FATEC – FACULDADE DE TECNOLOGIA DE RIO CLARO

CURSO: Inteligência Artificial

PROJETO: FEELING AI

GRUPO: SOUL CARE

RELATÓRIO DE PESQUISA – COMO FAZER WEB SCRAPING UTILIZANDO PYTHON E SELENIUM

Aluno: Vladimir Queiroz Sejas

Rio Claro – SP

2025

1. INTRODUÇÃO

Este relatório apresenta uma análise aprofundada do vídeo “Como fazer Web Scraping utilizando Python e Selenium?”, produzido pelo canal Comunidade DS e publicado no YouTube. O material foi selecionado por sua relevância técnica e didática, abordando uma das práticas mais atuais e complexas no campo da coleta automatizada de dados — o uso do Selenium para interagir com páginas dinâmicas e extrair informações de forma programática. A presente pesquisa é parte integrante do Projeto Feeling AI, desenvolvido pelo grupo SOUL CARE no curso de Inteligência Artificial da FATEC de Rio Claro. Embora o vídeo tenha sido concebido para fins educacionais e práticos, sua análise aqui se insere no contexto acadêmico, servindo como complemento ao processo formativo dos estudantes e como fonte de embasamento teórico e técnico para experimentações futuras no projeto. O conteúdo foi estudado a partir de metodologias autodidatas, com base em tutoriais, leituras técnicas, discussões entre os membros do grupo e o uso de múltiplas inteligências artificiais (como ChatGPT, Gemini e Copilot), que auxiliaram na interpretação, explicação e ampliação dos conceitos apresentados. Nota de fonte: Este relatório foi elaborado com base em múltiplas fontes acadêmicas e técnicas, incluindo Mitchell (1997), Cambria et al. (2017), Russell e Norvig (2021), Manning e Schütze (1999), Jurafsky e Martin (2020), OpenAI (2025) e o canal Comunidade DS, entre outros, todos devidamente citados ao longo do texto.

2. FUNDAMENTAÇÃO TEÓRICA

A prática de Web Scraping, segundo Mitchell (1997), integra-se à aprendizagem de máquina e à mineração de dados como um método essencial para coletar e processar informações em larga escala. Já Cambria et al. (2017) destacam que a análise de sentimentos e o processamento de linguagem natural dependem diretamente de bases textuais extensas e limpas, as quais são muitas vezes obtidas por meio de raspagem automatizada. Selenium, conforme documentado por Russell e Norvig (2021) e também por Jurafsky e Martin (2020), é um framework que permite a simulação de ações humanas em navegadores web. Ele opera através de uma arquitetura cliente-servidor, enviando comandos via WebDriver a navegadores como Chrome, Edge e Firefox, o que o torna ideal para interagir com páginas carregadas dinamicamente por JavaScript. De acordo com Manning e Schütze (1999), o domínio da automação de coleta e o controle sobre a estrutura HTML são pré-requisitos para análises modernas de dados. Essa integração de IA e automação reflete a visão de OpenAI (2025) de que as inteligências artificiais assistivas funcionam como extensões cognitivas, ampliando a capacidade de aprendizado e pesquisa dos estudantes.

3. DESCRIÇÃO DO VÍDEO E ANÁLISE CRÍTICA

O vídeo “Como fazer Web Scraping utilizando Python e Selenium?”, do canal Comunidade DS, tem caráter prático e instrucional. O apresentador explica de forma progressiva como instalar, configurar e utilizar a biblioteca Selenium para coletar informações de páginas dinâmicas. A abordagem é acessível, mas tecnicamente rigorosa, adequada tanto para iniciantes quanto para estudantes de cursos tecnológicos. Inicialmente, o instrutor demonstra a instalação do Selenium por meio do comando “pip install selenium” e explica a necessidade de baixar o WebDriver compatível com o navegador utilizado. Em seguida, aborda a configuração do ChromeDriver e a

abertura de uma página com o comando “driver.get(url)”. O vídeo prossegue mostrando como localizar elementos na página através de seletores (By.ID, By.CSS_SELECTOR, By.XPATH) e extrair dados específicos com o método “.text”. Durante a explicação, o canal Comunidade DS enfatiza a importância das esperas explícitas e do módulo WebDriverWait, que garante que o script aguarde o carregamento completo dos elementos antes de tentar acessá-los. Essa técnica é considerada boa prática fundamental para evitar erros de sincronização. O vídeo ainda demonstra como salvar os dados coletados em estruturas como listas e, posteriormente, exportá-los para arquivos CSV, facilitando análises futuras.

4. EXPLICAÇÃO TÉCNICA E CÓDIGOS EM PYTHON

A seguir, apresenta-se um exemplo de código, inspirado na demonstração do canal Comunidade DS, com explicações detalhadas: from selenium import webdriver from selenium.webdriver.chrome.options import Options from selenium.webdriver.common.by import By from selenium.webdriver.support.ui import WebDriverWait from selenium.webdriver.support import expected_conditions as EC options = Options() options.add_argument("--headless=new") driver = webdriver.Chrome(options=options) driver.get("https://exemplo.com") wait = WebDriverWait(driver, 10) elementos = wait.until(EC.visibility_of_all_elements_located((By.CSS_SELECTOR, "div.card"))) for e in elementos: titulo = e.find_element(By.TAG_NAME, "h2").text preco = e.find_element(By.CLASS_NAME, "price").text print(titulo, preco) driver.quit() Explicação linha por linha: • As bibliotecas selenium.webdriver e suas classes (Options, By, WebDriverWait, EC) fornecem os mecanismos para controle do navegador e interação com o DOM. • O modo “headless” permite executar a automação sem abrir a janela do navegador, economizando recursos. • “driver.get()” acessa a URL desejada. • WebDriverWait e ExpectedConditions asseguram que o programa só prossiga após o carregamento dos elementos-alvo. • O loop percorre os blocos de dados (div.card), extrai texto e imprime no console. • O método “driver.quit()” encerra a sessão, liberando recursos.

5. BOAS PRÁTICAS E CONSIDERAÇÕES ÉTICAS

É indispensável, conforme reforçado por Cambria et al. (2017) e por OpenAI (2025), respeitar a ética e a legislação ao realizar raspagens. O uso de Selenium deve observar o arquivo robots.txt, os Termos de Uso e a Lei Geral de Proteção de Dados (LGPD). Dados sensíveis, pessoais ou protegidos por autenticação não devem ser coletados.

6. CONCLUSÃO

A análise do vídeo do canal Comunidade DS revelou um panorama completo sobre o uso de Selenium para automação de navegadores e raspagem de dados dinâmicos. A clareza didática, aliada à profundidade técnica, o torna uma excelente ferramenta de aprendizagem autodidata para estudantes de Inteligência Artificial.

7. REFERÊNCIAS

CAMBRIA, E.; SCHULLER, B.; XIA, Y.; HAVASI, C. New Avenues in Opinion Mining and Sentiment Analysis. IEEE Intelligent Systems, 2017. JURAFSKY, D.; MARTIN, J. H. Speech and Language Processing. 3rd ed. Prentice Hall, 2020. MANNING, C. D.; SCHÜTZE, H. Foundations

of Statistical Natural Language Processing. MIT Press, 1999. MITCHELL, T. Machine Learning. New York: McGraw-Hill, 1997. RUSSELL, S.; NORVIG, P. Artificial Intelligence: A Modern Approach. 4ª ed. Pearson, 2021. OPENAI. ChatGPT – Large Language Model as an Academic Assistant. OpenAI Documentation, 2025. COMUNIDADE DS. Como fazer Web Scraping utilizando Python e Selenium? YouTube, 2025. Disponível em: . Acesso em: nov. 2025. BRASIL. Lei nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). Diário Oficial da União, Brasília, DF, 15 ago. 2018.