para textos, etc. Qualquer coisa que é útil e pode ser reutilizada deve ficar dentro desse diretório.

**Pasta de views**

A screenshot of a computer

Description automatically generated

A pasta de views contém todas as páginas do site, nela contém diversos componentes e elas mesmo também são um componente, então na prática ela é o início de uma hierarquia de componentes e códigos HTML, CSS e Typescript. Cada página deve ter uma rota que deve ser mapeada no arquivo “router.ts” que se encontra na raiz do projeto, dentro desse arquivo contém uma lista com vários items na qual deve mapear um componente com um texto que nada mais é a rota do site.

**BACK-END**

O ambiente backend é responsável por armazenar os dados de todas as empresas e usuários, vagas, currículos e por disponibilizar esses dados para a interface. A aplicação do backend é uma API utilizando o framework v4.2.3 codificado na linguagem de programação Python v3.11.4. A aplicação também implementa o framework RESTful (também conhecido como rest) encapsulado como uma biblioteca otimizada e customizada para ser utilizada juntamente com o framework do Django.

A aplicação faz uso de um banco de dados relacional MySQL, para facilitar o processo e reduzir os recursos necessários durante o desenvolvimento, estamos utilizando uma variação do MySQL que não necessita de um servidor e é independente, o sqlite, o qual permite persistir os dados em um banco de dados relacional não precisando hospedar ou se conectar com um servidor.

A API disponibilizada possui 19 endpoints com 15 deles possuindo operações CRUD completas, para facilitar o desenvolvimento o backend possui uma documentação completa para todas as APIs disponibilizando uma interface gráfica utilizando o Swagger e Swagger UI, os quais são conjuntos de ferramentas de API na qual a especificação OpenAPI é baseada.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Além da interface gráfica, para facilitar ainda mais os testes durante o desenvolvimento, também implementamos uma coleção completa de endpoints hospedada no Postman, uma plataforma de API para desenvolvedores. Nessa coleção, o desenvolvedor pode realizar chamadas para os endpoints do backend para avaliar as respostas, os parâmetros e outros pontos necessários para o desenvolvimento de uma API.

Tela de computador com fundo preto

Descrição gerada automaticamente

A aplicação possui um método de autenticação do tipo Bearer Token, o qual é um tipo de token de acesso em uma API utilizado para autenticação e autorização para manter as credenciais do usuário e indicar autorização para requisições de acesso. Os tokens Bearer são gerados com base em protocolos e especificações como OAuth e JWT (JSON Web Token).

Toda requisição para o ambiente de backend e seus endpoints devem possuir um token localizado no cabeçalho (header) da requisição, caso a requisição não informe um token, o acesso não será autorizado e o usuário receberá um erro. Para gerar um token, o usuário deve possuir uma conta de administrador no ambiente de backend e solicitar a geração de um token de acesso a partir de um endpoint único, o qual deve se autenticar utilizando usuário e senha.

Tela de celular com aplicativo aberto

Descrição gerada automaticamente

Toda a camada de endpoint, autenticação e resposta é tratada pelo próprio framework do Django com a ajuda do framework do rest. Todos os endpoints possuem o mesmo esquema de declaração e funcionamento da API, o que difere entre um endpoint e outro são dos dados que são gerenciados pelo endpoint, a complexidade e a lógica para aquisição e tratamento dos dados.

A arquitetura do ambiente de backend foi modularizada e seccionada cuidadosamente para criar uma arquitetura onde possibilita um fácil escalonamento, atualização e manutenção.

Texto

Descrição gerada automaticamente com confiança média

A pasta **api** possui os arquivos responsáveis pela informação geral da API. A pasta **api\_admins** possui os arquivos responsáveis pelo armazenamento e gerenciamento de usuários administrativos, os quais poderão gerar seus tokens para realizar acessos remotos via endpoints. A pasta **company** é responsável pela lógica, armazenamento e gerenciamento de todos os dados (conta e perfil) das empresas cadastradas no site.

Assim como a **company**, a pasta **users** é responsável pela lógica, armazenamento e gerenciamento de todos os dados (conta e perfil) dos usuários cadastrados no site. A pasta **vacancy**, assim como o nome define, é responsável pelo tratamento de todas as vagas criadas. Por fim, a pasta **resumes** é responsável pela lógica, armazenamento e gerenciamento de todos os currículos gerados pelos usuários, os quais serão utilizados para se candidatar nas vagas, as quais por sua vez, serão publicadas pelas empresas.

A estrutura de urls e endereços da API são definidas em arquivos específicos dentro do framework do Django, os quais são comumente nomeados de **urls.py**. Nesses arquivos, são definidos os endereços os quais serão acessados pelos usuário através dos endpoints, essas definições irão lincar a parte lógica da API (também conhecidas como views) e a área externa da aplicação (os endpoints).

Texto

Descrição gerada automaticamente

As views, por sua vez, possuem uma abordagem orientada a objetos, também conhecida como **class-based views**, sendo ela uma das abordagens disponibilizada pelo framework do Django. A criação de uma API consiste em duas classes, um schema, o qual será responsável por produzir as informações necessárias para gerarmos a documentação do Swagger, e uma classe onde será responsável pelo endpoint.

A classe responsável pelo schema renderizado na documentação do Swagger deve conter duas principais funções, uma para a descrição do conteúdo e outra para os parâmetros do endpoint. Essa classe será utilizada apenas para renderizar as informações corretas, porém, apesar de sua funcionalidade simples, mostra-se essencial para o conhecimento da API e de seus endpoints.

Texto

Descrição gerada automaticamente

A classe responsável pelo endpoint deverá conter os métodos das requisições que deverão ser suportadas pelo endpoint, ou seja, se o endpoint permite uma requisição no método GET, uma função get deve existir dentro da classe pelo fato de que ela será executada ao receber uma requisição desse método.

Essa função deverá receber um parâmetro self, o qual será utilizado para referenciar a própria classe, e um parâmetro request, o qual será utilizado para referenciar a requisição que foi recebida, possuindo uma tipagem de um HttpRequest, uma estrutura baseada em dicionários disponibilizada no Python.

Essa função também deverá retornar uma resposta que possa ser serializada e enviada via HTTP, permitindo uma comunicação entre APIs sem a renderização de uma página. Essa resposta deverá receber um dicionário que irá conter os dados que devem ser retornados na resposta, assim como um código de status, o qual será utilizado para definir se a requisição retornou os dados com sucesso ou se ocorreu algum erro no processo.

Texto

Descrição gerada automaticamente

Nesse caso em específico, o conteúdo retornado é apenas um texto (string) que irá informar a versão da API e qual o ambiente que está sendo executado (desenvolvimento ou produção). Outro modelo de resposta que também é muito utilizado é o retorno de objetos do banco de dados, porém, diferente de uma string, um objeto que seria uma classe na linguagem Python não pode ser serializado automaticamente pelo framework do rest.

Para permitir a serialização de um objeto, é necessário definir um serializador que realizará essa conversão de um objeto para algo que possa ser retornado via protocolos HTTP, o que no caso seriam dicionários.

Texto

Descrição gerada automaticamente

O exemplo mostrado na imagem **<insert image number here>** é o serializador de objetos de vagas, ele é responsável por converter uma entrada do banco de dados em um dicionário que possa ser serializado e transmitido para o requisitante via protocolo HTTP. Em resumo, o serializador possui uma meta classe que irá definir duas principais variáveis, **model** e **fields.** A variável **model** será responsável por definir qual a tabela do banco de dados a qual os dados deverão ser buscados, já a variável **fields** irá informar quais informações (colunas) devem ser selecionadas e serializadas para poderem ser transmitidas.

Além dessas duas variáveis, o exemplo exibe também algumas variáveis computadas, os quais seriam **addresses** e **company\_name**. Essas variáveis são variáveis únicas geradas durante a serialização, geralmente utilizadas para obter informações de outras tabelas que estão lincadas por chaves estrangeiras, nesse caso, a variável **addresses** é responsável por obter o endereço disponível para essa vaga em específico, a qual é armazenada em uma outra tabela de endereços da empresa. Já a variável **company\_name** é responsável por obter o nome da empresa a qual criou a vaga, a qual também é armazenada em uma outra tabela de contas e perfis das empresas.

O retorno desse serializador é um dicionário que poderá ser transmitido via protocolo HTTP, caso algum erro ocorra durante a serialização, uma exceção será levantada, alertando o usuário sobre o erro.

No framework Django, o banco de dados é tratado completamente pelo próprio framework, sendo necessária a criação do chamado modelo. O modelo é uma classe a qual possui uma estrutura específica para que possa ser interpretada pelo framework e manipulada como tabelas em um banco de dados relacional.

Texto

Descrição gerada automaticamente

O exemplo mostrado na imagem **<insert image number here>**, é o modelo da tabela das vagas criadas por uma empresa, a classe do modelo herda de uma classe específica do framework que permitirá que o Django interprete suas variáveis, métodos e metaclasses criadas e gere comandos SQL a partir dessas interpretações.

As variáveis declaradas na classe são as variáveis que irão definir as colunas das tabelas, os seus valores são os tipos / atributos nos quais os valores dessa coluna serão tratados, assim como a herança da classe, os valores das variáveis são classes específicas do framework, os quais permitirão a interpretação e manipulação conforme necessário. Das classes, a classe **ForeignKey** é responsável por gerar uma relação de chave estrangeira com outra tabela, o que neste caso, seria outro modelo gerado a partir do framework. A classe **ManyToManyField** é responsável por gerar uma relação muitos-para-muitos com outro modelo.

Devido a responsabilidade de manipulação do banco de dados ser completamente do framework, o Django disponibiliza nativamente uma interface de administração, também conhecido como “Django administration”, para que os dados do banco possam ser facilmente acessados e modificados sem necessitar uma conexão manual com o banco, podendo causar problemas e incompatibilidades com o tratamento do framework.

Tela de computador

Descrição gerada automaticamente

O registro de um modelo para exibição na página de administração do framework é realizado por um arquivo específico de sua estrutura chamado de **admin.py**, nele são criadas classes que irão exibir, além das seções presentes na imagem **<insert image number here>**, as páginas de gerenciamento e exibição dos dados da tabela.Texto

Descrição gerada automaticamente

O exemplo mostrado na imagem **<Insert image number here>** é o processo de registro da exibição das vagas criadas no banco de dados, a classe **VacancyAdmin** é criada herdando, assim como seu modelo, de uma classe específica do próprio framework, permitindo interpretar as informações e renderizar a página conforme solicitado.

As variáveis declaradas na classe são responsáveis pela configuração da exibição do modelo, a variável **list\_display** será responsável pela configuração de quais colunas da tabela serão exibidas na listagem e o **list\_display\_links** serão os links para acessar a página de gerenciamento dessa entrada em específico. O **list\_per\_page** é responsável por configurar o número máximo de entradas permitidas sem paginação, o **list\_filter** será um filtro localizado na extrema direita da página que irá permitir a filtragem dos dados listados, o **search\_fields** será responsável por informar ao framework quais são as colunas as quais o texto buscado deverá ser pesquisado e o **readonly\_fields** é responsável pela configuração de quais campos não devem ser modificados pelo usuário manualmente.

**METODOLOGIA**

A metodologia de pesquisa utilizada foi bibliográfica, a análise de dados sobre imigrantes foi feita em cima de dados oficiais obtidos de documentos do governo brasileiro. Com o objetivo de criar o site, os métodos utilizados foram definir a arquitetura cliente e servidor, onde toda parte visual do site foi desenvolvida utilizando o framework React com a linguagem TypeScript e as regras de negócio junto aos dados foram estabelecidas na parte do servidor, com uma API REST feita com Django e a linguagem Python. O sistema foi desenvolvido separado por funcionalidade, implementando e unificando as alterações a cada funcionalidade finalizada.

Sobre a parte de UI do sistema, foi criado um protótipo utilizando a ferramenta Figma para dar a possibilidade de ajuste de layout e design de todo site com facilidade de visualização, visando oferecer uma boa experiência aos usuários. Os dados foram salvos em um banco de dados único que utiliza a convenção relacional. O sistema foi dividido em tabelas e o Sistema de Gerenciamento de Banco de Dados escolhido foi o MySQL, pois ele possui funcionalidades importantes para criação de websites. Ambos cliente e servidor são gerenciados pelo GitHub e todo código está hospedado em um repositório privado no mesmo, onde todos os desenvolvedores envolvidos têm acesso.

O material de estudo foi obtido a partir de notícias, artigos, fóruns e livros que foram publicados nos últimos 5 anos. O conteúdo coletado foi utilizado para identificar as dificuldades e necessidades dos imigrantes nesse processo de transição, tendo como objetivo criar e fornecer ferramentas para ajudá-los a superar essas dificuldades. O controle de qualidade do site foi feito de forma manual durante o desenvolvimento e aplicação das ferramentas e funcionalidades, com o intuito de mitigar erros e melhorar a experiência do usuário durante a sua utilização.

**REFERENCIAL TEÓRICO**

No Brasil, o número de estrangeiros buscando oportunidades de emprego tem aumentado cada vez mais, isso se deve principalmente devido a crises em outros lugares do mundo. Segundo o relatório do Observatório das Migrações Nacionais (2022), o volume de estrangeiros trabalhando no Brasil foi de 62.423 para 187.985 de 2011 para 2021, quase triplicando.

Além dos trabalhadores com visto legal, o Brasil tem muitos refugiados e o marco legal para o início dessa imigração foi em 22 de julho de 1997 com a Lei nº 9.474 que define que será reconhecido como refugiado todo indivíduo que:

“I – devido a fundados temores de perseguição por motivos de raça, religião, nacionalidade, grupo social ou opiniões políticas encontre-se fora de seu país de nacionalidade e não possa ou não queira acolher-se à proteção de tal país;

II – não tendo nacionalidade e estando fora do país onde antes teve sua residência habitual, não possa ou não queira regressar a ele, em função das circunstâncias descritas no inciso anterior;

III – devido a grave e generalizada violação de direitos humanos, é obrigado a deixar seu país de nacionalidade para buscar refúgio em outro país.” (LEI Nº 9.474, DE 22 DE JULHO DE 1997).

Segundo os dados do Observatório de Migrações Nacionais (2022) entre os anos de 1997-2009 o Brasil recebeu 2.488 solicitações de refúgio e entre os anos 2010-2021 esse número aumentou mais de 100 vezes, saltando para 298.331 solicitações, dos quais 59% foram venezuelanos e 13,3% haitianos. Este fato se deve principalmente devido as crises humanitárias tanto na Venezuela quanto no Haiti, que fez com que parte da população buscasse refúgio em outros lugares do mundo.

**Tabela 1** **–** Número de solicitações de reconhecimento da condição de refugiado, segundo principais países de nacionalidade ou de residência habitual, Brasil, 2016-2021

Tabela

Descrição gerada automaticamente

Fonte: Relatório Anual 2022 (OBMigra)

Além de todos os dados já levantados, existem muitos refugiados e imigrantes que estão ilegalmente no país e acabam não entrando nos dados oficiais. É evidente que o Brasil tem um volume considerável de imigrantes tentando iniciar uma vida aqui e todos os dias novas vagas de emprego aparecem tentando ser preenchidas.

Com todos esses aspectos, é necessário criar um facilitador para os imigrantes que já vivem no Brasil e os que tem interesse em imigrar, que contenha desde informações sobre documentação até busca de vagas de emprego de confiança, para que seja evitada a exposição a golpes e vagas falsas.

Para isso o nosso site tem como objetivo auxiliar estrangeiros com interesse de vir trabalhar no Brasil, também os que já estão no país e precisam de um emprego e empresas que buscam expandir suas buscas de profissionais com aptidões para suas vagas abertas.

Existem diversos meios de divulgação de empregos no Brasil, com vagas disponíveis para os interessados se inscreverem e tentarem preencher. Os sites e aplicativos estão cada vez mais sendo utilizados como ponte para essa conexão entre empresas e pessoas buscando vagas.

Foi utilizado como base de estudo sites que de acordo com o portal Guia da Carreira, estão entre os 11 melhores sites de busca de emprego do Brasil. Dentro deles, existem sites com vagas mais gerais, para qualquer tipo de pessoa e existem outros mais focais, com vagas específicas para estagiários, pessoas com deficiência, áreas específicas ou para empregos fora do Brasil.

Figura 1 – Site Linkedin

Ícone

Descrição gerada automaticamente

Fonte: Linkedin, 2023

O mais conhecido é o Linkedin, lançado na Califórnia em 2003 ele funciona como uma rede social, currículo online e plataforma de empregos, muito utilizado para criar conexões com outras pessoas no mercado de trabalho, mesmo não tendo tantas ofertas de vagas quanto em outros concorrentes. É gratuito e tem a possibilidade de assinatura para amplificar a visibilidade do seu currículo.

Figura 2 – Site Catho

Logotipo

Descrição gerada automaticamente

Fonte: Catho, 2023

Existe também a Catho, uma das mais tradicionais e com mais ofertas de emprego no Brasil, o serviço tem a opção de teste gratuito por 7 dias, mas é pago após esse tempo. Pessoas com deficiência conseguem usar gratuitamente.

Figura 3 – Site Indeed

Logotipo

Descrição gerada automaticamente

Fonte: Indeed, 2023

Já o Indeed é um agregador de vagas totalmente gratuito, nele é possível buscar vagas em diversos sites de empregos diferentes ao mesmo tempo. Dessa forma poupa tempo do usuário em ficar entrando em diversos sites diferentes para realizar novas buscas.

Figura 4 – Trovit

Desenho de um círculo

Descrição gerada automaticamente com confiança baixa

Fonte: Trovit, 2023

O Trovit é um site internacional muito utilizado na Europa e na América Latina, mas diferente dos outros, ele não só tem vagas de emprego como também funciona como um classificado, tendo imóveis e carros também.

Figura 5 – Site CIEE

Desenho de um círculo

Descrição gerada automaticamente com confiança média

Fonte: CIEE, 2023

Para os estagiários, existe o Centro de Integração Empresa-Escola (CIEE), o qual é completamente gratuito e tem vínculo com diversas empresas que divulgam seus programas de estágio pela sua plataforma, facilitando a busca de quem está na faculdade querendo iniciar a vida profissional com estágio na sua área.

Figura 6 – Deficiente Online

Ícone

Descrição gerada automaticamente

Fonte: Deficiente Online, 2023

E para as pessoas com deficiência, foi criado o Deficiente Online, conhecido como PCD.com.br, dá a possibilidade de o usuário filtrar pelo tipo de deficiência que possui, além de possuir também guias sobre documentação para facilitar a organização dos candidatos. De todas as opções citadas anteriormente, essa é a mais próxima do que é necessário para os imigrantes no Brasil.

**RESULTADOS**

O resultado do Brazil Career Hub foi um projeto que facilitou muito para a aplicação de vagas de todos que pretendem trabalhar no Brasil, sua interface trouxe uma experiência fácil para o usuário, sem fazer com que o usuário passe muito tempo procurando por um emprego, a interação visual do site com o servidor ficou muito boa, com vários pontos dinâmicos, sem problemas de performance etc. O projeto se tornou uma boa oportunidade de investimento para trazer mão de obra qualificada de qualquer lugar do mundo para o Brasil, assim como promover o trabalho para Brasileiros residentes no qual resulta em geração de empregos.

O projeto resolve o problema de falta de emprego em cidades pequenas pois dentro do site já possuí a possibilidade de criação de vagas remotas, fazendo com que uma pessoa com ou sem uma especialidade possa trabalhar de qualquer lugar.

Um outro problema que o site facilita é a comunicação, pois no cadastro do usuário eles podem registrar algum dado que a empresa possa vir a se comunicar, como e-mail, telefone, website etc. A empresa também pode fornecer alguns dados na qual o usuário pode pesquisar mais sobre ela.

**CONCLUSÃO (CARLOS)**

**REFERÊNCIAS**

ALEXANDRIA, K. A um dia do fim, 80 mil goianos ainda não declararam o Imposto de Renda. O Popular. Disponível em: <https://opopular.com.br/economia/a-um-dia-do-fim-80-mil-goianos-ainda-n-o-declararam-o-imposto-de-renda-1.2464589>. Acesso em 25 de março de 2023.

FREITAS, E. O número de imigrantes nos Estados Unidos. Brasil Escola. Disponível em: <https://brasilescola.uol.com.br/geografia/o-numero-imigrantes-nos-estados-unidos.htm>. Acesso em 25 de março de 2023.

IBGE. Desemprego. Disponível em: <https://www.ibge.gov.br/explica/desemprego.php>. Acesso em 25 de março de 2023.

TST. Brasil tem mais de 180 mil imigrantes no mercado de trabalho formal. Disponível em: [https://www.tst.jus.br/-/brasil-tem-mais-de-180-mil-imigrantes-no-mercado-de-trabalho-formal#:~:text=Em%20uma%20década%2C%20o%20volume,das%20Migrações%20Nacionais%20(OBMigra)](https://www.tst.jus.br/-/brasil-tem-mais-de-180-mil-imigrantes-no-mercado-de-trabalho-formal#:~:text=Em uma década%2C o volume,das Migrações Nacionais (OBMigra)). Acesso em 25 de março de 2023.

SILVA, L. Os imigrantes no Brasil, sua vulnerabilidade e o princípio da igualdade. Disponível em: <https://www.researchgate.net/profile/Leda-Maria-Silva/publication/320203829_OS_IMIGRANTES_NO_BRASIL_SUA_VULNERABILIDADE_E_O_PRINCIPIO_DA_IGUALDADE/links/5dc589ff299bf1a47b23d708/OS-IMIGRANTES-NO-BRASIL-SUA-VULNERABILIDADE-E-O-PRINCIPIO-DA-IGUALDADE.pdf>. Acesso em 7 de maio de 2023.

CAVALCANTI, Leonardo; OLIVEIRA, Antonio Tadeu; TONHATI, Tânia (Orgs.) **A Inserção dos Imigrantes no Mercado de Trabalho Brasileiro**. Cadernos OBMigra, Ed. Especial, Brasília, 2015.

CAVALCANTI, L; OLIVEIRA, T.; SILVA, B. G. **Relatório Anual OBMigra 2022**. Série Migrações. Observatório das Migrações Internacionais; Ministério da Justiça e Segurança Pública/ Conselho Nacional de Imigração e Coordenação Geral de Imigração Laboral. Brasília, DF: OBMigra, 2022.

GUIA DA CARREIRA. Descubra quais são os melhores sites de emprego do Brasil. Disponível em: <https://www.guiadacarreira.com.br/blog/os-melhores-sites-de-emprego>. Acesso em 7 de maio de 2023.

Brasil. Lei no 9.474, de 22 de julho de 1997. Define mecanismos para a implementação do Estatuto dos Refugiados de 1951, e determina outras providências. Brasília, DF: Presidência da República. Disponível em: <https://www.planalto.gov.br/ccivil_03/leis/l9474.htm>. Acesso em 7 de maio de 2023.