Software design

As mentioned under the specifications, our first goal is a C++ library which can generate fractals. More precisely we want to generate density maps, stored in arrays whose dimensions match the specified resolution. The values in the density map can later be interpreted to mean different colors for example, but this will not be the responsibility of this library. We will provide the ability to save the density map in binary format to a file. This will be accompanied by a command line application. This will have the purpose of providing an easy way to generate these density maps and to output the result into a file. We also want to provide a brief user guide which will appear on the screen when the help keyword is used. For the python wrapper we will be using pybind11, which is an excellent tool for wrapping C++ code. The main reason is that it works nicely with cmake, which we will be using anyway and so by writing a short wrapping code in C++ we will have the ability to simply install wrapper as a package using PIP.

Finally the web application will use the aforementioned Python package. It will take the resulting density map and postprocess it so we will get a nice colored image that can be displayed on the webpage. Fractal generation will run on the server while state management and zooming will be handled locally in javascript. The schematic workings of our web application is summarized in Figure 1.

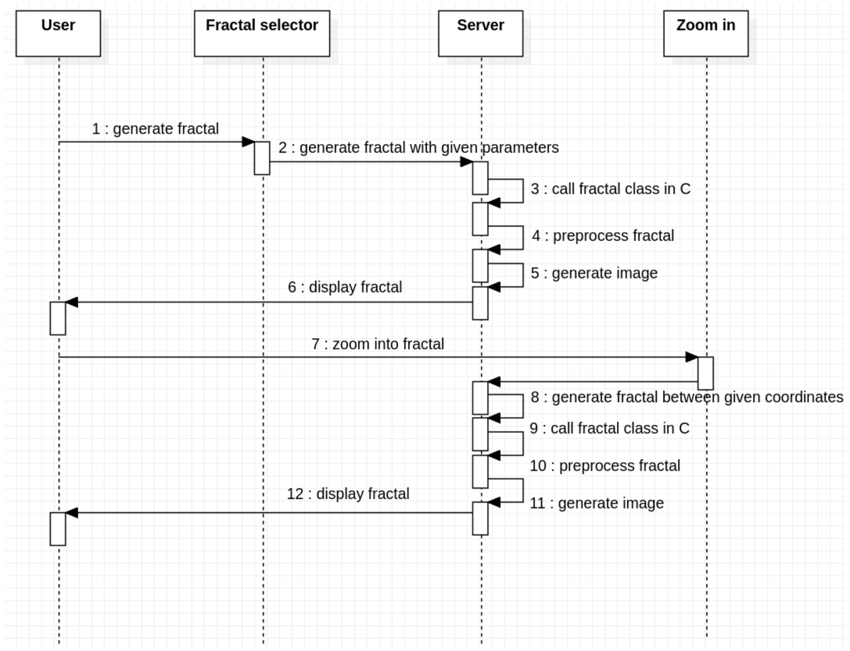

Figure 1.: Sequence diagram of the project. Detailed explanation of the implemented functionalities which are executed in case of selecting a fractal or zooming into mandelbrot. The numbers indicate the order of execution of functionalities. The first (numbers from 1 to 6) and second (numbers from 7 to 12) cycles can be executed as many times as required by the user.

The server side will be based on the Flask toolkit. Flask is a micro-web framework written in Python which enables us to easily establish the server application. It provides us easy-to maintain functionalities to communicate with HTML through POST requests [1]. The server, written in Python, will contain three main functionalities ("home", "updateFractal" and "zoomInFractal") which handle the creation of appropriate HTML templates to display for the user. The functions rely on the data obtained from the POST request and on the “generateFern”, “generateMandelbrot” and “generateEncodedFractalImage” functions. These functions generate the desired fractal images with the given parameters and encode them in “base64” format [2]. The result is communicated with Javascript in Json format.

Javascript handles the actions taken place by the user on the webpage. It will contain the on click event for the “generate” button and image holder of the fractal, a class for the mandelbrot fractal and a functionality for the fancy moving background. All functionalities will be based on jQuery, a feature-rich Javascript library [3]. Data with Python is communicated asynchronously using Ajax [4]. In the zooming part the mandelbrot class variables are modified through its methods.

Resources:

[1] https://flask.palletsprojects.com/en/1.1.x/quickstart/
[2] https://en.wikipedia.org/wiki/Base64
[3] https://jquery.com/
[4] https://en.wikipedia.org/wiki/Ajax_(programming)