UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

GABRIEL MORAES RAMOS STUDART

UMA ABORDAGEM AUTOMATIZADA PARA INTEGRAÇÃO E ANÁLISE DE DADOS ACADÊMICOS E PROFISSIONAIS DE ALUNOS DOS CURSOS DE ENGENHARIA DA UFC

FORTALEZA

2025

GABRIEL MORAES RAMOS STUDART

UMA ABORDAGEM AUTOMATIZADA PARA INTEGRAÇÃO E ANÁLISE DE DADOS ACADÊMICOS E PROFISSIONAIS DE ALUNOS DOS CURSOS DE ENGENHARIA DA UFC

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em ENGENHARIA DE COMPUTAÇÃO do CENTRO DE TECNOLOGIA da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em ENGENHARIA DE COMPUTAÇÃO.

Orientador: Prof. Dr. José Marques Soares

FORTALEZA

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S922a Studart, Gabriel Moraes Ramos.
Uma abordagem automatizada para integração e análise de dados acadêmicos e profissionais de alunos dos cursos de Engenharia da UFC / Gabriel Moraes Ramos Studart. – 2025.
114 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Tecnologia, Curso de Engenharia de Computação, Fortaleza, 2025.
Orientação: Prof. Dr. José Marques Soares.

1. Python. 2. Business intelligence. 3. Power BI. 4. Web scraping. 5. LinkedIn. I. Título.
CDD 621.39

GABRIEL MORAES RAMOS STUDART

UMA ABORDAGEM AUTOMATIZADA PARA INTEGRAÇÃO E ANÁLISE DE DADOS ACADÊMICOS E PROFISSIONAIS DE ALUNOS DOS CURSOS DE ENGENHARIA DA UFC

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em ENGENHARIA DE COMPUTAÇÃO do CENTRO DE TECNOLOGIA da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em ENGENHARIA DE COMPUTAÇÃO.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. José Marques Soares (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Ricardo Jardel Nunes da Silveira
Universidade Federal do Ceará (DETI/UFC)

Prof. Wellington Wagner Ferreira Sarmento
Universidade Federal do Ceará (Instituto UFC Virtual)

Aos meus pais, meus irmãos, meus familiares, minha namorada e amigos.

AGRADECIMENTOS

"Sozinho você pode ir mais rápido, mas juntos vamos mais longe."

Definitivamente, essa frase descreve perfeitamente a essência deste agredecimento. Durante toda a minha jornada acadêmica, sempre pude contar com pessoas que me apoiaram, tanto nos momentos felizes quanto, principalmente, nos períodos mais difíceis, quando mais precisei. Dedico, portanto, este trabalho a todos vocês que acreditaram em mim e que são, e sempre serão,importantes na minha vida.

Em primeiro lugar, dedico aos meus pais, que sempre estiveram presentes, aconselhando-me e sendo o alicerce para que eu me tornasse quem sou hoje. Minha mãe, uma mulher guerreira, forte, trabalhadora e exemplar em tudo o que faz, sempre desejou o melhor para os filhos. Meu pai, um homem bom, alegre, brincalhão e engenhoso, além de ser meu melhor amigo, esteve ao meu lado sempre que precisei.

Dizem que o irmão mais velho é quase um segundo pai, pois ensina e aconselha. Tive a sorte de ter três irmãos mais velhos: Daniel, Diego e Adalberto. Agradeço a cada um deles por tudo o que fizeram. Sempre cuidaram de mim, ajudaram quando eu precisava e me aconselharam nos momentos necessários, sendo, acima de tudo, grandes parceiros, verdadeiros irmãos.

Agradeço à minha namorada, que, sem ela, eu não teria conseguido chegar até aqui. Ela sempre me ajudou, apoiou, confiou em mim e me ofereceu todo o amor que me faz sentir especial.

Aos meus familiares, e em especial à minha avó Maria Idilva, que faleceu no início de 2025, que sempre me apoiou e investiu na minha educação, sem jamais medir esforços para tanto. Ela sempre sonhou que seus netos tivessem uma formação, e esta conquista é dedicada à senhora. É apenas o começo, prometo. Presto, também, minha sincera homenagem ao meu tio Edgar, que faleceu em 2023, cujas últimas palavras foram um pedido para que eu me formasse. Dedico este trabalho a ele, com profundo reconhecimento e carinho.

A todos os professores do curso e da UFC, com os quais tive a honra de ter ótimas aulas e valiosos aprendizados para a vida, deixo aqui o meu sincero agradecimento. E em especial, expresso minha gratidão ao Prof. José Marques, que aceitou ser meu orientador, apoiou-me na escolha do projeto e, em todos os momentos, se mostrou presente, paciente e cuidadoso. Meu muitíssimo obrigado!

Não posso deixar de mencionar o Bob, a Nina, o Frajola, o Biro Biro e o Bruthinhos,

que são os queridos animais de estimação que fazem parte da família e que eu tenho muito amor.

Aos meus amigos, sejam eles da escola, do prédio, da faculdade ou do trabalho, agradeço por fazerem parte dessa trajetória. Junto de vocês, cresci como pessoa e como profissional, e cada um tem um lugar especial no meu coração.

Por fim, sou grato a todas as experiências que vivi na faculdade, as quais me auxiliaram diariamente a superar cada desafio dessa jornada acadêmica: o CdH (Clube do Hardware),o RAITec, a GTi Engenharia Jr., a FEJECE e o SAS. Vocês me transformaram e trouxeram vivências inesquecíveis, pelas quais serei eternamente grato.

“Excelência é uma arte conquistada com treino e hábito”

(Aristóteles)

RESUMO

Desde a antiguidade, observa-se a construção do conceito de escola e a preocupação com a formação de cidadãos capazes de impactar a sociedade. Nesse processo evolutivo, surgiram universidades e faculdades. No mundo atual, essas instituições de ensino enfrentam constantes transformações devido aos avanços tecnológicos da quarta revolução industrial, também conhecida como Indústria 4.0, o que exige que busquem atualizações para acompanhar as rápidas mudanças do mercado de trabalho. Isso é evidenciado no relatório anual do Fórum Econômico Mundial, intitulado *The Future of Jobs* (ou *O Futuro do Trabalho*, em tradução livre), no qual são analisadas tendências relacionadas ao mercado de trabalho e às habilidades necessárias para os profissionais do futuro, destacando as mudanças que ocorrem globalmente ano após ano. Com base nisso, foram desenvolvidas neste trabalho ferramentas em *Python* capazes de realizar *web scraping* nos dados disponíveis publicamente no SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas) sobre a situação acadêmica dos alunos e egressos, como: concluído, ativo e cancelado, de cada estudante matriculado no curso via SISU (Sistema de Seleção Unificada), e outro para coletar dados do *LinkedIn*, como experiência profissional e formação acadêmica, rastreando o máximo possível de alunos e egressos dos cursos de Engenharia da UFC. Por fim, elaborou-se um painel de *Power BI* capaz de relacionar os dados de situação acadêmica e profissional, gerando indicadores e *insights*. Dos resultados obtidos, demonstrou-se a viabilidade da automação na extração e atualização de dados dos perfis do SIGAA e do *LinkedIn*, utilizando *Selenium* e *Python*, superando desafios de navegação em ambientes web dinâmicos, padronização de formatos de datas, automação de *login* e entre outros. Como resultado, obteve-se um conjunto de dados relevantes para a elaboração do relatório de *Business Intelligence*, incluindo indicadores sobre experiências e competências alinhadas às demandas do mercado de trabalho, bem como taxas de conclusão e desistência dos cursos. Observou-se que, por exemplo, em média, os alunos demoram mais para concluir a formação do que o estipulado no currículo. Esses dados fornecem insumos valiosos para apoiar as tomadas de decisões acadêmicas, permitindo aprimorar os cursos e melhor preparar os estudantes para as exigências do mercado profissional.

Palavras-chave: *Python*. *Business Intelligence*. *Power BI*. *Web scraping*. *LinkedIn*. Competências. Situação Acadêmica. Mercado de Trabalho.

ABSTRACT

Since antiquity, the concept of school and the concern with shaping citizens capable of impacting society have been observed. Over time, universities and colleges emerged from this evolutionary process. In today's world, these educational institutions face constant transformations due to technological advances of the Fourth Industrial Revolution, also known as Industry 4.0, which require them to seek continuous updates to keep pace with rapid changes in the job market. This is evidenced in the World Economic Forum's annual report, The Future of Jobs, which analyzes labor market trends and the skills required for future professionals, highlighting global changes year after year. Based on this context, this study developed Python tools and scripts to perform web scraping on publicly available SIGAA data regarding the academic status of students and graduates—such as "completed," "active," or "canceled"—for each individual admitted through SISU, as well as to collect LinkedIn data on professional experience and academic background, tracking as many students and graduates from UFC's Engineering programs as possible. Finally, a Power BI dashboard was created to correlate academic and professional data, generating indicators and insights. The results demonstrated the feasibility of automating the extraction and updating of SIGAA and LinkedIn profile data using Selenium and Python, overcoming challenges related to dynamic web navigation, date-format standardization, automated login, and more. As a result, a relevant dataset was obtained for the creation of a Business Intelligence report, including indicators related to skills aligned with market demands, as well as course completion and dropout rates. It was observed, for example, that students typically take longer to finish their degrees than stipulated in the curriculum. These data provide valuable support for academic decision-making, enabling improvements to programs and better preparing students to meet professional demands.

Keywords: Python. Business Intelligence. Power BI. Web Scraping. LinkedIn. Skills. Academic Status. Job Market.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A introdução do ensino profissionalizante, paradigma adotado no Brasil, remonta aos séculos XVIII e XIX, tendo como referências pioneiras a França – exemplificada pela *École Polytechnique* – e a Alemanha – representada pela *Technische Hochschule*. No Brasil, esse modelo educacional emergiu em 1792, com a criação da Real Academia de Artilharia, Fortificação e Desenho, no Rio de Janeiro, marcando o início de um processo voltado à formação de profissionais capacitados para atender às demandas do mercado e do Estado, conforme destacado por (CORDãO; MORAES, 2020) e (CORDEIRO, 2021).

Com o passar do tempo, as universidades e as faculdades foram crescendo e acompanhando a evolução das cidades e as suas novas necessidades, empresas nascendo e novas profissões surgindo. Atualmente, estamos passando por um grande avanço tecnológico, Schwab (2016), autor de "A Quarta Revolução Industrial", descreve como o mundo está em momento de mudanças profundas na forma de como trabalhamos, nos comportamos, nos comunicamos, nos entretemos e nos relacionamos. O surgimento de novos modelos de negócios está impactando em todos os setores da sociedade, algumas profissões ficaram ultrapassadas e novas vão surgindo. Essas mudanças tornaram-se cada vez mais aceleradas,havendo a necessidade crescente de que as faculdades e seus cursos se atualizem, enfrentando o desafio de acompanhar essa evolução.

Com os avanços tecnológicos, o mundo está cada vez mais conectado. Surgem grandes plataformas que aproximam as pessoas, conhecidas como "redes sociais". Em 5 de maio de 2003, surgiu uma nova rede social, o LinkedIn, com o objetivo de promover conexões profissionais e gerar oportunidades de emprego. Sua missão é "conectar profissionais do mundo todo, tornando-os mais produtivos e bem-sucedidos" e sua visão consiste em "criar oportunidades econômicas para cada integrante da força de trabalho global" (LINKEDIN, 2025).

Atualmente, a plataforma se apresenta como a maior rede profissional do mundo, atuando em 200 países e territórios, com mais de 850 milhões de usuários (LINKEDIN, 2025) e, somente no Brasil, cerca de 75 milhões de usuários (CNN, 2024). Considerando os desafios e objetivos das universidades, bem como a relevância da plataforma LinkedIn, é possível estabelecer uma relação que favoreça o desenvolvimento profissional dos estudantes.

A pesquisa desenvolvida neste trabalho buscou construir ferramentas capazes de coletar dados de alunos e egressos dos cursos de Engenharia da Universidade Federal do Ceará, referentes à situação acadêmica (por meio da plataforma da faculdade – SIGAA) e à situação profissional (através da plataforma LinkedIn), gerando, a partir desses dados, um painel de BI

(*Business Intelligence*),um relatório interativo para a comunidade acadêmica.

Com isso, busca-se ajudar a conectar ainda mais a Universidade com o mercado de trabalho, acompanhando o ritmo acelerado das mudanças, o surgimento de novas profissões e identificando quais são as competências que as empresas esperam de recém-graduados, e como essas exigências se relacionam com as habilidades desenvolvidas durante a jornada universitária.

1.1 Objetivos

Esse trabalho tem como objetivo geral construir ferramentas reutilizáveis capazes de coletar e visualizar dados profissionais de alunos e egressos da UFC e os seus dados de situação acadêmica.

1.1.1 Objetivos específicos

- Desenvolver um software de *Web Scraping* (Raspagem de dados da Web) que permita coletar, do SIGAA, os dados abertos sobre a situação acadêmica dos alunos e dos egressos, como: concluído, ativo, cancelado, de cada aluno matriculado no curso via SISU.
- Desenvolver um software de *Web Scraping* (Raspagem de dados da Web) que permita coletar, do Linkedin, os dados de ex-alunos dos cursos de engenharia da UFC.
- Propor um painel de BI capaz de relacionar os dados de situação acadêmica e dados profissionais, gerando indicadores e conclusões.

1.2 Estrutura do trabalho

O presente trabalho está estruturado em 5 capítulos. O primeiro capítulo aborda a introdução, envolvendo a contextualização, a problematização, a motivação e os objetivos do projeto. O segundo capítulo trata dos fundamentos, que constituem a base teórica utilizada como alicerce para a realização do trabalho. O terceiro capítulo apresenta a construção das ferramentas do projeto, detalhando como foram elaboradas, quais tecnologias foram utilizadas, as dificuldades enfrentadas e todo o passo a passo do desenvolvimento. O quarto capítulo discorre sobre os resultados obtidos, apresentando os dados coletados, as análises realizadas a partir das visualizações criadas e os estudos de caso. Por fim, o quinto capítulo apresenta a conclusão do

trabalho e os próximos passos, incluindo a atualização dos dados coletados, a utilização das ferramentas para a coleta de novos dados e possíveis evoluções do projeto.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, é abordada toda a base teórica para o desenvolvimento deste trabalho e suas referências, bem como os principais conceitos, teorias, técnicas e ferramentas relacionadas à *Web Scraping* (raspagem de dados web) e à construção de relatórios de *Business Intelligence*. Além disso, serão discutidos os desafios associados à evolução do mercado de trabalho.

2.1 Evolução do mercado de trabalho e os desafios das mudanças tecnológicas

As mudanças no mercado de trabalho estão cada vez mais rápidas, pois estamos vivenciando o ápice da quarta revolução industrial com o avanço tecnológico. Segundo Klaus Schwab (SCHWAB, 2016), novas profissões estão surgindo e muitas outras, que nem imaginávamos, emergem à medida que as necessidades evoluem, juntamente com as transformações aceleradas que o mundo vem sofrendo. Para Schwab, novas habilidades serão essenciais, as "Habilidades do Futuro" acompanharão essas mudanças aceleradas e as profissões mais promissoras estão conectadas diretamente com setores ligados à inovação.

Anualmente, o Fórum Econômico Mundial publica um relatório intitulado *The Future of Jobs – o Futuro do Trabalho*. Nesse documento, são analisadas as tendências relacionadas ao mercado de trabalho e às habilidades necessárias para os profissionais no futuro, relacionando as mudanças que vêm acontecendo no mundo ano após ano. O relatório de 2025 reforça que as inovações tecnológicas, juntamente com o contexto geopolítico, continuarão a potencializar a criação e a eliminação de vagas de emprego. Além disso, o relatório ressalta que as habilidades e as competências do futuro serão cada vez mais integradas à interação homem-máquina, com tarefas cada vez mais relacionadas à tecnologia, seja por meio de inteligência artificial, automação e robótica.De acordo com essa publicação, empresas e governos, representados por faculdades e universidades, devem investir constantemente na formação e requalificação dos profissionais (FORUM, 2025).

2.2 Plataforma de Rede Social Profissional – LinkedIn, como meio de estudo do Mercado de Trabalho atual

Com o avanço da era digital, surgiram as redes sociais, como Facebook e Instagram. No entanto, uma delas se destacou ao adotar a premissa de conectar as pessoas profissionalmente, com o intuito de gerar empregos. A visão da plataforma LinkedIn é "Criar oportunidades

econômicas para cada integrante da força de trabalho global" (LINKEDIN, 2025).

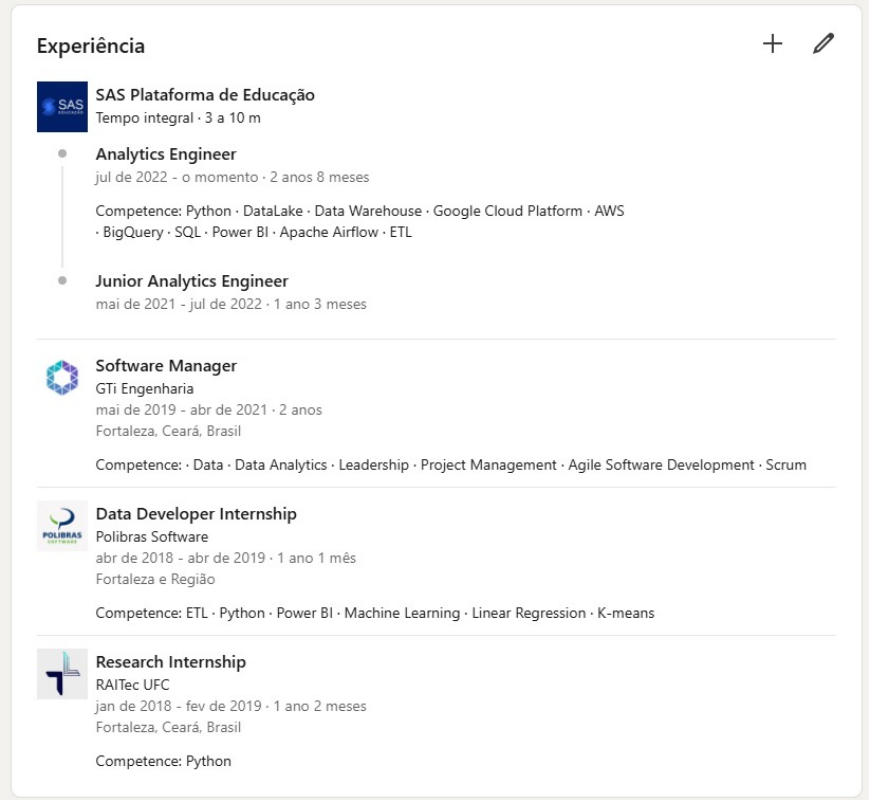
A plataforma vem ganhando magnitude ao longo dos anos, atingindo mais de 850 milhões de usuários em 200 países e territórios, sendo cerca de 75 milhões no Brasil (CNN, 2024). Na plataforma, o usuário consegue se conectar com outros profissionais, enviar mensagens, buscar vagas em empresas e publicar projetos e contribuições.

A seção de perfil é uma das mais importantes, pois funciona como uma vitrine profissional do usuário. É a partir dela que as empresas costumam fazer o primeiro contato. Nessa seção, constam dados pessoais, informações sobre a área de atuação, habilidades e uma lista de suas experiências profissionais e acadêmicas.

Figura 1 – Informações do Perfil - Nome e Subtítulo

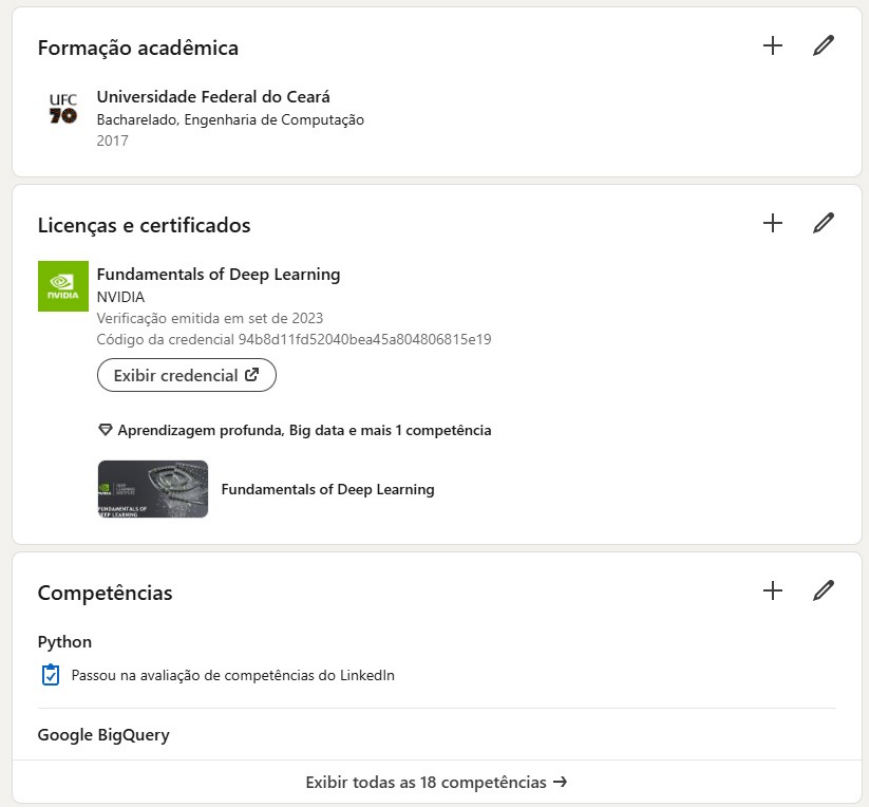
Fonte: elaborado pelo autor (2025).

Figura 2 – Informações do Perfil - Experiências



Fonte: elaborado pelo autor (2025).

Figura 3 – Informações do Perfil - Formações e Competências



Fonte: elaborado pelo autor (2025).

Em Rodrigues (2018), realizou-se uma análise do LinkedIn como ferramenta social e profissional no grupo de administradores, com o objetivo de investigar como o LinkedIn é utilizado como rede social profissional, focando no grupo de ex-alunos de Administração da Faculdade de Gestão e Negócios (FAGEN) da Universidade Federal de Uberlândia (UFU). Concluiu-se que o LinkedIn se mostra uma ferramenta relevante tanto para a busca de empregos quanto para a construção de *networking*. Além disso, o estudo ressalta a importância de um perfil bem estruturado para alcançar empregabilidade e construir a imagem profissional. Também foi apontado que a rede social não é utilizada apenas por quem está ativamente buscando recolocação; muitos profissionais a utilizam para se manter conectados, acompanhar tendências e observar a evolução do mercado de trabalho.

2.3 Conceitos de Web Scraping

Para GLEZ-PEÑA *et al.* (2014), o *Web Scraping* é um processo automatizado para extrair o conteúdo da internet de maneira estruturada, simulando a interação humana em um site e navegando pelas páginas para obter as informações desejadas. Com isso, os dados são extraídos e transformados em um formato estruturado, como tabelas, para que, posteriormente, sejam armazenados e utilizados em análises futuras.

Badaró (2023) ressalta, a importância de conhecer a estrutura do HTML para definir como será realizada a extração dos dados via *web scraping* e quais estratégias serão empregadas, conforme os estudos de Krijnen, Bot e Lampropoulos (2014) citados em seu trabalho. O HTML é uma linguagem de marcação que organiza o conteúdo da página web em uma hierarquia de elementos, isto é, as "tags" (como `<div>`, `<body>`, `<p>`, etc.) que representam os elementos da página, as quais podem ser observadas nas Figuras 4 e 5. Essas tags podem conter atributos, como classes e IDs, que são fundamentais para a identificação dos elementos durante a extração de dados.

Figura 4 – Exemplo de um HTML de uma página Web

```html
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Pensando na Web</title>
</head>
<body>
  <h1>Aprendendo HTML</h1>
  <p>
    O HTML é uma linguagem de marcação utilizada para construir páginas web.
    Ela foi criada por Tim Berners-Lee (...)
  </p>
</body>
</html>
```

Fonte: elaborado pelo autor (2025).

Figura 5 – Informações do Perfil - Formações e Competências

Fonte: pensandonaweb (2025).

O processo de raspagem de dados pode ter, principalmente, duas abordagens: o *scraping* sintático e o *scraping* semântico.

- O *scraping* sintático foca na estrutura física do documento HTML, baseando-se

na análise dos elementos de marcação, ou seja, as “tags” e seus atributos, como IDs e classes, mas sem considerar o conteúdo dos dados extraídos.

- O *scraping* semântico é uma abordagem mais complexa que vai além da estrutura HTML e procura atribuir significado aos dados extraídos, podendo utilizar até técnicas de Processamento de Linguagem Natural (NLP) para interpretar, identificar e categorizar o conteúdo.

O *Selenium* é uma ferramenta de automação de navegadores que permite a criação de testes e a extração de dados, possibilitando interações com páginas web, tais como cliques, o preenchimento de formulários e rolagem. Conforme Zhan (2015), a utilização do *Selenium WebDriver* na raspagem de dados é relevante pela capacidade de lidar com páginas que apresentam conteúdos dinâmicos, presentes na maioria dos sites modernos. O autor afirma que esse recurso torna o processo de *web scraping* mais robusto e eficaz em situações que exigem a simulação do usuário, como a necessidade de aguardar o carregamento de elementos ou revelar informações dinamicamente.

2.4 Conceito de Business Intelligence e o uso de Power BI como ferramenta

Como descrito pelo próprio fabricante (Microsoft, 2024), o Power BI é “uma coleção de serviços de software, aplicativos e conectores que trabalham juntos para transformar suas fontes de dados não relacionadas em informações coerentes, visualmente envolventes e interativas. Com o Power BI, você pode se conectar facilmente a fontes de dados, visualizar e descobrir conteúdo importante e compartilhá-lo com todas as pessoas que quiser.” Dentre os recursos oferecidos, destaca-se a possibilidade de conectar a diferentes fontes de dados, o que permite visualizar e analisar informações de modo integrado.

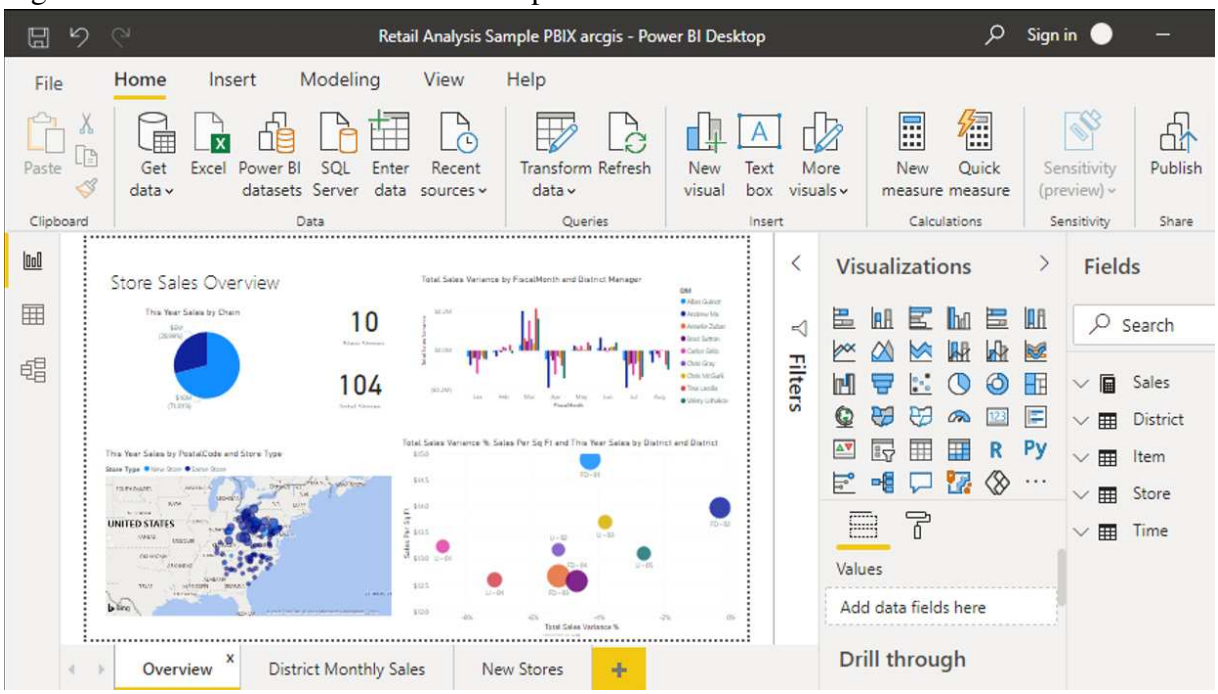
Segundo Meireles (2022), *Business Intelligence* (BI) é um sistema orientado a dados que dá suporte a análises e decisões fundamentadas em fatos. De acordo com **??**), essa abordagem combina a coleta e o armazenamento de dados com a gestão do conhecimento, permitindo às organizações utilizar informações confiáveis em seus processos decisórios. Meireles (2022) também destaca que o BI evoluiu de relatórios estáticos para sistemas dinâmicos e interativos, capazes de fornecer *insights* e operar em tempo quase real.

Nunes (2021), ressalta que, segundo Silva (2019 *apud* Nunes, 2021), a ferramenta se destaca pela simplicidade e facilidade de uso, oferecendo recursos de arrastar e soltar, além

da possibilidade de criar relatórios atraentes e customizados. Também, de acordo com Alves (2019 *apud* Nunes, 2021), as funcionalidades do Power BI, aliadas a um ambiente seguro, de confiabilidade e disponibilidade, tornam a ferramenta indispensável para a gestão e análise de dados.

Um dos principais serviços da ferramenta é o Power BI Desktop, que pode ser visto na Figura 6. De acordo com Microsoft (2024), "É um aplicativo gratuito que pode ser instalado no computador local e que permite que você se conecte aos seus dados, transforme-os e visualize-os. Com o Power BI Desktop, você pode se conectar a várias fontes de dados diferentes e combiná-las (geralmente chamado de modelagem) em um modelo de dados. Esse modelo de dados permite que você crie visuais e coleções de visuais [...]".

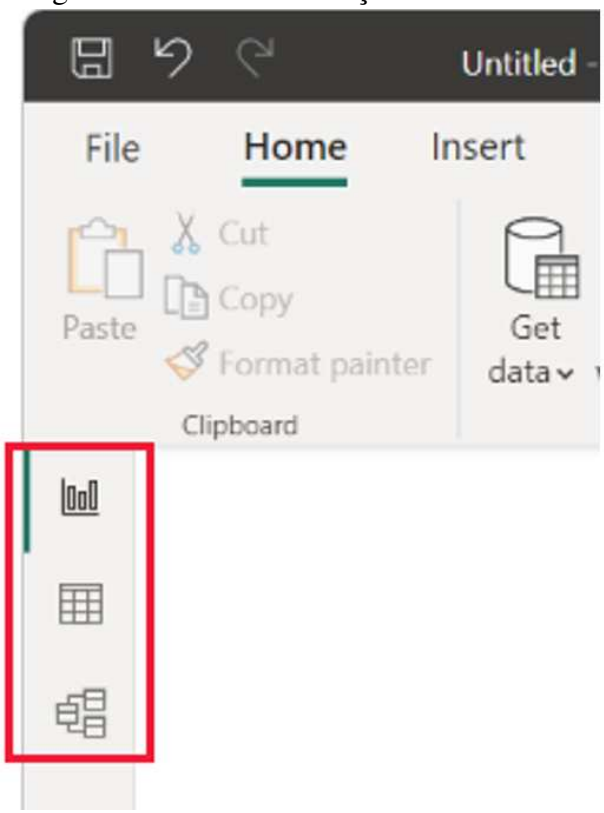
Figura 6 – Interface do Power BI Desktop



Fonte: microsoft (2024).

No Power BI Desktop, existem três formas de exibições, as quais podem ser visualizadas na Figura 7 e são fundamentais para a construção do relatório, a saber: Relatório, Dados e Modelo, nas quais se encontra o que é essencial para o desenvolvimento do projeto.

Figura 7 – Menu de exibições



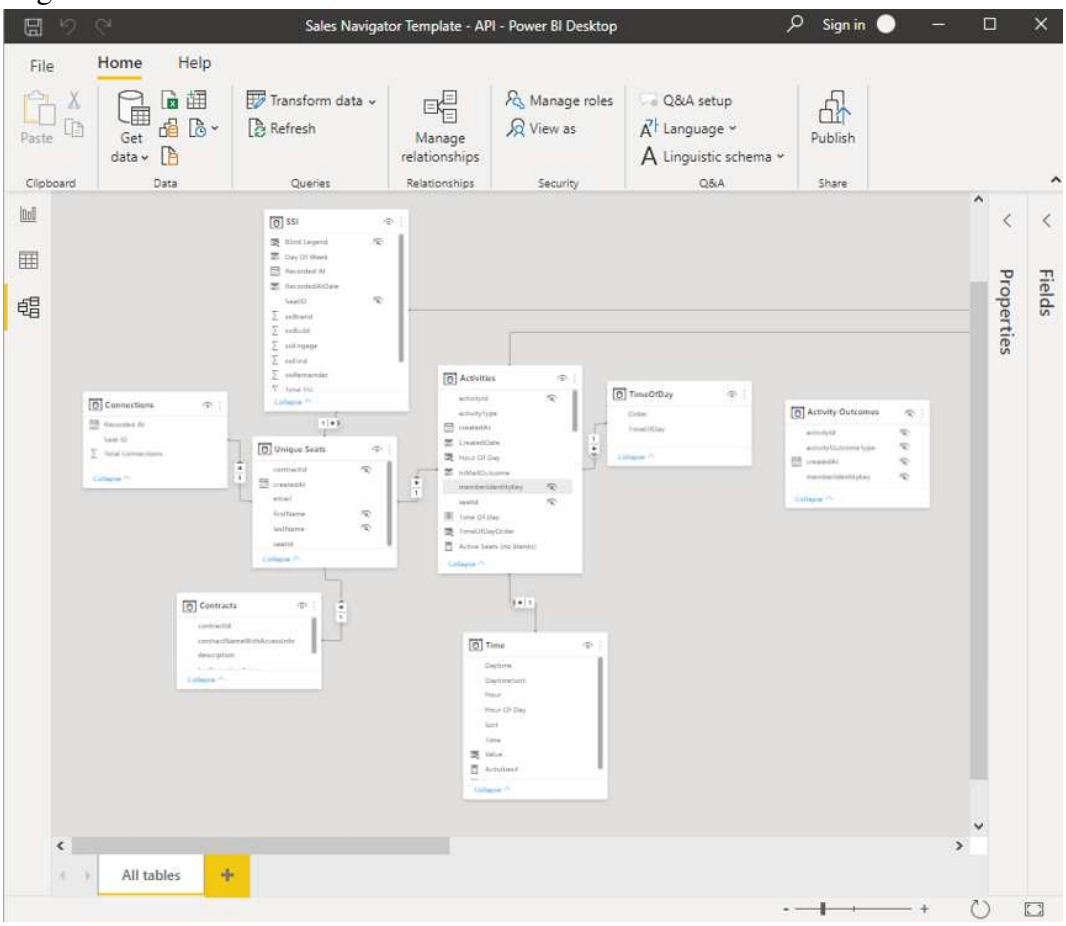
Fonte: microsoft (2024).

Na primeira forma de exibição, encontram-se as páginas do relatório com os visuais criados, é nela que são construídos os gráficos, os quais selecionam os campos das tabelas que serão usados e definem as medidas DAX, que, de acordo com a Microsoft (2024) é "uma linguagem de expressão de fórmula usada nos Analysis Services, no Power BI e no Power Pivot no Excel. As fórmulas DAX incluem funções, operadores e valores para realizar cálculos avançados e consultas em dados nas tabelas e colunas relacionadas nos modelos de dados tabulares."

Na segunda forma de exibição, são os Dados, é possível visualizar as tabelas importadas, as medidas, as colunas criadas e todo o resultado da transformação das estruturas de dados realizadas no Power BI.

Por fim, a terceira forma de exibição e fundamental para o funcionamento do relatório, denominada Modelo, na qual é possível gerenciar todos os relacionamentos entre as tabelas, garantindo que a modelagem esteja correta.

Figura 8 – Modelo de dados



Fonte: microsoft (2024).

2.5 Considerações finais

Todos os fundamentos teóricos apresentados,incluindo a rede social LinkedIn, os conceitos de web scraping, o conceito de Business Intelligence (BI) e a utilização da ferramenta Power BI, serviram como base conceitual para a elaboração e o desenvolvimento das ferramentas propostas neste trabalho. No próximo capítulo, serão detalhadas as metodologias adotadas e os resultados obtidos, evidenciando como esses referenciais foram aplicados na prática. Assim, a fundamentação teórica sustenta as etapas subsequentes do projeto, tornando claro o caminho conceitual percorrido até a implementação das soluções sugeridas.

3 PROJETO

Neste capítulo, são apresentadas as etapas de implementação do projeto, que se fundamentam nas técnicas discutidas no capítulo 2, bem como no objetivo do projeto e no resultado esperado ao final do mesmo. Serão discutidos os desafios encontrados durante o desenvolvimento de cada ferramenta e os contextos nos quais elas ocorreram.

3.1 Objetivo do Projeto

O projeto tem como foco desenvolver ferramentas que possibilitem a extração de dados sobre a situação dos alunos atuais e dos egressos dos cursos da UFC, considerando seus status acadêmicos e perfis na rede social LinkedIn. Esses dados serão utilizados para a construção e análise de um relatório de *Business Intelligence*, com o objetivo de criar indicadores que possam orientar e auxiliar os professores e coordenadores no entendimento das novas demandas acadêmicas e expectativas do mercado de trabalho. O projeto também busca proporcionar autonomia para que eles possam alimentar o relatório com novos dados, sejam de cursos ainda não contemplados, de novos alunos ingressantes ou de atualizações periódicas sobre a situação dos discentes e egressos da Universidade.

3.2 Etapa 1: Entendimento das informações

Nesta etapa inicial, buscou-se identificar quais informações estavam disponíveis e de que forma elas eram apresentadas em seus sites.

3.2.1 Dados de situação acadêmica

Para o desenvolvimento do projeto, foi planejado extrair as bases de dados da situação acadêmica dos alunos matriculados da UFC de 2015 até 2024. Para isso, era necessário extrair os dados disponíveis publicamente no SIGAA, na página <https://si3.ufc.br/sigaa/public/sisu/resultadoMatriculaAnosAnteriores.jsf>, selecionar o curso e a edição (ano) desejados conforme ilustrado na Imagem 9.

Figura 9 – Página de seleção de curso e ano

Fonte: SIGAA (2025).

A ferramenta deve oferecer autonomia na escolha do curso e do período, permitindo a extração de dados para uma nova graduação ou a atualização de uma graduação já existente, além de definir se a extração será completa para o período ou abrangerá apenas um recorte específico.

Os dados a serem extraídos pela ferramenta serão as seguintes informações: Nome do Curso, Ano, Nome do Aluno e Status.

O status pode ser:

- Conclusão: alunos que concluíram o curso;
- Ativos: alunos que ainda possuem matrículas ativas;
- Cancelado: alunos que abandonaram o curso;
- Trancado: alunos que trancaram o curso.

Além destes, foram identificados alguns casos raros, que indicam alunos com status em branco, significando que eles não têm status definido. Isso pode representar um tipo de bug no SIGAA ou o aluno está passando por um processo não finalizado na Universidade, e, enquanto não é realizado, não possuem status. Na Figura 10, pode-se observar como a página gera essas informações.

Figura 10 – Tabela com informações dos Alunos

Universidade Federal do Ceará — Fortaleza, 21 de Fevereiro de 2025

SIGAA
Sistema Integrado de Gestão de Atividades Acadêmicas

ENTRAR NO SISTEMA

SELECIONE UMA EDIÇÃO E UM CURSO E DEPOIS CLIQUE EM "BUSCAR" PARA PESQUISAR O RESULTADO.

FILTRAR RESULTADOS POR EDIÇÃO E CURSO

Edição: 2015

Curso: ENGENHARIA DE COMPUTAÇÃO/INTEGRAL/FORTALEZA

Buscar

CANDIDATOS SELECIONADOS - ENGENHARIA DE COMPUTAÇÃO/INTEGRAL/FORTALEZA - 1º SEMESTRE

Modalidade	CPF	Nome (Ordem Alfabética)	Situação atual
L4	019******84	AIRTON SALES BAYMA NETO	CANCELADO
AC	068******03	ALEF CARNEIRO DE SOUSA	CONCLUÍDO
L2	604******00	ANTONIO BATISTA DE OLIVEIRA	CONCLUÍDO
AC	045******50	ANTONIO CLAUDIO GUIMARAES NETO	CANCELADO
L2	064******50	ARIEL DA SILVA BARROS	CONCLUÍDO
AC	607******60	CAIO CID SANTIAGO BARBOSA	CONCLUÍDO
AC	608******60	CAIO COSTA DO AMARAL	CONCLUÍDO
AC	981******04	CAIO RODRIGO DE ALMEIDA VIEIRA	CONCLUÍDO
AC	059******08	DANIEL FILHO COELHO RAMOS	CONCLUÍDO
AC	603******03	DANIEL HORTA MAXIMO	CANCELADO
L2	434******24	DANILO MOREIRA DE ALMEIDA	CANCELADO
AC	020******80	DAVI BARRETO FACANHA	CONCLUÍDO
AC	050******03	DIEGO NOGUEIRA FEIJO	CONCLUÍDO
L4	045******01	EDILSON SILVA	CANCELADO
AC	999******91	EDUARDO FERNANDES MONTESUMA	CONCLUÍDO
L4	063******10	FELIPE ARAUJO MAGALHAES	ATIVO
L4	025******99	FERNANDO DIEGO SILVEIRA NASCIMENTO	CANCELADO
AC	062******38	FRANCISCO ALESON BARRETO GABRIEL	CONCLUÍDO
L1	067******60	FRANCISCO ROMEU DE SOUSA RIBEIRO	CANCELADO
L3	063******70	GABRIEL DA ROCHA SILVA	CONCLUÍDO
AC	608******02	GABRIEL HENRIQUE ALENCAR MEDEIROS	CONCLUÍDO

Fonte: SIGAA (2025).

3.2.2 Dados da plataforma LinkedIn

Além dos dados da situação acadêmica, fazia parte do planejamento construir uma ferramenta para extrair, via web scraping, os dados de perfil dos alunos e dos egressos no LinkedIn, que podem ser visto nas Figuras 1, 2 e 3.

O perfil do usuário possui algumas seções que são muito importantes para a conexão com recrutadores de empresas e fornecem informações relevantes para o desenvolvimento do relatório, como:

- **Informação de identificação e atividade profissional:** Nesse campo, constará o nome – dado crucial para identificar se a pessoa é, de fato, um aluno ou egresso da UFC –, a foto de perfil, que auxilia na identificação, e, por fim, o subtítulo, onde são inseridas palavras-chave relacionadas ao trabalho, como profissão e ferramentas. Essas informações facilitam o rankeamento do perfil e tornam mais fácil sua localização por recrutadores que buscam profissionais com as características adequadas à atividade desempenhada.

Figura 11 – Informações de Perfil

Fonte: elaborado pelo autor (2025).

- **Informações de experiências profissionais:** Nesse campo, consta uma lista de todas as experiências profissionais que o usuário listou, contendo informações sobre o nome da empresa, o cargo, a duração da experiência e um campo de texto livre no qual o profissional pode detalhar suas atividades.

Figura 12 – Informações de Experiência

Fonte: elaborado pelo autor (2025).

- **Informações de formação acadêmica:** Nesse campo, consta uma lista de todas as formações que o usuário definiu, contendo informações sobre o nome da instituição, o nome do curso (o qual pode ser opcional ou apresentado em inglês) e o período da formação. Esse campo é muito importante para o sucesso do projeto, pois é nele que se cria o relacionamento entre o perfil do usuário e o perfil da Universidade.

Figura 13 – Informações de Formação Acadêmica

Fonte: elaborado pelo autor (2025).

- **Informações de competências:** Nesse campo, consta a listagem de todas as competências mencionadas pelo usuário, abrangendo desde habilidades técnicas, como programação e desenvolvimento web, até ferramentas, como Excel e Power BI, além de habilidades mais relacionais, como liderança e comunicação. Esse campo é fundamental para o relatório do projeto, pois, com essas informações, é possível compreender quais competências são atualmente importantes no mercado de trabalho, quais aparecem com mais frequência e quais estão emergindo.

Figura 14 – Informações de Competência

Fonte: elaborado pelo autor (2025).

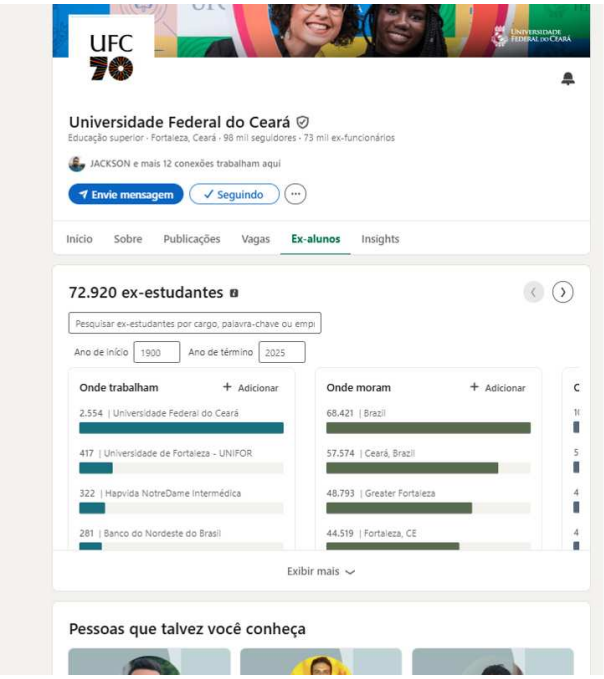
3.2.3 Encontrar os perfis de Alunos e Ex-alunos no Linkedin

Contudo, um grande desafio para a extração é identificar os perfis dos egressos e correlacioná-los com os dados da UFC. Por exemplo, no SIGAA consta o registro do aluno Gabriel Moraes Ramos Studart, do curso de Engenharia de Computação em Fortaleza, turma de 2017, mas localizar no LinkedIn um perfil que corresponda exatamente a essa pessoa é uma tarefa complexa. Ao superar esse obstáculo, o relatório poderá conter informações relevantes e auxiliar na resposta de questões como: “Os egressos daquele curso estão trabalhando? Daquela turma, quantos estão empregados? Em quais profissões atuam?”.

Com isso, e considerando que existe uma página da UFC no LinkedIn na qual constam todos os alunos e ex-alunos que se declaram parte da universidade na formação acadêmica,

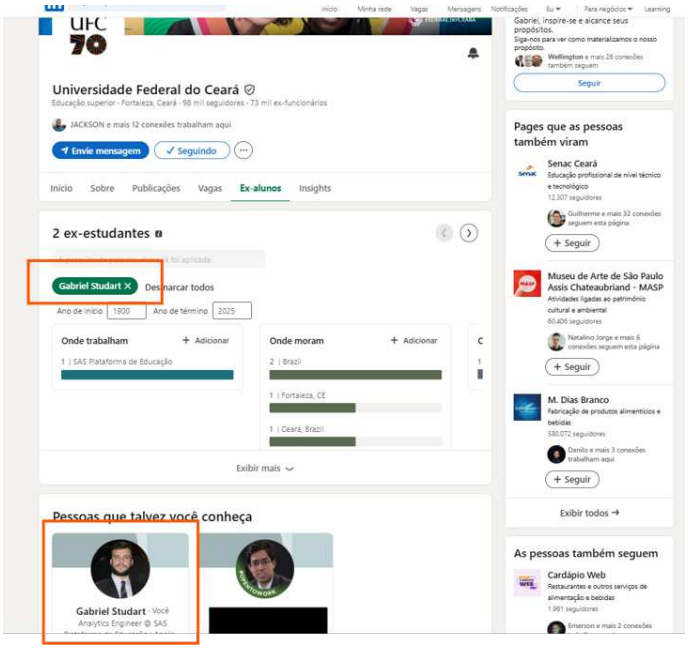
a estratégia inicialmente adotada foi pesquisar todos os usuários relacionados a essa página na rede social, utilizando o filtro pré-estabelecido do curso selecionado, como ilustrado nas Figuras 15 e 16.

Figura 15 – Página da UFC no LinkedIn



Fonte: elaborado pelo autor (2025).

Figura 16 – Filtrando um aluno na página da UFC no LinkedIn



Fonte: elaborado pelo autor (2025).

3.3 Etapa 2: Levantamento de requisitos

A partir dos objetivos e das informações existentes, identificou-se a necessidade de duas ferramentas, uma que pudesse automatizar a navegação na página da situação acadêmica no SIGAA e outra para o LinkedIn, facilitando o acesso aos perfis de ex-alunos de forma eficiente.

3.3.1 Ferramenta script_UFC

Como primeira etapa do projeto, realizou-se o levantamento de requisitos para a implementação da ferramenta de raspagem de dados da página do SIGAA. Nesta fase, tornou-se essencial conhecer o site e suas funcionalidades, assim como sua estrutura HTML e seus elementos. Verificou-se a necessidade de um mecanismo automatizado capaz de:

- Navegar automaticamente na página específica do SIGAA, acessando seus elementos pertinentes, como a escolha do ano, do curso e selecionando o botão "Buscar".
- Permitir a seleção dinâmica de cursos e intervalos de anos, por meio da leitura de arquivos de configuração e planilhas, a fim de tornar a ferramenta flexível para diferentes cenários de análise, dando autonomia ao usuário para escolher.
- Realizar a extração dos dados contidos nas tabelas, especificamente os dados relativos à formação, tais como o nome do aluno, o curso e a situação acadêmica.
- Atualizar as informações já armazenadas, evitando a criação de duplicidades e garantindo a integridade dos dados consolidados.
- Lidar com problemas inerentes à navegação automatizada, como *timeouts* e falhas de conexão, utilizando técnicas de espera explícita para contornar as características da página do SIGAA.

3.3.2 Ferramenta script_LinkedIn

Nesta etapa, realizou-se o levantamento dos requisitos para a implementação da ferramenta de raspagem de dados do LinkedIn. A partir de uma análise detalhada dos processos existentes, identificou-se a necessidade de um sistema capaz de navegar automaticamente na plataforma, extrair informações relevantes dos perfis, como dados pessoais, informações acadêmicas, experiências profissionais e competências, e atualizar registros já armazenados sem gerar duplicidades. Nesta fase, tornou-se essencial conhecer o site e suas funcionalidades,

assim como sua estrutura HTML e seus elementos e lidar com desafios encontrados, como problemas de autenticação, *timeouts* e páginas dinâmicas. Verificou-se a necessidade de um sistema automatizado que pudesse:

- **Navegar automaticamente na plataforma:** acessar o LinkedIn, efetuar login (utilizando *cookies* para manter a sessão ativa ou realizando login automático quando necessário) e direcionar-se às páginas dos perfis dos ex-alunos de um curso específico.
- **Localizar e extrair elementos a partir do HTML:** coletar informações de diferentes seções (perfil, educação, experiência e competências), padronizando os dados e tornando-os compatíveis com os demais processos de análise.
- **Atualizar registros de forma inteligente:** comparar dados novos e já armazenados, evitando duplicações por meio de um identificador único (por exemplo, *aluno_id*).
- **Garantir resiliência e autonomia:** diante da volatilidade da interface do LinkedIn e de possíveis falhas no Selenium (como *timeouts* ou travamentos), o sistema deve possuir um mecanismo de recuperação automática, reiniciando a instância do Selenium e retomando a execução a partir da última URL visitada.

Dessa forma, o levantamento de requisitos orientou a proposta de uma solução capaz de tornar a automação mais robusta, atendendo às necessidades observadas na coleta manual de dados e contribuindo para a melhoria contínua das bases de informações dos ex-alunos.

Dessa forma, o levantamento de requisitos orientou a proposta de uma solução que alia automação à robustez, alinhada com as necessidades observadas na coleta manual de dados, contribuindo para a melhoria contínua das bases de informações dos ex-alunos.

3.4 Etapa 3: Testes

Durante a etapa de testes, foram realizadas diversas rodadas de experimentação para validar as funcionalidades, a robustez e a resiliência das ferramentas desenvolvidas, principalmente para a extração e atualização de dados de perfis do LinkedIn devido à complexidade envolvida. O processo de desenvolvimento não se limitou à escrita do código, mas envolveu uma abordagem iterativa, ajustando detalhes, incorporando tratativas para exceções e garantindo que as ferramentas fossem capazes de lidar com as peculiaridades dos sites, como as páginas dinâmicas do LinkedIn e os desafios inevitáveis da extração automatizada de informações.

Inicialmente, os testes concentraram-se na verificação dos fluxos básicos, como o login automático e a navegação entre as páginas. Foi escolhido o uso do Selenium como framework de Web Scraping; esses testes permitiram identificar problemas de carregamento e a necessidade de estratégias como o WebDriverWait, recurso que aguarda, de forma programática, até que determinados elementos sejam carregados ou certas condições sejam atendidas antes de prosseguir com a execução do script, aumentando a confiabilidade e a sincronização na coleta de dados. Além disso, foi necessário implementar comandos como:

```
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
```

para garantir que os elementos fossem carregados completamente antes da extração dos dados.

Paralelamente, foram avaliadas as técnicas de extração, principalmente os seletores XPath , que consistem em expressões capazes de localizar, de modo estruturado e preciso, elementos específicos em documentos HTML ou XML. Sua aplicação possibilitou a captura de informações referentes a perfil, formação, experiência e competências. Durante os ensaios, verificou-se que alguns seletores não eram suficientemente robustos, levando à necessidade de ajustes para lidar com a estrutura dinâmica das páginas do LinkedIn. Além disso, foi implementada uma função de tratamento de datas e testada em diversos cenários, resultando em melhorias para converter corretamente valores nulos e garantir a integridade dos dados extraídos.

Outro aspecto fundamental foi a verificação da correspondência entre os nomes extraídos e os dados presentes em fontes complementares (dados extraídos do SIGAA). Foram testadas abordagens para lidar com abreviações, acentuação e variações na composição dos nomes, aprimorando a função de comparação para considerar não apenas a igualdade textual, mas também a correspondência parcial de iniciais e nomes abreviados, que serão descritas nas seções seguintes.

A funcionalidade de atualização dos arquivos também foi exaustivamente testada. Conduziram-se ensaios para garantir que o sistema identificasse e removesse registros duplicados, mantendo apenas as informações mais recentes dos perfis com base no identificador único de cada aluno.

Por fim, testes de resiliência foram realizados, simulando falhas como timeouts e travamentos do Selenium. O mecanismo de reinicialização automática foi avaliado para garantir que a execução pudesse ser retomada a partir da última URL acessada, mantendo a sessão ativa por meio do carregamento de cookies previamente salvos, abordado nos próximos tópicos. Além

disso, foram implementadas rotinas de reinicialização do driver para assegurar a continuidade do processo diante de eventuais interrupções inesperadas.

Portanto, a etapa de testes foi decisiva para refinar as ferramentas, identificando e corrigindo problemas durante o desenvolvimento. Os resultados apresentados no próximo capítulo confirmam a viabilidade da abordagem adotada, demonstrando que as ferramentas são capazes de extrair, tratar e atualizar os dados dos perfis do LinkedIn de forma confiável e eficiente, atendendo aos requisitos estabelecidos e fornecendo uma base sólida para futuras análises e aprimoramentos.

3.5 Etapa 4: Extração dos dados

3.5.1 Extração dos dados da UFC

Nesta etapa, o objetivo foi a extração automatizada dos dados da situação acadêmica dos alunos da UFC a partir do SIGAA. Pretendeu-se substituir os processos manuais por uma solução eficiente, que coletasse, tratasse e armazenasse as informações de maneira estruturada e confiável. Identificou-se a necessidade de uma ferramenta que fosse capaz de realizar as seguintes funções:

- **Navegação e Seleção Dinâmica:** O sistema inicia com a leitura de um arquivo de configuração, que armazena parâmetros essenciais, como URLs e seletores (XPaths), e permite a seleção dos cursos considerados no relatório a partir de uma planilha Excel. Além disso, possibilita a definição do período (de 2015 a 2024) de interesse, dando autonomia ao usuário para definir o recorte temporal da extração e a escolha do curso.
- **Extração Automatizada:** A extração dos dados é realizada por meio do Selenium WebDriver, que automatiza a navegação no SIGAA e interage com os elementos da página. A função responsável, denominada `fetch_students`, executa a seleção do ano e do curso desejados, simula o clique no botão de busca e aguarda o carregamento completo da tabela contendo os registros dos alunos.
- **Tratamento e Consolidação dos Dados:** Uma vez extraídos, os dados – que incluem, entre outras informações, o nome do aluno e o status acadêmico – são organizados em um `DataFrame` do Pandas. O código trata casos de duplici-

dade, garantindo a integridade das informações. Posteriormente, os dados são consolidados e salvos nos formatos Excel e CSV. Em um primeiro momento, a ferramenta armazenava os dados seguindo uma estrutura de pastas organizada por curso, o que facilita a consulta e o uso posterior em análises e relatórios. Contudo, não era o melhor formato para o desenvolvimento posterior do relatório no Power BI, razão pela qual se decidiu salvar todos os cursos e anos no mesmo arquivo.

3.5.1.1 Desafios Encontrados e Soluções Implementadas

Durante o desenvolvimento, alguns desafios técnicos foram superados para garantir a robustez da ferramenta:

- **Instabilidade no Carregamento das Páginas:** Implementação de espera explícita (WebDriverWait) para evitar falhas de carregamento, pois foram encontradas lentidões na página do SIGAA. Com esse recurso, define-se um tempo máximo de espera para o surgimento dos elementos no site.
- **Problemas de Sobrescrita de Dados** Implementação da estratégia de atualização seletiva na função `save_data`, garantindo que apenas os dados mais recentes sejam preservados em uma eventual reexecução do script.

3.5.1.2 Estratégias Utilizadas para a Extração

A extração de dados foi planejada considerando os desafios técnicos da navegação no SIGAA:

- **Carregamento Dinâmico:** Utilização de WebDriverWait para garantir que os elementos sejam completamente carregados antes da extração.
- **Parametrização dos Seletores:** Seletores XPath foram parametrizados via arquivo de configuração, permitindo rápida adaptação a mudanças no HTML do SIGAA.

3.5.1.3 Estrutura do Código e Principais Funções

A implementação se baseou em uma série de funções desenvolvidas em Python com Selenium, que juntas automatizaram a coleta e a organização dos dados, e os códigos podem ser

encontrados no apêndice.

- `load_config`: Lê o arquivo de configuração contendo parâmetros essenciais como URLs, caminhos de acesso e seletores XPath.
- `load_courses`: Carrega a lista de cursos disponíveis a partir de uma planilha.
- `get_year_selection`: Dá autonomia ao usuário para definir o recorte temporal da extração (de 2015 a 2024).
- `fetch_students`: Direciona o Selenium para a página do SIGAA e interage com os elementos da interface para selecionar o ano, o curso e disparar a busca dos alunos.
- `save_data`: Consolida os dados extraídos sem sobrescrever informações de outros cursos, removendo duplicatas e garantindo a integridade.

A etapa de extração dos dados da UFC estabeleceu os fundamentos para a criação de uma base de dados robusta e atualizada, essencial para a construção de relatórios e análises acadêmicas. As dificuldades encontradas, como o carregamento dos elementos, problemas de formatação e atualização de registros, foram superadas por meio de técnicas de espera, tratamento de exceções e funções de manipulação de dados. Esse conjunto de soluções não atesta a viabilidade da automação, mas também reforça a importância de uma implementação iterativa, flexível e adaptável aos desafios do ambiente web.

3.5.2 Extração dos dados do Linkedin

A etapa de extração dos dados dos perfis do LinkedIn foi fundamental para complementar as informações obtidas do SIGAA, proporcionando uma visão mais abrangente sobre a trajetória acadêmica e profissional dos alunos e ex-alunos da UFC. Este processo envolveu a criação de uma nova ferramenta “Script_Linkedin.py” para a automação da coleta de dados abertos e disponíveis nos perfis do LinkedIn, enfrentando desafios específicos relacionados à dinâmica da plataforma e à autenticação.

3.5.2.1 Desafios Encontrados e Soluções Implementadas

- **Autenticação e Manutenção de Sessão:** O LinkedIn possui mecanismos de segurança que dificultam a automação de logins. Para contornar esse obstáculo, implementou-se o uso de cookies para manter a sessão ativa, reduzindo a necessidade de autenticações repetidas. Além disso, desenvolveu-se uma rotina de login

automatizado para os casos em que os cookies expirassem ou não estivessem disponíveis.

- **Carregamento Dinâmico de Conteúdo:** A estrutura dinâmica das páginas do LinkedIn, com elementos carregados de forma assíncrona, exigiu a adoção de técnicas de espera explícita, como o uso de *WebDriverWait*, garantindo que todos os elementos estivessem completamente carregados antes da extração.
- **Variação na Estrutura dos Perfis:** As diferenças nos layouts dos perfis apresentaram desafios na localização consistente dos dados. Para superar isso, foram utilizados seletores *XPath* parametrizados, permitindo ajustes rápidos conforme alterações na estrutura da página.
- **Correspondência de Nomes:** Diante dos desafios de variação nos nomes, foram desenvolvidas funções específicas para gerar combinações de nomes e realizar comparações que considerem abreviações e diferenças na composição dos nomes. Essa abordagem foi crucial para estabelecer a correspondência entre o perfil extraído e os registros dos egressos da UFC, possibilitando o cruzamento de informações no relatório do Power BI.
- **Processamento das Datas:** Apresentou desafios particulares, pois os dados de tempo estavam em formatos textuais variados, precisando ser convertidos de forma padronizada para que pudessem ser comparados e utilizados na consolidação dos dados. Para isso, foi desenvolvida a função `processar_tempo`, que mapeia as abreviações e converte as datas para um formato uniforme.

3.5.2.2 Estratégias Utilizadas para a Extração

- **Automação com Selenium:** Utilizou-se o *Selenium WebDriver* para automatizar a navegação e interação com as páginas do LinkedIn, configurando opções específicas para contornar bloqueios e garantir a estabilidade da sessão.
- **Gerenciamento de Sessão:** A manutenção da sessão ativa foi alcançada por meio do armazenamento e carregamento de cookies, minimizando a necessidade de novos logins e reduzindo o risco de bloqueios por parte da plataforma.
- **Extração Estruturada de Dados:** Implementaram-se funções dedicadas para extrair informações específicas, como dados de identificação, formação acadêmica, experiências profissionais e competências, utilizando seletores precisos e

técnicas de tratamento de exceções para lidar com possíveis inconsistências.

- **Tratamento e Consolidação dos Dados:** Após a extração, os dados são organizados em *DataFrames* utilizando o Pandas. A função `processar_tempo` foi fundamental para padronizar a conversão de datas, enquanto funções auxiliares para a normalização dos nomes permitiram a correta correspondência entre os perfis extraídos e os registros existentes dos alunos e egressos da UFC.
- **Armazenamento Organizado:** Os dados coletados foram estruturados em *DataFrames* e salvos em formatos como Excel e CSV, seguindo uma organização em pastas que facilita futuras análises e atualizações.

3.5.2.2 Estrutura do Código e Principais Funções

- `start_driver` e `setup_driver`: Responsáveis por configurar e inicializar o *Selenium WebDriver*, incluindo opções que ignoram erros de SSL e garantem a estabilidade da conexão.
- `auto_login`, `save_cookies` e `load_cookies`: Essas funções gerenciam o processo de autenticação, utilizando cookies para manter a sessão ativa e reduzindo a necessidade de login manual, garantindo maior autonomia e continuidade na extração dos dados.
- `safe_get` e `restart_selenium`: Funções auxiliares que asseguram o carregamento completo da página e a recuperação da sessão em caso de falhas, contribuindo para a resiliência do sistema.
- `processar_tempo`: Fundamental para padronizar e converter os dados de data, permitindo que formatos diversos sejam transformados em objetos *datetime*.
- `generate_name_combinations` e `check_nome`: Auxiliam na identificação correta dos perfis, considerando variações, abreviações e diferenças na composição dos nomes.
- **Funções de Extração:** As funções `extract_profile_data`, `extract_education_data`, `extract_experience_data` e `extract_competence_data` foram desenvolvidas para capturar, tratar e organizar as informações dos perfis do LinkedIn. Entre elas, a função `processar_tempo` se destaca por padronizar a conversão dos dados textuais referentes a datas, superando a variabilidade encontrada nos formatos apresentados.

- `visit_profiles`: Gerencia a navegação entre os perfis, controlando a quantidade de perfis a serem visitados com base na entrada do usuário e garantindo que os links já visitados não sejam processados novamente.
- `update_or_append_file` e `save_to_file`: Responsáveis por consolidar os dados extraídos e armazená-los em arquivos estruturados (Excel e CSV), organizados em uma hierarquia de pastas que facilita a análise e a atualização contínua do banco de dados.

A implementação da extração de dados do LinkedIn, apesar dos desafios encontrados na automação em plataformas dinâmicas e protegidas, resultou em uma ferramenta robusta capaz de extrair e atualizar informações relevantes dos perfis de LinkedIn dos alunos e ex-alunos da UFC. As estratégias adotadas, como o uso de cookies para manutenção de sessão, técnicas de espera explícita e extração estruturada de dados, asseguraram a eficácia do processo. Com isso, foi criada uma base de dados enriquecida para análises futuras e para apoiar decisões estratégicas no âmbito acadêmico e profissional.

É importante destacar que, embora tenha sido encontrada uma solução para relacionar cada aluno de uma turma de um curso da UFC, durante os testes realizados nem todos os estudantes foram identificados. Isso evidencia algumas limitações que ainda podem ser superadas em trabalhos futuros, tais como:

- Nem todos os alunos e egressos necessariamente possuem contas na plataforma LinkedIn, e todos os dados apresentados no próximo capítulo são baseados em um espaço amostral.
- A relação entre alunos e egressos é feita por meio dos nomes extraídos da página do SIGAA e dos nomes encontrados nos perfis da página da UFC no LinkedIn, juntamente com o curso no filtro. Além disso, o ano da turma da pessoa na base de dados da UFC deve ser o mesmo ano que o usuário coloca na seção de Formação Acadêmica no LinkedIn referente à formação na Universidade Federal do Ceará. No entanto, como este segundo é um campo aberto, o usuário pode inserir outro ano ou simplesmente não colocar a data, dificultando o relacionamento.
- Por fim, a combinação para o relacionamento utiliza tanto o nome da pessoa quanto o nome do curso na base de dados da UFC e na plataforma como premissas. No entanto, o usuário na plataforma tem a liberdade de escrever seu nome e o nome do curso como desejar, podendo, por exemplo, colocar o nome do curso

em outro idioma.

Apesar dessas limitações, como será abordado no capítulo subsequente, foi encontrado um número relevante de perfis, o que fundamenta a construção do relatório no Power BI.

3.5.3 Atualização dos dados existentes do Linkedin

A atualização dos dados dos perfis no LinkedIn tornou-se essencial no fluxo de trabalho, pois os perfis de alunos e egressos estão sempre mudando. Foi desenvolvida uma ferramenta específica, partindo do princípio de que os perfis já foram coletados; basta acessar suas páginas e atualizar os dados existentes. Esse módulo foi criado para garantir que as informações extraídas sejam revisadas periodicamente e incorporadas ao banco de dados, evitando duplicidades e assegurando a integridade dos registros que servirão de base para indicadores estratégicos.

3.6 Etapa 5: Construção do relatório de BI

Nessa etapa, o foco foi sobre a construção do relatório de dados, a partir das extrações realizadas nas fases anteriores. Com as extrações feitas, é possível ter uma análise holística de todos os dados obtidos e iniciar o planejamento da modelagem de dados, porém, antes, foi necessário fazer uma definição dos indicadores e requisitos de visualização. Com a visão de todos os dados, é possível definir quais métricas seriam relevantes para o objetivo do relatório e quais visualizações seriam necessárias para tanto.

Assim, a etapa foi dividida em três partes:

- **Definição de Indicadores e Visualizações**
- **Modelagem de Dados**
- **Construção das Visualizações**

3.6.1 Definição de Indicadores e Visualizações

Com as bases de dados extraídas, `UFC_Status` (dados de situação acadêmica dos estudantes) e as tabelas resultantes da extração do LinkedIn (`experience`, `education`, `profiles` e `competence`), foi possível estabelecer uma base estruturada para a construção de indicadores. Segue abaixo as tabelas e seus campos:

Tabela 1 – Estrutura da Tabela *UFC_Status*

Nome do Campo	Tipo de Dado	Descrição	Restrições
`id`	INT	Identificador único do aluno	Chave Primária
`Curso`	VARCHAR	Nome do curso	Não Nulo
`Ano`	INT	Ano de ingresso	Não Nulo
`Nome`	VARCHAR	Nome completo do aluno	Não Nulo
`Situação`	VARCHAR	Status do aluno (Ativo, Concluído, etc.)	Não Nulo
`linkedin`	INT	Indicador de presença no LinkedIn (1 = Sim, -1 = Não)	Pode ser nulo

Fonte: o autor.
Nota: Tabela que relaciona informações acadêmicas e status dos alunos da UFC.
Anotações: Possibilita filtrar alunos ativos, concluídos e identificados no LinkedIn.

Tabela 2 – Estrutura da Tabela *Experience*

Nome do Campo	Tipo de Dado	Descrição	Restrições
`col_tipo`	VARCHAR	Tipo de experiência (ex.: "experiências")	Não Nulo
`col_link_empresa`	VARCHAR	Link para página da empresa no LinkedIn	Pode ser nulo
`col_img_empresa`	VARCHAR	URL da imagem (logo) da empresa no LinkedIn	Pode ser nulo
`col_nome_empresa`	VARCHAR	Nome da empresa	Não Nulo
`col_cargo`	VARCHAR	Cargo ocupado	Não Nulo
`col_tempo`	VARCHAR	Período em formato textual (ex.: "De mar de 2016 até set de 2017")	Pode ser nulo
`Nome`	VARCHAR	Nome do usuário associado	Não Nulo
`link_perfil`	VARCHAR	Link para o perfil profissional	Não Nulo
`aluno_id`	INT	Identificador único do usuário	Chave Estrangeira
`data_inicio`	DATETIME	Data de início da experiência	Pode ser nulo
`data_fim`	DATETIME	Data de fim da experiência	Pode ser nulo

Fonte: o autor.
Nota: Campos utilizados para armazenar informações profissionais do usuário.
Anotações: Pode ser estendida para outras formas de experiência, como voluntariado.

Tabela 3 – Estrutura da Tabela *Education*

Nome do Campo	Tipo de Dado	Descrição	Restrições
`Tipo`	VARCHAR	Tipo de formação (ex.: "Formação Acadêmica")	Não Nulo
`Local`	VARCHAR	Instituição de ensino	Não Nulo
`Ocupação`	VARCHAR	Curso ou formação específica	Não Nulo
`Tempo`	VARCHAR	Período em formato textual (ex.: "mar de 2018 - jul de 2023")	Pode ser nulo
`Nome`	VARCHAR	Nome do usuário associado	Não Nulo
`link_perfil`	VARCHAR	Link para o perfil profissional	Não Nulo
`Curso`	VARCHAR	Nome do curso (ex.: ENGENHARIA ELÉTRICA)	Não Nulo
`data_inicio`	DATETIME	Data de início da formação	Pode ser nulo
`data_fim`	DATETIME	Data de fim da formação	Pode ser nulo
`Ano`	INT	Ano de ingresso ou ano base	Pode ser nulo
`aluno_id`	INT	Identificador único do usuário	Chave Estrangeira

Fonte: o autor.
Nota: Informações referentes à formação acadêmica ou cursos complementares.
Anotações: Os cursos podem está escrito em outro idioma.

Tabela 4 – Estrutura da Tabela *Profiles*

Nome do Campo	Tipo de Dado	Descrição	Restrições
`Nome`	VARCHAR	Nome completo do usuário	Não Nulo
`Link_img`	VARCHAR	URL da foto de perfil do usuário	Pode ser nulo
`Subtítulo`	VARCHAR	Descrição breve do usuário	Pode ser nulo
`link_perfil`	VARCHAR	Link para o perfil profissional	Não Nulo
`aluno_id`	INT	Identificador único do usuário	Chave Estrangeira

Fonte: o autor.
Nota: Tabela principal que agrupa informações gerais do perfil do usuário.

Tabela 5 – Estrutura da Tabela *Competence*

Nome do Campo	Tipo de Dado	Descrição	Restrições
`Tipo`	VARCHAR	Categoria da competência (ex.: "Competências")	Não Nulo
`Competencia`	VARCHAR	Nome da competência	Não Nulo
`Nome`	VARCHAR	Nome do usuário associado	Não Nulo
`link_perfil`	VARCHAR	Link para o perfil profissional	Não Nulo
`aluno_id`	INT	Identificador único do usuário	Chave Estrangeira

Fonte: o autor.
Nota: Utilizada para armazenar competências e habilidades declaradas pelos usuários.

Portanto, primeiramente, pensou-se na definição dos indicadores principais, responsáveis por fornecer uma visão quantitativa abrangente dos alunos presentes no banco de dados:

- **Quantidade total de alunos**, permitindo uma visão global da população analisada;
- **Quantidade de perfis no LinkedIn e sua proporção em relação ao total**, importante para saber o espaço amostral;
- **Quantidade de alunos que concluíram o curso e a porcentagem deles que estão no LinkedIn**, importante para compreender a inserção dos egressos no mercado de trabalho;
- **Quantidade de alunos ativos e a porcentagem deles que estão no LinkedIn**, possibilitando a análise da interação dos alunos em formação com a rede profissional;
- **Quantidade de alunos que cancelaram e a porcentagem deles que estão no LinkedIn**, contribuindo para a avaliação do impacto do abandono na continuidade profissional.

Com base nesses dados, foi realizada a segmentação dos alunos em grupos relacionados à sua situação acadêmica e seu status no mercado de trabalho, resultando em indicadores adicionais:

- Quantidade de alunos que concluíram e estão trabalhando;
- Quantidade de alunos que concluíram e não estão trabalhando;
- Quantidade de alunos que estão ativos e trabalhando;
- Quantidade de alunos que estão ativos e não estão trabalhando;
- Quantidade de alunos que cancelaram e estão trabalhando;
- Quantidade de alunos que cancelaram e não estão trabalhando.

Além disso, com o intuito de compreender melhor a qualificação dos alunos e sua relação com o mercado de trabalho, foram definidas visualizações de competências, ranqueando-as conforme a quantidade de alunos que as possuem, e representações de empresas, classificando-as de acordo com o número de alunos e egressos que atuaram nelas. Para os indicadores quantitativos principais, optou-se por gráficos de colunas, enquanto para competências e empresas utilizou-se uma organização ranqueada em tabelas, facilitando a identificação de tendências.

3.6.2 Modelagem dos dados

A modelagem de dados é uma etapa fundamental na construção de relatórios de BI, pois estabelece a estrutura que organiza e relaciona as diversas tabelas, garantindo cálculos

precisos, criação de medidas e interatividade nas visualizações. No contexto de modelagem, existe a classificação das tabelas em "tabelas dimensão"e "tabelas fato", elas têm papéis bem definidos: as tabelas dimensão armazenam atributos descritivos (como informações dos alunos), enquanto as tabelas fato guardam eventos ou dados mensuráveis, que podem estar relacionados a atividades profissionais, acadêmicas ou comportamentais dos perfis do LinkedIn dos alunos.

Outro ponto importante é a criação de tabelas derivadas no Power BI, utilizando o *Power Query*. Esse recurso possibilita a transformação e o enriquecimento dos dados a partir de outras tabelas já existentes, sem a necessidade de alterar a fonte original. No caso específico, criou-se uma tabela de dimensão de empresas, integrando informações estratégicas relevantes para as análises dos dados do LinkedIn e para futuros indicadores. Essa prática não apenas aprimora o modelo ao adicionar novas perspectivas analíticas, como também facilita a manutenção e a escalabilidade do sistema, visto que as transformações realizadas via *Power Query* podem ser atualizadas e replicadas conforme a evolução dos indicadores e das fontes de dados.

3.6.3 Construção de Visualizações

Por fim, a construção das visualizações compreende tanto a elaboração dos gráficos quanto a criação de medidas em DAX (*Data Analysis Expressions*). Essas medidas possibilitam a criação de indicadores precisos e interativos, fundamentais para a análise dos dados, empregando funções e expressões capazes de calcular, filtrar e agregar as informações de forma dinâmica.

Com isso, parte-se para a escolha de visualizações que façam sentido para a construção dos indicadores e para a interpretação estratégica das informações coletadas. Os resultados foram organizados e integrados em *dashboards* interativos desenvolvidos no Power BI, os quais foram divididos em quatro visões distintas, cada uma com foco em um aspecto específico da análise. Essa segmentação visa facilitar a compreensão dos dados e apoiar a tomada de decisões pelos gestores acadêmicos e de mercado.

Páginas do Relatório:

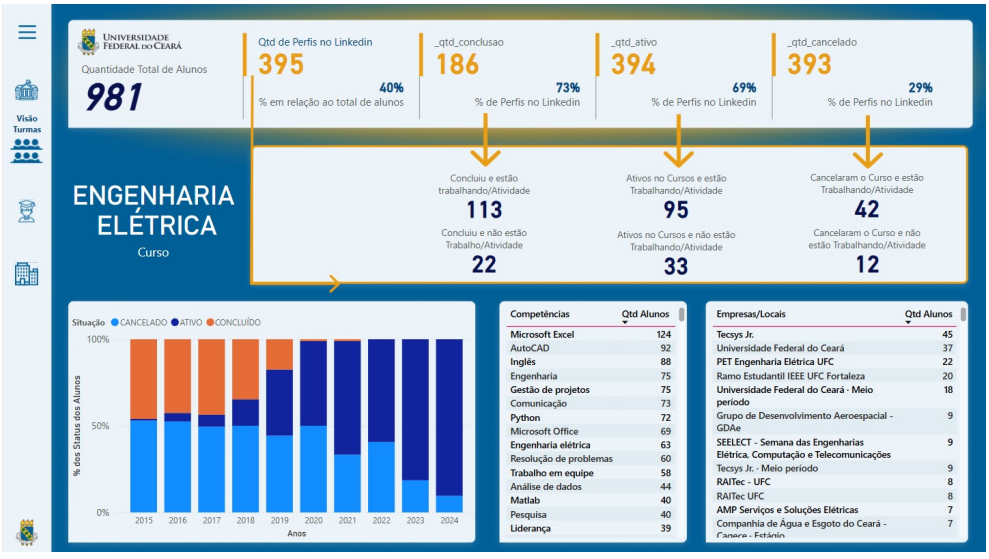
- **Visão Cursos:** Página focada em uma visualização mais macro dos indicadores, agrupados em turmas e permitindo comparações entre os cursos.

Figura 17 – Página no BI de Visão Cursos

Fonte: elaborado pelo autor (2025).

- **Visão Turmas:** Página com visualização semelhante à de "Cursos", mas que apresenta indicadores específicos que mostram se os alunos estão trabalhando ou não, além de um gráfico de comparação entre as turmas em relação ao status acadêmico e tabelas de empresas e competências classificadas pela quantidade de alunos.

Figura 18 – Página no BI de Visão Turmas



Fonte: elaborado pelo autor (2025).

- **Visão Empresas:** Página focada em dados sobre empresas, incluindo o número de alunos que trabalharam nelas, quais alunos atuaram em determinadas empresas, um gráfico de média de dias trabalhados por turma e tabelas de competências

relacionadas à quantidade de alunos. Todas essas visualizações podem ser filtradas pelo nome das empresas.

Figura 19 – Página no BI de Visão Empresas

Fonte: elaborado pelo autor (2025).

3.7 Considerações finais

Ao longo deste capítulo, foram descritas as etapas de extração de dados a partir do SIGAA e do LinkedIn, o levantamento de requisitos para ambas as ferramentas, a realização de testes para validar a robustez e a confiabilidade dos métodos de extração, além da idealização e implementação do relatório de BI. Com isso, demonstrou-se a importância de se adotar uma abordagem cuidadosa para lidar com plataformas dinâmicas, como o LinkedIn, e com dados que se encontram em constante transformação.

A adoção de técnicas de Web Scraping, aliada a uma estrutura sólida de modelagem de dados, permitiu não apenas a coleta das informações, mas também o tratamento necessário para garantir a qualidade e a confiabilidade dos dados. Foram implantadas estratégias para lidar com problemas de autenticação, carregamento dinâmico de conteúdo, correspondência de nomes e atualização periódica de dados.

Por fim, a construção de relatórios no Power BI permitiu que as informações extraídas fossem apresentadas de maneira nítida e orientada a indicadores. Dessa forma, a comunidade acadêmica pode contar com uma base analítica mais rica, capaz de gerar insights sobre a trajetória dos alunos e egressos da Universidade.

No próximo capítulo, serão expostos os resultados obtidos a partir dessas ferramentas

de extração, tratamento e visualização dos dados, bem como as análises resultante desses processos.

4 ANÁLISES

Este capítulo apresenta e discute os principais resultados obtidos a partir das visualizações e indicadores do BI, construídos com base nos dados extraídos do SIGAA e do LinkedIn por meio das ferramentas de web scraping. A intenção é analisar, de forma fundamentada e comparativa, como determinadas informações acadêmicas e profissionais se manifestam em diferentes cursos de Engenharia de Computação, Engenharia Elétrica e Engenharia de Telecomunicações de modo a entender a trajetória dos estudantes, sua inserção no mercado de trabalho e o impacto de competências específicas em suas carreiras.

Para facilitar a compreensão, a análise está organizada em tópicos que refletem diferentes dimensões do perfil estudantil e profissional:

- **Perfil e Trajetória Acadêmica dos Estudantes:** Observa a evolução dos alunos durante o curso, correlacionando dados de matrículas, conclusões e eventuais desistências.
- **Empregabilidade e Inserção no Mercado de Trabalho:** Apura quantos egressos estão efetivamente trabalhando e em quais atividades ou empresas, apontando as possíveis razões de maior (ou menor) empregabilidade.
- **Análise de Competências e Habilidades:** Examina a distribuição das competências mais mencionadas nos perfis, correlacionando-as com indicadores de empregabilidade e relevância no mercado.

4.1 Perfil e Trajetória Acadêmica dos Estudantes – Engenharia de Telecomunicações

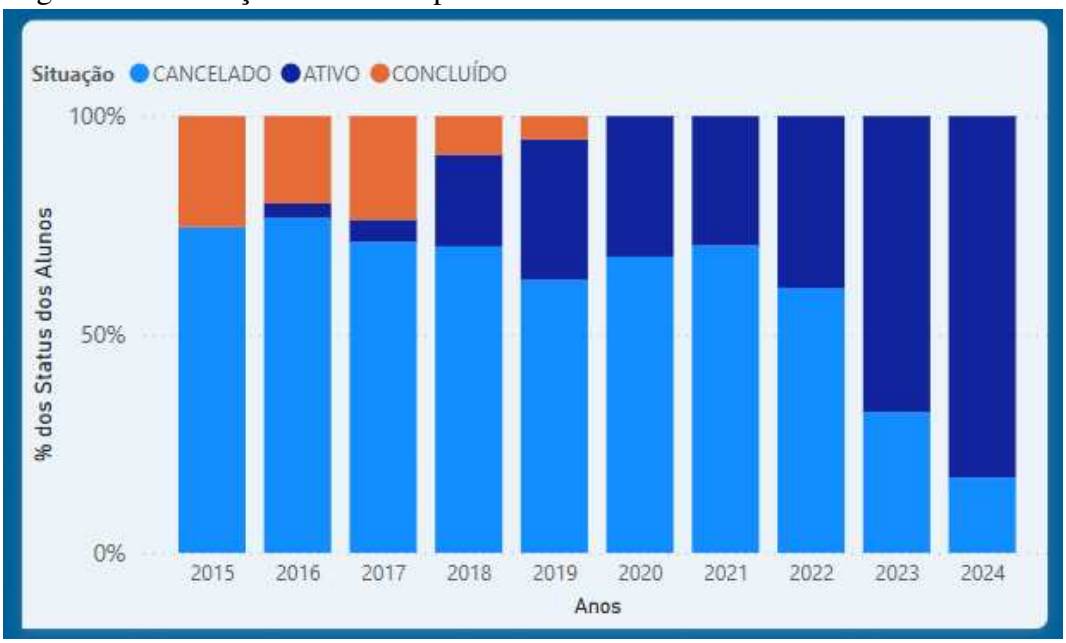
Ao observar os dados referentes às matrículas de 2015 a 2024 para Engenharia de Telecomunicações, percebe-se um perfil acadêmico marcado por uma taxa expressiva de cancelamentos. O gráfico de colunas empilhadas na Figura 20 exibe a proporção de alunos ativos, concluídos e cancelados em cada Turma, indicando que, na maioria das turmas, mais de 50% dos estudantes acabam cancelando o curso. Essa alta incidência de cancelamentos sugere, por um lado, a possível dificuldade dos estudantes em prosseguir nos semestres seguintes e, por outro, a necessidade de avaliar possíveis fatores que desencadeiam desistências, como carga horária intensa ou menor afinidade com a área.

Outro ponto a destacar é a existência de matrículas ainda ativas em anos distantes (como 2016, 2017 e 2018) que pode ser visto pela Figura 21, mesmo considerando que a

formação tem duração prevista de cinco anos. A presença desses alunos pode indicar situações de prolongamento da graduação, seja por reprovações em disciplinas, trancamentos pontuais ou mudanças no ritmo de estudos.

Já na Figura 22 é mostrada a comparação entre a quantidade de concluintes e cancelados, reforçando a disparidade quantitativa entre esses grupos. Enquanto o número de formados em determinadas turmas se mostra relativamente baixo, a soma de cancelamentos é nitidamente superior. Essa diferença notável enfatiza o desafio de manter estudantes engajados até a conclusão do curso.

Figura 20 – Situação acadêmica por turma

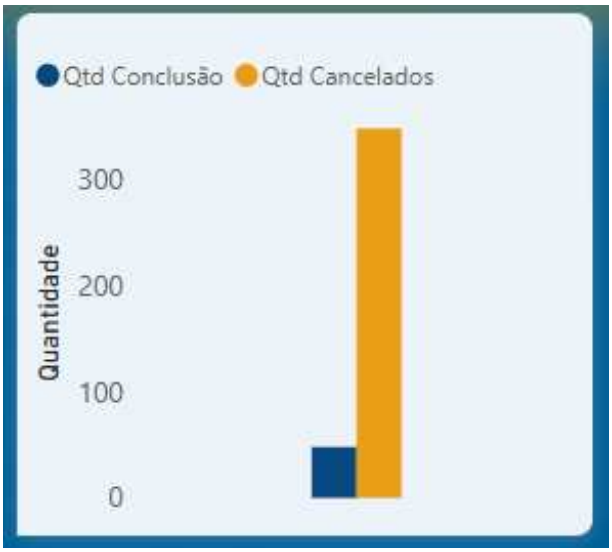


Fonte: elaborado pelo autor (2025).

Figura 21 – Porcentagem de ativos por turma

Fonte: elaborado pelo autor (2025).

Figura 22 – Quantidade de concluintes e de cancelados no total



Fonte: elaborado pelo autor (2025).

Portanto, os resultados demonstram que a Engenharia de Telecomunicações enfrenta desafios significativos, com taxas elevadas de evasão e um número considerável de alunos que permanecem além do período ideal de formação. Nesse contexto, a aplicação de *Business Intelligence* (BI) revela-se fundamental, pois os *dashboards* interativos permitem a visualização detalhada dos indicadores e facilitam a identificação de padrões críticos. Através da consolidação dos dados, como a proporção de alunos ativos, concluídos e cancelados, o BI fornece insumos que podem orientar a coordenação acadêmica na implementação de políticas de acompanhamento e estratégias de apoio.

4.2 Empregabilidade e Inserção no Mercado de Trabalho – Engenharia de Computação

Na análise de Engenharia de Computação, os dados exibidos na Figura 23 indicam que, dos 101 alunos que concluíram o curso, 52% foram encontrados no LinkedIn até então. Destes, 47 já estão exercendo alguma atividade profissional, enquanto 6 não apresentam qualquer vínculo de trabalho atualmente. Esse contraste sugere que, embora a maioria dos egressos tenha se posicionado no mercado ou em projetos acadêmicos/profissionais, ainda existe uma parcela que, por motivos diversos, não registra atividade atual. Ressalta-se, porém, a possibilidade de perfis desatualizados na plataforma, o que pode subestimar o real índice de empregabilidade.

Já para os alunos ativos, que somam 278 alunos, a proporção de perfis no LinkedIn sobe para 84% como mostra na Figura 24. Esse incremento sugere uma adesão crescente à rede

por parte dos estudantes mais novos, possivelmente motivada pela busca de oportunidades de estágio, vivências em empresas juniores ou atividades de pesquisa e extensão. Do total de ativos no LinkedIn, 74 estão se envolvendo em alguma atividade profissional ou acadêmica, reforçando o interesse de grande parte dos discentes em vivenciar o mercado ou projetos práticos ainda durante a graduação. Em contrapartida, 11 não exibem registro de trabalho, o que pode indicar dedicação exclusiva aos estudos ou, novamente, perfis não atualizados.

De forma geral, esses indicadores apontam para uma trajetória de empregabilidade relativamente positiva, sobretudo para alunos que ainda não concluíram o curso e já buscam experiência prática, mas que, por outro lado, podem postergar a conclusão no curso . A presença expressiva no LinkedIn serve de ponte para contatos profissionais, e os números indicam que muitos estudantes aproveitam esse potencial. Ainda assim, é essencial considerar que parte dos alunos pode não manter seu perfil atualizado, tornando necessário um olhar crítico sobre os dados para que, no futuro, a coordenação do curso e demais gestores acadêmicos possam traçar ações mais efetivas de apoio ao registro de informações como forma de aumentar as oportunidades de inserção no mercado de trabalho.

Figura 23 – Indicadores de conclusão e empregabilidade em Engenharia de Computação

Fonte: elaborado pelo autor (2025).

Figura 24 – Indicadores de ativos e empregabilidade em Engenharia de Computação

Fonte: elaborado pelo autor (2025).

4.3 Análise de Competências e Habilidades – Engenharia Elétrica

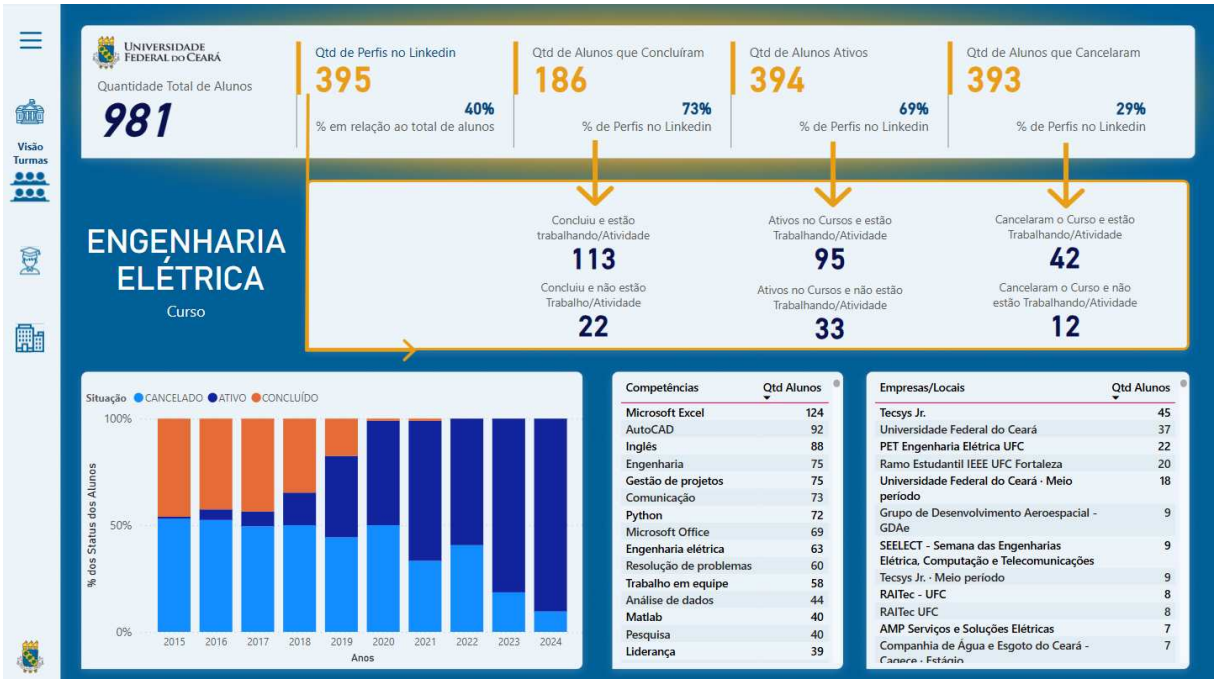
O curso de Engenharia Elétrica, como pode ser visto na Figura, contabilizou-se o total de 981 alunos matriculados, e foram identificados 395 perfis no LinkedIn. Entre os 186 alunos que concluíram o curso, 73% marcam presença na plataforma, com 113 já inseridos em alguma atividade profissional e 22 ainda sem registro de ocupação. No grupo de 394 alunos ativos, 69% possuem perfis no LinkedIn, sendo que 95 já demonstram envolvimento em alguma atividade,estágio, empresa júnior ou pesquisa, enquanto 33 não apresentam qualquer menção profissional.

A tabela de competências, extraída desses perfis, revela uma combinação interessante de habilidades técnicas e gerenciais. Recursos como Excel, AutoCAD e Python destacam-se como ferramentas fundamentais. Ao mesmo tempo, *soft skills* como liderança, trabalho em equipe e resolução de problemas evidenciam a busca por engenheiros capazes de atuar em grupos multidisciplinares e em ambientes que demandam criatividade e comunicação eficaz. Observa-se ainda a recorrência de conhecimentos voltados à gestão de projetos, indicando a relevância de

profissionais que conciliem formação técnica com aptidões de planejamento e coordenação.

Essas informações sugerem que o mercado de Engenharia Elétrica valoriza um perfil profissional versátil, capaz de transitar tanto em funções mais gerais de engenharia e liderança quanto em áreas específicas de programação ou domínio técnico elétrico. Por fim, o fato de uma parcela significativa dos estudantes já se encontrar em atividades profissionais ou acadêmicas reforça a utilidade do LinkedIn como ferramenta para *networking* e para a apresentação das competências adquiridas. Ferramentas como esta, portanto, podem apoiar a tomada de decisões pelas coordenações de curso ou pela administração superior da universidade no que concerne à aquisição de competências e habilidades para a formação profissional.

Figura 25 – Visão geral das turmas Engenharia Elétrica



Fonte: elaborado pelo autor (2025).

5 CONCLUSÕES E PERSPECTIVAS

Ao longo deste trabalho, foram desenvolvidos ferramentas automatizadas em python para a extração, consolidação e atualização de dados dos perfis do LinkedIn e dados do SIGAA, com o objetivo de construir uma base robusta para o monitoramento das trajetórias acadêmicas e profissionais de alunos e egressos da UFC. O projeto teve seu início na identificação de uma demanda real: De acompanhar as rápidas mudanças do mercado de trabalho e a dificuldade de se obter, de maneira manual e sistemática, informações atualizadas sobre a situação dos alunos e egressos. Essa necessidade motivou a integração de técnicas de web scraping, utilizando o Selenium, possibilitando a construção de uma base de dados e de indicadores que abrangem desde a trajetória acadêmica até a inserção dos perfis no mercado de trabalho e a análise de competências.

Durante o desenvolvimento, o projeto enfrentou desafios inerentes à dinâmica das páginas web, como a complexidade dos seletores XPath, a variabilidade dos formatos de dados, especialmente a padronização de datas, e problemas relacionados a autenticação e timeouts. Estratégias como o uso de WebDriverWait, mecanismos de recuperação automática e o gerenciamento de *cookies* foram necessárias para superar esses desafios. Os testes realizados evidenciaram ganhos significativos em termos de eficiência e qualidade na extração e atualização dos dados, mostrando a viabilidade da solução proposta mesmo em ambientes complexos e voláteis.

Além disso, a integração dos resultados em *dashboards* interativos no Power BI possibilitou a visualização dos indicadores, fornecendo insumos valiosos para apoiar decisões estratégicas por parte dos gestores acadêmicos.

Apesar dos avanços alcançados, o trabalho apresenta alguns limites, como a dependência dos usuários em manter seus perfis atualizados no LinkedIn e as dificuldades de estabelecer correspondências perfeitas entre registros do SIGAA com o LinkedIn. Esses desafios apontam para oportunidades de aprimoramento. Trabalhos futuros poderão explorar a incorporação de técnicas para melhorar a correspondência de perfis, ampliar a integração com outras fontes de dados, como dados das empresas, e refinar os processos de busca e atualização, deixando-os mais assertivos. Essas melhorias não apenas potencializariam a qualidade dos indicadores, mas também serviriam de base para inovações e estratégias, contribuindo para o desenvolvimento contínuo do ambiente acadêmico e a formação de profissionais alinhados com as demandas do mercado.

Diante disso, o projeto demonstrou que é possível automatizar com sucesso o processo de extração e atualização de dados em ambientes desafiadores, gerando informações estratégicas que podem orientar decisões institucionais e fomentar novas pesquisas e aplicações práticas no monitoramento do alunos no mercado de trabalho.

REFERÊNCIAS

BADARó, P. C. **Web Scraping e Análise de Dados no Aperfeiçoamento do Processo Seletivo de Programadores**. 2023. <http://monografias.ice.ufjf.br/tcc-web/exibePdf?id=756>. Monografia. ICE - UFJF.

CNN. **Com geração Z, mais de 60% da força de trabalho nacional tem um perfil no LinkedIn**. 2024. Publicado em 09 jun. 2024, 07:30. Acesso em: 24 fev. 2025. Disponível em: <https://www.cnnbrasil.com.br/economia/negocios/com-geracao-z-mais-de-60-da-forca-de-trabalho-nacional-tem-um-perfil-no-linkedin/>.

CORDEIRO, J. F. P. **História da educação: de Confúcio a Paulo Freire**. 2. ed. [S.l.]: História da educação, 2021.

CORDãO, F. A.; MORAES, F. de. **Educação profissional no Brasil: síntese histórica e perspectivas**. [S.l.]: Editora SENAC São Paulo, 2020.

FORUM, W. E. **The Future of Jobs Report 2025**. 2025. <https://www.weforum.org/reports/the-future-of-jobs-report-2025>. Acesso em: 24 fev. 2025.

GLEZ-PEÑA, D. *et al.* Web scraping technologies in an api world. **Briefings in Bioinformatics**, Oxford University Press, v. 15, n. 5, p. 788–797, 2014.

LINKEDIN. **Sobre o LinkedIn**. 2025. Disponível em: <https://about.linkedin.com/pt-br>. Acesso em: 24 fev. 2025.

MEIRELES, D. P. **Business Intelligence e automação de pipeline de dados: Implementação para o Cross-S**. 2022. <http://link-da-monografia>.

Microsoft. **Visão geral do Power BI**. 2024. <https://learn.microsoft.com/pt-br/power-bi/fundamentals/power-bi-overview>. Acesso em: 25 abr. 2025.

NUNES, R. I. da S. **Uso do Power BI para Tratamento de Dados de Aviação Operacional no CBMDF**. 2021. <https://biblioteca.cbm.df.gov.br/jspui/bitstream/123456789/263/1/1150%20-%20TCC%20-Raquel.pdf>. Monografia. CBM DF.

RODRIGUES, S. V. **Uma Análise do LinkedIn como ferramenta social e profissional no grupo de adiministradores**. 2018. <https://repositorio.ufu.br/handle/123456789/24841>. Monografia. Acesso em: 24 fev. 2025.

SCHWAB, K. **A Quarta Revolução Industrial**. Genebra, Suíça: World Economic Forum, 2016. Tradução de: Daniel Moreira Miranda.

ZHAN, Z. **Selenium WebDriver Recipes in C#**. [S.l.]: Springer, 2015.

APÊNDICE A – CÓDIGOS-FONTES SCRIPT UFC

Código-fonte 1 – Script UFC

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions
   as EC
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.common.exceptions import
   NoSuchElementException
import re
import os
from unidecode import unidecode
import time
import pandas as pd

from selenium.webdriver.common.action_chains import
   ActionChains
from bs4 import BeautifulSoup



def sanitize_filename(course_name):
    # Remover acentua  o
    unaccented = unidecode(course_name)
    # Substituir espa os, barras normais, barras
       invertidas e outros caracteres especiais por h fens
    sanitized = re.sub(r'[ /\\]', '-', unaccented)
```

```
26      # Remover quaisquer outros caracteres n o
           alfanum ricos exceto h fens
27      sanitized = re.sub(r'[^a-zA-Z0-9-]', '', sanitized)
28      return sanitized
29
30  def load_config():
31      config = {}
32      with open('config.txt', 'r') as file:
33          for line in file:
34              parts = line.strip().split('=', 1)
35              if len(parts) == 2:
36                  key, value = parts
37                  config[key] = value
38      return config
39
40  def get_year_selection():
41      years = list(range(2015, 2025))  # Cria uma lista de
           anos de 2015 a 2024
42      print("Selecione os anos para processar:\n0. Todos")
43      for index, year in enumerate(years):
44          print(f"{index + 1}. {year}")
45
46      selection = input("Digite os n meros dos anos
           selecionados, separados por v rgula, ou '0' para
           todos: ")
47      if selection.strip() == '0':
48          return years  # Retorna todos os anos
49      else:
50          selected_indices = [int(x.strip()) - 1 for x in
               selection.split(',')]  # Converte para  ndices
51          return [years[i] for i in selected_indices]  #
               Retorna os anos selecionados
```

```
52 def load_courses():
53     df = pd.read_excel('dicionario_cursos_id.xlsx')
54     print("Selecione os cursos para processar:\n0. Todos")
55     for index, row in df.iterrows():
56         print(f"{index + 1}. {row['Curso']}")
57     return df
58 
59 def get_user_selection(df):
60     selection = input("Digite os n meros dos cursos
          selecionados, separados por v rgula, ou '0' para
          todos: ")
61     if selection.strip() == '0':
62         return [(row['id'], row['Curso']) for index, row in
              df.iterrows()]  # Retorna todos os IDs e nomes
63     else:
64         selected_indices = [int(x.strip()) - 1 for x in
             selection.split(',')]  # Convertendo para
              ndices
65         return [(df.iloc[index]['id'], df.iloc[index]['
             Curso']) for index in selected_indices]
66 
67 
68 def setup_driver():
69     chrome_options = Options()
70     service = Service(ChromeDriverManager().install())
71     driver = webdriver.Chrome(service=service, options=
          chrome_options)
72     return driver
73 
74 def fetch_students(driver, config, course_info):
75 
76     course_id, course_name = course_info
```

```
77      # Navega para a URL configurada (sup e-se que seja a
            p gina onde voc  pode selecionar o curso)
78      driver.get(config['URL'])
79
80      WebDriverWait(driver, 10).until(EC.
            element_to_be_clickable((By.XPATH, config['
            XPATH_Ano_Curso']))).click()
81      WebDriverWait(driver, 10).until(EC.
            element_to_be_clickable((By.XPATH, config['
            XPATH_Ano_Opcao'].format(int(config['Ano']) - 2013))
            )).click()
82
83      # Seleciona o curso especificado pelo ID do curso
84      course_option_xpath = config['XPATH_Curso_Opcao'].
            format(course_id)
85      WebDriverWait(driver, 30).until(EC.
            element_to_be_clickable((By.XPATH,
            course_option_xpath))).click()
86
87      # Clica no bot o de buscar para carregar os alunos do
            curso
88      WebDriverWait(driver, 10).until(EC.
            element_to_be_clickable((By.XPATH, config['
            XPATH_Buscar']))).click()
89
90      # Aguarda pela presen a de todos os elementos que
            representam as linhas de dados dos alunos
91      rows = WebDriverWait(driver, 60).until(EC.
            presence_of_all_elements_located((By.XPATH, config['
            XPATH_Linha'])))
92
93      # Extrai os dados dos alunos das linhas
```

```
94      data_alunos = []
95      for row in rows:
96          cells = row.find_elements(By.TAG_NAME, 'td')
97          if len(cells) > 1:  # Verifica se a linha tem
                c lulas suficientes
98              nome = cells[2].text  # Ajuste o  ndice
                    conforme a disposi  o das colunas
99              situacao = cells[3].text  # Ajuste o  ndice
                    conforme a disposi  o das colunas
100             data_alunos.append({
101                 'Ano': config['Ano'],
102                 'Curso': course_name,  # Usando o ID do
                        curso diretamente
103                 'Nome': nome,
104                 'Situa  o': situacao
105             })
106
107     return data_alunos
108
109 def save_data(data_alunos, config, course_name):
110     base_dir = "Dados/UFC_Status"
111     os.makedirs(base_dir , exist_ok=True)
112     csv_dir = os.path.join(base_dir , 'CSV')
113     excel_dir = os.path.join(base_dir , 'Excel')
114     os.makedirs(csv_dir, exist_ok=True)
115     os.makedirs(excel_dir, exist_ok=True)
116
117     csv_filename = f"UFC_Status.csv"
118     excel_filename = f"UFC_Status.xlsx"
119
120     csv_path = os.path.join(csv_dir, csv_filename)
121     excel_path = os.path.join(excel_dir, excel_filename)
```

```
122
123     # Tenta carregar os dados existentes, se o arquivo
            existir
124     if os.path.exists(csv_path):
125         existing_data = pd.read_csv(csv_path)
126     else:
127         existing_data = pd.DataFrame()
128
129     # Concatena os novos dados
130     updated_data = pd.concat([existing_data, data_alunos],
            ignore_index=True)
131
132     # Remove duplicatas
133     updated_data.drop_duplicates(keep='last', inplace=True)
134
135     # Salva os arquivos atualizados
136     updated_data.to_csv(csv_path, index=False)
137     updated_data.to_excel(excel_path, index=False)
138
139 def main():
140     config = load_config()
141     driver = setup_driver()
142     courses_df = load_courses()
143     selected_courses = get_user_selection(courses_df)
144     selected_years = get_year_selection()
145
146     fato_ufc = []
147
148     try:
149         for course_info in selected_courses:
150             for year in selected_years:
```

```
151                     config['Ano'] = year  # Atualiza o ano no
                            config para o ano selecionado
152                     _, course_name = course_info
153                     print(f"Processando {course_name} para o
                            ano {year}")
154                     data_alunos = fetch_students(driver, config
                            , course_info)
155                     data_alunos_df = pd.DataFrame(data_alunos)
156                     fato_ufc.append(data_alunos_df)
157                     save_data(data_alunos_df, config,
                            course_name)
158
159             fato_ufc_df = pd.concat(fato_ufc, ignore_index=True
                    )
160             save_data(fato_ufc_df, config, "todos")
161         finally:
162             driver.quit()
163             print("Programa finalizado")
164
165 if __name__ == '__main__':
166     main()
167
168 }
```

APÊNDICE B – CÓDIGOS-FONTES SCRIPT LINKEDIN

Código-fonte 2 – Script LinkedIn

```
1
2  import pickle
3  import time
4  import re
5  import os
6  from datetime import datetime
7  from pathlib import Path
8  import itertools
9  import warnings
10 import pandas as pd
11
12 from selenium import webdriver
13 from selenium.webdriver.chrome.service import Service
14 from selenium.webdriver.chrome.options import Options
15 from selenium.webdriver.support.ui import WebDriverWait
16 from selenium.webdriver.support import expected_conditions
      as EC
17 from selenium.webdriver.common.by import By
18 from selenium.webdriver.common.keys import Keys
19 from selenium.common.exceptions import
      NoSuchElementException, TimeoutException,
      WebDriverException
20 from selenium.webdriver.common.action_chains import
      ActionChains
21
22 from webdriver_manager.chrome import ChromeDriverManager
23 from bs4 import BeautifulSoup
24 from unidecode import unidecode
25
```

```
26 import configparser
27
28 warnings.filterwarnings("ignore", category=FutureWarning)
29
30 # Arquivo para salvar os cookies
31 COOKIES_FILE = "cookies.pkl"
32 SELECTED_COURSE = None
33 SELECTED_NUM_ALUNOS = None
34 ALUNOS_VISITED = None
35
36 def start_driver():
37     """Inicia uma nova inst ncia do Selenium."""
38     options = Options()
39     options.add_argument('--ignore-ssl-errors=yes')
40     options.add_argument("--disable-gpu")
41     options.add_argument('--ignore-certificate-errors')
42     # Outras op   es podem ser adicionadas aqui para
         evitar detec   o
43     service = Service(ChromeDriverManager().install())
44     driver = webdriver.Chrome(service=service, options=
         options)
45     driver.set_page_load_timeout(30)  # 180 segundos
46     return driver
47
48 def save_cookies(driver, path=COOKIES_FILE):
49     """Salva os cookies atuais do Selenium em um arquivo
         pickle."""
50     with open(path, "wb") as file:
51         pickle.dump(driver.get_cookies(), file)
52     print("       Cookies salvos!")
53
54 def load_cookies(driver, path=COOKIES_FILE):
```

```
55      """Carrega os cookies salvos para manter a sess o
            ativa."""
56      try:
57          with open(path, "rb") as file:
58              cookies = pickle.load(file)
59          for cookie in cookies:
60              # Algumas chaves (como expiry) podem precisar
                    de ajustes
61              driver.add_cookie(cookie)
62          print("       Cookies carregados!")
63      except FileNotFoundError:
64          print("       Nenhum cookie salvo encontrado!")
65
66  def restart_selenium(driver, last_url):
67      """
68      Reinicia o Selenium e volta para a  ltima  URL acessada
            .
69      Tenta carregar os cookies para manter a sess o ou, se
            necess rio , refazer o login.
70      """
71      try:
72          driver.quit()
73      except Exception:
74          pass
75      print(f"       Reiniciando o Selenium e retornando para
            {last_url}")
76      driver = start_driver()
77      # Acesse a p gina de login para que os cookies possam
            ser adicionados
78      driver.get("https://www.linkedin.com/login")
79      load_cookies(driver)
80      driver.get(last_url)
```

```
81      return driver
82
83  def auto_login(driver, config):
84      """
85      Tenta utilizar cookies para manter a sess o.
86      Se os cookies n o estiverem presentes ou estiverem
            expirados, realiza o login autom tico.
87      """
88      driver.get("https://www.linkedin.com/login")
89      # Tenta carregar os cookies salvos
90      load_cookies(driver)
91      time.sleep(2)
92      # Redireciona para a p gina inicial para validar a
            sess o
93      driver.get("https://www.linkedin.com/feed/")
94      time.sleep(2)
95      if "login" in driver.current_url.lower():
96          # Se ainda estiver na p gina de login, refaz o
                login
97          print("       Realizando login autom tico...")
98          email_login = driver.find_element(By.XPATH, config[
                'xpath_username'])
99          email_login.clear()
100         email_login.send_keys(config['email'])
101
102         senha_login = driver.find_element(By.XPATH, config[
                'xpath_password'])
103         senha_login.clear()
104         senha_login.send_keys(config['senha'])
105         senha_login.send_keys(Keys.ENTER)
106         time.sleep(5)  # Aguarda o login
107         save_cookies(driver)
```

```
108     else:
109         print("    Sess o ativa com cookies!")
110 
111 
112 
113 def safe_get(driver, url):
114     try:
115         driver.get(url)
116         WebDriverWait(driver, 10).until(lambda d: d.
               execute_script("return document.readyState") ==
               "complete")
117         return True
118     except TimeoutException as e:
119         '''
120         try:
121             driver.execute_script("window.stop();")
122         except Exception as e:
123             print("Erro ao executar window.stop():", e)
124 
125         time.sleep(2)
126         '''
127         raise e
128 
129 
130     return False
131 
132 def processar_tempo(tempo):
133     if not isinstance(tempo, str) or pd.isna(tempo) or
          tempo.strip() == "":
134         return pd.Series({"data_inicio": pd.NaT, "data_fim"
               : pd.NaT}, dtype="datetime64[ns]")
135 
```

```
136     meses_map = {
137         "jan": "01", "fev": "02", "mar": "03", "abr": "04",
                "mai": "05", "jun": "06",
138         "jul": "07", "ago": "08", "set": "09", "out": "10",
                "nov": "11", "dez": "12"
139     }
140 
141     tempo = tempo.lower()
142     tempo = re.sub(r'\bde\b', '', tempo)
143     tempo = re.sub(r'\s+', ' ', tempo).strip().strip(" ,.")
144 
145     if "at " in tempo:
146         partes = tempo.split("at ")
147     elif "-" in tempo:
148         partes = tempo.split("-")
149     else:
150         partes = [tempo]
151 
152     partes = [p.strip(" ,") for p in partes]
153     inicio = partes[0]
154     fim = partes[1] if len(partes) > 1 else None
155 
156     if fim is not None and re.search(r'\b(momento|o momento
            |at  o momento)\b', fim):
157         fim = datetime.strptime("1900-01-01", "%Y-%m-%d")
158 
159     def converter_data(data_texto):
160         try:
161             if data_texto.isdigit():
162                 dt = datetime.strptime(f"{data_texto}-01-01
                        ", "%Y-%m-%d")
163                 return dt
```

```
164             elif "-" in data_texto and " " not in
                    data_texto:
165                 ano_inicio, ano_fim = data_texto.split("-")
166                 dt1 = datetime.strptime(f"{ano_inicio
                        }-01-01", "%Y-%m-%d")
167                 dt2 = datetime.strptime(f"{ano_fim}-01-01",
                        "%Y-%m-%d")
168                 return (dt1, dt2)
169             else:
170                 partes = data_texto.split(" ")
171                 if len(partes) >= 2:
172                     mes = meses_map.get(partes[0], "01")
173                     ano = partes[1]
174                     dt = datetime.strptime(f"{ano}-{mes}-01
                            ", "%Y-%m-%d")
175                     return dt
176                 else:
177                     return None
178         except Exception:
179             return None
180
181     if "-" in inicio and (" " not in inicio):
182         conv = converter_data(inicio)
183         data_inicio = conv[0] if isinstance(conv, tuple)
                else conv
184     else:
185         data_inicio = converter_data(inicio)
186
187     data_fim = converter_data(fim) if fim else None
188     if isinstance(data_fim, tuple):
189         data_fim = data_fim[0]
190
```

```
191     return pd.Series({"data_inicio": data_inicio, "data_fim
           ": data_fim})
192
193 def sanitize_filename(course_name):
194     unaccented = unidecode(course_name)
195     sanitized = re.sub(r'[ /\\]', '-', unaccented)
196     sanitized = re.sub(r'[^a-zA-Z0-9-]', '', sanitized)
197     return sanitized
198
199 def load_config():
200     config = {}
201     with open('config.txt', 'r') as file:
202         for line in file:
203             parts = line.strip().split('=', 1)
204             if len(parts) == 2:
205                 key, value = parts
206                 config[key] = value
207     return config
208
209 def get_user_input():
210     num_alunos = int(input("Digite o n mero de alunos que
           deseja buscar (digite 0 para todos): "))
211     return num_alunos
212
213 def prepare_new_tab(driver):
214     window_main = driver.current_window_handle
215     driver.execute_script("window.open('');")
216     tab_profile = driver.window_handles[1]
217     driver.switch_to.window(window_main)
218     return window_main, tab_profile
219
220 def switch_to_profile_tab(driver, tab_profile, link):
```

```
221     driver.switch_to.window(tab_profile)
222     time.sleep(1)
223     try:
224         safe_get(driver, link)
225         WebDriverWait(driver, 10).until(lambda d: d.
               execute_script("return document.readyState") ==
               "complete")
226     except TimeoutException as e:
227         raise e
228
229 def switch_to_main_tab(driver, window_main):
230     driver.switch_to.window(window_main)
231     time.sleep(1)
232
233 def login(driver, config):
234     safe_get(driver, "https://www.linkedin.com/login")
235     email_login = driver.find_element(By.XPATH, config['
           xpath_username'])
236     email_login.send_keys(config['email'])
237
238     senha_login = driver.find_element(By.XPATH, config['
           xpath_password'])
239     senha_login.send_keys(config['senha'])
240     senha_login.send_keys(Keys.ENTER)
241
242 def load_courses():
243     df = pd.read_excel('dicionario_cursos_id.xlsx')
244     print("Selecione o curso para extrair dados:")
245     for index, row in df.iterrows():
246         print(f"{index + 1}. {row['Curso']}")
247     index = int(input("Escolha o n mero do curso: ")) - 1
248     return df.iloc[index]
```

```
249
250 def navigate_to_course(driver, course):
251     #driver.get('https://www.linkedin.com/school/ufcinforma
        /people/')
252     driver.get('https://www.linkedin.com/school/ufcinforma/
        people/?facetFieldOfStudy=100347')
253
254 def extract_student_data(driver, config):
255     try:
256         elemento_encontrado = False
257         tentativas = 0
258         max_tentativas = 10
259         driver.execute_script("window.scrollTo(0, document.
            body.scrollHeight);")
260         while not elemento_encontrado and tentativas <
            max_tentativas:
261             try:
262                 time.sleep(2)
263                 alunos_box = driver.find_element(By.XPATH,
                    "//div[contains(@class,'org-people__card
                    -margin-bottom')]")
264                 driver.execute_script("arguments[0].
                    scrollIntoView(true);", alunos_box)
265                 elemento_encontrado = True
266                 print("Elemento encontrado e rolado at
                    ele.")
267             except TimeoutException:
268                 print("Elemento n o encontrado, rolando
                    mais para baixo...")
269                 driver.execute_script("window.scrollBy(0,
                    400);")
270                 tentativas += 1
```

```
271         if not elemento_encontrado:
272             print("Elemento n o encontrado ap s v rias
                tentativas.")
273     except Exception as e:
274
275         print(f"Erro inesperado: {e}")
276         raise e
277
278 def save_to_file(df, file_type, course_name):
279     base_path = f"dados/linkedin/{course_name}/{file_type}/
        "
280     os.makedirs(base_path, exist_ok=True)
281     file_path = os.path.join(base_path, f"{file_type}.xlsx"
        )
282     df.to_excel(file_path, index=False)
283     df.to_csv(file_path.replace(".xlsx", ".csv"), index=
        False)
284
285 def extract_competence_data(driver, person_name,
    course_name):
286     data = {
287         "Tipo": [],
288         "Competencia": []
289     }
290     many_competencias = driver.find_elements(By.XPATH, "//a
        [contains(@id, 'navigation-index-Exibir-todas')]")
291     aux_many_competencias = 0
292     if many_competencias == []:
293         competencias = driver.find_elements(By.XPATH, "//
            div[@id='education']/following-sibling::div[2]/
            ul/li")
294     else:
```

```
295         aux_many_competencias = 1
296         link_competencias = many_competencias[0].
               get_attribute('href')
297         safe_get(driver, link_competencias)
298         time.sleep(2)
299         competencias = driver.find_elements(By.XPATH, "//
               div[@class='scaffold-finite-scroll__content']/ul
               /li")
300     for competencia in competencias:
301         data["Tipo"].append("Compet ncias")
302         try:
303             competencia_text = competencia.find_element(By.
                   CSS_SELECTOR, "div.t-bold span").text
304             data["Competencia"].append(competencia_text)
305         except NoSuchElementException:
306             data["Competencia"].append(None)
307     df = pd.DataFrame(data)
308     df["Nome"] = person_name
309     if aux_many_competencias == 1:
310         try:
311             driver.back()
312             time.sleep(1)
313         except TimeoutError:
314             print("Timeout ao voltar a p gina , tentando
                   uma alternativa...")
315             driver.execute_script("window.history.go(-1)")
316     return df
317
318 def extract_education_data(driver, person_name, course_name
       , ano):
319     data = {
320         "Tipo": [],
```

```
321         "Local": [],
322         "Ocupa  o": [],
323         "Tempo": []
324     }
325     many_formacoes = driver.find_elements(By.XPATH, "//a[
            @id='navigation-index-see-all-education']")
326     aux_many_formacoes = 0
327     if many_formacoes == []:
328         formacoes = driver.find_elements(By.XPATH, "//div[
                @id='education']/following-sibling::div[2]/ul/li
                ")
329     else:
330         aux_many_formacoes = 1
331         link_formacoes = many_formacoes[0].get_attribute('
                href')
332         safe_get(driver, link_formacoes)
333         time.sleep(2)
334         formacoes = driver.find_elements(By.XPATH, "//div[
                @class='scaffold-finite-scroll__content']/ul/li"
                )
335     for formacao in formacoes:
336         data["Tipo"].append("Forma  o Acad mica")
337         try:
338             instituicao = formacao.find_element(By.
                    CSS_SELECTOR, "div.t-bold span").text
339             data["Local"].append(instituicao)
340         except NoSuchElementException:
341             data["Local"].append(None)
342         ocupacao = None
343         spans = formacao.find_elements(By.TAG_NAME, "span")
344         for span in spans:
```

```
345             if span.get_attribute("class") == "t-14 t-
                    normal":
346                 ocupacao = span.find_element(By.TAG_NAME, "
                        span").text
347                 break
348         data["Ocupa  o"].append(ocupacao if ocupacao else
                None)
349         try:
350             periodo = formacao.find_element(By.CSS_SELECTOR
                    , "span.t-black--light span").text
351             data["Tempo"].append(periodo)
352         except NoSuchElementException:
353             data["Tempo"].append(None)
354     df = pd.DataFrame(data)
355     df["Nome"] = person_name
356     if aux_many_formacoes == 1:
357         try:
358             driver.back()
359             time.sleep(1)
360         except TimeoutError:
361             print("Timeout ao voltar a p gina , tentando
                    uma alternativa...")
362             driver.execute_script("window.history.go(-1)")
363     return df
364
365 def extract_experience_data(driver, person_name,
    course_name):
366     data = {
367         "col_tipo": [],
368         "col_link_empresa": [],
369         "col_img_empresa": [],
370         "col_nome_empresa": [],
```

```
371         "col_cargo": [],
372         "col_tempo": []
373     }
374     many_XP = driver.find_elements(By.XPATH, "//a[@id='
            navigation-index-see-all-experiences']")
375     aux_many_XP = 0
376     if many_XP == []:
377         experiencias = driver.find_elements(By.XPATH, "//
                div[@id='experience']/following-sibling::div[2]/
                ul/li")
378     else:
379         aux_many_XP = 1
380         link_XP = many_XP[0].get_attribute('href')
381         safe_get(driver, link_XP)
382         time.sleep(2)
383         experiencias = driver.find_elements(By.XPATH, "//
                div[@class='scaffold-finite-scroll__content']/ul
                /li")
384     for experiencia in experiencias:
385         data["col_tipo"].append("experi ncias")
386         try:
387             link_empresa = experiencia.find_element(By.
                    XPATH, "//*[@data-field='
                    experience_company_logo']").get_attribute('
                    href')
388             data["col_link_empresa"].append(link_empresa)
389         except NoSuchElementException:
390             data["col_link_empresa"].append(None)
391         try:
392             link_img_empresa = experiencia.find_element(By.
                    XPATH, "//*[@data-field='
                    experience_company_logo']/div/div/img").
```

```
                get_attribute('src')
393             data["col_img_empresa"].append(link_img_empresa
                )
394         except NoSuchElementException:
395             data["col_img_empresa"].append(None)
396         try:
397             if len(experiencia.find_elements(By.XPATH, ".//
                div[contains(@class, 'pvs-entity__sub-
                components')]/ul/li[*[1][self::span]]")) >
                1:
398                 data["col_nome_empresa"].append(experiencia
                    .find_element(By.XPATH, ".//span[1]").
                    text)
399             else:
400                 data["col_nome_empresa"].append(experiencia
                    .find_element(By.XPATH, ".//div/span[
                    contains(@class, 't-14') and not(
                    contains(@class, 't-black--light'))]/
                    span[2]").text)
401         except:
402             data["col_nome_empresa"].append(None)
403         list_cargos = []
404         try:
405             cargos = experiencia.find_elements(By.XPATH, "
                .//div[contains(@class, 'pvs-entity__sub-
                components')]/ul/li[*[1][self::span]]")
406             if len(cargos) > 1:
407                 for cargo in cargos:
408                     list_cargos.append(cargo.find_element(
                        By.XPATH, ".//span[2]").text)
409                 data["col_cargo"].append(list_cargos)
410             else:
```

```
411                 data["col_cargo"].append(experiencia.
                        find_element(By.XPATH, ".//div[contains(
                        @class, 'mr1') and *[1][self::span]]/
                        span[1]").text)
412             except:
413                 data["col_cargo"].append(None)
414             list_tempos = []
415             try:
416                 if len(cargos) > 1:
417                     for cargo in cargos:
418                         list_tempos.append(cargo.find_element(
                            By.XPATH, ".//span[contains(@class,
                            't-black--light')]/span[2]").text)
419                     data["col_tempo"].append(list_tempos)
420                 else:
421                     data["col_tempo"].append(experiencia.
                        find_element(By.XPATH, ".//div/span[
                        contains(@class, 't-black--light')]/span
                        [2]").text)
422             except:
423                 data["col_tempo"].append(None)
424         df = pd.DataFrame(data)
425         df["Nome"] = person_name
426         df = df.explode(["col_cargo", "col_tempo"],
               ignore_index=True)
427         if aux_many_XP == 1:
428             try:
429                 driver.back()
430                 time.sleep(1)
431             except TimeoutError:
432                 print("Timeout ao voltar a p gina , tentando
                        uma alternativa...")
```

```
433                 driver.execute_script("window.history.go(-1)")
434         return df
435
436 def extract_profile_data(driver, person_name, course_name):
437         data = {
438             "Nome": [person_name],
439             "Link_img": [],
440             "Subt tulo": []
441         }
442         try:
443             link_img = driver.find_element(By.CSS_SELECTOR, '
                    div.pv-top-card__non-self-photo-wrapper img').
                    get_attribute('src')
444         except NoSuchElementException:
445             link_img = None
446         try:
447             subtitle = driver.find_element(By.CSS_SELECTOR, '
                    div.text-body-medium').text
448         except NoSuchElementException:
449             subtitle = None
450         data["Link_img"].append(link_img)
451         data["Subt tulo"].append(subtitle)
452         df = pd.DataFrame(data)
453         return df
454
455 def update_or_append_file(new_data, base_path, file_name,
       key_column):
456         os.makedirs(base_path, exist_ok=True)
457         file_path = os.path.join(base_path, f"{file_name}.xlsx"
               )
458         if os.path.exists(file_path):
459             try:
```

```
460             df_existing = pd.read_excel(file_path)
461         except Exception as e:
462             print(f"Erro ao ler {file_path}: {e}")
463             df_existing = pd.DataFrame()
464         df_combined = pd.concat([df_existing, new_data],
                ignore_index=True)
465         df_combined = df_combined.drop_duplicates( keep='
                last')
466     else:
467         df_combined = new_data
468     df_combined.to_excel(file_path, index=False)
469     df_combined.to_csv(file_path.replace(".xlsx", ".csv"),
        index=False)
470
471 def extract_profile_info(driver, config, tab_profile,
    window_main, profile_link, aluno_name, ano, course_name,
     base_profiles, base_education, base_experience,
    base_competence, aluno_id):
472     switch_to_profile_tab(driver, tab_profile, profile_link
        )
473     sanitized_course_name = sanitize_filename(course_name)
474     person_name = driver.find_element(By.CSS_SELECTOR, 'h1'
        ).text
475     time.sleep(2)
476     profile_df = extract_profile_data(driver, person_name,
        sanitized_course_name)
477     education_df = extract_education_data(driver,
        person_name, sanitized_course_name, ano)
478     experience_df = extract_experience_data(driver,
        person_name, sanitized_course_name)
479     competence_df = extract_competence_data(driver,
        person_name, sanitized_course_name)
```

```
480
481     profile_df["link_perfil"] = profile_link
482     education_df["link_perfil"] = profile_link
483     experience_df["link_perfil"] = profile_link
484     competence_df["link_perfil"] = profile_link
485
486     profile_df["aluno_id"] = aluno_id
487     education_df["aluno_id"] = aluno_id
488     experience_df["aluno_id"] = aluno_id
489     competence_df["aluno_id"] = aluno_id
490
491     education_df["Curso"] = course_name
492
493     if education_df["Tempo"].empty:
494         education_df["data_inicio"] = pd.Series(dtype='
                object')
495         education_df["data_fim"] = pd.Series(dtype='object'
                )
496     else:
497         education_df[["data_inicio", "data_fim"]] =
                education_df["Tempo"].apply(processar_tempo)
498
499     education_df["Ano"] = pd.to_datetime(education_df["
            data_inicio"], format="%d/%m/%Y %H:%M:%S", errors='
            coerce').dt.year
500
501     condicao = (education_df["Local"] == "Universidade
            Federal do Cear ") & (education_df["Ano"] == ano)
502
503     if not condicao.any():
504         switch_to_main_tab(driver, window_main)
505         return False
```

```
506
507     if experience_df["col_tempo"].empty:
508         experience_df["data_inicio"] = pd.Series(dtype='
               object')
509         experience_df["data_fim"] = pd.Series(dtype='object
               ')
510
511     else:
512         experience_df[["data_inicio", "data_fim"]] =
               experience_df["col_tempo"].apply(processar_tempo
               )
513
514     print(f"Salvando informa  o de: {person_name}")
515     update_or_append_file(profile_df, base_profiles, "
           profiles", "link_perfil")
516     update_or_append_file(education_df, base_education, "
           education", "link_perfil")
517     update_or_append_file(experience_df, base_experience, "
           experience", "link_perfil")
518     update_or_append_file(competence_df, base_competence, "
           competence", "link_perfil")
519     switch_to_main_tab(driver, window_main)
520     return True
521
522 def clean_name(name):
523     name = re.sub(r"[\(\[\{].*?[\)\]\}]", "", name)
524     name = re.sub(r"[*&@]", "", name)
525     name = unidecode(name).strip().lower()
526     name = re.sub(r'\s+', ' ', name)
527     return name
528
529 def generate_name_combinations(full_name):
```

```
530     name_parts = [word for word in full_name.split() if len
          (word) >= 3]
531     if len(name_parts) < 2:
532         return [full_name]
533     first_name, second_name = name_parts[:2]
534     valid_combinations = [
535         f"{first} {other}" for first in (first_name,
              second_name)
536         for other in name_parts if first != other and
              name_parts.index(other) > name_parts.index(first
              )
537     ]
538     return [full_name] + valid_combinations
539
540 def check_nome(name_combination, person_name):
541     name_combination = clean_name(name_combination)
542     person_name = clean_name(person_name)
543     name_combination_parts = name_combination.split()
544     person_name_parts = person_name.split()
545     if all(word in name_combination_parts for word in
          person_name_parts):
546         return True
547     for word in person_name_parts:
548         if len(word) == 1:
549             if not any(n.startswith(word) for n in
                  name_combination_parts):
550                 return False
551         elif word not in name_combination_parts:
552             return False
553     return True
554
```

```
555 def visit_profiles(driver, config, tab_profile, window_main
    , num_alunos, course_name, links_visitados,
    df_alunos_course, caminho_arquivo, df_completo,
    qtd_alunos_visitados):
556     url_base = driver.current_url
557     base_profiles = f"dados/linkedin/{course_name}/profiles
        /"
558     base_education = f"dados/linkedin/{course_name}/
        education/"
559     base_experience = f"dados/linkedin/{course_name}/
        experience/"
560     base_competence = f"dados/linkedin/{course_name}/
        competence/"
561     perfil_visitados = set()
562     perfil_visitados.update(links_visitados)
563
564     if qtd_alunos_visitados is None:
565         alunos_visited = 0
566     else:
567         alunos_visited = qtd_alunos_visitados
568
569     try:
570         for index, (aluno_name, ano) in enumerate(zip(
            df_alunos_course["Nome"], df_alunos_course["Ano"
            ])):
571             print(aluno_name)
572             profile_found = False
573             aluno_id = df_alunos_course.iloc[index]['id']
574             if df_alunos_course.iloc[index]['linkedin'] !=
                0:
575                 pass
576             else:
```

```
577                 for name_combination in
                        generate_name_combinations(aluno_name):
578                     safe_get(driver, url_base + f'&keywords
                            ={course_name.replace(" ", "%20")
                            }%20{name_combination.replace(" ",
                            "%20")}')
579                     qtd_alunos_str = WebDriverWait(driver,
                            20).until(
580                         EC.presence_of_element_located((By.
                                XPATH, "//div[@class='org-
                                people__header-spacing-carousel
                                ']/div/h2"))
581                     ).text.split(' ')[0]
582                     qtd_alunos_split = qtd_alunos_str.split
                            (' ')[0].split('.')
583                     if len(qtd_alunos_split) > 1:
584                         qtd_alunos = int(qtd_alunos_split
                                [0] + qtd_alunos_split[1])
585                     else:
586                         qtd_alunos = int(qtd_alunos_split
                                [0])
587                     while True:
588                         if qtd_alunos == 0:
589                             break
590                         alunos_encontrados = driver.
                                find_elements(By.CLASS_NAME, "
                                org-people-profile-card__profile
                                -card-spacing")
591                         last_len = len(alunos_encontrados)
592                         for aluno in alunos_encontrados:
593                             person_name = aluno.
                                    find_element(By.XPATH,".//
```

```
                section/div/div/div[2]//*[
                normalize-space()][1]").text
            if check_nome(aluno_name,
                person_name):
                profile_link =  aluno.
                    find_element(By.TAG_NAME
                    , 'a').get_attribute('
                    href')
                if profile_link not in
                    perfil_visitados:
                    perfil_visitados.add(
                        profile_link)
                    sucess_etl =
                        extract_profile_info
                        (driver, config,
                        tab_profile,
                        window_main,
                        profile_link, aluno,
                         ano, course_name,
                        base_profiles,
                        base_education,
                        base_experience,
                        base_competence,
                        aluno_id)
                    if sucess_etl:
                        profile_found =
                            True
                        break
            else:
                pass
        if profile_found == True:
            break
```

```
606                         while True:
607                             driver.execute_script("
                                   arguments[0].scrollIntoView(
                                   true);", alunos_encontrados
                                   [-1])
608                             driver.execute_script("window.
                                   scrollBy(0, 800);")
609
610                             new_alunos = driver.
                                   find_elements(By.CLASS_NAME,
                                    "org-people-profile-
                                   card__profile-card-spacing")
611                             if len(new_alunos) >= (
                                   qtd_alunos -2 ) or len(
                                   new_alunos) > last_len: #
                                   coloquei um -2, porque o
                                   linkedin buga e ai seria uma
                                    margem de erro
612                                 break
613                         if  last_len >= (qtd_alunos -2 )
                               or alunos_visited >= num_alunos:
614                             break
615                         last_len = len(new_alunos)
616                     if profile_found == True:
617                         break
618
619                 if profile_found:
620                     df_completo.loc[df_completo["id"] ==
                           aluno_id, "linkedin"] = 1
621                 else:
622                     df_completo.loc[df_completo["id"] ==
                           aluno_id, "linkedin"] = -1
```

```
623
624                 atualizar_dados_curso(caminho_arquivo,
                        df_completo, df_alunos_course)
625
626                 alunos_visited += 1
627                 ALUNOS_VISITED = alunos_visited
628
629             if num_alunos != 0 and alunos_visited >=
                    num_alunos:
630                 break
631         print(f'Visitados {alunos_visited} perfis.')
632     except (WebDriverException, TimeoutException) as e:
633         raise e
634
635 def get_visited_links(course_name):
636     base_profiles = f"dados/linkedin/{course_name}/profiles
            /"
637     file_xlsx = os.path.join(base_profiles, "profiles.xlsx"
            )
638     file_csv = os.path.join(base_profiles, "profiles.csv")
639     if os.path.exists(file_xlsx):
640         try:
641             df = pd.read_excel(file_xlsx)
642         except Exception as e:
643             print(f"Erro ao ler {file_xlsx}: {e}")
644             return set()
645     elif os.path.exists(file_csv):
646         try:
647             df = pd.read_csv(file_csv)
648         except Exception as e:
649             print(f"Erro ao ler {file_csv}: {e}")
650             return set()
```

```
651     else:
652         return set()
653     if "link_perfil" in df.columns:
654         links = df["link_perfil"].dropna().tolist()
655     elif "links" in df.columns:
656         links = df["links"].dropna().tolist()
657     else:
658         links = []
659     return set(links)
660 
661 def atualizar_dados_curso(caminho_arquivo, df_original,
    df_atualizado):
662     df_original.to_excel(caminho_arquivo, index=False)
663 
664 def carregar_dados_curso(nome_curso):
665     caminho_base = Path(__file__).resolve().parent
666     caminho_arquivo = caminho_base / 'Dados' / 'UFC_Status'
        / 'Excel' / 'UFC_Status.xlsx'
667     if not caminho_arquivo.exists():
668         raise FileNotFoundError(f"O arquivo {
            caminho_arquivo} n o foi encontrado.")
669     df = pd.read_excel(caminho_arquivo)
670     df_filtrado = df[df['Curso'] == nome_curso]
671     return caminho_arquivo, df, df_filtrado
672 
673 
674 def main(selected_course, selected_num_alunos,
    qtd_alunos_visitados):
675     config = load_config()
676     driver = start_driver()
677     try:
```

```
678         # Tenta manter a sess o com cookies ou realizar o
                login autom tico
679         auto_login(driver, config)
680
681         # Utiliza o curso previamente selecionado
682         course = selected_course
683         course_name = course['Curso'].split("/")[0]
684         print(f"Curso selecionado: {course_name}")
685
686         caminho_arquivo, df_completo, df_alunos_course =
                carregar_dados_curso(course['Curso'])
687         navigate_to_course(driver, course)
688
689         links_visitados = get_visited_links(course_name)
690         # Em vez de chamar get_user_input(), usa o valor
                armazenado
691         num_alunos = selected_num_alunos
692         print(f"N mero de alunos a buscar: {num_alunos}")
693
694         window_main, tab_profile = prepare_new_tab(driver)
695
696         visit_profiles(driver, config, tab_profile,
                window_main, num_alunos, course_name,
697                             links_visitados, df_alunos_course,
                                caminho_arquivo, df_completo,
                                qtd_alunos_visitados)
698
699         print("Finalizado!")
700
701     except (WebDriverException, TimeoutException) as e:
702         last_url =  "https://www.linkedin.com/"
```

```
703             print(f"       Erro detectado: {e}. Reiniciando
                    Selenium...")
704             driver = restart_selenium(driver, last_url)
705             # Lan a a exce  o para que o loop principal
                    reinicie o main() mantendo as escolhas
706             raise e
707         finally:
708             driver.quit()
709
710 if __name__ == '__main__':
711         # Se o curso ainda n o foi selecionado, pede ao
                usu rio
712         if SELECTED_COURSE is None:
713             SELECTED_COURSE = load_courses()
714         # Se o n mero de alunos ainda n o foi informado, pede
                ao usu rio
715         if SELECTED_NUM_ALUNOS is None:
716             SELECTED_NUM_ALUNOS = get_user_input()
717
718         # Loop principal que garante a reinicializa  o do
                processo mantendo as escolhas
719         while True:
720             try:
721                 main(SELECTED_COURSE, SELECTED_NUM_ALUNOS,
                        ALUNOS_VISITED)
722                 break  # Se tudo ocorrer bem, sai do loop
723             except Exception as e:
724                 print(f"Erro inesperado: {e}. Reiniciando o
                        processo em 5 segundos...")
725                 time.sleep(5)
726
727 }
```

APÊNDICE C – CÓDIGOS-FONTES SCRIPT UPDATE LINKEDIN

Código-fonte 3 – Script Update LinkedIn

```
1  import pickle
2  import time
3  import re
4  import os
5  from datetime import datetime
6  from pathlib import Path
7  import itertools
8  import warnings
9  import pandas as pd
10 import logging
11
12 from selenium import webdriver
13 from selenium.webdriver.chrome.service import Service
14 from selenium.webdriver.chrome.options import Options
15 from selenium.webdriver.support.ui import WebDriverWait
16 from selenium.webdriver.support import expected_conditions
      as EC
17 from selenium.webdriver.common.by import By
18 from selenium.webdriver.common.keys import Keys
19 from selenium.common.exceptions import
      NoSuchElementException, TimeoutException,
      WebDriverException
20 from selenium.webdriver.common.action_chains import
      ActionChains
21
22 from webdriver_manager.chrome import ChromeDriverManager
23 from bs4 import BeautifulSoup
24 from unidecode import unidecode
25
```

```
26 import configparser
27
28 warnings.filterwarnings("ignore", category=FutureWarning)
29
30 COOKIES_FILE = "cookies.pkl"
31 SELECTED_NUM_ALUNOS = 400
32 LAST_ALUNO = None
33 ID_ALUNO = None
34
35
36
37 def start_driver():
38     """Inicia uma nova inst ncia do Selenium."""
39     options = Options()
40     options.add_argument('--ignore-ssl-errors=yes')
41     options.add_argument("--disable-gpu")
42     options.add_argument('--ignore-certificate-errors')
43     # Outras op   es podem ser adicionadas aqui para
         evitar detec   o
44
45     # Suprimir logs desnecess rios:
46     options.headless = True
47     options.add_experimental_option("excludeSwitches", ["
         enable-logging"])
48     options.add_argument("--log-level=3")
49
50     service = Service(ChromeDriverManager().install(),
         log_path="NUL")
51     driver = webdriver.Chrome(service=service, options=
         options)
52
```

```
53      logger = logging.getLogger('selenium.webdriver.remote.
            remote_connection')
54      logger.setLevel(logging.WARNING)  # or any variant from
            ERROR, CRITICAL or NOTSET
55
56      driver.set_page_load_timeout(30)  # 180 segundos
57      return driver
58
59  def save_cookies(driver, path=COOKIES_FILE):
60      """Salva os cookies atuais do Selenium em um arquivo
            pickle."""
61      with open(path, "wb") as file:
62          pickle.dump(driver.get_cookies(), file)
63      print("       Cookies salvos!")
64
65  def load_cookies(driver, path=COOKIES_FILE):
66      """Carrega os cookies salvos para manter a sess o
            ativa."""
67      try:
68          with open(path, "rb") as file:
69              cookies = pickle.load(file)
70          for cookie in cookies:
71              # Algumas chaves (como expiry) podem precisar
                    de ajustes
72              driver.add_cookie(cookie)
73          print("       Cookies carregados!")
74      except FileNotFoundError:
75          print("       Nenhum cookie salvo encontrado!")
76
77  def restart_selenium(driver, last_url):
78      """
```

```
79      Reinicia o Selenium e volta para a  ltima  URL acessada
            .
80      Tenta carregar os cookies para manter a sess o ou, se
            necess rio , refazer o login.
81      """
82      try:
83          driver.quit()
84      except Exception:
85          pass
86      print(f"       Reiniciando o Selenium e retornando para
            {last_url}")
87      driver = start_driver()
88      # Acesse a p gina de login para que os cookies possam
            ser adicionados
89      driver.get("https://www.linkedin.com/login")
90      load_cookies(driver)
91      driver.get(last_url)
92      return driver
93
94  def auto_login(driver, config):
95      """
96      Tenta utilizar cookies para manter a sess o.
97      Se os cookies n o estiverem presentes ou estiverem
            expirados, realiza o login autom tico.
98      """
99      driver.get("https://www.linkedin.com/login")
100     # Tenta carregar os cookies salvos
101     load_cookies(driver)
102     time.sleep(1)
103     # Redireciona para a p gina inicial para validar a
            sess o
104     driver.get("https://www.linkedin.com/feed/")
```

```
105     time.sleep(1)
106     if "login" in driver.current_url.lower():
107         # Se ainda estiver na p gina de login, refaz o
                login
108         print("       Realizando login autom tico...")
109         email_login = driver.find_element(By.XPATH, config[
                'xpath_username'])
110         email_login.clear()
111         email_login.send_keys(config['email'])
112 
113         senha_login = driver.find_element(By.XPATH, config[
                'xpath_password'])
114         senha_login.clear()
115         senha_login.send_keys(config['senha'])
116         senha_login.send_keys(Keys.ENTER)
117         time.sleep(2)  # Aguarda o login
118         save_cookies(driver)
119     else:
120         print("    Sess o ativa com cookies!")
121 
122 def login(driver, config):
123     driver.get( "https://www.linkedin.com/login")
124     email_login = driver.find_element(By.XPATH, config['
            xpath_username'])
125     email_login.send_keys(config['email'])
126 
127     senha_login = driver.find_element(By.XPATH, config['
            xpath_password'])
128     senha_login.send_keys(config['senha'])
129     senha_login.send_keys(Keys.ENTER)
130 
131 def processar_tempo(tempo):
```

```
132     if not isinstance(tempo, str) or pd.isna(tempo) or
           tempo.strip() == "":
133         return pd.Series({"data_inicio": pd.NaT, "data_fim"
               : pd.NaT}, dtype="datetime64[ns]")
134
135     meses_map = {
136         "jan": "01", "fev": "02", "mar": "03", "abr": "04",
               "mai": "05", "jun": "06",
137         "jul": "07", "ago": "08", "set": "09", "out": "10",
               "nov": "11", "dez": "12"
138     }
139
140     tempo = tempo.lower()
141     tempo = re.sub(r'\bde\b', '', tempo)
142     tempo = re.sub(r'\s+', ' ', tempo).strip().strip(" ,.")
143
144     if "at " in tempo:
145         partes = tempo.split("at ")
146     elif "-" in tempo:
147         partes = tempo.split("-")
148     else:
149         partes = [tempo]
150
151     partes = [p.strip(" ,") for p in partes]
152     inicio = partes[0]
153     fim = partes[1] if len(partes) > 1 else None
154
155     if fim is not None and re.search(r'\b(momento|o momento
           |at  o momento)\b', fim):
156         fim = datetime.strptime("1900-01-01", "%Y-%m-%d")
157
158     def converter_data(data_texto):
```

```
159         try:
160             if data_texto.isdigit():
161                 dt = datetime.strptime(f"{data_texto}-01-01
                        ", "%Y-%m-%d")
162                 return dt
163             elif "-" in data_texto and " " not in
                    data_texto:
164                 ano_inicio, ano_fim = data_texto.split("-")
165                 dt1 = datetime.strptime(f"{ano_inicio
                        }-01-01", "%Y-%m-%d")
166                 dt2 = datetime.strptime(f"{ano_fim}-01-01",
                         "%Y-%m-%d")
167                 return (dt1, dt2)
168             else:
169                 partes = data_texto.split(" ")
170                 if len(partes) >= 2:
171                     mes = meses_map.get(partes[0], "01")
172                     ano = partes[1]
173                     dt = datetime.strptime(f"{ano}-{mes}-01
                            ", "%Y-%m-%d")
174                     return dt
175                 else:
176                     return None
177         except Exception:
178             return None
179
180     if "-" in inicio and (" " not in inicio):
181         conv = converter_data(inicio)
182         data_inicio = conv[0] if isinstance(conv, tuple)
                else conv
183     else:
184         data_inicio = converter_data(inicio)
```

```
185
186     data_fim = converter_data(fim) if fim else None
187     if isinstance(data_fim, tuple):
188         data_fim = data_fim[0]
189
190     return pd.Series({"data_inicio": data_inicio, "data_fim
           ": data_fim})
191
192 def load_config():
193     config = {}
194     with open('config.txt', 'r') as file:
195         for line in file:
196             parts = line.strip().split('=', 1)
197             if len(parts) == 2:
198                 key, value = parts
199                 config[key] = value
200     return config
201
202 # Fun   es auxiliares para extra   o de dados
203
204 def extract_competence_data(driver, person_name,
       course_name):
205     data = {
206         "Tipo": [],
207         "Competencia": []
208     }
209     many_competencias = driver.find_elements(By.XPATH, "//a
           [contains(@id, 'navigation-index-Exibir-todas')]")
210     aux_many_competencias = 0
211     if many_competencias == []:
212         competencias = driver.find_elements(By.XPATH, "//
               div[@id='skills']/following-sibling::div[2]/ul/
```

```
            li")
213     else:
214         aux_many_competencias = 1
215         link_competencias = many_competencias[0].
               get_attribute('href')
216         driver.get( link_competencias)
217         time.sleep(1)
218         competencias = driver.find_elements(By.XPATH, "//
               div[@class='scaffold-finite-scroll__content']/ul
               /li")
219     for competencia in competencias:
220         data["Tipo"].append("Compet ncias")
221         try:
222             competencia_text = competencia.find_element(By.
                   CSS_SELECTOR, "div.t-bold span").text
223             data["Competencia"].append(competencia_text)
224         except NoSuchElementException:
225             data["Competencia"].append(None)
226     df = pd.DataFrame(data)
227     df["Nome"] = person_name
228     if aux_many_competencias == 1:
229         try:
230             driver.back()
231             time.sleep(1)
232         except TimeoutError:
233             print("Timeout ao voltar a p gina , tentando
                   uma alternativa...")
234             driver.execute_script("window.history.go(-1)")
235     return df
236
237 def extract_education_data(driver, person_name, course_name
    , ano):
```

```
238     data = {
239         "Tipo": [],
240         "Local": [],
241         "Ocupa  o": [],
242         "Tempo": []
243     }
244     many_formacoes = driver.find_elements(By.XPATH, "//a[
           @id='navigation-index-see-all-education']")
245     aux_many_formacoes = 0
246     if many_formacoes == []:
247         formacoes = driver.find_elements(By.XPATH, "//div[
               @id='education']/following-sibling::div[2]/ul/li
               ")
248     else:
249         aux_many_formacoes = 1
250         link_formacoes = many_formacoes[0].get_attribute('
               href')
251         driver.get( link_formacoes)
252         time.sleep(1)
253         formacoes = driver.find_elements(By.XPATH, "//div[
               @class='scaffold-finite-scroll__content']/ul/li"
               )
254     for formacao in formacoes:
255         data["Tipo"].append("Forma  o Acad mica")
256         try:
257             instituicao = formacao.find_element(By.
                   CSS_SELECTOR, "div.t-bold span").text
258             data["Local"].append(instituicao)
259         except NoSuchElementException:
260             data["Local"].append(None)
261         ocupacao = None
262         spans = formacao.find_elements(By.TAG_NAME, "span")
```

```
263             for span in spans:
264                 if span.get_attribute("class") == "t-14 t-
                        normal":
265                     ocupacao = span.find_element(By.TAG_NAME, "
                            span").text
266                     break
267             data["Ocupa  o"].append(ocupacao if ocupacao else
                    None)
268             try:
269                 periodo = formacao.find_element(By.CSS_SELECTOR
                        , "span.t-black--light span").text
270                 data["Tempo"].append(periodo)
271             except NoSuchElementException:
272                 data["Tempo"].append(None)
273         df = pd.DataFrame(data)
274         df["Nome"] = person_name
275         if aux_many_formacoes == 1:
276             try:
277                 driver.back()
278                 time.sleep(1)
279             except TimeoutError:
280                 print("Timeout ao voltar a p gina, tentando
                        uma alternativa...")
281                 driver.execute_script("window.history.go(-1)")
282         return df
283
284 def extract_experience_data(driver, person_name,
        course_name):
285     data = {
286         "col_tipo": [],
287         "col_link_empresa": [],
288         "col_img_empresa": [],
```

```
289         "col_nome_empresa": [],
290         "col_cargo": [],
291         "col_tempo": []
292     }
293     many_XP = driver.find_elements(By.XPATH, "//a[@id='
            navigation-index-see-all-experiences']")
294     aux_many_XP = 0
295     if many_XP == []:
296         experiencias = driver.find_elements(By.XPATH, "//
               div[@id='experience']/following-sibling::div[2]/
               ul/li")
297     else:
298         aux_many_XP = 1
299         link_XP = many_XP[0].get_attribute('href')
300         driver.get( link_XP)
301         time.sleep(1)
302         experiencias = driver.find_elements(By.XPATH, "//
               main[@aria-label='Experi ncia']/section/div[2]/
               div/div/ul/li")
303     for experiencia in experiencias:
304         data["col_tipo"].append("experi ncias")
305         try:
306             link_empresa = experiencia.find_element(By.
                   XPATH, ".//*[contains(@data-view-name, '
                   profile-component-entity')]/div/a").
                   get_attribute('href')
307             data['col_link_empresa'].append(link_empresa)
308         except NoSuchElementException:
309             data['col_link_empresa'].append(None)
310
311         try:
```

```
312             link_img_empresa = experiencia.find_element(By.
                    XPATH, ".//*[contains(@data-view-name, '
                    profile-component-entity')]/div/a/div/div/
                    img").get_attribute('src')
313             data['col_img_empresa'].append(link_img_empresa
                    )
314         except NoSuchElementException:
315             data['col_img_empresa'].append(None)
316 
317         if aux_many_XP == 1:
318             cargos = experiencia.find_elements(By.XPATH, "
                    .//div[contains(@class, 'pvs-entity__sub-
                    components')]/ul/li/div/div/div/ul/li")
319         else:
320             cargos = experiencia.find_elements(By.XPATH, "
                    .//div[contains(@class, 'pvs-entity__sub-
                    components')]/ul/li[*[1][self::span]]")
321 
322         try:
323             if len(cargos) > 1:
324                 data["col_nome_empresa"].append(experiencia
                        .find_element(By.XPATH, ".//span[1]").
                        text)
325             else:
326                 data["col_nome_empresa"].append(experiencia
                        .find_element(By.XPATH, ".//div/span[
                        contains(@class, 't-14') and not(
                        contains(@class, 't-black--light'))]/
                        span[2]").text)
327         except:
328             data["col_nome_empresa"].append(None)
329         list_cargos = []
```

```
330         try:
331             if len(cargos) > 1:
332                 for cargo in cargos:
333                     list_cargos.append(cargo.find_element(
                            By.XPATH, ".//span[2]").text)
334                 data["col_cargo"].append(list_cargos)
335             else:
336                 data["col_cargo"].append(experiencia.
                        find_element(By.XPATH, ".//div[contains(
                        @class, 'mr1') and *[1][self::span]]/
                        span[1]").text)
337         except:
338             data["col_cargo"].append(None)
339         list_tempos = []
340         try:
341             if len(cargos) > 1:
342                 for cargo in cargos:
343                     list_tempos.append(cargo.find_element(
                            By.XPATH, ".//span[contains(@class,
                            't-black--light')]/span[2]").text)
344                 data["col_tempo"].append(list_tempos)
345             else:
346                 data["col_tempo"].append(experiencia.
                        find_element(By.XPATH, ".//div/span[
                        contains(@class, 't-black--light')]/span
                        [2]").text)
347         except:
348             data["col_tempo"].append(None)
349
350
351
352     df = pd.DataFrame(data)
```

```
353     df["Nome"] = person_name
354     df = df.explode(["col_cargo", "col_tempo"],
            ignore_index=True)
355     if aux_many_XP == 1:
356         try:
357             driver.back()
358             time.sleep(1)
359         except TimeoutError:
360             print("Timeout ao voltar a p gina , tentando
                uma alternativa...")
361             driver.execute_script("window.history.go(-1)")
362     return df
363
364 def extract_profile_info(driver, profile_link, aluno_name,
    ano, course_name, base_profiles, base_education,
    base_experience, base_competence, aluno_id):
365     driver.get(profile_link)
366     WebDriverWait(driver, 10).until(EC.
            presence_of_element_located((By.CSS_SELECTOR, 'h1'))
            )
367
368     person_name = driver.find_element(By.CSS_SELECTOR, 'h1'
            ).text
369     time.sleep(1)
370
371     education_df = extract_education_data(driver,
            person_name, course_name, ano)
372     experience_df = extract_experience_data(driver,
            person_name, course_name)
373     competence_df = extract_competence_data(driver,
            person_name, course_name)
374
```

```
375     education_df["link_perfil"] = profile_link
376     experience_df["link_perfil"] = profile_link
377     competence_df["link_perfil"] = profile_link
378
379     education_df["aluno_id"] = aluno_id
380     experience_df["aluno_id"] = aluno_id
381     competence_df["aluno_id"] = aluno_id
382
383     education_df["Curso"] = course_name
384
385     # Convertendo datas
386     if not education_df["Tempo"].empty:
387         education_df[["data_inicio", "data_fim"]] =
              education_df["Tempo"].apply(processar_tempo)
388         education_df["Ano"] = pd.to_datetime(education_df["
              data_inicio"], errors='coerce').dt.year
389
390     if not experience_df["col_tempo"].empty:
391         experience_df[["data_inicio", "data_fim"]] =
              experience_df["col_tempo"].apply(processar_tempo
              )
392
393     print(f"       Atualizando informa  es de: {
          person_name}")
394
395     update_or_append_file(education_df, base_education, "
          education", "link_perfil")
396     update_or_append_file(experience_df, base_experience, "
          experience", "link_perfil")
397     update_or_append_file(competence_df, base_competence, "
          competence", "link_perfil")
398
```

```
399     return True
400
401 def update_or_append_file(new_data, base_path, file_name,
    key_column):
402     os.makedirs(base_path, exist_ok=True)
403     file_path = os.path.join(base_path, f"{file_name}.xlsx"
        )
404
405     if os.path.exists(file_path):
406         try:
407             df_existing = pd.read_excel(file_path)
408         except Exception as e:
409             print(f"Erro ao ler {file_path}: {e}")
410             df_existing = pd.DataFrame()
411
412         # Remover as linhas cujo aluno_id j  esteja em
            new_data
413         if "aluno_id" in df_existing.columns and "aluno_id"
             in new_data.columns:
414             df_existing = df_existing[~df_existing["
                aluno_id"].isin(new_data["aluno_id"])]
415
416         if "col_img_empresa" in df_existing.columns:
417             df_combined = pd.concat([df_existing, new_data
                ], ignore_index=True).drop_duplicates(subset
                =[col for col in df_existing.columns if col
                != "col_img_empresa"], keep='last')
418         else:
419             df_combined = pd.concat([df_existing, new_data
                ], ignore_index=True).drop_duplicates(keep='
                last')
420     else:
```

```
421             df_combined = new_data
422
423         df_combined.to_excel(file_path, index=False)
424         df_combined.to_csv(file_path.replace(".xlsx", ".csv"),
                index=False)
425
426 # Fun   o principal para atualizar os perfis
427 def update_profiles(driver, course_name,qtd_total_alunos,
        last_aluno,id_aluno):
428         global LAST_ALUNO
429
430         base_profiles = f"dados/linkedin/{course_name}/profiles
                /"
431
432         alunos_visted = 0
433
434         if LAST_ALUNO is None:
435             last_aluno = 0
436         else:
437             alunos_visted = last_aluno
438
439         file_xlsx = os.path.join(base_profiles, "profiles.xlsx"
                )
440         if not os.path.exists(file_xlsx):
441             print(f"    Arquivo {file_xlsx} n  o encontrado!")
442             return
443
444         df_profiles = pd.read_excel(file_xlsx)
445
446         for index, row in df_profiles.iterrows():
447             aluno_id = row.get("aluno_id", None)
448
```

```
449             if (alunos_visted  < (index + 1)) and (id_aluno is
                    None or aluno_id in id_aluno ):
450                 profile_link = row["link_perfil"]
451                 aluno_name = row["Nome"]
452                 ano = row.get("Ano", None)
453
454                 if pd.isna(profile_link):
455                     continue
456
457                 success = extract_profile_info(
458                     driver, profile_link, aluno_name, ano,
                            course_name,
459                     base_profiles, f"dados/linkedin/{
                            course_name}/education/",
460                     f"dados/linkedin/{course_name}/experience/"
                            , f"dados/linkedin/{course_name}/
                            competence/", aluno_id
461                 )
462
463                 if not success:
464                     print(f"       Perfil {aluno_name} n o
                            atualizado.")
465
466                 alunos_visted += 1
467                 LAST_ALUNO = alunos_visted
468
469                 if alunos_visted >= qtd_total_alunos:
470                     break
471
472                 time.sleep(2)  # Pequeno delay entre as
                        requisi  es para evitar bloqueios
473
```

```
474 # Configura  o do WebDriver
475 def setup_driver():
476     options = Options()
477     options.add_argument("--headless")
478     options.add_argument("--disable-gpu")
479     options.add_argument("--no-sandbox")
480     options.add_argument("--disable-dev-shm-usage")
481 
482     service = Service(executable_path="chromedriver.exe")
483     driver = webdriver.Chrome(service=service, options=
        options)
484     return driver
485 
486 
487 def main(selected_course, qtd_total_alunos, last_aluno,
    id_aluno):
488     config = load_config()
489     driver = start_driver()
490     try:
491         # Tenta manter a sess o com cookies ou realizar o
            login autom tico
492         auto_login(driver, config)
493 
494         update_profiles(driver, selected_course,
            qtd_total_alunos,last_aluno,id_aluno)
495 
496         print("Finalizado!")
497 
498     except (WebDriverException, TimeoutException) as e:
499         last_url =  "https://www.linkedin.com/"
500         print(f"       Erro detectado: {e}. Reiniciando
            Selenium...")
```

```
501             driver = restart_selenium(driver, last_url)
502             # Lan a a exce  o para que o loop principal
                    reinicie o main() mantendo as escolhas
503             raise e
504         finally:
505             driver.quit()
506
507 # Execu  o principal
508 if __name__ == "__main__":
509     course_name = "ENGENHARIA EL TRICA"
510
511     # Loop principal que garante a reinicializa  o do
            processo mantendo as escolhas
512     while True:
513         try:
514             main("ENGENHARIA EL TRICA",
                    SELECTED_NUM_ALUNOS,LAST_ALUNO,ID_ALUNO)
515             break  # Se tudo ocorrer bem, sai do loop
516         except Exception as e:
517             print(f"Erro inesperado: {e}. Reiniciando o
                    processo em 5 segundos...")
518             time.sleep(5)
519
520
521
522
523 }
```