15-716-954

Date: 27.03.2022

Name: Michael Ziörjen

Keywords: Architecture, Concept, Branching Policy, Lexing Function, Deployment Automation, API Implementation

Overall Progress: 3

Activities completed:

- US1a: Setup of azure cloud function for the java language, incorporating the provided lexing library, creating an endpoint for lexing provided java code, deployment of function to azure, unit tests to test functionality
- US1b: Extension of cloud function to support language parameter for Java, Kotlin and Python, loading correct resolver class from lexing library based on parameter, unit tests to test functionality
- US2a: Writing unit test for incomplete code sample
- Automation of deployment of lexing function to azure using GitHub actions
- Unifying all repos in mono repo (instead of one distinct repo for each function / component of service)

Current activities:

- US3a: Research where and how to store the lexed data (most likely CosmosDB), finalizing decision, document and then setup / implementation for the next deliverable (to be done before 31.03.2022) [together with Sebastian Richner]

Goals or Activities for next deliverable:

- US3a: Implement the storage of the lexed data, split training and test data (required for US3b) in the process
- Research potential design for split of training / test data in storage
- Research options for proper API documentation for both public facing API (mainly provided by nest.js webserver) and internal APIs (provided by cloud functions e.g.

lexing / prediction). Potential solutions involve using Swagger or OpenAPI documentation

- Automate execution of tests before deploying the cloud functions to azure using GitHub actions
- Support for US3b / US9 in case other team members need help

Immediate attention: -

General remark: In the first week of the sprint I got sick with Covid-19 and was thus not able to participate in the project. However, I invested a substantial amount of time in the second part of the sprint to make up for the lost time. We decided fairly late in the sprint to merge all the codebases into a single repo since this would facilitate the automated deployment of the components and allowed us to streamline the merge / pull-request processes because only a single repo is affected.