KUNAL ZAVERI
Python Developer

ABOUT ME

As a Python Developer with over 2 years of professional experience, I have a robust track record in designing, developing, and implementing applications based on Python, Django, DRF, Flask, and client-server technologies. I am highly skilled in RESTful services and have extensive experience delivering web scraping solutions using Scrapy, Playwright, and Selenium. My expertise in Python, Django, DRF, and SQL has enabled me to build high-quality, scalable, and reliable software solutions.

In addition to my technical skills, I possess excellent communication, analytical problem-solving, decision-making, time management, and teamwork abilities. I am constantly seeking opportunities to enhance my skills and knowledge, and I am passionate about leveraging technology to solve complex problems and create innovative solutions.

PROFESSIONAL SUMMARY

- Designed, coded, and debugged operations, reporting, data analysis, and web applications using Python.
- Demonstrated expertise in delivering web scraping solutions.
- Proficient in handling client inquiries and providing accurate time estimations for projects.
- Utilized Version Control Systems like GIT to organize code versions and configurations.
- Specialized in developing RESTful services using Python.
- Applied Object-Oriented Python, Django, PostgreSQL, Exception Handling, and Collections in software development.
- Extensive experience in Python frameworks (Django, Django Rest Framework) and IDEs such as PyCharm, Sublime Text, Spyder, and VS Code.
- Hands-on experience with various Relational Database Management Systems (RDBMS) including PostgreSQL, Sqlite3, and Non-Relational Database Management Systems like MongoDB.
- Familiarity with web scraping libraries such as BeautifulSoup, Scrapy, and Playwright.
- Proficient in implementing Scheduled Tasks in the background using Celery.
- Experienced in scraping data and automated testing using Selenium.
- Developed a generic scraper for scraping with Scrapy and Selenium.

EDUCATION

U.V.PATEL College of Engineering, Ganpat University, Kherva, Mehsana — 2018 - 2022
BE, Information Technology Bachelor of Engineering

TECHNICAL SKILLS

Languages/ Technologies	Python3, Sql, HTML, CSS, Javascript
Frameworks	Django, Flask, Django REST framework, Scrapy, Selenium
Libraries/APIs	NumPy Pandas, Scikit-learn, Beautiful Soup, Boto3, Pytest
Tools	Git/GitHub, Jira, Jupyter, Slack, PyCharm, VS Code
Paradigms	Object-oriented Programming (OOP),Model View Controller (MVC),

KUNAL ZAVERI
Python Developer

Storage	Storage MySQL, PostgreSQL, AWS S3, MongoDB

PROJECT SUMMARY

1. Words Processing Scrapping

Overview

In this project, we are tasked with scraping data from two different websites. Our main focus is to extract words from one website and compare them with the content of the other. If a match is found, we proceed to create a docx file that accurately reflects the structure and format of the original website. This includes preserving the layout, font, and any other relevant formatting details. Furthermore, we will be checking if the source sentences in the existing docx match with any of the words we have scraped. If a match is identified, we will add the corresponding sentences to the docx file we are creating. To ensure the words are easily readable, we will also convert any phonetic letters present in the scraped words into their normal form.

Roles & Responsibilities

- Created the documentation of scrap words from the website
- We also need to change phonetic letters into normal letters while scrapping words from the website
- Created the docx from the given words scraped by and created individually docs for every individual word
- While scrapping from a website and creating a docx of every word we should need to make sure the docx should have some content as the website has where bold or italic letters are present there should be bold and italic letters available in the docx file also
- We also need to check some phonetic words should remain the same while saving the docx files
- We need to create different variations of the source words and match all the variations in the given source sentences docx if any word matched with any sentences present in the word and send that sentences in our created target docx. There are also some conditions while pasting the source sentences in the docx.

Technology:- Python, scrapy, selenium, beautiful soup

2. Scrapping Tool

Overview:

This advanced system extracts valuable data from diverse websites, utilizing Scrapy for structured information retrieval and Selenium for dynamic content capture. The scraped data seamlessly integrates into an ElasticSearch-based search engine through Django Rest Framework. The project stands as a testament to my skills in web scraping, automation, and building efficient search solutions, providing users with a refined and efficient experience in accessing and navigating vast amounts of data.

Role & Responsibility:

- Employed Scrapy for structured data extraction and Selenium for dynamic content capture, ensuring comprehensive coverage of the target website. Additionally, successfully tackled the challenging task of extracting data from PDFs by identifying common structures and implementing a strategy for data retrieval.

- Implemented data transformation processes to organize and structure the scraped information for efficient storage and retrieval.
- Designed and implemented integration with ElasticSearch, ensuring optimal indexing and mapping of scraped data into Elasticsearch for fast and accurate search functionalities.
- Developed a user-friendly search engine using ElasticSearch, enabling users to interact with and retrieve information effortlessly.
- Implemented advanced search functionalities, such as full-text search, filters, and sorting, to enhance the user experience.
- Conducted thorough testing of the scraping tool to ensure accurate data extraction, handling edge cases to improve data quality.

Technology: Selenium, Scrapy, ElasticSearch, BeautifulSoup, Django Rest Framework.

3. E-Commerce Web Scrapping

Overview:

In this project, our objective was to scrape product numbers from various E-commerce websites using Scrapy and Selenium. The task involved extracting data for specific URLs provided. When a user pastes a URL from the target site, the information for that particular product is scraped and made visible to the end user. Additionally, we implemented an API to send the scraping response, ensuring that the end user can view this information seamlessly

Role & Responsibility:

1. **Scrapping with Selenium:**
    - Initially, the web scraping process began using Selenium for more than 100 ecommerce sites
    - Selenium, while effective, posed challenges when deployed on a server, leading to memory usage errors.
2. **Transition to Scrapy:**
    - To address the memory consumption issues, there was a strategic shift to Scrapy as the scraping tool.
    - Scrapy was chosen for its efficiency in resource utilization and server-friendly characteristics.
3. **Challenges with Dynamic Content:**
    - Ecommerce websites often load data dynamically using JavaScript, complicating the scraping process.
    - The transition from Selenium to Scrapy became necessary due to the limitations of Selenium in handling dynamic content.
4. **Innovative Solution for Dynamic Content:**
    - A solution was devised by closely examining website structures and leveraging script tags, specifically those containing the application/ld+json format.
    - This approach allowed for the extraction of valuable data embedded within script tags, contributing to a more comprehensive scraping strategy**.**
5. **Development of Universal Scraper:**
    - A significant achievement was the creation of a Universal Scraper, designed to extract data from various ecommerce sites.
    - The Universal Scraper is capable of utilizing meta tags to obtain relevant information, offering versatility across different website structures.
6. **Shopify-Specific Scraper:**

- Recognizing the prevalence of Shopify-powered ecommerce sites, a specialized Shopify-specific scraper was developed.
- This scraper is engineered to effectively extract data from any Shopify website, providing a tailored solution for this widely used ecommerce platform.

Technology: **Scrapy, Selenium, Flask**

4. Iron Depot Scrapping

Overview:

In this project, I utilized Python, Scrapy, and Selenium to extract data from Facebook and various other sites. I developed a PostgreSQL pipeline to efficiently store and update the extensive data. Implemented a mechanism to drop incomplete data lacking essential fields. Demonstrates expertise in large-scale data extraction, storage, and maintenance.

Role & Responsibility:

- When scraping data from websites, it's important to ensure that if required fields' data is not available, it is not scraped and added to the database.
- When scraping data from Facebook, we need to ensure that our session ID is not detected.
- I've implemented a pipeline in the code to save data into the database, ensuring there are no duplicate records, and removing data from the database which are not avaiable on the sites.
- Additionally, I'm downloading images while scraping from the URL and storing them in an S3 bucket. The links to these images are also stored in the S3 bucket.
- The client needs to monitor the scraping process for any errors. We've set up email notifications and Spidermon for monitoring during the scraping process.
- To ensure ease of future modifications, I've structured the project so that if anything changes on the website, the client only needs to modify one section of the code, and the code will run smoothly again

Technology: Scrapy, Selenium, AWS, Spidermon