Vergabepilot.AI – LLM-based Tender Scraping for State-of-the-Art Tender Search

1. Project Context and Methodology

Vergabepilot.AI is an emerging German startup that enables **high-precision tender search** using **detailed natural-language requirements and exclusion criteria**. This leads to significantly better search effectiveness than traditional keyword-based tender portals. While the platform currently focuses on the **German market**, an extension to the **European market** is planned.

A core dependency of the tender search pipeline is the **tender document set**: each tender can consist of **multiple documents** (PDFs, attachments, forms, etc.). These documents are the **foundation** for search relevance and other downstream tasks such as requirements matching, summarization, and ranking. Therefore, it is **critical** that – alongside **Tender metadata** (title, buyer, deadlines, CPV codes, locations, procedure type, etc.) – scraping reliably extracts **all associated documents**.

In practice, tender platforms are heterogeneous and constantly changing, which makes scraping **fragile** and costly to maintain. The project focuses on improving the scraping layer of Vergabepilot.AI using **LLM-based methods**.

Core Idea:

Instead of writing scrapers manually, students will explore and build a framework to **generate, validate, and improve tender scrapers automatically** using LLMs and LLM-based agents. The project treats scraper creation as an iterative and automated engineering loop: **observe website, generate scraper, test, diagnose failures, improve.**

This includes comparing:

- **manually implemented scrapers** (baseline production approach)
- **LLM-generated scrapers** (automation approach)
- **MLLM agent-based scraping** (fallback when classical scraping breaks)

Knowledge Share Format:

Students will document and explain decisions and trade-offs across the team, including:

- scraper generation strategies and prompting choices
- evaluation design for document completeness
- failure analysis (what broke and why)
- comparison of manual vs. generated approaches

Primary Domain:

We will use **LLMs, LLM-based agents, and web automation/scraping methods** to address real-world data acquisition challenges in the public procurement domain.

2. Project Timeline

Weeks 1–4: Setup, Data Acquisition & Evaluation Foundations

Project setup and reproducible development environment. Collection of data from **both successful and failed scraping runs** (provided by Vergabepilot.AI). Definition of data processing pipelines and evaluation metrics with a strong focus on **tender document completeness**. Initial small-scale analysis of scraping failures. Initial preparation for the LLM-based scraping approaches.

Artifacts: Setup guide, data processing pipeline, evaluation definition, initial failure analysis.

Weeks 5–8: LLM-based Scraper Generation & Comparative Evaluation

Exploration of different **LLM-based approaches to automatically generate scrapers** for diverse tender providers. Systematic evaluation of generated scrapers using the previously defined metrics on the test dataset. Comparative analysis of manual vs. LLM-generated scrapers to understand strengths, weaknesses, and robustness under changing website conditions. Initial preparation for agent-based scraping approaches.

Artifacts: LLM-based scraper generation pipeline, evaluation benchmark, comparative study, iteration report.

Weeks 9–12: MLLM Agent-based Scraping & Failure Classification

In-depth exploration of **MLLM agent-based scraping** for dynamic and hard-to-scrape tender platforms. Evaluation of different agent configurations and in-depth analysis of failure cases. Development of an **automatic failure classification method** to diagnose scraping issues and recommend fallback strategies. Final system evaluation and documentation.

Artifacts: Agent-based scraping baselines, failure taxonomy and distribution, automatic failure classifier, final report and reusable evaluation kit.

3. Technical Scope

- Python and/or Java
- Programming with LLMs
- LLM tool calling and structured generation (schema-guided outputs)
- Web scraping engineering (Playwright/Selenium, DOM parsing, downloads)
- Agent-based web automation with MLLMs (GUI navigation, dynamic sites) (e.g., OpenAI CUA)
- Virtual GUI environments (e.g., *xvfb* virtual displays, *Xfce* desktop environments, *xdotool* for simulating user actions)
- Software engineering: REST APIs, VS Code/Cursor

4. Links

- https://www.vergabepilot.ai/
- https://github.com/openai/openai-cua-sample-app
- https://github.com/microsoft/playwright-python