GraffLib Technical Specification

Mažvila Mantas, Namajūnas Joris, Švedas Gintautas,
Voroncov Ivan, and Supervisor: dr. Linas Bukauskas

Vilnius University Faculty of Mathematics and Informatics

October, 2021

Contents

1 Introduction

1.1 Overview

This is the technical specification for the project GraffLib. GraffLib is a piece of software for sharing graffiti and seeing its changes.

1.2 Purpose

This is the document describing the technical specifications of GraffLib. The purpose of this software is to allow users to upload, analyze and compare images of graffiti.

2 Overview of the whole system

2.1 Integration with external systems

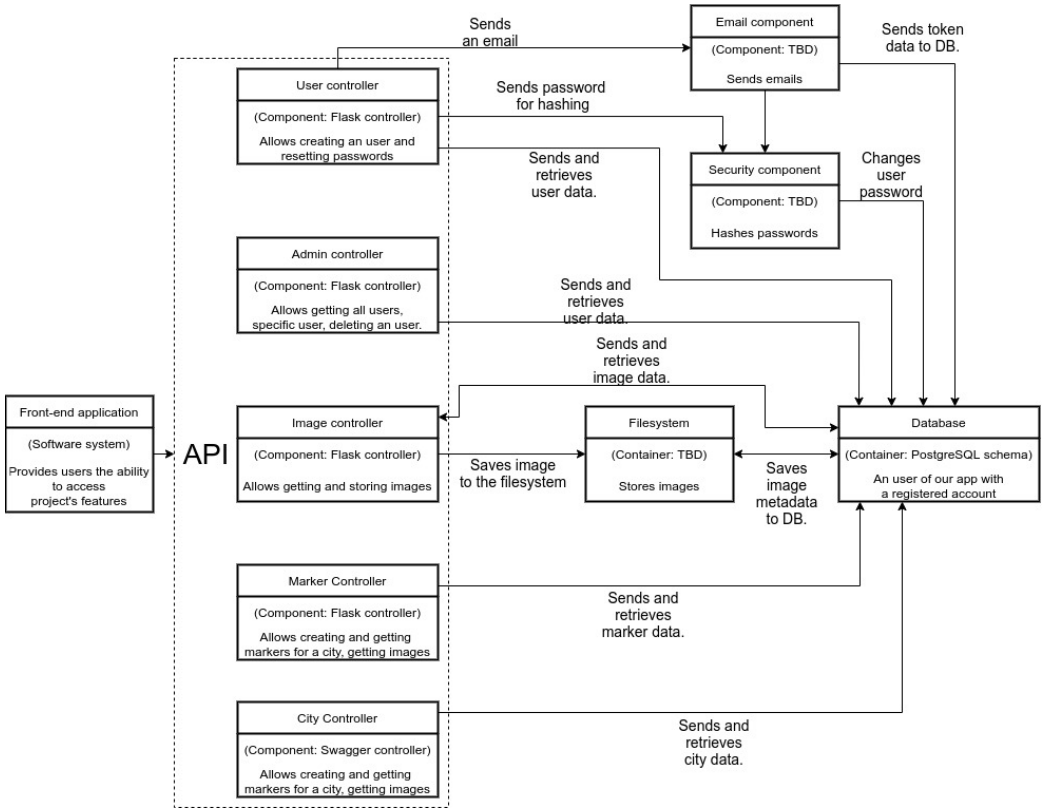

Figure 1: Partial context diagram

Our system can be accessed through an API, that will be hosted on a web server. The requests will be routed to the back-end where the magic happens. In figure 1, we see our context diagram, where the communication between our features is explained. As of writing this specification, the image uploading (Figure 2) is very basic (feature-wise), and it will stay simple for now. The images get stored in our filesystem, and the metadata gets sent to the database. Eventually we might add forced conversion to .jpg files or compression to save space. The users controller goes through a security component where passwords get hashed using bcrypt, otherwise it stores most of the data. The other controllers simply send and receive data to and from the database.

2.2 System artifacts

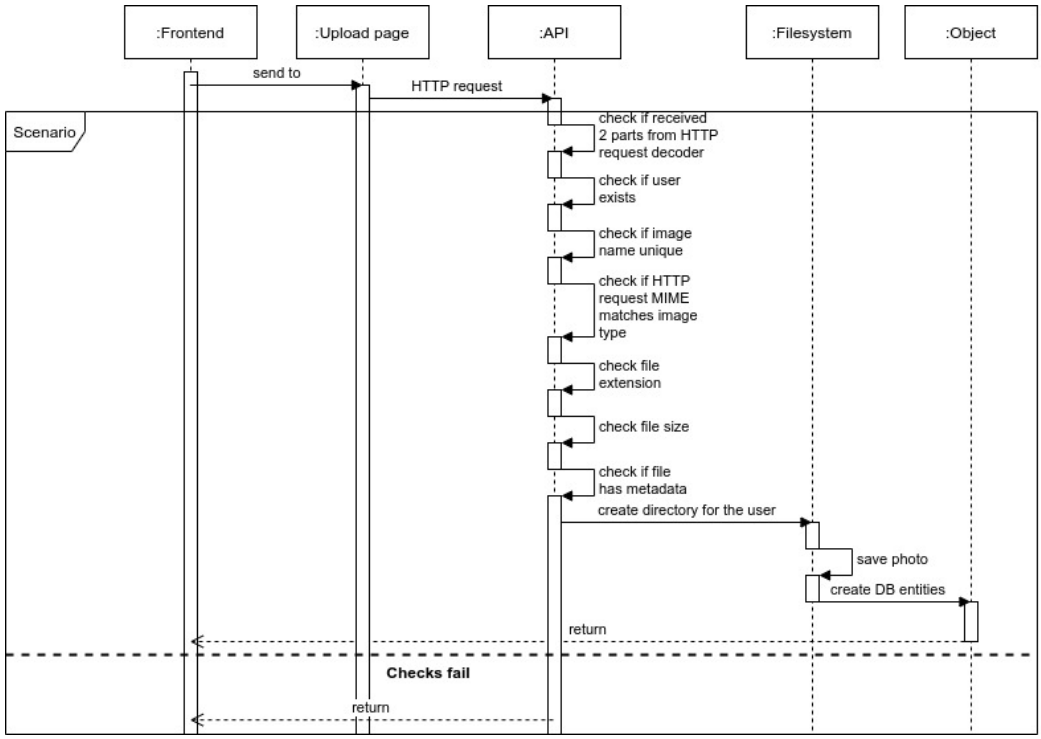

Figure 2: Image upload sequence diagram

The process of uploading an image is pictured in Figure 2. In the first scenario, it assumes that a picture gets through all the checks. The checks limit the size of the file, verify that the file is an image, has metadata and that it's sent by an user. If the checks fail, it returns to the frontend, where it will show an error message.

3 High-level overview of the system internals

3.1 Structural aspects

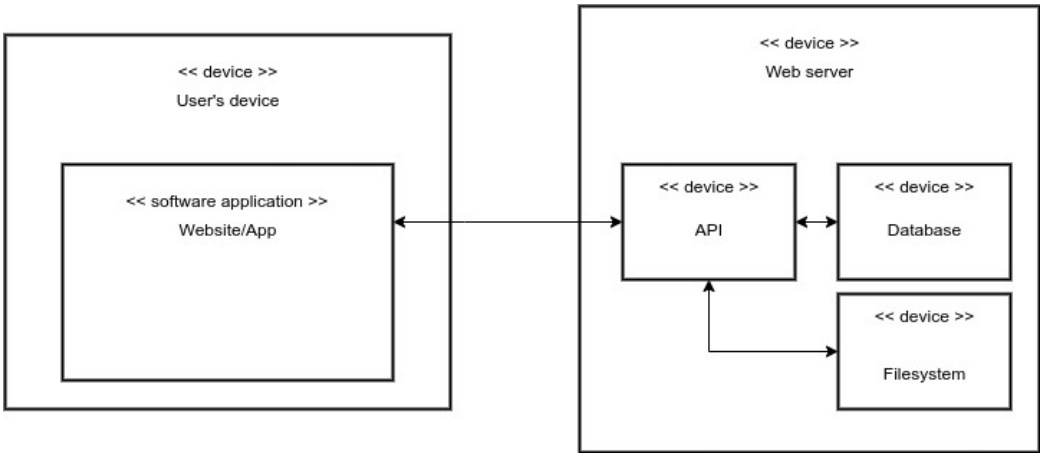

Figure 3: Structural aspects

We may very well fit the API, the Database and the Filesystem into a single virtual machine, running on Vilnius University Institute of Mathematics and Informatics infrastructure.

3.2 Dynamic aspects

4 Tools and technologies

4.1 Back-end

The following are software tools being used for the GraffLib project.

- Flask
- Python3
- Swagger

We're using the micro web framework Flask for development and HTTP request handling.
Swagger is our interface description language. We're using it for our REST API.

4.2 Storage

These are the systems in place for storage of the GraffLib project.

- PostgreSQL
- A filesystem

The database will store user accounts, image metadata, and other content. We chose PostgreSQL as our RDBMS because it allows for easy interaction with GIS.
We will also have a filesystem for image uploads. We feel like it will make the job of image storage significantly easier with no significant drawbacks.