UNIVERSIDADE DO ESTADO DO AMAZONAS
ESCOLA SUPERIOR DE TECNOLOGIA

EDILSON RIBEIRO DOS SANTOS JÚNIOR

DESENVOLVIMENTO DE SOFTWARE PYTHON PARA COLETA DE PREÇOS DE MATERIAIS ELÉTRICOS PARA CONSTRUÇÕES RESIDENCIAIS DE BAIXA TENSÃO

Orientação: Fábio de Sousa Cardoso, Dr.

Manaus
2023

EDILSON RIBEIRO DOS SANTOS JÚNIOR

DESENVOLVIMENTO DE SOFTWARE PYTHON PARA COLETA DE PREÇOS DE MATERIAIS ELÉTRICOS PARA CONSTRUÇÕES RESIDENCIAIS DE BAIXA TENSÃO

Projeto de pesquisa desenvolvido durante a disciplina de Trabalho de Conclusão de Curso I e apresentado à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro Eletricista.

Orientação: Fábio de Sousa Cardoso, Dr.

Manaus

2023

Universidade do Estado do Amazonas – UEA
Escola Superior de Tecnologia - EST

Reitor:
André Luiz Nunes Zogahib
Vice-Reitor:
Kátia Nascimento Couceiro
Diretora da Escola Superior de Tecnologia:
Ingrid Sammyne Gadelha Figueiredo
Coordenador do Curso de Engenharia Elétrica:
Israel Londres Torné

Banca Avaliadora composta por: *Data da defesa: 20/03/2023.*
Prof. Fábio de Sousa Cardoso, Dr. *(Orientador)*
Prof. Antonio Luiz Alencar Pantoja, Dr.
Prof. Arlindo Pires Lopes, Ph.D.

CIP – Catalogação na Publicação

Junior, Edilson Ribeiro dos Santos

Desenvolvimento de software na linguagem python para coleta de preços de materiais elétricos para construções residenciais de baixa tensão / Edilson Ribeiro dos Santos Júnior; [orientado por] Fábio de Sousa Cardoso, Dr. – Manaus: 2023.
45 p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica). Universidade do Estado do Amazonas, 2023.

1. Coleta de dados. 2. Desenvolvimento em Python. 3. Automação de processos. I. Cardoso, Fábio de Sousa.

EDILSON RIBEIRO DOS SANTOS JÚNIOR

DESENVOLVIMENTO DE SOFTWARE PYTHON PARA COLETA DE PREÇOS DE MATERIAIS ELÉTRICOS PARA CONSTRUÇÕES RESIDENCIAIS DE BAIXA TENSÃO

Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro Eletricista.

Nota obtida: 9,9 (nove pontos e nove décimos)

Aprovada em 20 / 03 / 2023.

Área de concentração: Automação de Processos

BANCA EXAMINADORA

Fábio de Sousa Cardoso

Orientador: Fábio de Sousa Cardoso, Dr.

Antonio Pantoja

Avaliador: Antonio Luiz Alencar Pantoja, Dr.

Arlindo Pires Lopes

Avaliador: Arlindo Pires Lopes, Ph.D.

Manaus
2023

Dedicatória

Dedico este trabalho aos meus pais, Edilson e Leia, que nunca mediram esforços desde meu primeiro caderno para que minha graduação no Ensino Superior fosse possível. Sou grato a Deus pela oportunidade e saúde que me dá dia após dia para obter êxito nos meus objetivos.

AGRADECIMENTOS

Agradeço aos meus familiares, à minha noiva e aos meus amigos por todo apoio e incentivo durante todos os anos da graduação. Agradeço ao meu orientador Fábio de Sousa Cardoso e ao professor Arlindo Pires Lopes por todo o suporte durante o desenvolvimento do meu Trabalho de Conclusão de Curso, além do meu irmão Jason Silva dos Santos que criou a necessidade da resolução de um problema que gerou a ideia para este trabalho. Agradeço ainda à Universidade do Estado do Amazonas e à Escola Superior de Tecnologia por viabilizar este estudo no ramo de desenvolvimento de software, contribuindo com materiais de pesquisa, participação de colaboradores e conhecimento sólido para abordagem do tema.

RESUMO

O mercado de trabalho de Engenharia Elétrica exige uma capacitação cada vez maior dos profissionais da área, não bastando apenas o diploma de formação. A experiência dentro da área de instalações elétricas passa, por exemplo, por ferramentas capazes de auxiliar na construção de orçamentos e busca de preços materiais, porém muitos estudantes e engenheiros recém formados não têm acesso por falta de disponibilidade a preços acessíveis. Buscar por soluções de softwares utilizados no ambiente de trabalho que funcionem de maneira simples e efetiva permite que os discentes de engenharia elétrica da Universidade do Estado do Amazonas possam entrar no mercado de trabalho de maneira competitiva. Este trabalho tem por objetivo o desenvolvimento de um software capaz de coletar preços de materiais elétricos e gerar orçamentos de maneira automatizada, utilizando Python. A coleta será realizada utilizando a biblioteca Selenium, capaz de automatizar a interação do usuário com um navegador, a interface será feita utilizando a biblioteca Py Simple GUI, otimizando a janela para uma aplicação leve e simples, e o orçamento dos materiais será gerado com o auxílio da biblioteca Pandas, capaz de interagir com os dados provindos da planilha de preços. Ao final do desenvolvimento, uma análise sobre o potencial de desenvolvimento de uma aplicação para outras engenharias da UEA.

Palavras-Chave: Coleta de dados, Desenvolvimento em Python, Automação de processos.

LISTA DE FIGURAS

SUMÁRIO

ABSTRACT

The job market for Electrical Engineering demands an increasing level of qualification from professionals in the field, with just a degree not being enough. Experience in the area of electrical installations, for example, requires tools that can assist in budget construction and material price searching, but many students and newly graduated engineers do not have access due to unaffordable prices. Seeking solutions for software used in the work environment that function in a simple and effective way allows electrical engineering students at the State University of Amazonas to enter the job market competitively. This work aims to develop a software capable of collecting prices of electrical materials and generating automated budgets, using Python. Collection will be done using the Selenium library, capable of automating user interaction with a browser, the interface will be made using the Py Simple GUI library, optimizing the window for a lightweight and simple application, and the material budget will be generated with the aid of the Pandas library, capable of interacting with data from the price spreadsheet. At the end of the development, an analysis will be conducted on the potential for developing an application for other engineering fields at UEA.

Keywords: Data Collection, Python Development, Process Automation.

1. INTRODUÇÃO

O mercado de trabalho tem se qualificado cada vez mais, a ponto de faltar pessoas qualificadas em 2023 (CNN, 2021). Dentro do mercado de instalações elétricas, é possível verificar a dificuldade de novos engenheiros adquirirem ferramentas que os auxiliem na execução de seus projetos, como softwares orçamentários, a custos acessíveis, distanciando a experiência da sala de aula da experiência demandada pelo mercado. Dessa forma, é necessário desenvolver soluções que auxiliem os jovens do mercado de trabalho.

Os softwares orçamentários para itens de instalações elétricas comumente são atrelados aos programas para construção civil, uma vez que o primeiro compõe o segundo. Dessa forma, os preços também costumam acompanhar o aglomerado de ferramentas que o software permite, resultando em opções com recursos e preços acima do necessário para engenheiros eletricista, uma vez que engenheiros civis estão aptos a projetar a parte elétrica de uma construção civil.

Ainda, a procura por preços de distintos materiais a cada novo projeto resulta em um acréscimo significativo no tempo do retorno do orçamento, tendo em vista que nem todos os sites dispõem de uma biblioteca completa de materiais ou mesmo de um mecanismo prático de procura de itens. A existência de uma ferramenta para controle de gastos ainda não sanaria a deficiência na procura de materiais.

Com o contexto anteriormente exposto, a disponibilidade de uma solução tecnológica que possa pesquisar, mensurar e reportar materiais elétricos utilizados em construção civil auxiliará tanto jovens engenheiros quanto estudantes de engenharia elétrica, que aplicariam a ferramenta em disciplinas de Instalações Elétricas, por exemplo.

Este projeto de pesquisa tem como objetivo final o desenvolvimento de um software Python capaz de gerar orçamentos de materiais elétricos para construção civil utilizando um banco de dados local, alimentado pelo Sistema Nacional de Pesquisa de Custos e Índices da Construção Civil (Sinapi) através da documentação mensal com preços dos insumos disponibilizada por ele.

2. REFERENCIAL TEÓRICO

Neste capítulo serão abordados os aspectos teóricos dos assuntos relacionados ao desenvolvimento do projeto, como Python, Selenium, biblioteca Pandas e outras tecnologias, além de uma contextualização relacionada às principais opções de softwares que apresentam a funcionalidade do projeto.

2.1 SOFTWARES ORÇAMENTÁRIOS PARA INSTALAÇÕES ELÉTRICAS

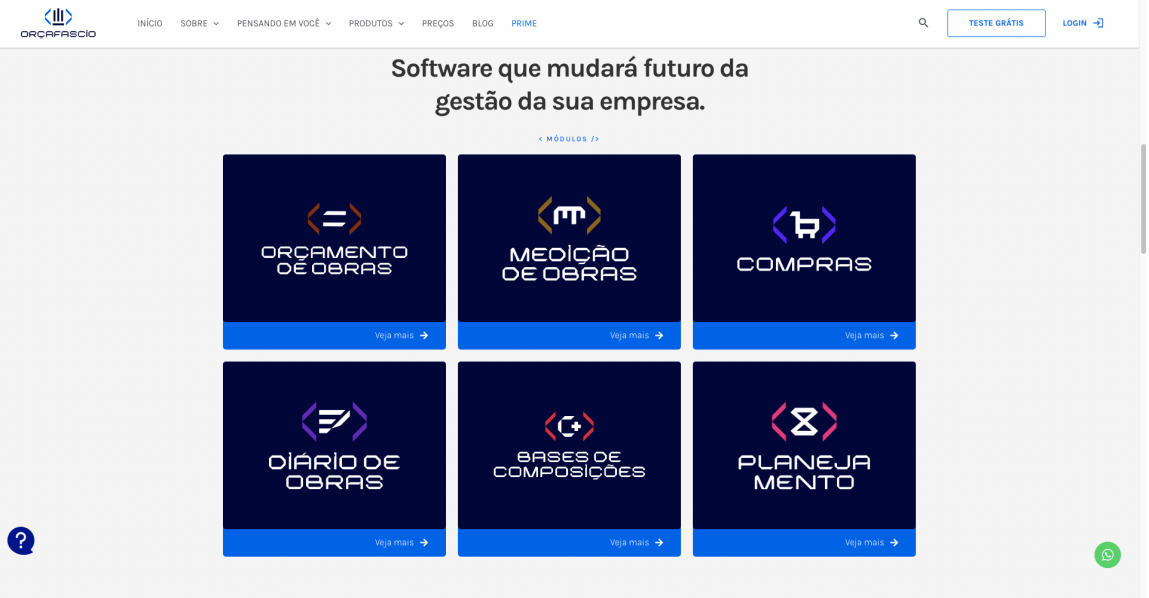
Com auxílio dos discentes de Engenharia Elétrica e Civil da Universidade do Estado do Amazonas (UEA), foram selecionadas as principais ferramentas utilizadas com o intuito de fazer levantamento de materiais e orçamentos, comentados a seguir sobre funcionalidades e base de preços.

2.1.1. ORÇAFASCIO

O OrçaFascio surgiu em 2015 para auxiliar no mercado de orçamentos seguindo os padrões e normas do Tribunal de Contas da União, utilizando diversas bases de composição como SINAPI, SICRO e SBC, atualizadas de forma automática.

Dentre as funcionalidades do site, é possível criar orçamentos de maneira ilimitada, gerar relatórios em XLSX, formato de planilha de um arquivo Excel, ajuste automático no valor de orçamentos e consultas à base de dados de insumos e serviços, dentre outras.

Figura 1: OrçaFascio



Fonte: (OrçaFascio, 2021)

Para o módulo mais simples, com funcionamento voltado para orçamento, o OrçaFascio pede R$999,00 anuais para acesso de cinco usuários, com possibilidade de acesso gratuito por 7 dias (ORÇAFASCIO, 2021).

Figura 2: Orçamento de aquisição de licença

Fonte: (OrçaFascio, 2021)

2.1.2. ALTOQI VISUS

O AltoQi Visus é um software para orçamento e planejamento integrado ao fluxo de desenvolvimento de projetos BIM (Building Information Modeling), permitindo a atualização do orçamento em qualquer fase do projeto.

O AltoQi Visus não possui preço explícito, sendo necessário solicitar uma proposta junto ao site, com necessidade de pelo menos 5 pessoas para aceite de projeto (ALTO QI, 2022).

Figura 3: AltoQi Visus

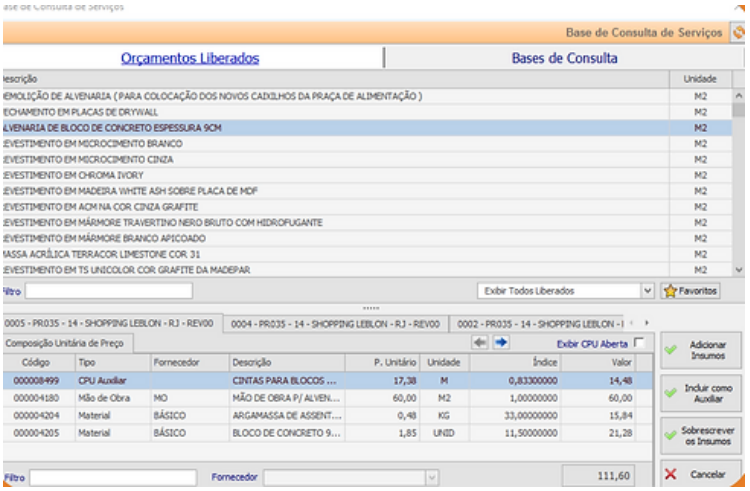
Fonte: (AltoQi, 2022)

2.1.3. LUMINA ERP

O Lumina ERP é o principal produto do portfólio da Lumina IT, empresa de tecnologia da informação. Similar ao OrçaFascio, também compõe orçamento a partir de um banco de dados atualizado mas não é claro no site quanto a fonte das informações, gerando relatórios para o cliente também.

O Lumina ERP não possui preço explícito, sendo necessário solicitar uma proposta junto ao site (Lumina ERP, 2022).

Figura 4: Software Lumina



Fonte: (Lumina, 2022)

2.1.4 BANCO DE DADOS DOS SISTEMAS DE ORÇAMENTO

Todas as soluções digitais de orçamento disponíveis para a área de Engenharia Elétrica trabalham com bancos de dados disponibilizados gratuitamente de maneira oficial, com a escolha a depender do projeto em questão. Um desses bancos é o SINAPI, Sistema Nacional de Pesquisa de Custos e Índices da Construção Civil, mantido pelo governo brasileiro. Esse sistema realiza pesquisas periódicas sobre os custos de materiais, mão de obra, equipamentos e serviços utilizados em projetos de construção civil em todo o país. Com base nessas pesquisas, o Sinapi disponibiliza informações atualizadas e confiáveis sobre os custos e índices de diversos materiais e serviços de construção, que são utilizados como referência para a definição de preços e orçamentos de obras em todo o país. O SINAPI é amplamente utilizado por empresas, órgãos públicos, como o Governo do Estado do Amazonas, e profissionais da construção civil como uma importante fonte de informações para a realização de estudos de viabilidade, elaboração de projetos e execução de obras.

Figura 5: SINAPI

Fonte: (Caixa, 2023)

2.2. PYTHON

Lançado por Guido van Rossum em 1991, Python é uma linguagem de programação de alto nível, com modelo de desenvolvimento comunitário. A linguagem foi projetada a fim de enfatizar a importância do esforço do desenvolvedor sobre o esforço da máquina, isto é, computacional, fazendo combinação de uma sintaxe concisa e clara com os recursos de sua rica biblioteca padrão e módulos e frameworks desenvolvidos por terceiros.

A linguagem é a preferida dentre os cientistas de dados, uma vez que vários recursos são necessários para entrar na área, o que torna uma linguagem objetiva e eficaz um grande atrativo na hora de optar por uma (CEDROTECH, 2019). Python possui uma enorme comunidade, em constante crescimento, que discute e compartilha experiências com a linguagem em diversos portais na internet. Ainda, a depuração é mais simples uma vez que poucas linhas de códigos são exigidas para execução de comentários, aumentando produtividade na criação e manutenção dos roteiros de programação.

Figura 6: Python

Fonte: (Coodesh, 2021)

A linguagem possui uma biblioteca, Pandas, que fornece estruturas de dados expressivas, flexíveis e eficientes, projetadas para facilitar o trabalho com dados relacionais (COMMUNITY, 2019). É uma base importante para interagir com arquivos como excel sem necessidade de usar a aplicação nativa.

2.2.1. Pandas

A biblioteca Pandas é uma biblioteca popular de análise de dados em Python, projetada para trabalhar com dados em formato de tabela ou planilha. Ela oferece uma ampla variedade de ferramentas para ler, manipular, analisar e visualizar dados, tornando-se uma ferramenta valiosa para cientistas de dados e analistas de dados.

Algumas das funcionalidades oferecidas pelo Pandas incluem:

- Leitura e gravação de arquivos em diversos formatos, incluindo CSV, Excel, SQL, JSON, HTML e muitos outros;
- Manipulação de dados como filtragem, ordenação, agrupamento e fusão;
- Tratamento de dados faltantes ou ausentes, como preenchimento com valores padrão ou interpolação;
- Visualização de dados em gráficos e tabelas, usando biblioteca de visualização de dados Matplotlib.

A biblioteca Pandas é altamente otimizada e pode trabalhar com grandes conjuntos de dados de forma rápida e eficiente. Além disso, é uma biblioteca de código aberto e gratuita,

com uma grande comunidade de desenvolvedores e usuários que contribuem para o seu desenvolvimento e melhoria contínuos.

2.2.2. PySimpleGui

PySimpleGUI é uma biblioteca de interface gráfica do usuário (GUI) para Python, que permite criar interfaces gráficas de forma simples e rápida, sem a necessidade de aprender uma nova linguagem de marcação, como HTML ou XML. A biblioteca é projetada para ser fácil de usar e intuitiva, com uma sintaxe simples e legível, permitindo que desenvolvedores iniciantes possam criar interfaces gráficas de usuário facilmente.

Figura 7: Aplicação de interface

Fonte: (Youtube, 2020)

A biblioteca é baseada na biblioteca de interface gráfica do usuário Qt, o que o torna altamente portátil e suporta várias plataformas, incluindo Windows, Linux e macOS. A biblioteca inclui uma ampla variedade de widgets gráficos, como botões, campos de entrada de texto, caixas de seleção, barras de progresso, entre outros, tornando possível criar interfaces gráficas para uma ampla gama de aplicações.

2.3. WEB SCRAPING

Dados são gerados constantemente na Internet, com diferentes fontes e formatos de diferentes informações para o usuário. Nesse contexto, a necessidade da utilização de robôs se faz cada vez mais presente para acesso a grandes volumes de dados simultâneos. A definição de Web Scraping pode ser:

> O processo de extração e combinação de conteúdos de interesse da Web de uma forma sistemática. Em tal processo, um agente de software, também conhecido como robô, imita a interação de navegação entre os servidores da Web e o humano. Passo a passo, o robô acessa os sites conforme necessário, analisa seu conteúdo para encontrar e extrair dados de interesse e estruturar esses conteúdos conforme desejado (LOURENÇO, 2013).

Figura 8: Web Scraping

Fonte: (Toptal, 2020)

Dentro do conceito de web scraping existe o *Crawler*, robôs rastreadores que cumprem a função de realizar varredura em sites ou em bancos de dados digitais. O uso do método contempla o projeto com aumento de produtividade, otimização de recursos, redução de custos operacionais, aprimoramento da inteligência do negócio, dentre outros benefícios, sendo a pesquisa de preços uma aplicação direta de crawler (CRAWLY, 2022).

2.3.1. PRINCIPAIS BIBLIOTECAS

Sendo a preterida para este projeto, a linguagem de programação mais utilizada para desenvolver softwares de Web Scraping é o Python, seja pela ampla disponibilidade de bibliotecas, tendo recursos poderosos como Scrapy e Selenium, ou pelos recursos nativos que permitem interação com arquivos de maneira simplificada de integração.

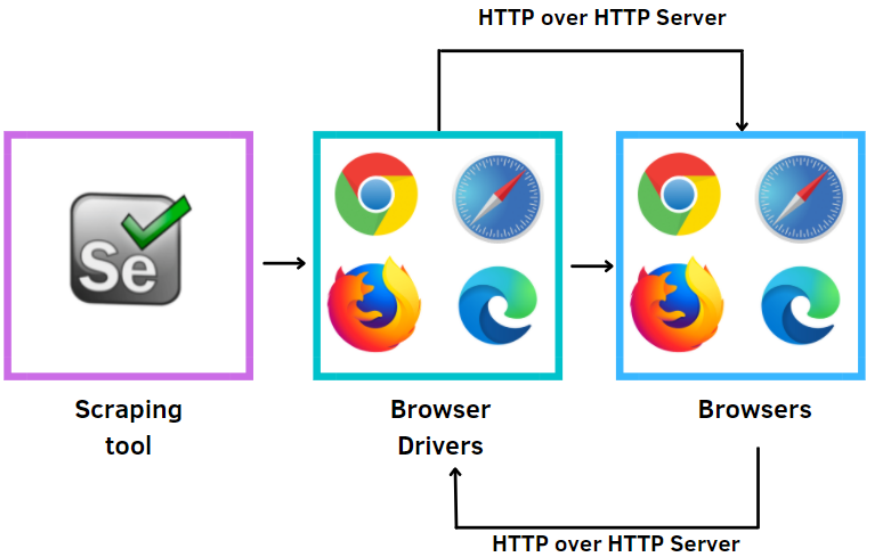
2.3.1.1. Scrapy

A biblioteca funciona utilizando classes que definem como um site ou grupo terá suas informações extraídas, denominadas *spiders*, possibilitando usar as requisições de maneira assíncrona, tratando erros sem parar a execução, sendo bem efetiva, porém não é capaz de capturar elementos renderizados por Javascript, tendo a necessidade do uso de ferramentas de apoio para tal.

2.3.1.2 Selenium

É um conjunto de ferramentas de código aberto com a finalidade de testar aplicações web pelo navegador de forma automatizada. Ele utiliza o próprio driver do navegador (seja ele Chrome, Firefox, Explorer) para permitir a interação entre o roteiro de teste e o respectivo navegador em tempo real. Essa ferramenta permite a manipulação do browser como, maximizar e minimizar tela, eventos de clique, navegação entre diferentes abas, comandos de *back* e *forward*, dentre outros. O Selenium é a ferramenta preterida para o desenvolvimento deste projeto, sem a necessidade de itens de apoio para o funcionamento da automação.

Figura 9 - Web Scraping usando Selenium



Fonte: (Analytics Vidhya, 2021)

De acordo com um relatório divulgado pela empresa Net Applications, feito através do Google Analytics, programa de análise de dados do Google, o Google Chrome foi o navegador mais utilizado no ano de 2021, com 66,64% do uso dos navegadores em geral, seguido pelo Safari em segundo lugar com 13,92%. Dessa forma, o Google Chrome será o navegador modelo para desenvolvimento da ferramenta deste projeto, bem como seu webdriver.

2.4. BANCO DE DADOS

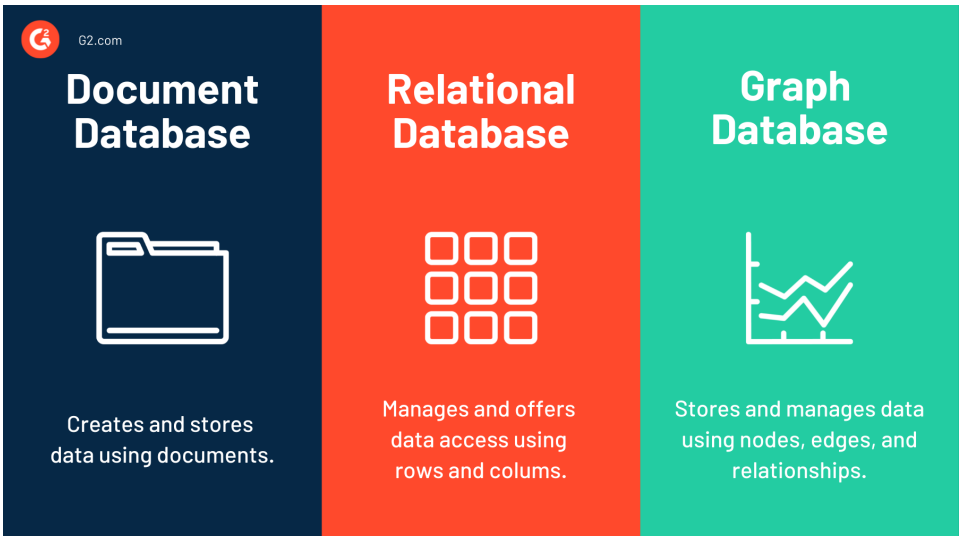
Uma coleção organizada de informações ou dados estruturados pode ser definida como banco de dados, normalmente armazenada em um sistema de computador eletronicamente. Geralmente o banco de dados é controlado por um sistema de gerenciamento de banco de dados (DBMS), compondo então um sistema de banco de dados (ORACLE, 2022).

Os dados normalmente são modelados em linhas e colunas em uma série de tabelas para garantir eficiência em processamento e consulta de dados. Diferentemente de uma

planilha comum, bancos de dados são projetados para conter coleções grandes de informações organizadas, permitindo acesso de múltiplo de usuários com rapidez.

O tipo de banco de dados preterido para este projeto é o de documentos, projetado para armazenar, recuperar e gerenciar informações orientadas a documentos ao invés de linhas e colunas, de maneira a aproveitar o material disponibilizado via internet (G2, 2022). A otimização do acesso e da taxa de transferência aos dados é crítica para os softwares que necessitam de rastreio de dados, o que adiciona ao projeto também conceitos de banco de dados autônomos, capaz de fazer manutenção no próprio escopo com machine learning, automatizando processos manuais e demorados.

Figura 10: Tipos de banco de dados



Fonte: (G2, 2022)

2.4.1. Banco de Dados Relacional

Um banco de dados relacional é um tipo de banco de dados que armazena informações em tabelas compostas por linhas e colunas. Cada tabela é composta por registros, onde cada registro representa uma entrada de dados específica e as colunas da tabela representam os atributos desses dados. Os bancos de dados relacionais são usados em uma ampla gama de aplicativos, incluindo aplicativos de comércio eletrônico, gerenciamento de estoque, sistemas de gerenciamento de conteúdo e muitos outros. Eles são projetados para suportar transações

em grande escala, garantir a integridade dos dados e permitir a recuperação rápida de informações.

2.4.2. Banco de Dados não Relacional (Documento)

Um banco de dados de documento é um tipo de banco de dados não relacional (NoSQL) que armazena informações em documentos semiestruturados, geralmente em formato JSON (JavaScript Object Notation), BSON (Binary JSON) ou XML (Extensible Markup Language). Nesse tipo de banco de dados, as informações são armazenadas em um documento, que é uma unidade básica de armazenamento e pode ser organizado em coleções, semelhante a uma tabela em um banco de dados relacional.

Os bancos de dados de documento são usados para aplicativos da web modernos que precisam de alta escalabilidade, flexibilidade e desempenho, como aplicativos de comércio eletrônico, mídia social e análise de dados. Eles são frequentemente usados em conjunto com outras tecnologias, como sistemas de cache, servidores web e ferramentas de análise de dados para fornecer um sistema completo e eficiente para armazenamento e recuperação de dados.

2.5. TESTES DE SOFTWARE

Teste de software é o processo de avaliação do software com o objetivo de identificar defeitos, falhas e erros no sistema em desenvolvimento. O teste envolve a execução do software em um ambiente controlado e a comparação dos resultados obtidos com os resultados esperados, de acordo com as especificações e requisitos definidos. O objetivo do teste de software é garantir que o software esteja livre de defeitos, atenda aos requisitos do cliente, seja confiável, seguro, fácil de usar e de manter. Existem diferentes tipos de testes de software, como testes unitários, testes de integração, testes de sistema, testes de aceitação, entre outros, cada um com sua finalidade e abordagem específicas. O teste de software é uma etapa fundamental do ciclo de desenvolvimento de software e é essencial para garantir a qualidade do produto final.

2.5.1. Teste Unitário

Um teste unitário é um tipo de teste de software que tem como objetivo verificar a funcionalidade de uma unidade de código (geralmente uma função ou método) de forma isolada, ou seja, sem a dependência de outras unidades de código. Em outras palavras, um teste unitário é uma técnica de teste que verifica se uma unidade de código produz o resultado esperado para uma entrada específica.

Eles ajudam a detectar erros de programação em um estágio inicial do processo de desenvolvimento de software e podem ajudar a prevenir a propagação de erros para outras partes do sistema. Além disso, os testes unitários ajudam a documentar a funcionalidade do código e facilitam a manutenção e evolução do software.

2.5.2. Teste de Integração

Testes de integração são testes de software que verificam se os diferentes módulos ou componentes de um sistema funcionam de forma integrada, ou seja, se as interfaces entre eles estão corretas e se eles interagem entre si de forma adequada. Os testes de integração são realizados após os testes unitários e antes dos testes de sistema, e têm como objetivo identificar falhas de integração entre os diferentes módulos do sistema.

Esses testes podem ser realizados de várias maneiras, incluindo testes de interface, testes de unidade combinada e testes de sistemas parciais. Os testes de integração são frequentemente realizados em etapas, começando com a integração de dois módulos e progredindo para a integração de todos os módulos do sistema.

2.5.3. Teste de Sistema

Testes de sistema são testes de software que verificam se um sistema ou aplicação inteira atende aos requisitos funcionais e não-funcionais definidos, ou seja, se o sistema se comporta conforme o esperado em relação aos seus requisitos de uso, desempenho, segurança, usabilidade, compatibilidade e outras características.

Os testes de sistema geralmente são realizados após os testes de integração e têm como objetivo avaliar o sistema como um todo, em um ambiente próximo ao real, simulando o uso do sistema pelo usuário final. Eles podem incluir testes de aceitação do usuário, testes

de estresse, testes de segurança, testes de desempenho, testes de usabilidade e outros tipos de testes.

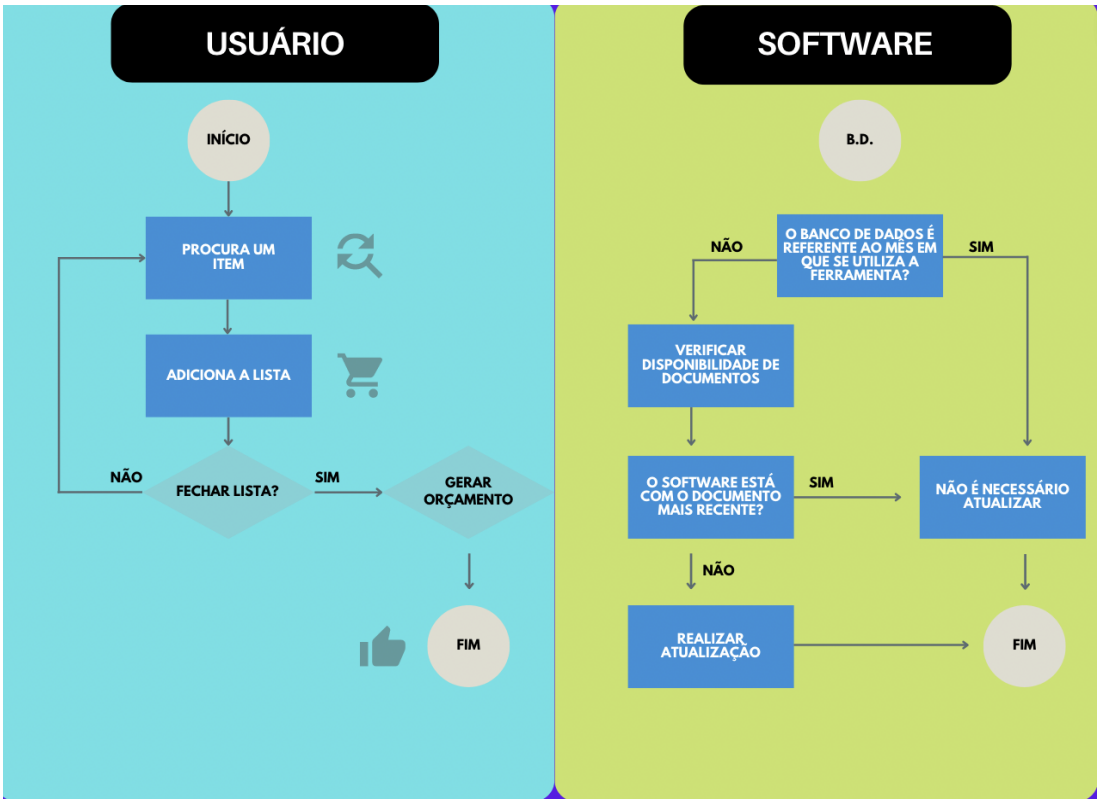
3. METODOLOGIA

O trabalho representa uma Pesquisa Aplicada, e tem como objetivo a realização de Pesquisa Exploratória sobre o material bibliográfico adquirido sobre o assunto. Está sendo utilizado o procedimento técnico de pesquisa bibliográfica. O método de abordagem utilizado é o hipotético-dedutivo e o método de procedimento de elaboração será o monográfico. Para coleta de dados será utilizada documentação indireta, com auxílio de documentos primários e secundários, e a análise e interpretação de seus dados será quantitativa.

Foram feitas pesquisas bibliográficas na área de coleta e processamento de dados, automação de navegadores de internet e desenvolvimento de software. As pesquisas foram realizadas para coletar dados sobre o estado da arte e as técnicas e soluções mais recentes, que sejam viáveis para elaboração do projeto. Ainda, utilizaram-se os dados para verificação de qualidade, eficiência, dentre outros critérios, uma vez que o sistema esteja finalizado.

O projeto foi desenvolvido utilizando o software PyCharm 2021.3.3, começando pela coleta dos dados de preços de materiais elétricos, disponibilizados pelo Sinapi no site da Caixa Econômica, utilizando Selenium para acessar a internet e manter um banco de dados local. O segundo passo foi tratar os dados obtidos a fim de obter apenas a informação desejada para o orçamento (nome, descrição, preço, data). A terceira etapa se baseia na criação de uma interface utilizando Python onde o usuário seja capaz de interagir com os itens desejáveis, seja para adicionar, remover, pesquisar ou multiplicar os itens sem necessidade de interação direta com a planilha. Com a interação completa, o quarto passo será a extração da planilha orçamentária do programa, utilizando a biblioteca Pandas de Python para a execução deste passo. O desenvolvimento do software chega às partes finais onde é necessário manter o banco de dados atualizado, criando uma rotina automatizada para renovação do arquivo de materiais disponibilizado na internet.

Figura 11: Diagrama do Software



Fonte: Próprio autor (2022)

O último procedimento é a etapa de testes. Para este projeto, o mesmo será validado com testes manuais em diferentes interações do usuário com o programa, a fim de encontrar possíveis erros, pontos de melhoria e verificar performance.

Figura 12: Etapa de Testes

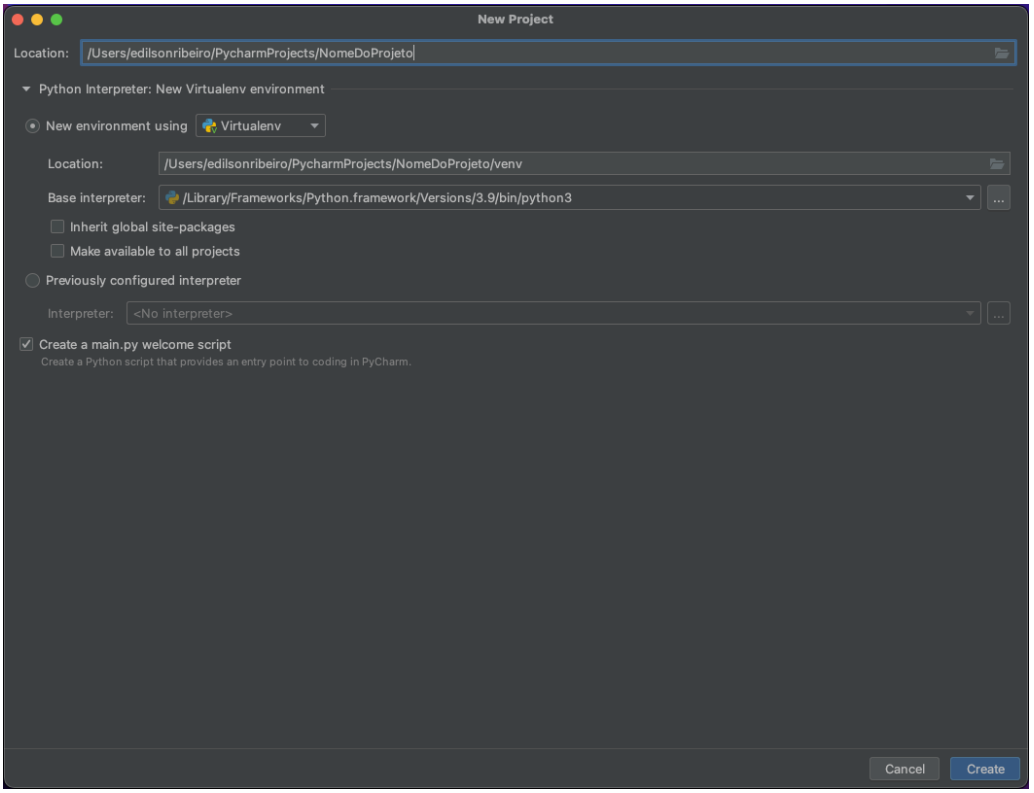
Fonte: (QAT Global, 2022)

3.1 Criação de projeto no PyCharm

Para desenvolver todas as etapas do projeto, é necessário iniciar um projeto através do PyCharm. Os procedimentos para isso é simples, o usuário deve, sequencialmente:

- Abrir o PyCharm;
- Clicar em "Create New Project";
- Selecionar o diretório do projeto;
- Escolher o nome do projeto;
- Apertar em "Create".

Figura 13: PyCharm



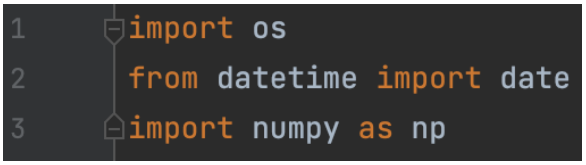
Fonte: Próprio Autor (2023)

Este documento utilizará o projeto "TCC_Edilson" como base de desenvolvimento do trabalho.

3.1.2. Preparação de ambiente

No decorrer do projeto, será necessário interagir com escopos além do já configurado no PyCharm, alguns já declarados como Pandas, Selenium, e outros que surgirão com a necessidade como date, numpy, os, dentre outros. A interação com eles só será concluída com a importação das bibliotecas com o ambiente em questão, comumente mantidos antes do início do programa a ser executado em si:

Figura 14: Importando Funções



Fonte: Próprio Autor (2023)

3.2 Criação e Manutenção do Banco de Dados Local

A aplicação terá desenvolvimento inicial baseado na criação do banco de dados, onde este depende de verificação no fornecedor dos dados: o Sinapi. O banco de dados será totalmente embasado no arquivo fornecido pelo site referido anteriormente, sendo então o foco da primeira etapa do desenvolvimento deste projeto, isto é, localizar e arquivar o material localmente, exatamente como ilustrado na figura 4. Para isso, utilizando o PyCharm, será criada uma função chamada verificar_atualizacao(), que seguirá uma estrutura simples.

3.2.1 Função para verificar atualização

A função receberá um argumento "datav", uma string responsável por salvar a referência da data presente combinada por mês e ano, no formato AAAAMM. A obtenção do argumento nesse formato se dá de maneira simples. Cria-se uma função que retorna dois valores, o mês atual e o ano atual:

Figura 15: Função para obter data

```
def obter_data():
    hoje = date.today()

    mes = hoje.strftime("%m")

    ano = hoje.strftime("%Y")

    return mes, ano
```

Fonte: Próprio Autor (2023)

A variável "datav" receberá os valores ano + mes, mantendo o formato desejado, através da biblioteda *date* e a função today() presente na mesma. Na sequência, verifica-se a referência anterior para comparação, onde se conclui se o mês em questão é o mês mais recente do banco de dados através de um arquivo txt chamado lastupdate, para fazer tomada de decisão: se for a mesma referência, o arquivo já está atualizado; se for uma referência mais antiga ou estiver nula em caso de nunca verificada, pode ter uma nova atualização (não necessariamente haverá) e não poderá ser uma referência mais nova pois a string datav utiliza sempre a data do dia da pesquisa. Será necessário duas funções distintas para isso: uma para criar e atualizar um arquivo txt e outra para ler e comparar a data atual com a data de referência, são as funções atualizar_arquivo() e verificar_atualizacao() respectivamente.

Figura 16: Funções visando atualização

```python
def verificar_atualizacao(datav):
    with open('/Users/edilsonribeiro/PycharmProjects/TCC_Edilson/lastupdate.txt', 'r') as f:

        linha = f.readline()

    if linha == datav:

        f.close()

        return False

    elif datav > linha:

        f.close()

        return True


def atualizar_arquivo(datav):
    with open('/Users/edilsonribeiro/PycharmProjects/TCC_Edilson/lastupdate.txt', 'w+') as f:
        f.write(datav)

        print("A nova referência é " + datav)

        f.close()
```

Fonte: Próprio Autor (2023)

Caso não haja resultados utilizando o parâmetro obtido por "datav", é feita uma varredura no mês anterior, até encontrar uma possibilidade de atualização (caso o banco de dados esteja nulo). Se não for nulo, a varredura será feita até o valor do registro no banco de dados, isto é: se o valor da data atual for maior que a do banco de dados, verifica se há uma atualização mais recente; se for o mesmo valor, já possui a versão mais atualizada.Por fim, não havendo possibilidade de nova atualização, a função retorna False, um valor booleano indicando uma resposta afirmativa, e retorna True em caso de uma resposta positiva sobre a análise.

3.2.2 Função para atualizar arquivo

Quando houver necessidade de atualizar ou verificar a atualização do arquivo, utilizaremos o Selenium para verificar junto ao Sinapi a presença de um material mais recente que o já presente no banco de dados. Dessa forma, dois processos diferentes são feitos: o de verificar no site e o de atualizar a nova referência, se existir, no arquivo lastupdate.

3.2.2.1 Verificação com Selenium

Essa etapa ainda seguirá o diagrama da figura 4 à risca, baseado no lado do software. Para isso, utilizaremos duas funções complementares em Python: "try"e "except", o primeiro para fazer uma tentativa (baixar um arquivo mais recente que a referência) e o segundo para seguir adiante caso não seja possível. As tentativas serão bem sucedidas ou não, baseada em se a nossa biblioteca do Selenium será capaz de encontrar os arquivos disponíveis no site.

Haverá então uma sequência de fatores a serem executados:

- Iniciar o driver do Selenium para o navegador a ser utilizado (neste caso, o Chrome Driver referente ao Google Chrome);

Figura 17: Iniciando driver

```
driver = webdriver.Chrome('/Users/edilsonribeiro/PycharmProjects/TCC_Edilson/chromedriver', options=chrome_options)
```

Fonte: Próprio Autor (2023)

- Com o driver inicializado, abrir a página da Caixa Econômica referente aos arquivos do Sinapi;

Figura 18: Abrindo página desejada

```
driver.get("https://www.caixa.gov.br/site/Paginas/downloads.aspx#categoria_640")
```

Fonte: Próprio Autor (2023)

- Aqui se inicia o try: através de funções da biblioteca do Selenium, verificar a presença do item em questão, isto é, arquivos mais atualizados. Os arquivos do Sinapi seguem uma sequência lógica mês por mês, no formato SINAPI_ref_Insumos_Composições_AM + data + Desonerado, sendo a data uma variável que será modificada a cada tentativa de procura do arquivo no site, até que ache um material mais recente ou que a data seja a mesma do último material atualizado. Sendo a mais recente, o arquivo txt lastupdate é atualizado com a versão mais recente;

Figura 19: Try

```
try:

    arquivo = WebDriverWait(driver, 30).until(
        EC.presence_of_element_located((
            By.PARTIAL_LINK_TEXT, "SINAPI_ref_Insumos_Composicoes_AM_"+data+"_Desonerado")))

    arquivo.click()

    print("Aguardando download")

    time.sleep(20)

    print("Download concluído com a versão mais recente")

    atualizar_arquivo(datav)
```

Fonte: Próprio Autor (2023)

- Caso não haja referência do mês da tentativa, ocorrerá o except: com auxílio de uma nova função chamada tentar_versao_anterior(), que utiliza referências do mês e do ano utilizados, a mesma retornará o mês anterior para uma nova tentativa no site;

Figura 20: Except

```
except:

    print('Ainda não há registros do mês ' + mes + ' '+ ano)

    mes, ano = tentar_versao_anterior(mes, ano)

    print('Verificando no mês anterior')

    datav = ano + mes

    data = mes + ano
```

Fonte: Próprio Autor (2023)

- Caso o mês anterior for o mesmo da referência, nosso script encerrará indicando a versão disponível como a mais recente. O script permanecerá em execução enquanto a função verificar_atualizacao() retornar True, o que não acontecerá mais se o nosso mês da tentativa for o mesmo da referência do txt;

Figura 21: Função de atualização retornando Falso

```
if verificar_atualizacao(datav) == False:
    print('A versão disponível é a mais recente')
    break
```

Fonte: Próprio Autor (2023)

- Por fim, finalizada a execução, é necessário fechar o driver do Selenium aberto no início do roteiro utilizando "driver.quit()".

A fim de ambientar o usuário, foi adotado o uso de prints durante cada tomada de decisão para haver um retorno de informações em tempo real durante a execução do script.

Figura 22: Mensagens de informação para o usuário

```
Procurando arquivo...
Ainda não há registros do mês 02 2023
Verificando no mês anterior
Ainda não há registros do mês 01 2023
Verificando no mês anterior
Aguardando download
Download concluído com a versão mais recente
A nova referência é 202212
A versão disponível é a mais recente
```

Fonte: Próprio Autor (2023)

A etapa final com o Selenium diz respeito ao diretório do arquivo a ser baixado: através de configurações customizadas, altera-se o endereço padrão da pasta de download do computador, visando concentrar o material do banco de dados em um único local, que será o mesmo diretório da pasta do projeto no PyCharm. As funções são pré existentes junto da biblioteca do Selenium, sendo necessário apenas interagir com as opções do navegador e adicionando a preferência na sequência:

- prefs = {'download.default_directory' : diretorio_desejado}
- webdriver.ChromeOptions().add_experimental_options('prefs', prefs).

3.3 Leitura do banco de dados

Dando prosseguimento, com o material do banco de dados devidamente armazenado, é possível montar o script para leitura e interação do arquivo. A principal biblioteca utilizada nesta parte do projeto é o Pandas, responsável por interação com arquivos Excel usando Python. A leitura se faz possível através de uma função da biblioteca:

- pandas.read_excel(nome_do_arquivo)

A partir da função, todo outro tipo de interação também é possível no arquivo selecionado. Atribuindo uma variável à leitura do arquivo através da função acima, foi possível desenvolver outras funções essenciais para a execução do projeto. Supondo a variável com o nome planilha, a função planilha.rename(columns={nome_antigo:nome_novo}) atribui novos nomes para as colunas da planilha. Para os dados do Sinapi, é importante nomear os tópicos ainda na primeira linha para poder referenciar cada coluna. Dessa forma, adiciona os títulos: Código, Descrição do Insumo, Unidade, Origem do Preço e Preço.

Por conta da formatação, algumas linhas precisaram ser excluídas para eliminar dados indesejados da planilha, para isso, a função planilha.drop(numero_linha) foi utilizada, utilizando como parâmetro o número da linha a ser removida.

3.3.1 Retornando valores da planilha

Facilitando a interação do usuário com a planilha, cria-se uma função responsável por retornar o nome, a unidade e o valor de um item através do seu id, número inteiro correspondente ao item na planilha. A função terá dois itens como parâmetros: o nome da planilha e o id do item. Utilizando a função planilha.values[linha][coluna] se tem como retorno o item das respectivas linhas e colunas na planilha, valores utilizados na função.

Figura 23: Função que retorna valores dos itens

```
def valores_itens(dataframe, id):
    nome = dataframe.values[id-6][1]
    unidade = dataframe.values[id-6][2]
    valor = dataframe.values[id-6][4]
```

Fonte: Próprio Autor (2023)

O valor negativo ao lado do id se dá por conta dos valores úteis da planilha não começarem no índice zero, tendo que ser feito um reajuste. Há também a possibilidade de pesquisa de um item dentro do arquivo, através da função:

- planilha.loc['Nome_da_coluna'].str.contaisn(nome_a_pesquisar)

A função retornará uma lista com os itens relacionados à pesquisa, junto do valor do id do item.

3.3.2. Gerando um orçamento com os itens desejados

Encontrado o item desejado, basta utilizar o id para obter seus valores. Todos os itens desejados serão armazenados em uma lista a parte da planilha já sendo utilizada, a fim de organizar uma lista apenas dos itens selecionados e gerar uma planilha a partir dela. Para isso, cada item gerado pelo id na função de values será incrementado em uma lista através da função lista.append(nome_do_item). É interessante ter um item para cada informação desejada para se ter em um orçamento, sendo as escolhidas deste projeto: nome do item, unidade, preço e quantidade. Em caso da necessidade de um reinício da lista, substitui-se as listas preenchidas por listas vazias novamente.

3.4. Exportação dos dados para o Excel

Com os itens definidos, é possível externar a lista para um arquivo excel com a função to_excel(nome_do_arquivo). Essa função irá imprimir os dados da lista na mesma disposição na planilha, então é necessário fazer ajustes como adição de título, bastando adicionar o nome no índice 0 de cada coluna da lista. Executada a função, o arquivo de nome a escolha do usuário será armazenado na pasta do projeto Python.

3.5. Desenvolvimento da Interface

A interface foi toda desenvolvida utilizando a biblioteca PySimpleGui, utilizada no desenvolvimento como "sg", junto das funções desenvolvidas ao longo deste item 7 do documento. A fim de facilitar a leitura visual do projeto, foram adicionados apenas itens considerados essenciais no layout do programa:

- Texto informando a versão do banco de dados presente no programa;
- Um botão para verificação de nova atualização

- Uma barra de pesquisa de itens acompanhada do botão "Pesquisar";
- Uma aba de escrita do item desejado acompanhado do botão "Ok";
- Uma aba de escrita de quantas unidades do item serão adicionados ao orçamento, junto de um botão de adição;
- Um botão para mostrar o orçamento atual, um botão para limpar o orçamento atual, uma aba para escrita do nome do arquivo do orçamento e um botão para gerar um arquivo Excel do orçamento;
- Um botão para encerrar a aplicação.

Esses itens se estruturam dentro de uma lista chamada layout, interagindo através das funções sg.Text(texto_a_ser_escrito), sg.Button(texto_a_ser_ilustrado_no_botao) e sg.InputText(texto_a_ser_ilustrado_no_campo_de_escrita). Após a definição dos elementos presentes na aplicação, cria-se a janela da aplicação com todas as informações acima:

- window = sd.Window(nome_da_janela, layout)

A seguir, é necessário criar algumas interações com o aplicativo, uma vez que os botões precisam realizar funções. Alguns passos são necessários para esta operação funcionar, sendo o primeiro a criação de um loop, isto é, uma ação em repetição intermitente, de modo a fazer o programa ficar em operação sem interrupções, o que pode ser facilmente resolvido com a função "while (ação)", ajustamos a ação para ser sempre verdadeira, em python, True. Porém, o usuário também necessita poder encerrar a aplicação, a fim de evitar o consumo desnecessário do processamento do dispositivo que estiver executando a aplicação. O botão já está definido, porém o mesmo necessita de um evento após a interação com o botão. A expressão a seguir supre tal necessidade:

- event, values = window.read()

As duas variáveis são responsáveis por dois pontos importantes do programa: "event" recebe as interações a partir do nome dos botões (isto é, se pressionado o botão "Encerrar", então a variável "event" adquire o valor "Encerrar"), enquanto values armazena todos os campos de preenchimentos de dados, sendo estes dinâmicos e enumerados em ordem a partir do índice zero como uma lista.

Cria-se então um roteiro condicional, onde se a variável "event"por igual a "Encerrar", então a função "while"deve receber "break", responsável por parar o loop.

A partir dessa configuração, é possível configurar todos os botões da aplicação:

Figura 24: Ações dos botões.

Fonte: Próprio autor, 2023.

3.6. Aplicação de Testes no Software

Os testes da aplicação estão concentrados em dois itens principais: as funções e a interface. Há uma diferença básica entre eles: as funções executam atividades previamente desenvolvidas dentro dos parâmetros estabelecidos. Dessa forma, uma função responsável por gerar um excel do orçamento não pode haver erros durante sua execução, o que faz com que seus possíveis impedimentos sejam tratados dentro do código fonte, sem interferência do usuário. Por outro lado, a maioria das falhas de interface estão relacionadas à maneira que o usuário utiliza a mesma, principalmente em casos de escrita de entrada de dados. Dos casos

desenvolvidos durante este Trabalho de Conclusão de Curso, é possível pontuar situações cruciais a fim de não haver problemas quando o usuário executar o programa:

i) Durante a busca de atualizações, é necessário considerar que não haverá registros de documentos no mês procurado, principalmente nos dias iniciais do mês em vigência;

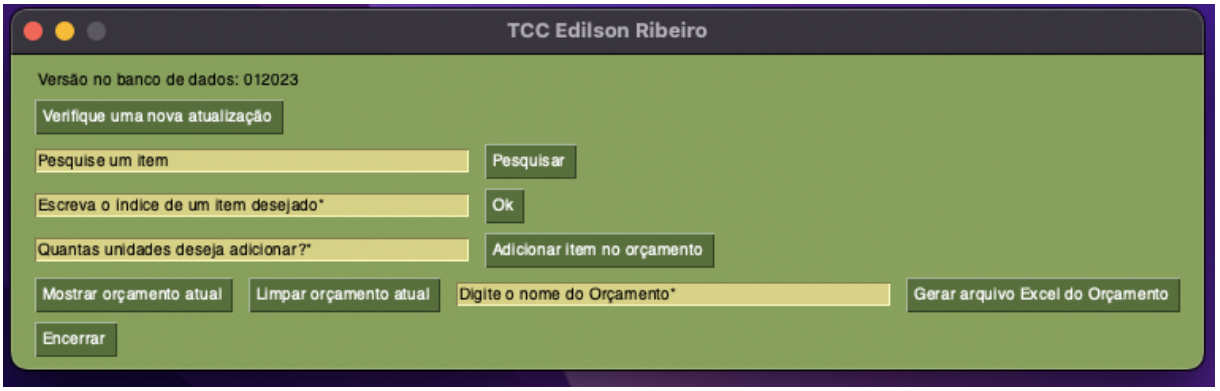
ii) Não será possível ler o arquivo se o banco de dados estiver sem material. É necessário criar uma condição onde sempre o programa procurará automaticamente um novo material caso seu banco de dados estiver nulo;

iii) Nos campos de pesquisa, é necessário delimitar ao usuário o tipo de dados: Não será possível deixar nulo o campo de nome do orçamento, haverá necessidade da pesquisa do item ser feito apenas com números inteiros (através do índice), não será possível adicionar um item ao orçamento se não houver indicação de quantidade, dentre outros a partir das escolhas do desenvolvedor.

4. RESULTADOS E DISCUSSÕES

No fim do desenvolvimento do projeto, há diversas variações a partir da escolha do desenvolvedor, seja por design, quantidade de botões, escolha de material para banco de dados, métodos de coletas, dentre outros. No desenvolvimento do autor, que seguiu exatamente como na instrução deste documento, pode-se obter o seguinte:

Figura 25: O aplicativo.

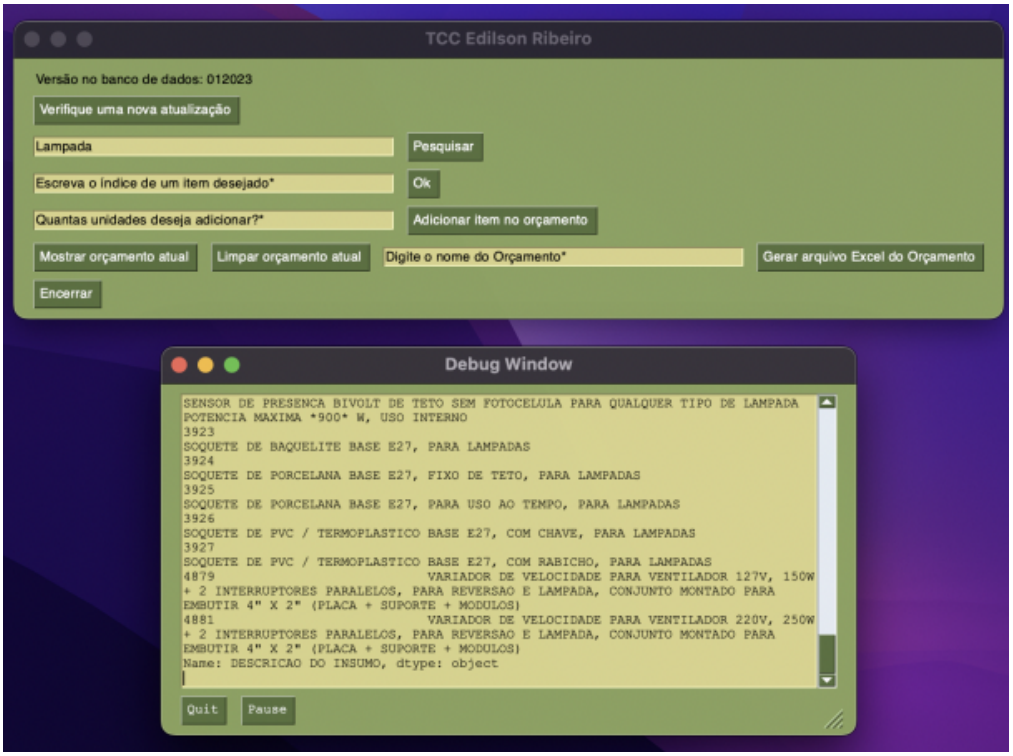


Fonte: Próprio autor, 2023.

O layout foi propositalmente desenvolvido para ser simples e objetivo, além de resultar em um arquivo leve e material prático para o usuário. Ao clicar em "Verifique uma

nova atualização", o programa se encarrega de fazer uma varredura no site do Sinapi em busca de uma versão de documento mais recente que a do mês 01 do ano 2023, como informa a primeira linha do aplicativo. Na sequência, há possibilidade de procura de um item:

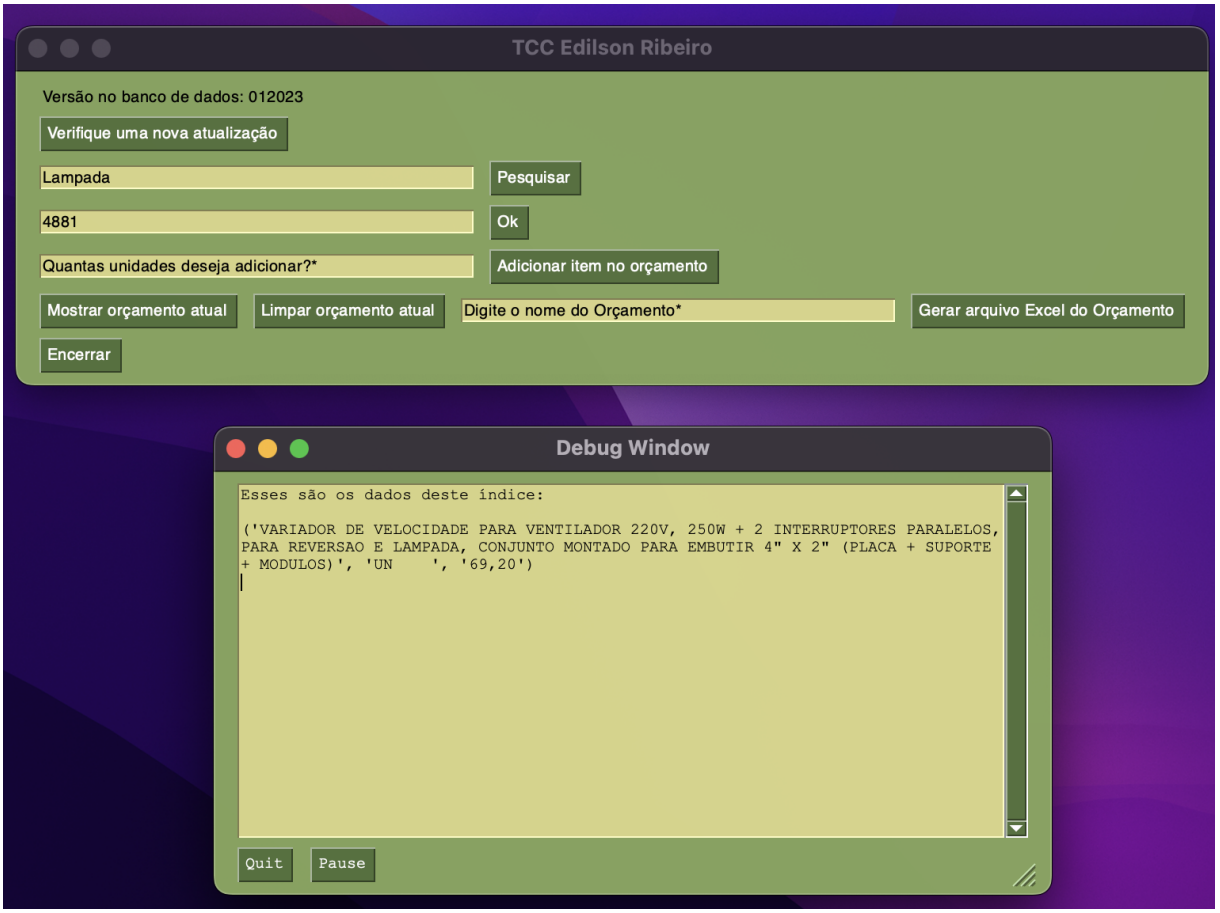
Figura 26: Pesquisa de itens.



Fonte: Próprio autor, 2023.

Ao procurar por "Lâmpada", uma série de itens é divulgada em uma nova janela, com o seu nome completo e o número do índice referente àquele item. Se o usuário optar pelo último item, ele pode fechar a janela da busca, adicionar o índice do item na segunda barra do programa e buscar mais informações sobre o mesmo:

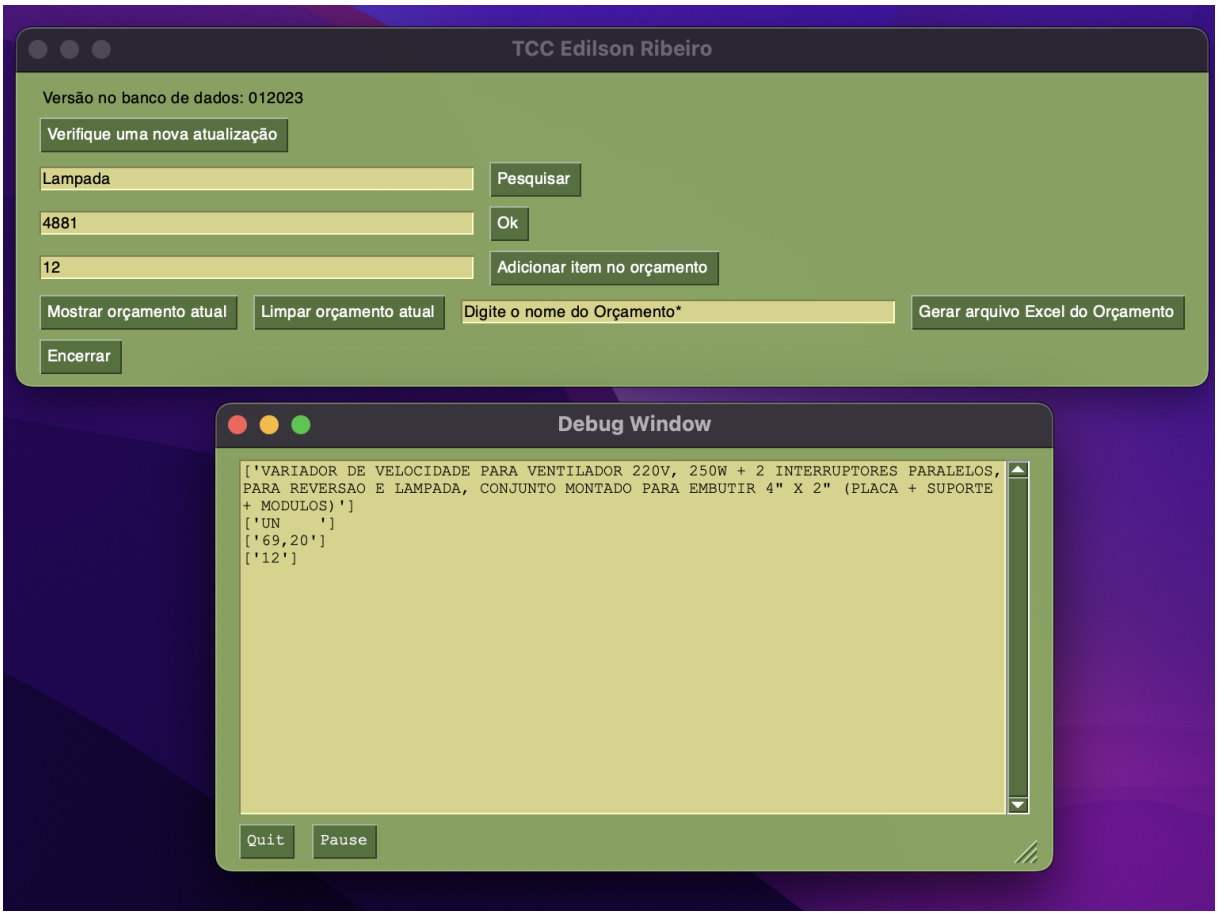
Figura 27: Busca através do índice.



Fonte: Próprio autor, 2023.

A pesquisa retorna dados completos do item: descrição, tipo de unidade e preço. Sendo um item a ser levado para o orçamento, adiciona-se a quantidade de unidades desejadas no terceiro campo de preenchimento e pressiona-se "Adicionar item ao orçamento". Essa ação se repete até o fim da montagem do orçamento. Caso haja necessidade de verificação dos itens até então adicionados, há possibilidade de visualizar previamente o orçamento, pressionando "Mostrar orçamento atual":

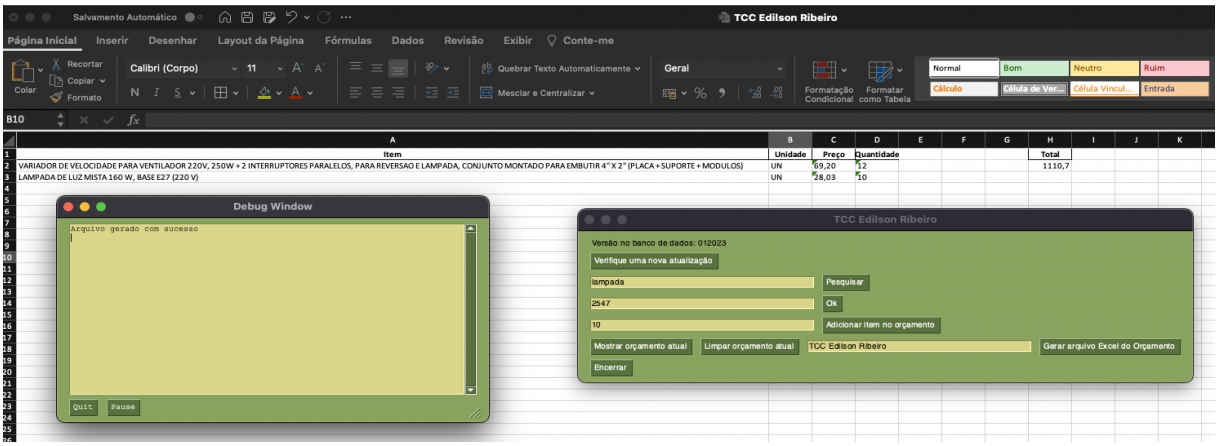
Figura 28: Visualização do orçamento.



Fonte: Próprio autor, 2023.

No fim do processo, para gerar o arquivo Excel é necessário adicionar um nome ao orçamento e pressionar o botão ao lado do campo, resultando então em um arquivo salvo dentro da pasta do projeto desenvolvido:

Figura 29: Orçamento em Excel.

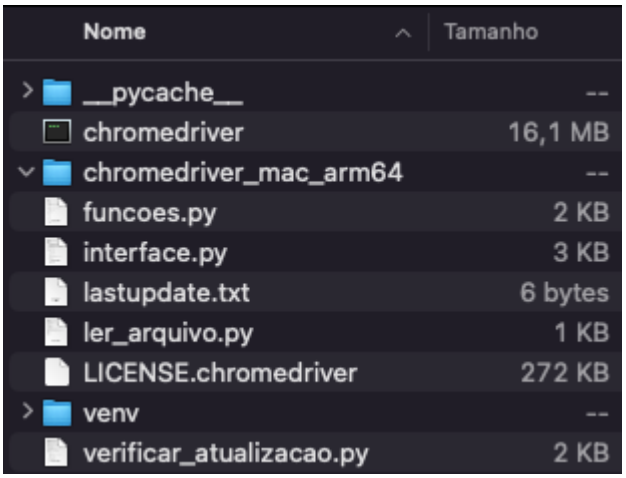


Fonte: Próprio autor, 2023.

Todos os arquivos gerados e utilizados para a execução desse projeto estão disponíveis na pasta do escritor desse trabalho através do GitHub, disponível nas referências bibliográficas.

A nível de complexidade de execução, os arquivos python gerados no script somam 20KB no disco rígido do computador, tamanho que não necessitará de um desempenho além de básico para sua execução. O maior somatório de tamanho vem do Chrome Driver com 16,1MB, sendo acionado apenas em caso de necessidade de atualização, ainda assim sendo considerado um tamanho pequeno para um software com esse potencial.

Figura 30: Arquivos criados no final do projeto.



Fonte: Próprio autor, 2023.

A função de um engenheiro pode variar dependendo da área em que ele atua, mas utilizando atribuição generalizada, um engenheiro é responsável por aplicar conhecimentos matemáticos, científicos e tecnológicos para projetar, construir e manter sistemas, produtos ou processos que atendam a uma necessidade específica.

O projeto de um sistema necessita também de um levantamento de material, seja na área de engenharia elétrica, civil, naval, mecânica, química etc. O projeto desenvolvido durante esse Trabalho de Conclusão de Curso foca em materiais elétricos, porém a discordância entre esse projeto e um com foco em materiais de engenharia naval, por exemplo, se passa apenas pelo tipo de banco de dados a ser armazenado e trabalhado.

O Sinapi, por exemplo, não é especializado apenas em engenharia elétrica, sendo também a principal fonte de parâmetro de dados de preço da engenharia civil. Dessa forma, o mesmo projeto desenvolvido pode também ser utilizado para itens de construção civil. Há

também sistemas de preços desenvolvidos para outros setores, como é o caso do SICRO, Sistema de Custos Referenciais de Obras, criado pelo Departamento Nacional de Infraestrutura de Transportes (DNIT) como referência de preços de insumos e composições de infraestruturas de transportes, ou também da tabela SEINFRA, Secretaria de Estado de Infraestrutura, responsável por unificar e padronizar os custos unitários de serviços de edificações, saneamentos, rodovias, portos e ferrovias, englobando as engenharias elétrica, civil, mecânica, naval, materiais e produção.

O projeto consiste em um modelo em cima da leitura de um arquivo, sendo facilmente expansível para as outras engenharias, bastando seguir o método presente neste documento e trabalhando em cima do banco de dados mais apropriado.

5. CONSIDERAÇÕES FINAIS

Neste Trabalho de Conclusão de Curso foi desenvolvida uma aplicação Python para montagem de orçamentos de materiais elétricos dentro do ramo da construção em baixa potência. Entre engenheiros e alunos de engenharia elétrica, antes eram poucas ou nulas as opções de software capazes de consultar um banco de dados atualizado e gerar um orçamento de maneira eficiente e gratuita, gerando trabalho manual em excesso ou custos muito acima do ideal. Dentro desta problemática, a solução desenvolvida trouxe resultados significativos: um programa capaz de realizar todas as funções propostas com visual simples, leitura rápida e de armazenamento em tamanho bem abaixo de programas considerados leves.

Dessa forma, a hipótese inicial de que é possível desenvolver um software para geração de orçamento de materiais elétricos, consultando, de maneira frequente e automatizada utilizando Python, o Sinapi, Sistema Nacional de Pesquisa de Custos e Índices da Construção Civil, e gerando um banco de dados local para fazer a leitura dos dados e gerar uma interface, facilitando a interação entre o usuário e os materiais selecionados por ele e produzindo um orçamento em excel com todo o relatório de custos já providenciado utilizando Pandas, é considerada verdadeira. O desenvolvimento finalizado da aplicação registrado nos resultados demonstra o alcance do objetivo previamente estabelecido.

Ainda, foi possível observar que o projeto também traz resultados relevantes não só para a engenharia elétrica, mas também para as outras engenharias oferecidas pela Universidade do Estados do Amazonas que necessitem de montagem de orçamentos. A adaptação não é imediata, uma vez que um estudo diferente seria feito para outros bancos de

dados. Entretanto, os ajustes não representam 20% do projeto final, o que deixa aberta a possibilidade de desenvolvimento futuro de ferramentas capazes de gerar orçamentos que auxiliem todos os campos da Escola Superior de Tecnologia utilizando a maior parte da base do projeto desenvolvido neste trabalho.

REFERÊNCIAS

CNN BRASIL. **Brasil deve ter déficit de trabalhadores qualificados até 2023 aponta pesquisa.** 2021. Disponível em: <https://www.cnnbrasil.com.br/business/brasil-deve-ter-deficit-de-trabalhadores-qualificados-ate-2023-aponta-pesquisa/> Acesso em: 02 ago. 2022.

BHARGAVA, Aditya. **Entendendo Algoritmos: Um Guia Ilustrado Para Programadores e Outros Curiosos**. MANNING, 2017.

LOURENÇO, A. **Web scraping technologies in an API world**. Briefings in Bioinformatics, v.15, n.5, 04 2013, p. 788–797. ISSN 1467-5463.

COMMUNITY, T. P. **Pandas Documentation**. 2019. Disponível em: <https://numpy.org/doc>. Acesso em: 29 ago. 2022.

ORACLE. **O que é um Banco de Dados?** 2022. Disponível em: <https://www.oracle.com/br/database/what-is-database/> Acesso em: 13 set. 2022.

CRAWLY. **O que é Crawly e como funcionam os robôs para coleta de dados.** 2022. Disponível em: <https://www.crawly.com.br/blog/o-que-e-crawler-robos-para-coleta-de-dados> Acesso em: 13 set. 2022.

G2. **What is a Document Database?** 2022. Disponível em: <https://www.g2.com/articles/document-databases> Acesso em 14 set. 2022.

CEDROTECH. **Python é a principal linguagem na ciência de dados?** 2019. Disponível em: <https://blog.cedrotech.com/python-e-a-principal-linguagem-na-ciencia-de-dados> Acesso em 14 set. 2022.

ORÇAFASCIO. **Faça orçamento de obras até 8x mais rápido com a OrçaFascio.** 2021. Disponível em: <https://www.orcafascio.com/> Acesso em 24 set. 2022.

ALTO QI. **Visus**. 2022. Disponível em : <https://altoqi.com.br/visus/> Acesso em 24 set. 2022.

LUMINA ERP. **O software específico da construção civil.** 2022. Disponível em: <https://www.luminaerp.com/> Acesso em 24 set. 2022.

TOPTAL. **Web Scraping Using Python Selenium**. 2020. Disponível em: <https://www.toptal.com/python/web-scraping-with-python> Acesso em 25 set. 2022.

ANALYTICS VIDHYA. **Guide to Web Scraping**. 2021. Disponível em: <https://www.analyticsvidhya.com/blog/2021/08/an-intuitive-guide-to-web-scraping-using-selenium/> Acesso em 25 set. 2022.

ESDS. **Testes e Qualidade de Software**. 2021. Disponível em: <https://www.qat.com/testes-e-qualidade-de-software/> Acesso em 28 fev. 2023.

PSGUI. **PySimpleGUI**. 2021. Disponível em: <https://www.pysimplegui.org/> Acesso em 28 fev. 2023.

CAIXA. **SINAPI - Sistema Nacional de Pesquisa de Custos e Índices da Construção Civil.** 2013. Disponível em: <https://www.caixa.gov.br/poder-publico/modernizacao-gestao/sinapi/Paginas/default.aspx>

COODESH. **Python: Linguagem do Ano**. 2021. Disponível em: <https://coodesh.com/blog/candidates/backend/python-linguagem-do-ano-veja-porque-ela-esta-em-alta/>

YOUTUBE. **Como criar uma tela em Python com PySimpleGUI**. 2020. Disponível em: <https://www.youtube.com/watch?v=UnfmxnFpfdM>

GITHUB. **Gerador de Orçamento de Materiais Elétricos**. 2023. Disponível em: <https://github.com/edilson-ribeiro/Gerador-de-Orcamento-de-Materiais-Eletricos>