BÄRNHÄCKT

Duck Squad
Welcher Wein zu meinem Menu

Technische Informationen für die Jury

Technische Informationen für die Jury

Aktueller Stand des Sourcecodes

- Link zu Github Repository
  https://github.com/nicololuescher/TruebliTipp

Ausgangslage

- Worauf habt ihr euch fokussiert?

Uns war es von Anfang an wichtig, dass wir uns kreativ mit Einbinden können und nicht nur den «offensichtlichen» Lösungsansatz wählen. Zudem hatten wir Lust auf neue Technologien, mit denen noch niemand von uns gearbeitet hat.

Nach der Initialen Brainstorming Phase haben wir die Aufgaben in die grösseren Teilgebiete Frontend, Backend und LLM aufgeteilt und diese dann untereinander verteilt. Dabei waren wir stehts kollaborativ, hatten aber einen «Federführer».

- Welche technischen Grundsatzentscheide habt ihr gefällt?

Der Wichtigste entscheid war wohl, dass wir nicht auf relationale Daten zurückfallen wollen, sondern die Schlüsse dynamisch von einem LLM generieren zu lassen.
Zudem haben wir uns dazu entschieden nicht zu viel Zeit für Funktionalitäten zu verschwenden die nicht nötig sind für das Endprodukt.

Technischer Aufbau

- Welche Komponenten und Frameworks habt ihr verwendet? / Wozu und wie werden diese eingesetzt?

Wir verwenden diverse Technologien und Frameworks. Einig sehr neu, einige bereits länger etabliert.
Im Frontend verwenden wir React mit dem MUI Styling Framework. Mit dieser Lösung können wir recht einfach eine Webapplikation erstellen, über die anschliessend mit unseren Diensten interagiert werden kann.
Im Backend verwenden wir NodeJS mit dem Express Framework. Mithilfe von Express können sehr schnell und effizient Endpunkte erstellt werden. Des Weiteren ist Express weit verbreitet und es gibt sehr viele Ressourcen/Tools. Als Datenbank verwenden wir PostgreSQL. Für die Orchestrierung verwenden wir Docker.
Wir wollten als erstes ein Self-Hosted LLM einsetzten, sind aber schnell zum Schluss gekommen, dass wir wahrscheinlich nicht über die benötigte Rechenleistung verfügen, damit wir das Model performant einsetzten können. Deshalb haben wir uns anschliessend für Google Gemini entschieden. Da wir deswegen eh bereits mit Google APIs interagierten, konnten wir auch direkt die Cloud Vision API von Google für die Bilderkennung verwenden.

Implementation

- Gibt es etwas Spezielles, was ihr zur Implementation erwähnen wollt?

Nahezu alle Daten, die in unserem Tool dargestellt werden sind prozedural generiert worden und stark vom Input des Benutzers abhängig. Durch Prompt Engineering konnten wir die Antworten des LLM in eine standardisierte Form bringen die anschliessend von unserem System ausgewertet werden kann. Wir verwenden das LLM zudem dazu Daten aus den vorhandenen Informationen zu schliessen. Wie z.B., dass Merlot ein Rotwein ist oder dass ein Rioja aus Spanien kommt.

- Was ist aus technischer Sicht besonders cool an eurer Lösung?

Unser Tool fühlt sich sehr «new Age» an. Das in so kurzer Zeit Daten aus einem Bild ausgelesen werden können und anschliessend direkt Daten generiert werden die Relevant sind für den jeweiligen Benutzer ist sehr aufregend und interessant. Auch dass niemand von uns Erfahrung hatte mit der Technologie und wir dennoch innerhalb der kurzen Zeit ein brauchbares Projekt auf die Beine stellen konnten ist sehr motivierend.

Abgrenzung / Offene Punkte

- Welche Abgrenzungen habt ihr bewusst vorgenommen und damit nicht implementiert? Weshalb?

Es gibt viele offene Punkte die unnötige Komplexität in den POC gebracht hätten. Dazu gehört zum Beispiel die Benutzerverwaltung oder die LLM-Sicherheit. Unser LLM kann einfach vom Benutzer missbraucht werden, dies ist aber bei einem POC kein Problem. Wir haben uns ganz klar auf die Dinge fokussiert die den POC einzigartig machen.