Versatile spatial raster dataset web viewer and analysis module

Purpose of the internship

We developed a desktop application for Mac and Windows called "GeoExplorer". This application is able to read GeoTIFF (spatial raster data image) files and display those in a zoomable and pannable interface. The user can query specific data points or draw a custom polygon to get a set of statistics. By combining several of these images it is possible to generate a Fourier Transform time series analysis. We would bundle these images onto a DVD and physically ship these to our partners.

Now we want to bring this same experience to the web. For projects we do and partners we work with we want to provide these spatial datasets in a web viewer with the same basic query and analysis capabilities, without the need to ship out any physical media.

We need a new web module that can be used in different web applications to view spatial outputs such as raster layers we provide from file or a database in the cloud, with the ability to perform basic queries such as looking up pixel values, simple statistics on a custom region, or a timeseries analysis over multiple custom points. In addition, the user may be able to add a layer by uploading their own raster datafile. The user can view multiple different layers in an interactive viewer that can pan and zoom. This will require the spatial data to be in a tiled format. The processing tool to prepare the data is part of this project. In addition to including this web GIS viewer module in multiple website we want it to be capable of being expanded with more advanced GIS functions in future.

Your responsibilities

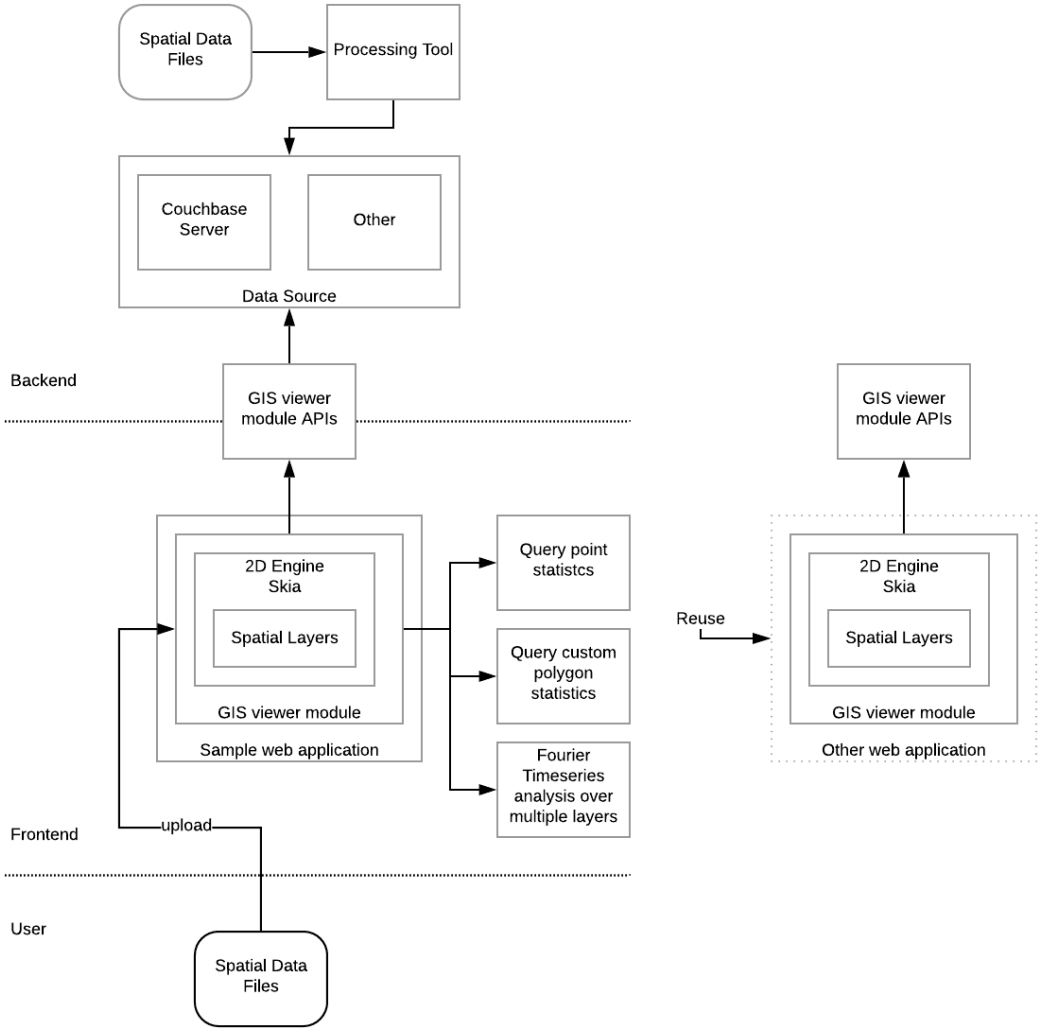
You will be responsible for the architecture, design, and implementation of the GIS viewer module and basic query functionality, and integrate this into a sample spatial data viewer website. This will include a front-end for the sample web application, a front-end for the GIS viewer module, and a back-end for the communication between the front-end and the data source in the back-end. You will write any processing tool required to pre-process/prepare spatial data into a format that can be used most optimally by the viewer web module. All code written should be well structured and organized, and allow for re-use in different websites and with the possibility of easy future expansion. All interfaces presented to the user should be well designed from a usability as well as a design standpoint.

You can propose the rendering engine to render and display the spatial data, especially the raster data. The default option will be to use Skia.

- The web module front-end and back-end, and the sample web application will be written in C# .NET using Blazor to be hosted from a Linux ASP.NET Core server
- The web module back-end should be able to retrieve data from files or a database in the cloud, with default minimum option a Couchbase Server NoSQL

database – the data source should be abstracted so different sources can be used

- All processing tools separate from the web module and sample web application are written in C# for .NET Core 3 and should run at least on Linux
- The web module front-end and back-end can be re-used by any number of websites to provide a GIS spatial data viewer functionality
- Data sources can contain raster layers of up to 1.5GB in size. The web module can also accept raster data files for upload with a maximum size still to be determined.
- All code is clean and readable, well-documented, and can be easily maintained and expanded



Expected result

- A sample web application that contains the GIS viewer module with sample spatial data

- The GIS viewer module front-end and back-end is easily re-usable for other projects
- The data source in the viewer module back-end is abstracted so spatial data can be loaded from a filesystem or from a database, with as default implementation a Couchbase Server database. The user can select from different layers in a set of data that is provided by the implemented data source, or upload their own, and these layers are then visualized in the viewer, or used with timeseries analysis.
- The user interface is friendly, modern, and well designed for usability and look-and-feel
- The software performs fast and fault-free
- All implementation software is well-documented. The architecture and design is documented, the source code is commented, the integration of the components and how they can be re-used for new web applications is documented as well as how it can be expanded with more features, the end-user interface and user functions are documented in a user guide