Geo Data Search

Find places you can't name yet

# SOFTWARE ARCHITECTURE

AMOS WiSe 20/21 – Project 1: Geo Data Search

# Table of Contents

# Runtime Components

## Diagram

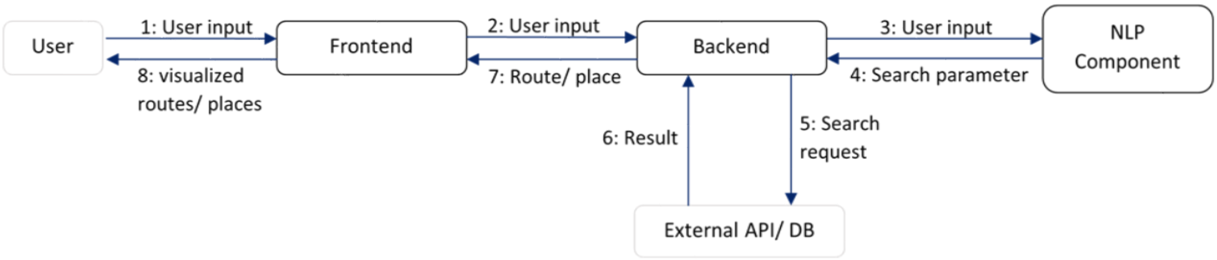

*Figure 1: Runtime Components (black) and external resources (grey).*

## Description

There are three different runtime components: the frontend, the backend and a NLP (Natural Language Processing) component.

The frontend provides the possibility to enter a user input as text and it shows the search results on a map.

The backend is responsible for the communication between the frontend and the NLP component as well as with external APIs or databases.

The NLP component converts the user input into machine-readable data for the backend.

Every component is deployed with Docker. The containers run locally on Linux systems, a cloud deployment and support of Windows is optional. The user is able to use the application through every common browser.

# Code Components

## Diagrams

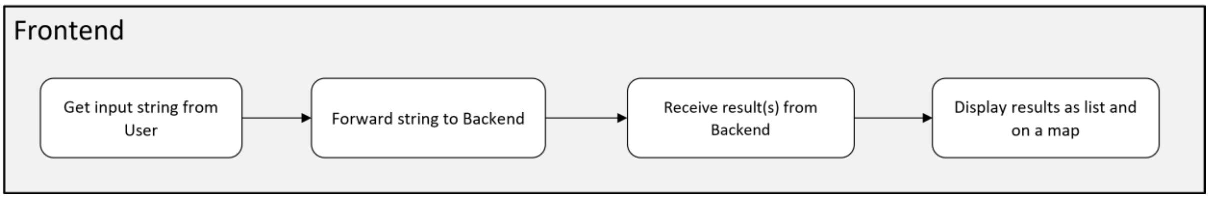
Frontend
Get input string from User
Forward string to Backend
Receive result(s) from Backend
Display results as list and on a map

*Figure 2: Code Components of Frontend.*

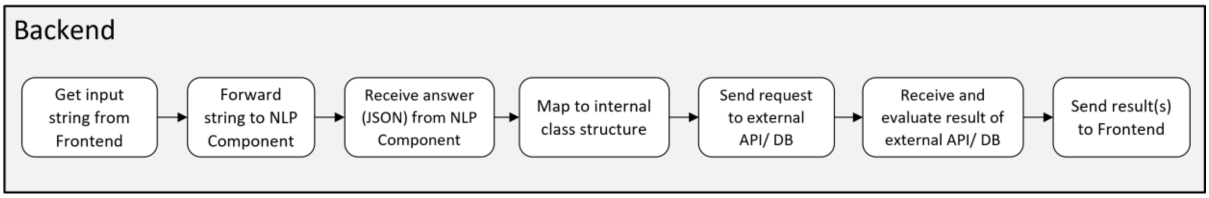
Backend
Get input string from Frontend
Forward string to NLP Component
Receive answer (JSON) from NLP Component
Map to internal class structure
Send request to external API/ DB
Receive and evaluate result of external API/ DB
Send result(s) to Frontend

*Figure 3: Code Components of Backend.*

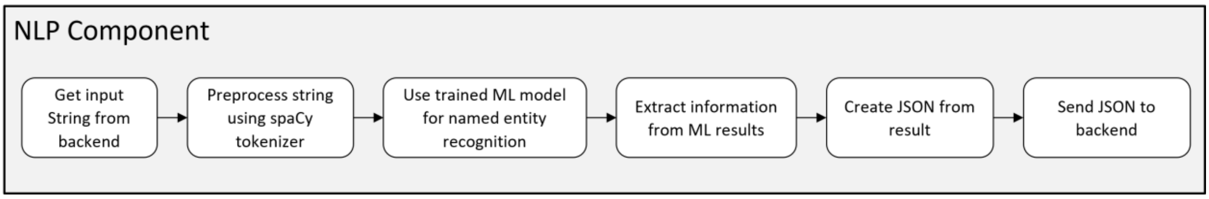
NLP Component
Get input String from backend
Preprocess string using spaCy tokenizer
Use trained ML model for named entity recognition
Extract information from ML results
Create JSON from result
Send JSON to backend

*Figure 4: Code Components of NLP Components.*

## Description

The frontend sends the new user input, a string, to the backend, the backend then forwards it to the NLP component.

The NLP component filters the relevant information, the search parameter, out of the user’s input, converts it into a JSON object and sends it back to the backend.

The backend converts the search parameter provided by the NLP component into the internal structure such that requests to external endpoints are easier to implement. The backend is also responsible to check the results it gets by external endpoints and whether they match with the user’s request. The result or results are sent to the frontend.

The frontend is then responsible to visualize the search results in a list and on a map.

# Technology Stack

## Frontend

### TypeScript

TypeScript is a strongly typed superset of JavaScript that compiles to plain JavaScript. It helps ensure that the code you write is always valid and works as expected before you deploy it to production.

### React

React will be used for the frontend. React is a JavaScript framework for developing single-page applications. React enables the building of reusable components and comes with features like one-way data binding, high performance, and a virtual DOM. There are many existing components that can be used due to the large developer community.

### Next.js

Next.js is an open-source React framework that helps you build fast, performant web apps. It supports both client-side and server-side-rendering, has TypeScript language support, provides a routing solution out-of-box, includes many optimizations on top of React and much more.

### Jotai

Jotai is a small state management library with great TypeScript support, which uses state atoms to share state across the components and provides support for derivd state with computed properties.

### Tailwind CSS

Tailwind is an extremely versatile utility-first low-level CSS framework. It's a powerful and flexible tool for building custom user interfaces with zero bloat.

## Backend

### Java

The latest Java version (probably 17) will be used as programming language.

## NLP Component

### Python

Python in the latest version is used as the base programming language, because of its rich technology stack that provides a large set of frameworks and libraries for Natural Language Processing and Machine Learning applications. In the following we present some of the main packages we will use in this component.

### virtualenv

virtualenv is a virtual environment builder that will keep all downloaded versions of Python packages project intern instead of setting the package versions global.

### spaCy

spaCy 3.1 is a library for Natural Language Processing built for production usage and provides pretrained models in different languages and sizes. It has a detailed documentation with examples and a preprocessing pipeline that saves all relevant information in the internal class structure. It is also fast as it uses C internally.

### FastAPI

FastAPI is a framework for building APIs for production usage. It has a documentation with step-by-step-guides and is standard-based. It also has a debugging interface that provides a basic documentation about the API, which can be extended.

## Deployment

### Docker

To make the deployment more flexible and scalable we use Docker to deploy our system in a consistent and isolated environment. Moreover, it enables the ability to run our system independently from the underlying operating system.