Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Departamento de Engenharia Elétrica

Alexsandra Macedo Souto

Relatório de Estágio Supervisionado

Campina Grande, Paraíba, Brasil
9 de outubro de 2024

Alexsandra Macedo Souto

Relatório de Estágio Supervisionado

Relatório de Estágio Supervisionado submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Gutemberg Gonçalves dos Santos Júnior, Dr.

Campina Grande, Paraíba, Brasil
9 de outubro de 2024

Alexsandra Macedo Souto

Relatório de Estágio Supervisionado

Relatório de Estágio Supervisionado submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovado em: 07/10/24

Prof. Gutemberg Gonçalves dos Santos Júnior, Dr.
Orientador

Prof. Danilo Freire de Souza Santos, Dr.
Avaliador

Campina Grande, Paraíba, Brasil
9 de outubro de 2024

Resumo

Este relatório apresenta as atividades desenvolvidas pela aluna de graduação em Engenharia Elétrica, Alexsandra Macedo Souto, durante o estágio supervisionado no Núcleo de PDI VIRTUS. O foco do trabalho foi o desenvolvimento de uma ferramenta web interativa, utilizando a linguagem de programação *Python*, as bibliotecas *Dash* e *Selenium*, e o banco de dados *MongoDB*. A ferramenta desenvolvida visa automatizar processos e facilitar a interação com sistemas web, integrando a biblioteca *Dash* para criar uma interface gráfica intuitiva e interativa. Além disso, a biblioteca *Selenium* foi empregada para automatizar tarefas no navegador, como o preenchimento de formulários em páginas web dinâmicas. O banco de dados *MongoDB*, por sua vez, foi utilizado para armazenar e gerenciar as informações coletadas, permitindo uma estrutura de dados flexível e escalável.

Palavras-chaves: Engenharia Elétrica, Estágio Supervisionado, *Python*, *Dash*.

Abstract

This report presents the activities carried out by the undergraduate Electrical Engineering student, Alexsandra Macedo Souto, during the supervised internship at the VIRTUS. The focus of the work was the development of an interactive web tool using the Python programming language, the Dash and Selenium libraries, and the MongoDB database.

The developed tool aims to automate processes and facilitate interaction with web systems, integrating the Dash library to create an intuitive and interactive graphical interface. Additionally, the Selenium library was employed to automate browser tasks, such as filling out forms on dynamic web pages. The MongoDB database, in turn, was used to store and manage the collected information, allowing for a flexible and scalable data structure.

Key-words: Electrical Engineering, Supervised Internship, Python, Dash.

Lista de abreviaturas e siglas

UAEE	Unidade Acadêmica de Engenharia Elétrica
CEEI	Centro de Engenharia Elétrica e Informática
UFCG	Universidade Federal de Campina Grande
PDI	Pesquisa, Desenvolvimento e Inovação
CITTA	Centro de Inovação e Tecnologia Telmo Araújo
ICT	Instituto de Ciência e Tecnologia

Lista de ilustrações

Sumário

1 Introdução

Este relatório descreve as atividades realizadas durante a execução da disciplina de Estágio Supervisionado pela aluna Alexsandra Macedo Souto, no curso de graduação em Engenharia Elétrica na Universidade Federal de Campina Grande (UFCG). O estágio ocorreu do dia 17 de junho a 13 de setembro de 2024, com uma carga horária semanal de 17 horas, totalizando 216 horas no VIRTUS (Núcleo de Pesquisa, Desenvolvimento e Inovação em Tecnologia da Informação, Comunicação e Automação da UFCG), sob orientação do professor Gutemberg Gonçalves dos Santos Júnior.

1.1 Objetivos

O Estágio Supervisionado é uma disciplina obrigatória integrante da grade curricular do curso de graduação em Engenharia Elétrica da Universidade Federal de Campina Grande. Seu principal objetivo é permitir que o aluno aplique os conhecimentos teóricos adquiridos ao longo de sua formação acadêmica nas demais disciplinas do curso. Além disso, proporciona um contato direto com o mercado de trabalho, permitindo adquirir conhecimentos essenciais para um profissional da área de tecnologia, contribuindo para uma formação abrangente.

Durante o período de estágio, a aluna integrou uma equipe de um projeto de Pesquisa, Desenvolvimento e Inovação (PDI) no VIRTUS, no qual atuou no processo de desenvolvimento de uma ferramenta web iterativa, que permita aos usuários visualizar, explorar e interagir com os dados de maneira simples e intuitiva.

Dessa forma, este relatório tem como principais objetivos documentar de maneira detalhada as experiências vivenciadas durante o período de estágio supervisionado na empresa parceira. Além de avaliar o desenvolvimento profissional do estagiário, destacando as habilidades e competências adquiridas, bem como identificar áreas que necessitam de aprimoramento. Ao mesmo tempo, serve como um registro formal das realizações e aprendizados obtidos, útil para futuras referências acadêmicas ou profissionais.

1.2 Estrutura do relatório

O presente relatório está organizado em capítulos para proporcionar uma visão abrangente de todo o processo desenvolvido, de forma que neste Capítulo 1, foram apresentadas uma introdução e objetivos do estágio, como também uma visão geral da organização deste trabalho. O Capítulo 2 contém a descrição do ambiente de trabalho no

qual o estágio foi realizado, incluindo um breve histórico e descrição de seus projetos. No Capítulo 3 será feita uma fundamentação teórica para melhor compreensão dos conceitos utilizados durante as atividades. O Capítulo 4 descreve as atividades desenvolvidas no período de estágio. Por fim, o Capítulo 5 finaliza este trabalho com as considerações finais.

2 Ambiente de Trabalho

Nesta Capítulo, será apresentado o VIRTUS, ambiente de desenvolvimento tecnológico e pesquisa na área de tecnologia da informação, onde o estágio supervisionado foi realizado.

2.1 VIRTUS

Fundado em 2014 na cidade de Campina Grande - PB, o VIRTUS é um Núcleo de Pesquisa, Desenvolvimento e Inovação em Tecnologia da Informação, Comunicação e Automação, um órgão suplementar da Universidade Federal de Campina Grande (UFCG) vinculado ao Centro de Engenharia Elétrica e Informática (CEEI).

O VIRTUS se encontra localizado fisicamente no ambiente pertencente ao Centro de Inovação e Tecnologia Telmo Araújo (CITTA), um centro de referência que atua de forma sistemática e cooperativa, organizando a oferta e a demanda de serviços para promover inovação tecnológica no Estado da Paraíba. Além de estruturar um portfólio diversificado de serviços, o CITTA é ocupado por empresas inovadoras, altamente qualificadas e conectadas com o mercado (CITTA, 2023).

O VIRTUS ocupa um importante papel na promoção do desenvolvimento e da inovação tecnológica em parceria com a indústria, abrangendo áreas como tecnologia da informação, comunicação e automação. Os projetos desenvolvidos no VIRTUS são executados através de mecanismos de incentivo como a Lei de Informática, EMBRAPII, entre outros (VIRTUS, 2023), uma vez que este está alocado na UFCG, uma ICT pública.

A equipe do VIRTUS é composta por professores, pesquisadores, estudantes e profissionais externos à instituição, que colaboram em diversas linhas de pesquisa, que incluem Sistemas Embarcados, Aprendizagem de Máquina, Internet das Coisas, entre outras.

Figura 1 – Fotografia do prédio do VIRTUS

Fonte: Autoria própria.

3 Fundamentação Teórica

Este capítulo apresenta uma revisão da literatura e os conceitos fundamentais que apoiarão o desenvolvimento das atividades durante o estágio. Ao final, espera-se que o leitor tenha obtido a compreensão teórica que norteia o uso das tecnologias e práticas aplicadas no período de estágio.

3.1 Metodologias ágeis: *Scrum*

Metodologias ágeis são abordagens de gerenciamento de projetos que tem como pilares a flexibilidade, a colaboração e a entrega contínua de valor. Esse tipo de metodologia foi desenvolvida em resposta às limitações dos métodos tradicionais de gestão de projetos, muitas vezes rígidos e ineficazes em lidar com mudanças e incertezas.

O *Scrum*[1] é um *framework* de gerenciamento de projetos com estrutura baseada em um conjunto de reuniões, ferramentas e funções para uma entrega eficiente de projetos. Entre as diversas metodologias ágeis, o *Scrum* se destaca por sua simplicidade e eficácia na entrega de produtos de alta qualidade. O *Scrum* é composto por papéis (*Product Owner, Scrum Master* e Equipe de Desenvolvimento), eventos (*Sprint, Daily Scrum, Sprint Planning, Sprint Review* e *Sprint Retrospective*) e artefatos (*Product Backlog, Sprint Backlog* e Incremento). O *Product Owner* representa os interesses de todos os envolvidos, define as funcionalidades do produto e gerencia o *backlog*; o *Scrum Master* facilita a equipe atua para resolver qualquer tipo de impedimento; e a Equipe de Desenvolvimento entrega incrementos funcionais do produto. (SCHWABER; SUTHERLAND, 2020)

O trabalho é dividido em *sprints*, ciclos curtos de duas a quatro semanas, onde versões funcionais do produto são entregues, permitindo *feedback* contínuo e ajustes no projeto. Cada *sprint* começa com a *Sprint Planning*, onde são definidos o objetivo do *sprint* e os itens do *backlog* que serão trabalhados. A *Daily Scrum* é uma reunião diária para revisar o progresso e planejar o trabalho do dia. A *Sprint Review* apresenta o incremento desenvolvido e coleta *feedbacks*, enquanto a *Sprint Retrospective* identifica melhorias para os próximos *sprints*.

Por fim, os artefatos são o *Product Backlog*, que se trata de uma lista priorizada de requisitos do produto; o *Sprint Backlog*, que contém as tarefas selecionadas para o *sprint* atual; e o Incremento, a soma de todos os itens completados durante o *sprint*. Uma ilustração do ciclo no *Scrum* é mostrada na figura 3.

O *Scrum* oferece benefícios como melhoria contínua, flexibilidade, transparência,

[1] <https://www.scrum.org/>

entrega rápida de valor e maior engajamento da equipe. Sua abordagem iterativa permite ajustes frequentes com base no *feedback*. Aliado a reuniões regulares, isso proporciona visibilidade do progresso e dos obstáculos, melhorando a comunicação e a gestão de projeto.

Figura 2 – Representação do ciclo da metodologia *Scrum*.

Fonte: G4 Educação, 2024.

3.2 Controle de versão: *Git*

Um sistema de controle de versão tem como objetivo gerenciar diferentes versões de um mesmo projeto. O *Git*[2] é um sistema de controle de versão amplamente utilizado para gerenciar o código-fonte em vários tipos de projetos, facilitando a colaboração entre desenvolvedores.

Uma das principais características do *Git* são os *branches*, que permitem o desenvolvimento de novas funcionalidades ou a correção de *bugs* sem afetar o código principal, possibilitando que os desenvolvedores realizem suas atividades de maneira simultânea, sem conflitos com outras versões que estão sendo desenvolvidas paralelamente. Além de possibilitar aos usuários visualizar o histórico de mudanças e reverter alterações, se necessário.

Os benefícios do *Git* incluem a facilitação do trabalho em equipe, o rastreamento de alterações, o backup de versões anteriores do código, proporcionando um controle de versão robusto, suporte à colaboração e uma maneira eficiente de gerenciar o histórico de alterações.

[2] <https://git-scm.com/>

3.3 Linguagem de programação *Python*

Python[3] é uma linguagem de programação de alto nível, conhecida por sua simplicidade e legibilidade, amplamente utilizada em diversas áreas, como desenvolvimento web, ciência de dados, inteligência artificial, automação, aplicações mobile, geoprocessamento, processamento de imagens, robótica, entre outras. (Silva; Silva 2019)

Uma das principais características de *Python* é sua sintaxe clara, que facilita o entendimento do código. Além disso, a linguagem possui uma vasta biblioteca padrão, oferecendo uma variedade de módulos e pacotes que facilitam o desenvolvimento de diferentes tipos de aplicações.

Python é uma linguagem interpretada, o que significa que seu código é executado linha por linha, tornando o processo de teste e depuração mais ágil. Também suporta múltiplos paradigmas de programação, como orientado a objetos, funcional e imperativo, permitindo flexibilidade no desenvolvimento de software.

3.3.1 *Pandas*

Pandas é uma biblioteca de manipulação e análise de dados, construída sobre a linguagem *Python*, que providencia uma abordagem rápida, flexível e robusta para trabalhar com dados relacionais.

A biblioteca oferece estruturas de dados eficientes, como *Series* e *DataFrames*, que permitem o armazenamento e manipulação de dados tabulares de forma intuitiva, possibilitando aos usuários realizarem operações como filtragem, agregação e transformação de dados de maneira simples. A biblioteca suporta a leitura e gravação de vários formatos de arquivos, incluindo CSV, Excel e SQL. Sua capacidade de lidar com grandes volumes de dados e operações complexas, torna o Pandas amplamente utilizado em tarefas de análise de dados, estatísticas e preparação de dados para modelagem.

3.3.2 *Dash*

Dash é uma biblioteca versátil que permite a criação de visualizações interativas usando a linguagem de programação *Python*. A biblioteca *Dash* possibilita que desenvolvedores construam *dashboards* com facilidade, utilizando componentes de interface do usuário prontos. Seus componentes incluem gráficos, tabelas dinâmicas, botões e menus, possibilitando visualizações personalizadas e interações intuitivas com os dados.

Uma de suas principais vantagens é o suporte a dados em tempo real, permitindo atualizações contínuas sem recarregar a página. O *Dash* oferece personalização avançada, permitindo ajustar cores, estilos e layouts dos componentes.

[3] <https://www.python.org/downloads/>

Sua integração com bibliotecas populares da comunidade *Python*, como *Plotly* e Pandas, faz com que seja amplamente utilizado em ciência de dados, permitindo a apresentação dinâmica de análises e resultados através de gráficos interativos e layouts personalizáveis, facilitando a criação de aplicações ricas em dados sem a necessidade de profundo conhecimento em desenvolvimento web.

3.3.3 *Seleniun*

Selenium é uma biblioteca da linguagem de programação *Python* utilizada na automação de tarefas relacionadas à interação com navegadores web, sendo uma ferramenta essencial em processos de teste de software, *web scraping* e controle de interfaces. Originalmente desenvolvida para facilitar a automação de testes de aplicações web, a *Selenium* tornou-se uma plataforma robusta que suporta diversos navegadores e sistemas operacionais.

A principal função do *Selenium* é permitir a interação automatizida com navegadores, semelhante à forma que um usuário faria. Utilizando o *Selenium WebDriver*, é possível simular cliques, inserção de textos, navegação entre páginas e até operações mais complexas como o arrastar e soltar elementos. A biblioteca também garante a localização e manipulação dos elementos da maneira como são renderizados pelo navegador, permitindo que o *Selenium* execute testes em um ambiente que reproduz com precisão o comportamento de usuários finais.

3.4 *MongoDB*

Os bancos de dados são sistemas essenciais para o armazenamento e gerenciamento de dados, que viabilizam o acesso eficiente a grandes volumes de informações. Eles podem ser classificados em dois principais tipos: relacionais e não relacionais.

O *MongoDB*[4] é banco de dados não relacional, baseado em software livre, que utiliza documentos flexíveis para processar e armazenar várias formas de dados, diferente do que acontece nos bancos de dados relacionais, que manipula os dados em formatos de linhas e tabelas.

As unidades básicas de armazenamento no *MongoDB* são documentos e coleções de documentos, formatados em *Binary JSON* (*Java Script Object Notation*), capaz de armazenar vários tipos de dados e fornecer um modelo de armazenamento elástico desses dados. Sua estrutura permite que os dados sejam armazenados de maneiras mais intuitivas e fáceis de entender, resultando em uma maior flexibilidade no armazenamento de dados.

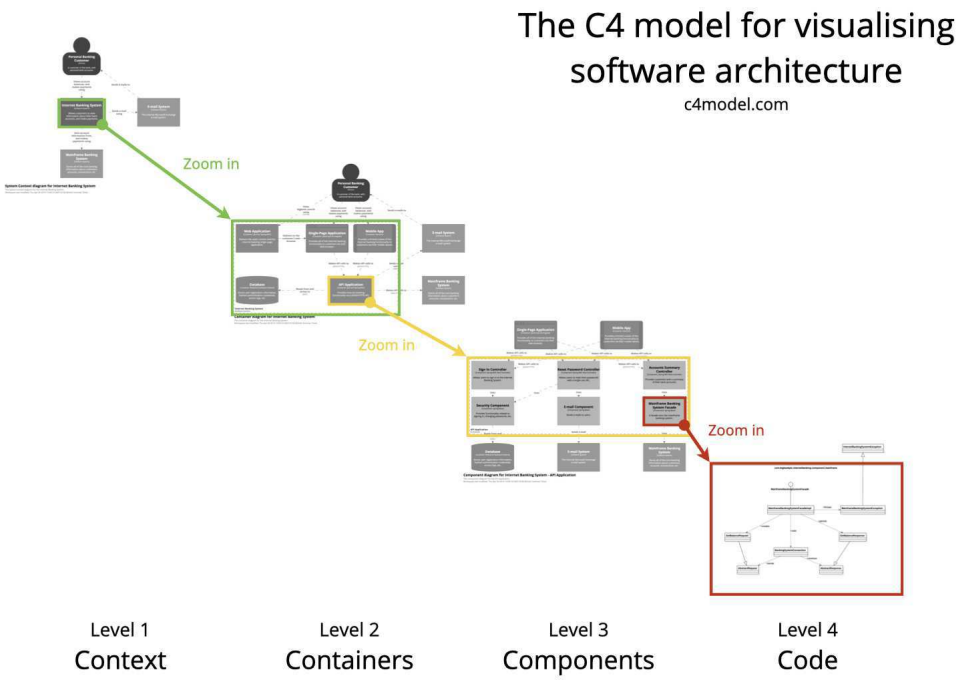
[4] <https://www.mongodb.com/try/download/community.>

O *MongoDB* é ideal para aplicações que requerem escalabilidade e desempenho, sendo capaz de lidar com grandes quantidades de dados e consultas complexas, por sua abordagem de esquema flexível. Sua facilidade de uso aliada à capacidade de manipular dados heterogêneos popularizou o *MongoDB*, tornando-o muito utilizado entre os desenvolvedores de software.

3.4.1 Arquitetura C4 *Model*

A arquitetura C4 *Model* é uma metodologia de arquitetura de software desenvolvida para facilitar a descrição e comunicação de arquiteturas de software de maneira clara e concisa, oferecendo uma abordagem pragmática e escalável para a modelagem arquitetônica. Essa metodologia se concentra em quatro níveis de abstração, da visão geral do sistema até os detalhes da implementação, permitindo que arquitetos e equipes técnicas comuniquem efetivamente a estrutura e o funcionamento de um sistema de software.

Figura 3 – Arquitetura C4 *Model*



Fonte: Brown, 2018

Os quatro níveis de abstração do C4 *Model* são:

1. Contexto (*Context*): Este é o nível mais alto de abstração e fornece uma visão geral do sistema e de seu contexto externo. Neste nível, são identificados os atores externos (usuários, sistemas externos, etc.) e suas interações com o sistema em questão.

2. Contêineres (*Containers*): No segundo nível, os sistemas são decompostos em contêineres, que são agrupamentos lógicos de componentes relacionados.

3. Componentes (*Components*): Os contêineres são então decompostos em componentes, que representam as unidades de software reutilizáveis e independentes. Esses componentes podem ser classes, módulos, serviços, entre outros.

4. Classes (*Code*): Este é o nível mais baixo de abstração, onde os componentes são decompostos em classes, métodos, funções, etc. Aqui é onde a implementação real do sistema é detalhada

4 Atividades Realizadas

Neste Capítulo, é fornecida uma visão geral das principais atividades realizadas durante o período de estágio, detalhando as etapas de levantamento de requisitos, definição da arquitetura, bem como o processo de desenvolvimento da ferramenta.

Devido às cláusulas de confidencialidade, este trabalho não trará os códigos desenvolvidos, as informações dos dados obtidos, as imagens dos gráficos e slides apresentados, tampouco o resultado final das atividades desenvolvidas, apenas o relato das sequências dos processos de desenvolvimento das atividades dos módulos descritos acima.

4.1 Pocesso de Engenharia de Software

O processo de engenharia de software é um conjunto estruturado de atividades necessárias para desenvolver, operar e manter um sistema de software. Segundo (Pressman, 2016), a Engenharia de Software abrange um processo, um conjunto de métodos e ferramentas que possibilitam aos profissionais desenvolverem software de altíssima qualidade. Ele abrange desde a concepção inicial do software até sua entrega e manutenção contínua. O objetivo principal desse processo é garantir que o software seja desenvolvido de maneira eficiente, atendendo aos requisitos dos usuários e mantendo alta qualidade.

4.1.1 Levantamento de requisitos

Uma das atividades cruciais durante o estágio foi o processo de levantamento de requisitos, essencial para garantir que o projeto atendesse às necessidades dos usuários e às funcionalidades do software. Este processo foi realizado em três etapas principais: a reunião com o cliente para coletar informações, a definição das regras do sistema e, por fim, a definição dos requisitos. Os requisitos foram levantados e agrupados em dois grupos: funcionais e não funcionais, e foram documentados para garantir uma base sólida para as fases seguintes do desenvolvimento.

Os requisitos funcionais descrevem as funcionalidades do sistema, ou seja, as ações e operações que ele deve realizar, especificando o comportamento esperado do sistema em resposta a determinadas entradas e em certas condições. Os requisitos não funcionais, por outro lado, definem os critérios que podem ser usados para julgar a operação do sistema, em vez de comportamentos específicos. Eles incluem atributos como desempenho, segurança, usabilidade, confiabilidade e escalabilidade.

4.1.2 Definição de arquitetura

A arquitetura do sistema foi definida utilizando o C4 Model para facilitar a descrição do projeto e a compreensão do impacto das mudanças e a identificação de possíveis riscos de forma antecipada, ao dividir o sistema em partes menores e mais gerenciáveis. Isso se alinha perfeitamente com uma abordagem de desenvolvimento ágil, pois permite que as equipes trabalhem em ciclos incrementais e adaptativos.

Por fim, a adoção do Modelo C4 também simplifica a documentação, tornando-a acessível tanto para novos membros da equipe quanto para aqueles que já estão envolvidos no projeto, promovendo uma comunicação clara e eficiente ao longo do ciclo de desenvolvimento.

4.2 Desenvolvimento

Esta seção descreve o caminho percorrido para transformar a teoria e o planejamento inicial em uma solução técnica funcional e eficiente, com base em uma abordagem prática e fundamentada nas melhores práticas de desenvolvimento de software.

4.2.1 Configuração do Ambiente de Desenvolvimento

A primeira etapa do desenvolvimento foi a configuração do ambiente de trabalho, utilizando o *PyCharm*[1] como IDE, escolhida pela sua robustez e suporte nativo para desenvolvimento com *Python* e *frameworks* de interface web. A configuração inicial incluiu a criação de um ambiente virtual para isolar dependências e garantir que as bibliotecas utilizadas não interferissem em outros projetos. Foram instaladas as bibliotecas essenciais, como *Dash* para criação da interface web, Pandas para manipulação de dados, *PyMongo* para a integração com o *MongoDB* e *Selenium*, para a função de exportação dos dados automatizada. Também foram adotadas práticas de controle de versão com o *Git*, permitindo o rastreamento de mudanças no código ao longo do ciclo de desenvolvimento.

4.2.2 Implementação da Interface com *Dash*

A segunda etapa consistiu na implementação da biblioteca *Dash* para criar uma interface web interativa que permitisse a visualização e interação com dados em tempo real. Durante o desenvolvimento, foram configuradas tabelas interativas que se atualizam automaticamente à medida que novos dados são inseridos no sistema, além de controles deslizantes e menus *dropdown* para a personalização da visualização. Essa interface foi projetada para oferecer uma experiência de usuário fluida e intuitiva, permitindo a exploração de grandes volumes de informações de maneira prática e visualmente atraente.

[1] <https://www.jetbrains.com/pt-br/pycharm/web-development/>

4.2.3 Integração com base de dados

A base de dados *MongoDB* foi implementada para armazenamento e recuperação dos dados necessários ao funcionamento da aplicação. A integração foi realizada com a biblioteca *PyMongo*, que facilitou a comunicação entre a aplicação e o banco de dados. Foram desenvolvidas funções que lidam com as principais operações de *CRUD* (*Create, Read, Update e Delete*), garantindo que a aplicação pudesse inserir, buscar, atualizar e remover dados de forma eficiente. Além disso, foram implementadas consultas otimizadas para lidar com grandes volumes de dados, permitindo que o sistema se mantenha performático mesmo com o crescimento do banco de dados. A estrutura do banco de dados foi desenhada para ser escalável, permitindo a fácil adaptação a novas funcionalidades futuras.

4.2.4 Documentação de código: *Docstring*

Para garantir a clareza e facilitar a manutenção futura, toda a base de código foi documentada utilizando *docstrings*. *Docstrings* são *strings* literais que aparecem logo após a definição de uma função, método, classe ou módulo, e são usadas para descrever seu propósito e comportamento. Elas permitem que desenvolvedores entendam rapidamente o que cada parte do código faz, quais são seus parâmetros, o que retorna, e qualquer exceção que possa ser levantada. Documentar o código com *docstrings* seguindo o padrão PEP 257, foi uma prática escolhida para assegurar que outros desenvolvedores (ou mesmo o próprio autor do código, em um momento futuro) possam manter, atualizar ou expandir a aplicação com facilidade.

4.2.5 Exportação de dados

A funcionalidade de exportação de dados foi implementada para permitir que os usuários possam gerar um arquivo CSV contendo os dados processados pela aplicação. O CSV é um formato amplamente utilizado para a troca de dados entre sistemas e softwares, garantindo compatibilidade e facilidade de uso em diversas ferramentas, como planilhas e sistemas de análise de dados.

Para realizar essa exportação, foi desenvolvida uma função específica que coleta os dados diretamente do banco de dados *MongoDB*, organizando-os em um formato estruturado. O usuário pode, então, baixar esses dados localmente na interface da aplicação. O processo de geração do arquivo CSV foi efetivado utilizando a biblioteca Pandas, que permite a manipulação eficiente dos dados e a conversão direta para o formato desejado, garantindo que a exportação seja rápida e eficaz, mesmo para grandes volumes de dados.

Além da exportação manual via interface, foi implementada uma automação para a exportação dos dados para um *website* externo, utilizando o *Selenium*, uma ferramenta

poderosa para automatizar interações com navegadores web. Essa automação permite que os dados gerados pela aplicação sejam automaticamente enviados para o site de destino, sem a necessidade de intervenção manual por parte do usuário.

O processo de automação com *Selenium* foi projetado da seguinte maneira: o usuário solicita a exportação dos dados via interface e, em seguida, o *script* do *Selenium* é executado para abrir o navegador, acessar o site especificado, fazer login, e navegar até o local onde os dados devem ser carregados. Por fim, os dados são preenchidos no *website* e o envio é concluído de forma automatizada. Essa funcionalidade elimina a necessidade de processos manuais repetitivos, agilizando a publicação de dados e garantindo maior eficiência e confiabilidade no fluxo de trabalho.

4.2.6 Demais Atividades

Além do desenvolvimento técnico da aplicação, diversas atividades complementares foram realizadas ao longo do estágio para garantir a qualidade, manutenibilidade e a organização do trabalho. Essas atividades incluem a revisão de código, participação em *plannings* e *reviews*, além da adoção de boas práticas de engenharia de software.

Uma parte essencial do projeto foi a criação de um guia do usuário, documentando de forma clara e acessível todas as funcionalidades da aplicação. O objetivo desse guia é fornecer instruções detalhadas para que todos os usuários possam utilizar a aplicação de maneira eficiente, independentemente do seu nível técnico. O documento cobre desde a instalação e configuração inicial, até a utilização de cada recurso da interface.

Essas atividades foram fundamentais para garantir um ciclo de desenvolvimento saudável e de alta qualidade durante o estágio, resultando em uma aplicação estável, bem documentada e alinhada às necessidades do cliente e da equipe.

5 Conclusões

Este relatório descreve as atividades realizadas durante a disciplina de Estágio Supervisionado, desenvolvidas no VIRTUS. O estágio proporcionou a aluna uma experiência de aprendizado focada no fortalecimento de habilidades comportamentais no trabalho em equipe, com ênfase na proatividade em planejamentos e execuções.

A orientação de um profissional do VIRTUS foi fundamental para o enriquecimento dessa experiência, uma vez que ele dedicou tempo para oferecer orientações e ensinamentos valiosos. Essa vivência permitiu a estagiária enfrentar diversas situações dentro de um projeto em desenvolvimento, promovendo o aprimoramento de habilidades técnicas (*hard skills*) e comportamentais (*soft skills*), ambas essenciais no mercado de trabalho.

As atividades desenvolvidas foram baseadas nas demandas do projeto principal, que é executado pelos profissionais da empresa parceira. É importante destacar que os conhecimentos adquiridos em disciplinas como Introdução à Programação, Técnicas de Programação e Informática Industrial do curso de Engenharia Elétrica, desempenharam um papel crucial na execução das atividades. Essas disciplinas forneceram uma base sólida, aplicável diretamente às tarefas desenvolvidas no estágio, demonstrando a relevância dos conteúdos presentes na grade curricular.

Referências

1 ALMEIDA, Marcus. Pandas Python: o que é, para que serve e como instalar. 2023. Disponível em: <https://www.alura.com.br/artigos/pandas-o-que-e-para-que-serve-como-instalar>. Acesso em 30 Jun. 2024.

2 BROWN, Simon. C4 Model for Visualising Software Architecture. 2018. Disponível em: <https://c4model.com>. Acesso em: 1 set. 2024.

3 CITTA. *Centro de Inovação e Tecnologia Telmo Araújo (CITTA)*. 2023. Disponível em: <https://www.linkedin.com/company/citta-centro-de-inovacao-e-tecnologia-telmo-araujo/?originalSubdomain=br>. Acesso em: 14 Jun. 2024.

4 VIRTUS. *VIRTUS - Núcleo de Pesquisa, Desenvolvimento e Inovação em Tecnologia da Informação, Comunicação e Automação*. 2023. Disponível em: <https://www.virtus.ufcg.edu.br/>. Acesso em: 14 Jun. 2024.

5 RAMOS, Alan. Controle de Versão,Git, Github e Bitbucket — Afinal, o que é tudo isso ? 2017. Disponível em: <https://medium.com/trainingcenter/controle-de-versÃčo-git-github-e-bitbucket-afinal-o-que-Ãľ-tudo-isso-9fa13fc13307>. Acesso em: 29 Jun. 2024.

6 G4 Educação. Metodologia Scrum: o que é, para que serve e exemplos para aplicar no seu negócio. G4 Educação. 2024. Disponível em: <https://g4educacao.com/portal/metodologia-scrum>. Acesso em: 27 Jun. 2024.

7 PRESSMAN, R. S. Software engineering: a practitioner's approach (9ª ed.). New York: Higher Education, 2016. Disponível em: <https://engenhariasoftwareisutic.wordpress.com/wp-content/uploads/2016/04/engenharia-software-pressman.pdf>. Acesso em: 04 Jul. 2024.

8 PEREIRA, Luiza Cherobini. *Guia Básico de Selenium com Python*. Tutoriais, Automações. Disponível em: <https://hub.asimov.academy/tutorial/guia-basico-de-selenium-com-python/>. Acesso em: 28 ago. 2024.

9 Selenium with Python Documentation. *Selenium with Python*. Disponível em: <https://selenium-python.readthedocs.io/>. Acesso em: 28 ago. 2024.

10 IBM. *O que é MongoDB?*. Disponível em: <https://www.ibm.com/br-pt/topics/mongodb>. Acesso em: 28 ago. 2024.

11 SCHWABER, Ken; SUTHERLAND, Jeff. The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game. Scrum.org, 2020. Disponível em: <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf>. Acesso em: 01 de set. 2024.