Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences

Unternehmenssoftware Summersemester 2024

Documentation - Project AOPSE

by

Adrian Tippe 584501
Christoph Nicklas Jänicke 584533
Ilkaan Bingöl 584398

Berlin, July 6, 2024

Contents

1 Introduction

In the summer semester of 2024, the task for the Unternehmenssoftware class was to develop an AI-powered application using either an OpenAI model or an open-source model.

This document outlines our idea, the concepts, the challenges we encountered during development, and the details of the project itself.

The code of the application and the documentation can be found in the GitHub repository c-jaenicke/project-aopse.

1.1 Project AOPSE

This project is called *Project AOPSE*, which stands for *AI-Driven OSINT People Search Engine.*

2 Idea

In todays digital age, online privacy is more critical than ever, as personal information is constantly at risk of being exposed, stolen, or misused.

Our project aims to enhance user privacy and online security by utilizing Open Source Intelligence (OSINT) techniques, service breaches, and search engines. By leveraging these tools, we can identify and address vulnerabilities, providing users with insights and strategies to safeguard their personal data.
This proactive approach not only helps in mitigating potential risks but also empowers users to take control of their online presence and protect their information.

3 User Story and Use Case

The following chapters will demonstrate the use case and explore a user story.

3.1 User Story

Character: Sarah Johnson, a 34-year-old marketing professional

Background: Sarah is a tech-savvy individual who values her online privacy and security. She regularly shops online, uses social media. Recently, Sarah has become increasingly concerned about the safety of her personal information after hearing about data breaches and privacy invasions in the news.

Goal: Sarah wants to ensure her online activities are secure and her personal data is protected from potential threats.

User Story: As Sarah Johnson, a marketing professional concerned about my online privacy, I want an application that can identify and fix weaknesses in my online presence. This way, I can understand potential risks, get useful recommendations, and improve my digital security to protect my personal information.

Scenario:

1. **Discovery:** Sarah learns about the AI-powered application through an online tech forum discussing the latest tools for improving online security.
2. **Setup:** She opens the website and registers for the service.
3. **Initial Scan:** Sarah inputs her email address and begins the scan.
4. **Results and Insights:** The service provides her an overview of findings using her email address as a query. It offers advice on how to improve potential issues.
5. **Action and Improvement:** Sarah asks the service some questions to clarify actions and potential issues. She follows the recommendations to improve her privacy and security.
6. **Result:** With the help of the service, Sarah feels more confident about her online privacy and security.

3.2 Use Case

The following use cases have been identified:

1. **Initial Assessment:** Using publicly available information on the user and cross-references his data against known breaches.
2. **Privacy Risk Evaluation:** Generate a report detailing the users online exposure, risks and the severity of identified breach.
3. **Mitigation Strategies:** Provide instructions on how to mitigate the risks and enhance the security of the user.

4 Concepts

The following concepts and paradigms are being used.

4.1 OSINT

Open Source Intelligence (OSINT) is intelligence produced by collecting and analyzing publicly available informationen [1]. Using manual research or automated tools.

The project leverages OSINT and automated OSINT tools to find information about the user.

4.2 Stateless

Given the the sensitive nature of the data, the application will not save any data connected to or requests made by the user. This ensures a high level of confidentiality and minimizes the risk of data breaches.

Users are responsible for saving the information themselves.

4.3 Living off the Land

Living off the Land (LOTL) is a technique employed by cybercriminals during malware attacks. It is based on the concept of ”using what you are given” meaning attackers leverage existing tools and resources on the victim's system rather than introducing additional malware or external tools [2].

The application employs this concept by working solely with the data provided by the user. This ensures that it does not seek out loosely connected information, thereby avoiding the risk of compromising other individuals and their data or inadvertently encouraging searches for unrelated individuals.

4.4 Running Locally

For the maximum amount of privacy and security the project has been made open-source and with the possilibity of running it locally on any machine in mind.

5 Development

The following sections will describe the development, the applications and frameworks we used an the challenges we faced.

5.1 OpenAI or Open-Source

We decided to use the OpenAI model *GPT-3.5 Turbo* [1]. Our testing has shown that this model produces the most accurate information related to privacy and security compared

[1] `https://platform.openai.com/docs/models/gpt-3-5-turbo`

to open-source models.
In addition to that its relativly cheap and easy to use.

5.2 Backend

The backend was developed using Python [2]. We used the FastAPI [3] framework to build the backend and used the Official OpenAI Python library to call the model [4].

5.3 Frontend

The frontend was built using SvelteKit [5] utilizing the Skeleton UI library [6].

5.4 Structure

The image below shows the internal structure of the application and the flow of information:

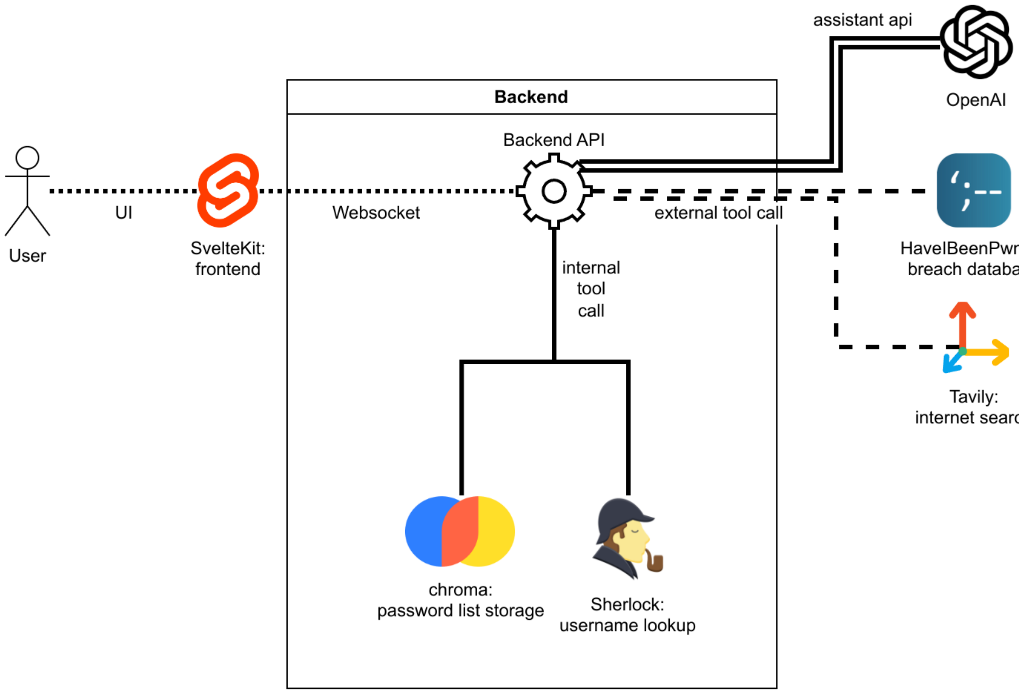

Figure 1: Structure of the application

[2] https://www.python.org/
[3] https://fastapi.tiangolo.com/
[4] https://github.com/openai/openai-python
[5] https://kit.svelte.dev/
[6] https://www.skeleton.dev/

The user interacts with the frontend.
The frontend then sends a request to the central backend API, which calls all the other APIs and tools.

5.5 Tools and APIs

We have decided to use a combination of different APIs and tools, the following chapters will explain the tools and their integration into the project.

5.5.1 chroma

we use chroma [7], a verctor based database to store list of leaked passwords. These lists were obtained from the GitHub repository 00xBAD/kali-wordlists which contains a number of different lists. We decided to use the following wordlists, which contain a mix of different passwords:

1. wifite
2. vnc_passwords
3. unix_passwords
4. sqlmap
5. password
6. keyboard-patterns
7. fasttrack
8. default_pass_for_services_unhash
9. common
10. adobe_top100_pass

More wordlists could be added at any time and will be loaded when a new thread is created.

5.5.2 Sherlock

Sherlock [8] is a command line interface tool which searches for usernames across 400 social media sites and networks.

The tool has been modified to be used as an internal tool, without the need to call it via a new process.

[7] `https://www.trychroma.com/`

[8] `https://github.com/sherlock-project/sherlock`

By utilizing Sherlock, users can scan for usernames across numerous social networks, identify old accounts, detect impersonation attempts, and take appropriate action.

5.5.3 HaveIBeenPwned

HaveIBeenPwned [9] is an online tool which searches for a given email in data breaches and informs the user of those breaches.

"A data breach is any security incident in which unauthorized parties access sensitive or confidential information, including personal data (Social Security numbers, bank account numbers, healthcare data) and corporate data (customer records, intellectual property, financial information)."[3]

We use the API provided by HaveIBeenPwned to inform the user about breaches.

5.5.4 tavily

We also use tavily [10], an API to search the internet, which is optimized for LLMs.

We use Tavily to look up new and up-to-date information on security and privacy-related topics.

5.6 Challenges

The following chapters show the different challenges we faced during the development and the solutions we implemented to overcome the challenges.

5.6.1 Frontend

One of the first challenges was the frontend. We had specific requirements and ideas to show the user the output of the application, to summarize it, and to get the user input in an intuitive way.

The goal was to show all the output on the left and have a chat on the right side, as shown in the following image:

[9] https://haveibeenpwned.com/

[10] https://tavily.com/

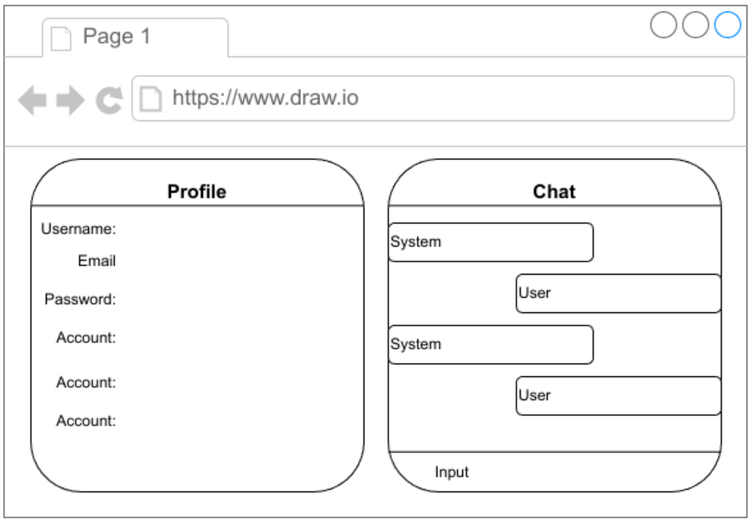

Figure 2: Draft of the frontend

None of the publicly available frontend libraries provided a solution to our requirements. To resolve this we created our own frontend using SvelteKit.

5.6.2 Frontend - Connection to the Backend

By building our own frontend, we faced the challenge of transporting information from the frontend to the backend, and back again. The first attempt, using a simple API on both sides failed.

We then transitioned to websockets, which allow the streaming of data to a listener. By doing so the frontend can send messages whenever and then listens to the response of the backend. This solved timeouts and other issues.

5.6.3 Finding Sources and APIs

Another challenge was the sourcing of information and tools to use in our application. While APIs and sources exist on the internet, most of them were behind a paywall or only available as tools to run locally.

We fixed the issues of local tooling by writing custom APIs for those tools, like spiderfoot [11]. Those APIs are being called by our backend.

In the end we ended up paying for the most important and useful APIs, like HaveIBeenPwned [12].

[11] https://github.com/smicallef/spiderfoot

[12] https://haveibeenpwned.com/

5.6.4 Integrating Sherlock

Since Sherlock is designed as a command-line interface tool, integrating it into the application was challenging.

We had to modify its code, removing most parts that control input and output, including argument parsing and scanning.

5.6.5 Giving the User a Score

We aimed to provide users with a score based on their level of privacy or lack thereof. However, coming up with a formula was not possible.

One issue was the "living off the land" concept we used, relying solely on the data the user provided. We had no way of knowing if the user had given us all relevant information.

Another challenge was determining the weight of the issues found by the tools. For instance, is a breach worse than a leaked password?

Additionally, we had no way of knowing if the user had already resolved an issue. For example, there was no way to verify if a user had changed a leaked password.

Given these issues, providing the user with a score would create a false sense of security.

6 Using Project AOPSE

The project can be run locally. The setup is described in the readme.md in the GitHub repository of the project on c-jaenicke/project-aopse.

The following prompts, or prompts similar to these, will execute a specific tool:

- `check the username <username>`: will start a search for the username on different social media networks
- `check the password <password>`: will check if the password is present in one of the wordlists and give advice
- `has the email <email address> been breached`: will check if the given email address has been found in any breaches
- `is there any new information on <topic>`: will check for new information or news on the topic

In addition to those prompts, any prompt will lead to a result based on the previous messages and context provided.

7 Future Plans

As this is the first version, we have exciting plants for future developments.

This includes the development of a custom AI-model, as to not rely on OpenAI. Improving the performance of the backend. Possibly a full rewrite in a more advanced and safe language, like Rust [13]. While also improving the security further. Implementing more tools and sources of information. And adding customizable search parameters.

[13] https://www.rust-lang.org/

References

[1] R. Gill, "What is open-source intelligence?." https://www.sans.org/blog/what-is-open-source-intelligence/, 2 2023. Accessed on 03.07.2024.

[2] B. Lenaerts-Bergmans, "What are living off the land (lotl) attacks?." https://www.crowdstrike.com/cybersecurity-101/living-off-the-land-attacks-lotl/, 2 2023. Accessed on 03.07.2024.

[3] M. Kosinski, "What is a data breach?." https://www.ibm.com/topics/data-breach, 5 2024. Accessed on 06.07.2024.