2025

**HAN University of Applied Sciences**

**MDD-Minor- Data Driven Decision Making in Business**

| | |
|---|---|
| Assignment type | **Individual Portfolio.** |
| Exam name | Individual Portfolio Data Science Tools and techniques |
| Report title | AI powered rental alert system (Initial idea)<br>AI agent for automation |
| Exam code | TOETS-08 |
| Student name | Showkot Nadim |
| Student number | 1679642 |
| Academic year | 2024/2025 |
| Semester | 1 or 2 |
| File or document name | Report Individual Project-Data Science Tools and Techniques-Showkot Nadim-1679642 |

## Contents

## AI Powered rental alert system

When I started exploring the videos I was not able pick a video. And then I read the assignment guidelines again, I found out the assignment should be something that I face problem currently or I interacted with, and my product should provide a solution. Then I realized that I have been struggling a lot find a room at Helix( one of HAN housing complex). As a former tenant at SSH&, I understand how competitive and time-sensitive the rental process is. especially since new listings are published every Wednesday at 12:00 PM and early applicants have a higher chance of being selected. However, I often miss these opportunities simply because I forget to react on time. To solve this problem, I wanted to develop an "AI-powered rental listing alert system" that sends real-time WhatsApp notifications as soon as a new listing is published.

There were some things that stood in the way when I was setting up the rental alert system. The main problem at first was that SSH& (https://www.sshn.nl/) does not provide an API for use. An API is the link websites use to allow developers to get data and use their functions safely and without difficulty. Services of this category, for example, OpenAI, Google Maps, and YouTube, usually give you an API key that certifies you have permission to access their API. For this reason, you can connect apps more smoothly and set access rights for your data. Because SSH& doesn't have either an API or a key, I couldn't access it directly. That is why I started to use web scraping to find the necessary data.

Initially, I used Python along with the requests and BeautifulSoup libraries to collect information from the SSH& website with web scraping. They are good for extracting data from HTML pages that are not complicated. However, it did not take long to understand that the website includes possibly JavaScript-rendered content or dynamic content that is beyond the scope of simple scraping tools. So, I was unable to get the listing info as expected, and it might be necessary to use advanced tools or methods for the task.

Another option I looked at was the Twilio API since it is both popular and reliable. I would have gotten a direct notification on my phone whenever something new was posted. While Twilio is a useful tool, I had to skip it since I didn't want to spend any money. I tried searching for other options, which did not work out for me. After that, then I decided to leave the messaging part out for now.

## AI agent for automation

However, I then continued with the AI agent for automation. I would say I have enjoyed the most making the AI agent as it was frustrating (creating the environment) and exciting as I finally made the workable ai agent. Why was it frustrating? While creating the environment I had to install lots of dependencies which required lots of space in my storage and my laptop crashed at some point. Then I borrowed my flatmate's mac, and I had to install everything from the scratch( anaconda navigator ,python, vs code) and a LLM called ollama and all the necessary python packages and dependencies( will be discussed and showed below)

I have used browser use, web-ui and python to create the ai agent. I did it manually, however, there are websites like n8n (https://n8n.io/ ) through this website it is possible to make an ai agent automatically and more easily(without any coding).

You might be wondering why creating an AI agent whereas there are paid options ?

While OpenAI offers a similar AI agent feature through its Pro plan, which costs around €200 per month and has some limitations, like where you can use it and how much you can do (Introducing Operator, 2025) The reference video of open ai operator/agent

(https://openai.com/index/introducing-operator/?utm_source=chatgpt.com&video=1049531514)

On the other hand, to create an AI agent using a web browser is free, has no limits, and it can be customized it however we want. This makes it a better and cheaper option for doing the same kinds of tasks.

Why did I want to build an AI agent, the motivation was from the rental alert listing via whatsapp, I planned that If I receive a message about the listing then I should automate more, I mean I should just tell my ai agent to login to SSH& website using my login credentials and find a room for me according to my preferences and apply for me.

Here I would like describe part by part how I built the agent for a person who does not have any ideas whatsoever about any sort of AI agent knowledge.

## Setting Up Python and BrowserUse for AI agent Automation

First, we'll have to make sure certain key steps are followed so that BrowserUse can be used.

**Step 1: Install Python**

Python is the starting point for everything we have to do. Ensure that Python is installed in our computer system. To download it, visit python.org and choose the installation instructions that are right for our operating system.

Since I opted for Anaconda Navigator, the Python language is pre-loaded, and it has many helpful packages for my development activities. I used Anaconda Navigator to start VS Code, so I could work on my projects.

**Step 2: Install BrowserUse**

Once Python is installed, the next step is to install BrowserUse, a key component for interacting with the web. Browser Use is the easiest way to connect our AI agents with the browser. It makes websites accessible for AI agents by providing a powerful, yet simple interface for browser automation. (Welcome to Browser Use - We enable AI to control your browser, 2025) To do this, by opening terminal (command prompt) and typed the following command:

```
pip install browser-use
```

For memory functionality (requires Python<3.13 due to PyTorch compatibility):

```
pip install "browser-use[memory]"
```

Install the browser:

```
playwright install chromium --with-deps --no-shell
```

**Install Playwright**

Playwright is another important tool that BrowserUse depends on to automate the browser. We Install it with this command in terminal:

pip3 install playwright

Then spin up the agent

```
import asyncio

from dotenv import load_dotenv

load_dotenv()

from browser_use import Agent

from langchain_openai import ChatOpenAI


async def main():

  agent = Agent(

    task="Go to SSH& website and loginto mijn sh& using my credentials username: mdshowkotalinadim@gmail.com password: Showkot@168 , then find a room for me with rental benefit and apply for me. Thank you",

    llm=ChatOpenAI(model="gpt-4o"),

  )

  await agent.run()
```

Add preferred API keys for the provider that we want to use to our .env file.

For example

**OPENAI_API_KEY=**

**ANTHROPIC_API_KEY=**

**AZURE_OPENAI_ENDPOINT=**

**AZURE_OPENAI_KEY=**

**GOOGLE_API_KEY=**

**DEEPSEEK_API_KEY=**

**GROK_API_KEY=**

**NOVITA_API_KEY=**

To know more about browser use please  check out the browser use documentation site please. **Introduction - Browser Use**

Now as we have our browser use setup and all the dependencies are installed we can continue with setting up our Web UI environment.

Firstly, what is web-ui and how it is going to help me build the AI agent. WebUI is an extensible feature-rich, and user-friendly self-hosted AI platform designed to operate entirely offline. It supports various LLM runners like Ollama and OpenAI-compatible APIs, with built-in inference engine for RAG, making it a powerful AI deployment solution. (Open web-ui, 2025)

## Set up the Web UI

BrowserUse comes with a Web User Interface (UI), which makes it easier to interact with the system. To set up the Web UI, we follow these steps:

1. **Clone the Repository**: Download the Web UI repository from GitHub or alternatively we can clone anything with GIT(download and then clone it locally- https://git-scm.com/)

   The repository I want to clone is this- https://github.com/browser-use/web-ui.git

2. **Navigate to the web_ui Folder**: So simply we can create a folder namely, ai-agent-demo and then in In the terminal, go to the web_ui folder within the cloned project directory.

## Install UV (Universal Versioning)

UV is a Python package manager, and it helps in managing dependencies. It is written in rust programming language. (Introduction, 2025) From uv we can get this:-

`curl -LsSf https://astral.sh/uv/install.sh | sh` and run in our terminal or command prompt.

It will manage our python environment.

After this we install:-

```
uv venv --python 3.11
```

Activate the virtual environment:

```
source .venv/bin/activate
```

**Install Dependencies**

Install Python packages:

```
uv pip install -r requirements.txt
```

Install Browsers in playwright.

```
playwright install --with-deps
```

Or install specific browsers by running:

`playwright install chromium --with-deps` – this creates a web browser, when we ask the agent our task it opens a chromium web browser then performs it’s task.

**Finally  Enjoy the web-ui**

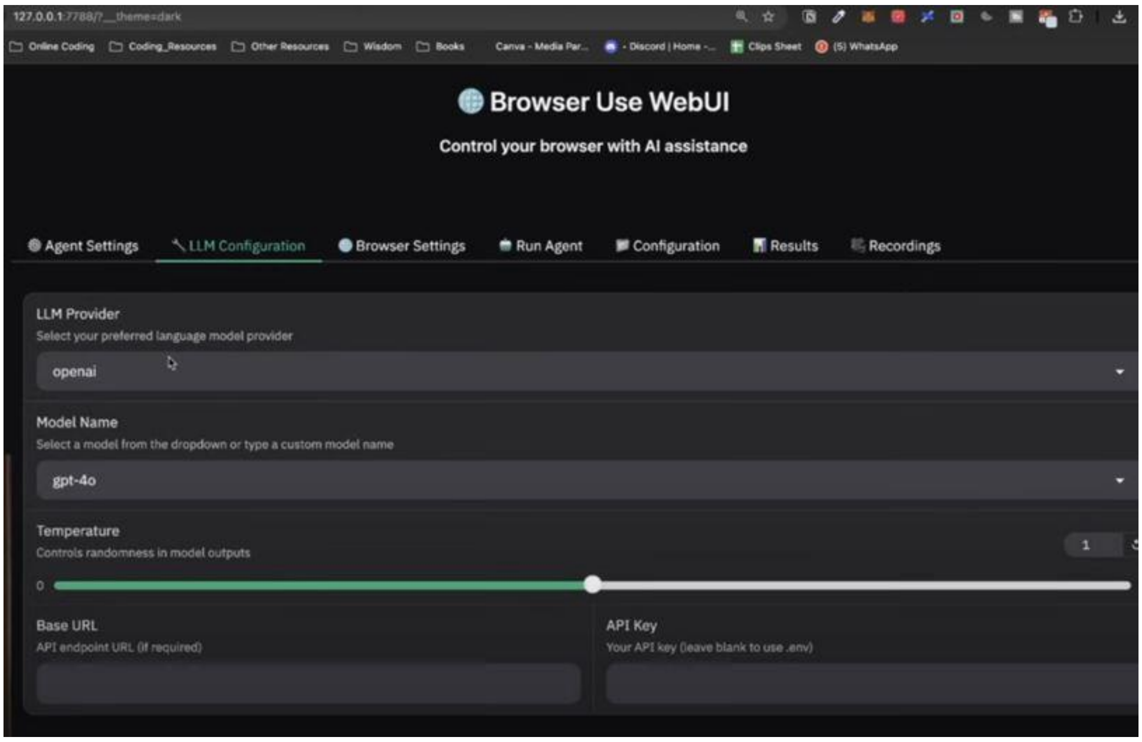
**Run the WebUI:**

```
python webui.py --ip 127.0.0.1 --port 7788
```

after running this in our terminal we get this local host created within our environment

http://127.0.0.1:7788  this local host only runs within the created environment.

This is the interface of the web-ui



Browser Use WebUI

Control your browser with AI assistance

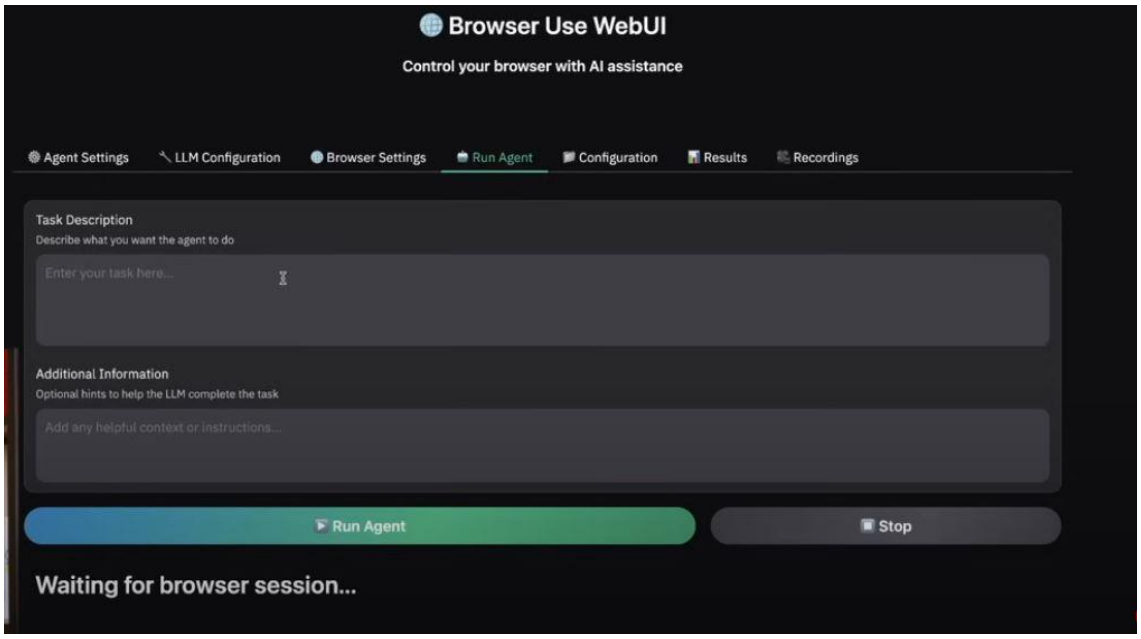
LLM Provider
Select your preferred language model provider
openai

Model Name
Select a model from the dropdown or type a custom model name
gpt-4o

Temperature
Controls randomness in model outputs
0 1

Base URL
API endpoint URL (if required)

API Key
Your API key (leave blank to use .env)



Browser Use WebUI

Control your browser with AI assistance

Task Description
Describe what you want the agent to do
Enter your task here...

Additional Information
Optional hints to help the LLM complete the task
Add any helpful context or instructions...

Waiting for browser session...

## Configure our AI Agent

When the new Web UI is ready, configuration of our AI agent will be possible. Several sections will be present in the Web UI to ensure this process moves smoothly.

1. **LLM Configuration**: In the LLM Configuration section, we can pick the preferred AI language model (LLM). A few examples are using models such as Gemini or DeepSeek. I received the API key for Gemini from Google AI studio . We should enter our API key and identify the model that we want to use. A feature called "temperature" makes it possible to control how much creativity or randomness is used in the output.
2. **Run Agent**: Here's where the magic happens. simply write a prompt describing what we want the AI to do. Please refer to the video link- https://drive.google.com/drive/folders/1uRgLT6x334wpXAR4yInoA8g8kvrS352O
   a. For example, here in this video you can see I instructed the agent to log into SSH& website, using my credentials and find a room for me, in the video you can see it logged into my account to the website and searched a room for me within my preference.
   b. Results Page: After the agent runs the task, the results page will show a recording of what the agent did.( refer to the video link please) You'll see the final output or any errors that occurred during the process. The first video in the drive is the record of what the agent did. And the second video was filmed by my phone from setting up and running till getting the result.

As I used my credentials with SSH to access the AI agent and find a room, I discovered that it can perform lots of other helpful activities on different platforms. The AI agent can do things like surfing the internet, just as quickly and better than a human can.

For example, here are some of the smart and high-impact tasks I can automate using this AI agent:

| Use Case | Tasks Can be Automated |
|---|---|
| 1. Online Shopping Automation | • Search for products (electronics, groceries, clothings)<br>• Add to cart & place orders using saved detaits(using pay account) |

| | |
|---|---|
| | • Track orders or initiate returns |
| 2. Travel & Booking Tasks | • Book hotels/flights based on custom criteria<br>• Fill out booking forms & complete process |
| 3. Login & Account Automation | • Auto login to websites using credentials<br>• Check past orders or navigate account settings |
| 4. Email & Communication Management | • Send AI-generated emails<br>• Organize inbox, mark important, delete spam |
| 5. Web Data Extraction | • Scrape prices, reviews, news articles<br>• Download reports, files, summaries |
| 6. Social Media Interactions | • Post updates, images, or stories<br>• Like, comment, follow/unfollow accounts |
| 7. Scheduling & Reminders | • Add calendar events<br>• Set up reminders or alerts |
| 8. Banking & Payments | • Check balances or recent transactions<br>• Pay bills or transfer funds |

It was lots of fun and I learned a great deal throughout this project. In this work, I figured out how AI agents can reach their full potential when applied with tools like BrowserUse. With Google AI Studio, I managed to work directly with LLMs such as Ollama, DeepSeek R1, and Gemini, setting the correct parameters through their APIs and configuring how creative the output would be through the temperature variable.

I managed to grasp the basics of web scraping through requests and BeautifulSoup, learning how data can be swiftly taken from different websites. Also, I was introduced to Playwright which allows browser automation, managing the setup of virtual environments, using requirements.txt to handle dependencies, and running a Web UI to manage my AI processes.

Moreover, I discovered how to set up intelligent automation prompts, engage with websites by automation, and plan to explore the basis of n8n’s workflow creation. (Flexible AI workflow automation, 2025) By developing these skills, I was able to experience how AI, automation, task handling, and real-time browser use combine to bring many possibilities.

All in all, I gained both useful skills and insights into AI, web automation, and workflow design, which has excited me even more. I plan to continue uncovering and developing more intelligent systems ahead.

## Link to the presentation:

https://youtu.be/SQyP5fWKhxo

## Link to the Drive for agent input and output-Video

https://drive.google.com/drive/folders/1uRgLT6x334wpXAR4yInoA8g8kvrS352O

## Link to the GitHub Repository:

https://github.com/Nadim8188/Individual-Project-.git

## Bibliography

*Flexible AI workflow automation*. (2025, June 5). Retrieved from n8n: https://n8n.io/

*Introducing Operator*. (2025, January 23). Retrieved from Open AI: https://openai.com/index/introducing-operator/?utm_source=chatgpt.com

*Introduction*. (2025). Retrieved from uv: https://docs.astral.sh/uv/

*Open web-ui*. (2025, June 6). Retrieved from Web-ui: https://docs.openwebui.com/

*Welcome to Browser Use - We enable AI to control your browser*. (2025, June 5). Retrieved from Browser Use: https://docs.browser-use.com/introduction