Datenbanken und Web-Techniken
Project Task
Summer Semester 2022

1. Introduction

In times of Web 2.0, many new digital artefacts are created every day and need to be shared with others. Simple web file sharing services may be the tools of choice to spread the content around the globe. Unfortunately, these one-click hosting services tend to be abused for sharing copyrighted material without consent of the copyright owner, so there need to be taken countermeasures to block the download of such files. As there are hundreds of file hosting services, a distributed blocklist might be useful to prevent a file taken down at one service to be uploaded again to another service.

Your task is to implement a prototype of a web file sharing service as a web application, which implements a distributed blocklist, that is available as a private web service. The focus will be to provide a simple way to upload and download files as well as the option to block (and possibly unblock) these files for download. The detailed task description is given on the following pages.

2. Task Description
Preliminary Remarks

There is only one practical project task for all students in this semester. This task may be processed in groups of up to two participants, but it is not allowed to share solutions with other groups. The detection of significant solution parts shared between submissions of different groups is considered an attempt to defraud and marked accordingly for all involved students (independent of who shared solutions or who used them).
If there is more than one student in a group, the additional tasks mentioned at the end are obligatory (individuals may of course solve them, too). Also, the work should be divided evenly between both members and the indication of the authorship is required for all parts.
The subtasks may be solved by the usage of any database management system, any programming language, any frameworks, any libraries and any web service API techniques, i.e. there are no restrictions in choice.
Supported languages are German and English.

2. Task Description
Submission

The submission consists of:

1. a PDF-file of the term paper in paper format A4 and
2. a separate ZIP-archive (with a maximum file size of 10 MiB) that contains
   - the program sources (i.e. the source code and additional files like pictures or other resources that are required by the program),
   - a script to initialize the database (i.e. the submission of the database itself is not required) and
   - a small manual on how to use the sources to get a working web application (i.e. the submission of an executable program is not required).

Only one submission is required per group (but if you still insist on uploading the same project twice to OPAL, the latest submission will count for your group).

The submission has to be in full (including all parts) on time, but late submission will still be accepted (see last slide for details). Updates after the submission deadline might be missed in case of previous on-time submissions, so please write an additional email to give a hint.

2. Task Description
2.1 Term Paper

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

1. There is an amount of about 12 pages of content (i.e. excluding cover, index, lists, appendix, bibliography, …) by using default values for font size, line spacing and so on.
2. A good form and a balanced ratio of pictures and text is kept (i.e. just including dozens of screenshots without referring to them in describing text is bad form and may also rarely be counted as content).
3. On the cover page, there is recorded the name, study course and matriculation number of all participating students.
4. An overview of all utilized technologies (i.e. DBMS, programming and other languages, frameworks, web service API implementation, ...) is given along with a short motivation, why they were chosen to solve a certain subtask. These technologies are also classified in the context of the lecture.

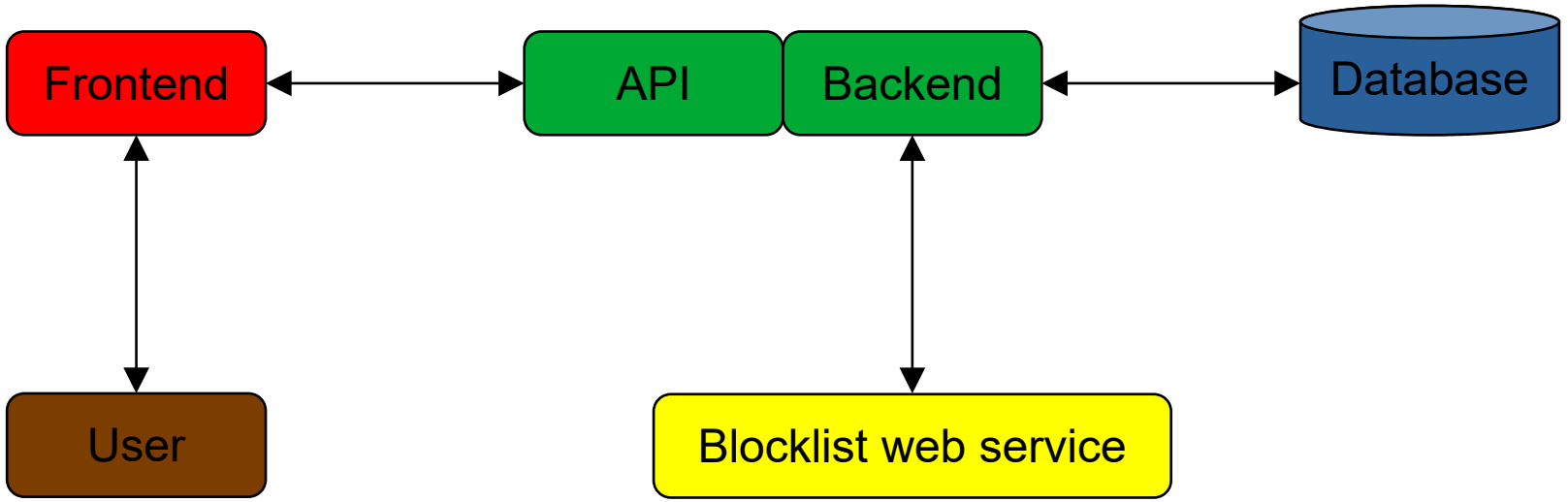
2. Task Description
2.1 Term Paper

5. 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.
6. All used sources, libraries and technologies are referenced. There are no restrictions on reference or citation style (but styles should not be mixed). There is no need for real scientific writing (i.e. citing scientific papers), but it is still forbidden to copy and paste something from Wikipedia without giving references.
7. The appendix includes a complete web service API documentation (i.e. just listing examples is not sufficient) containing:
   1. a list of all endpoints of the web service application programming interface,
   2. for every endpoint the list of the parameters and the return values each with type and meaning and
   3. for complex structures also the inner structure is to be documented respectively.

   The API documentation may be created using some tool, but it should fit with the other parts of the term paper in form and style (at least a bit, so there might be the need for reformatting things).

2. Task Description
2.2 Programming

The program consists of
1.a database for storing the data,
2.a backend for processing the data and interaction with the Blocklist web service,
3.a web service application programming interface for providing the data and
4.a frontend for displaying the data and user interaction.



2. Task Description

2.2.1 Database

1. All data is stored in the database, i.e. no additional storage (like files or web storage) is used, with the following exceptions:
   1. Program configuration like database connection data or university user account data may be stored in configuration file(s).
   2. Additional temporary storage is allowed within the workflow, but need to be cleaned up after user interaction is finished.
2. The frontend must never communicate directly with the database.

Disclaimer:
Storing binary data like random files in a database is considered bad practice. So please be aware that you should not do something like this in a real world scenario. This is just the usual academic task that lets you try something one time and meanwhile you hopefully learn, why this is a bad idea.

2. Task Description
2.2.2 Backend

1. The backend processes the data.
2. It handles all communication with the database.
3. It provides the web service API to be accessed by the frontend.
4. It handles the communication with the Blocklist web service, which is available at https://www.tu-chemnitz.de/informatik/DVS/blocklist/
    1. The web service is used as intended (i.e. only SHA-256 file hashes are provided).
    2. Possible web service errors are handled reasonably.
    3. Only files, whose hashes are not reported to be blocked by Blocklist web service, may be offered for download.
    4. The blocking status of files may be cached up to 12 hours.

Attention:
The Blocklist web service is secured by university login, so the login process to the Web Trust Center needs to be implemented in the application to communicate with it.

2. Task Description

2.2.3 Web Service Application Programming Interface

1. The web service application programming interface provides communication between the frontend and the backend as well as the database.
2. It delivers the data from the database to the frontend.
3. It provides the user input from the frontend for processing in the backend.
4. API access failures are caught and responded with meaningful error messages (and possibly corresponding HTTP status codes).

2. Task Description
2.2.4 Frontend

1. The frontend is a web application to be accessed by a web browser.
   1. All common web browsers are supported, i.e. Mozilla Firefox, Apple Safari and Chromium derivates.
   2. It is a responsive web application, i.e. different device and orientation types are supported.
2. Usability aspects are taken into account, i.e. a nice user interface is created (UI), which leads to a good user experience (UX), so for example, if a user has to click somewhere, scroll right and down, and finally click another position to solve a task that could be achieved by just one click, then it could be considered bad usability.
3. The design is appealing.
4. Users have the option to upload files in a fast and easy way.
   - see following slides for details
5. Users can download files by accessing a download URL.
   - see following slides for details
6. A special admin interface allows administrators to decide about administration requests.
   - see following slides for details

2. Task Description

2.2.4.4 Upload

1. All types of files may be uploaded.
2. The maximum file size is 10 MiB.
3. After successfully uploading a file, users are provided a download URL to download this file.
4. Each file has its own unique download URL, i.e. a download URL must not give access to more than one file.

2. Task Description
2.2.4.5 Download

1. There is no search option for files, i.e. the knowledge of the download URL is required for download.
2. Invalid download URLs (which currently do not point to a file) are handled reasonably.
3. The download page offers meta information about the file (at least file name, file size and upload date and time) and
    1. an option to download the actual file, if it is not blocked (see task 2.2.2.4), along with an option to request this file for blocking or
    2. an information, that the file is blocked, along with an option to request this file for unblocking.
4. The requests for blocking or unblocking a file require the user to provide some reason, why this file should be blocked or unblocked.
5. It is not possible to hot-link a file, i.e. starting a file download just by knowing a certain URL must not be allowed.
6. Inactive files (i.e. files which are not downloaded for 14 days) are removed by the system.

2. Task Description

2.2.4.6 Admin Interface

1. There is an overview of current pending requests along with the option to select one of these requests for processing.
2. Upon request selection the admin is offered sufficient information to decide about the request.
3. A request may either be accepted or declined and is removed from the overview about current pending requests after the decision is made.
4. If the request is accepted, the required action has to be performed using Blocklist web service (see task 2.2.2.4) and the request must not be finished until the Blocklist web service responded with the expected return value.

2. Task Description

2.2.5 Additional Tasks for Groups

1. There is the option to upload multiple files at once (but each file still gets its own download URL).
2. There is the option for users to register.
    1. No plaintext passwords are stored in the database.
3. Registered users, which are logged in to the system, may manage their file uploads, with the options
    1. to get an overview of all uploaded files including all relevant details (file name, file size, upload date and time, download URL and blocking status) and
    2. to remove their files from the system.
4. Unregistered users may still download files, but there are the following restrictions:
    1. The download speed is limited to 100 KiB per second.
    2. Only one download may be started within 10 minutes per IP address.
        1. Upon requesting another download, the remaining time, until this download can be performed, is shown.
5. The admin interface is secured by a login, too.

3. Examination

The examination consists of a 15-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.

Afterwards, some questions are given, which primary focus on the project and the term paper, but may also cover the lecture and exercise.

Finally, there will be a short consultation and you will be informed about your mark.

4. Dates

- Handout of task description:
  - starting 2022-06-02 16:00 CEST (UTC+2)
- Submission of project:
  - until 2022-07-07 16:00 CEST (UTC+2)
    - late submissions will be accepted, but your final mark will be reduced by 1/3 per 8 hours or parts thereof
      - your mark will be reduced by 1 per day, so you cannot pass, if you submit after 2022-07-10 16:00 CEST
  - via OPAL:
    - https://bildungsportal.sachsen.de/opal/auth/RepositoryEntry/297435137/CourseNode/16540506303827720004
- Oral exam and presentation:
  - between 2022-07-12 and 2022-07-22
  - appointment allocation will be done via OPAL, too
  - examination will take place via video conference in same room like lecture and exercise as default
    - we will try to offer personal meetings in a meeting room in the university as alternative if required