Datenbanken und Web-Techniken
Project Task
Summer Semester 2020

1. Introduction

When using services on the Internet, there are sometimes temporary or permanent restrictions on the availability or utilizability of individual offers, which only depend on from where you are connected to the Internet. One possible solution to remove the restrictions is to use proxies. It looks to the requested service as if the proxy is performing the request, which then forwards the answer to yourself. When using proxies, however, the problem arises that you have to know its address and that proxies are also not permanently available. There are various lists of proxies on the Internet from which the addresses can be queried (for example Proxy-Listen.de).

The aim of this practical task is to develop a web application that aggregates various proxy lists and presents them in a common user interface, as well as testing the functionality of the individual HTTP(S) proxies. The exact task description follows on the next pages.

2. Task Description
2.1 Preliminary remarks

The following practical task may be processed in groups of up to two participants. If more than one participant is present, it should be indicated for all parts (programming, term paper), who was responsible for which part. For groups, the additional tasks mentioned at the end are obligatory (individuals may of course solve them, too).

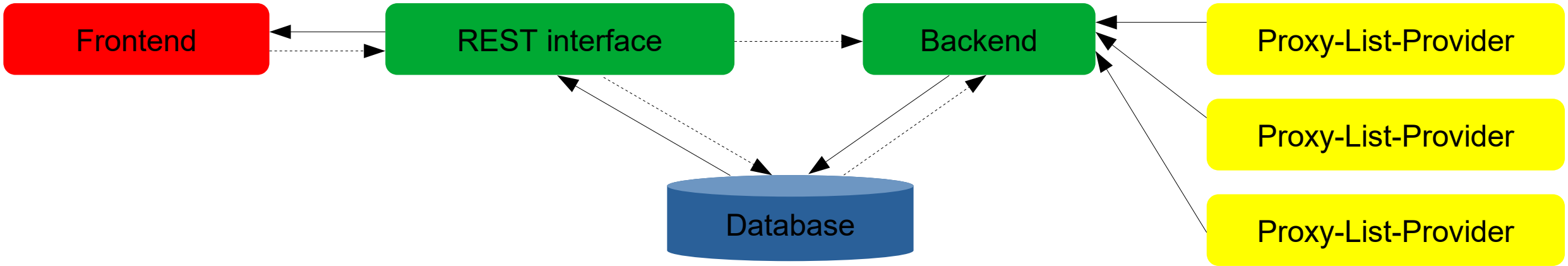
The submission consists of:

1. a PDF-file of the term paper in paper format A4 and
2. a separate ZIP-archive that contains the sources and a small manual on how to use them. The archive may not exceed a size of 10MiB.

2. Task Description
2.2 Programming

The program consists of

1. a backend for aggregating and processing the data of the proxy lists,
2. a database for storing the data of the proxies,
3. a REST based interface for providing the data from the database and
4. a frontend for displaying the data from the database.



2. Task Description
2.2.1 Backend

1. The backend aggregates and processes the data of the various proxy lists.
2. It prevents the updating of the proxy lists in a too small time interval. It waits at least 10 minutes between two requests to the same URL of a proxy list and there must not be more than one request per second to multiple URLs of one proxy list (for example for multi page lists).
3. It ensures a consistent storage of the data in a database, whereas duplicates of HTTP(S) proxy addresses are avoided.
4. It is able to test the functionality of HTTP(S) proxies in principle (basic functionality test).
5. It is able to test the functionality of HTTP(S) proxies in regard to a certain test URL.
6. HTTP(S) proxies are removed if they are not currently on a proxy list and the basic functionality test is currently negative and has not been positive within the past week.

2. Task Description
2.2.2 Database

1. The database stores the aggregated and processed data of the proxy lists.
2. It can also be used to store the configuration.
3. The frontend must never communicate directly with the database.

2. Task Description
2.2.3 REST based interface

1. The REST based interface provides communication between the frontend and the database.
2. It delivers the data from the database to the frontend.
3. It can also be used to store configuration settings in the database.
4. Furthermore, it is possible to use the REST based interface to initiate an update of the proxy lists as well as the functionality tests of the collected HTTP(S) proxy addresses by the backend. Therefore, it is not necessary to implement a proactive service which automatically performs the update.

2. Task Description
2.2.4 Frontend

1. The following detail information of all collected HTTP(S) proxies can be displayed:
   a) IP address
   b) port number
   c) date of the last successful basic functionality test
   d) date when the address was last found in any proxy list
   e) date when the address was first found in any proxy list
   f) for each available test URL the date of the last successful functionality test
2. The following information of all available test URLs can be displayed:
   a) test URL
   b) details of functionality test validation
3. Usability aspects are taken into account.
4. The design is appealing.

2. Task Description
2.2.4 Frontend

5. The following information can be displayed for each proxy list provider:
   a) base address (URL) of the proxy list
   b) details on extracting the HTTP(S) proxy addresses from the list
   c) date of the last successful update of the proxy list
   d) date of the last attempt to update with the number of records found or an indication of an error that occurred during the update attempt
6. At least three different working test URLs together with corresponding test validation information are available for demonstration purposes.
7. At least five different proxy list providers are available for demonstration purposes. At least 100 HTTP(S) proxy addresses have to be found by the system with each of this five providers. Duplicates between different lists count for each list.

2. Task Description
2.2.5 Additional task for groups

1. The basic functionality test (task 2.2.1.4) can also categorize the HTTP(S) proxies based on the anonymity achieved. This information can be displayed as a further detail for each proxy (task 2.2.4.1). The usual three levels are used:
   1. Level 1 (Elite): The proxy cannot be detected by the requested service.
   2. Level 2 (Anonymous): The proxy can be detected, but doesn't leak the original IP address.
   3. Level 3 (Transparent): The proxy can be detected and leaks the original IP address.
2. The time intervals for updating the proxy lists (task 2.2.1.2) and the age for deleting old HTTP(S) proxies (task 2.2.1.6) can be set via the front end.
3. In addition to the display option (task 2.2.4.5), complete proxy list management (creation, modification, deletion) is also carried out via the frontend, whereas at least the address (URL) and instructions for extracting the proxy addresses for each proxy list can be changed.
4. It is possible to filter the proxies according to suitable criteria. The result of the filtering can be exported as a new proxy list, whereby the IP address and the port number are provided for each HTTP(S) proxy.

2. Task Description
2.3 Term Paper

A term paper is to be written, that satisfies the following conditions:

1. Amount of about 10 pages (content!) for each project participant.
2. On the cover page there is the name, study course and matriculation number of all students recorded.
3. An overview of the utilized technologies is given and why they were used. They will be also classified in the context of the lecture.
4. The project is presented in such a way that, after reading, you know all parts of the program and their functionalities without having explicitly executed them.
5. A good form is kept and a balanced ratio of pictures and text is payed attention to.
6. All used sources, libraries and technologies are referenced.
7. In the Appendix (i.e. not part of the 10 pages) an API documentation is inserted, which contains the following:
   a) list of all endpoints of the REST based interface
   b) for every endpoint the list of the parameters and the return values each with type and meaning
   c) for complex structures also the inner structure is to be documented respectively
   d) documenting by only listing of examples is not sufficient!

3. Examination

The examination consists of a 10-minute presentation, which should meet the same criteria as the content of the term paper, but the API documentation is omitted here.

So the project and the used technologies should be presented and put into the lecture context. The presentation should also include a live demonstration of the project or a demonstration video demonstrating all parts of the practical task. For group work, the presentation time should be divided equally between both students.

After that some questions (primary concerning the project and the term paper) are given. Finally there will be a short consultation and you will be informed about your mark.

4. Dates

- Handout of task description:
  - starting 2020-05-25 00:00 MEST (UTC+2)
- Submission of project:
  - until 2020-06-28 23:59:59,9 MEST (UTC+2)
  - via OPAL:
    ➔https://bildungsportal.sachsen.de/opal/auth/RepositoryEntry/297435137/CourseNode/101474205454658
- Oral exam and presentation:
  - between 6. and 17. July 2020
  - modalities of appointment allocation will be announced after the project submission
    - appointment allocation most probably via OPAL, too
    - it will depend on the development in this Corona-thing whether the exam takes place via video conference or as a personal meeting in the meeting room