BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2021
http://excel.fit.vutbr.cz

Information System for Management of Geographical Data Visualizations

Jan Grossmann*

Abstract
The goal of this project is to create an information system for the visualization of geographical data. The main idea is to allow users to visualize their own geographical data, which they can import either from plain text files or directly attach their own database system as a source of data and make use of the data in real-time. By providing the necessary information like the type of database management system and credentials, the back-end of the information system will be able to connect, read or build up the schema of the database including all of the stored tables, objects, and its dimensions. Then, the user will choose the proper form of data through the graphical user interface with zero need for programming skill. These data will be used to create the visualizations, using an existing application for visualization of geographical data named Geovisto. Created visualizations could be also shared easily, by providing a specific link that will show the interactive result, allowing the user to embed the view into his personal webpage. This paper presents a new web information system that will act as a point of contact between users, geographical data, and visualizations.

Keywords: Information system — Data exploration — Geographical visualization

Supplementary Material: Development preview version

*xgross09@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

It is no longer a problem to automatically record and store large amounts of data. This data can be stored either in the form of text files in a readable form, but more often the data is stored in large database systems, for example in the form of tables. However, this type of data representation brings its drawback of understanding the stored information again. For example, when talking about geographical data, everyone can understand simple table of daily maximum temperatures in Africa, but using the exact coordinates of a particular place in the Sahara, most people will be clueless.

The way you can greatly increase the ability to understand the data, is to interpret the data visually, for example in some form of thematic map. In this case, the data is applied directly to the map layer linked with certain place and showing values or colors connected to graphical entities. This is one of the things the new information system will help the users with. But still, the missing piece is a connection between the database systems and tools for visualizations. The part that would allow everyone to link own sources of data to tools for visualizations, without the need to export the data into a file of predefined formats on the one side and import them on the other side.

This is the problem here, that user has access to the data in his own database, the user can have right tools to create visualizations, but the user has no platform to successfully combine these two things to achieve expected results. This is because it takes a certain amount of IT knowledge to dump different types of databases and store them in a format that would existing tools for visualizations understand. Another problem comes right after the data visualization is completed, when the user lacks a space to collaborate on his work or share the final result. So the main goal of this information system is to provide an easy-to-use way of creating interactive visualizations of geographic data, stored de facto anywhere (could be stored locally or in the cloud), accessible from all sorts of devices (such as desktops, smartphones, smart TV's, etc.) and able to share the results with the current state of interactivity, not losing any of it by transforming the visualization into a static image. All of this in just a few clicks.

2. Analysis of Requirements

2.1 Use case

The typical use case of this information system would be the moment, when the user comes to visualize his geographical data. The data can be imported from plain text files, or by directly attaching users database management system as a source of data. When the database system is chosen as the source, the user selects required dimensions in the user interface and the output is then used for visualization purposes. Then, there would be also a way to share the results, as described later.

Because of the Geovisto [1] application dependency, which is visualization module based on Leaflet[1] (library for creating interactive maps), this system could be also used to create configuration files in a more friendly way. This configuration file could be exported to skip the step of creating the settings when using the standalone Geovisto application directly.

2.2 Sources of data

We need to understand, what could be the possible storage of data used for visualization. Besides plain text files like JSON, CSV or XML that could be used to store data, it is often much more effective to use database systems. These database system could be divided into multiple types, e.g. by relation status.

Relational databases store the data with columns, rows, and tables [2], kind of following the way CSV files are used to store data. On the other hand, non-relational databases use a storage model that is optimized for the specific requirements of the type of data being stored. For example, data may be stored as simple key/value pairs, as JSON documents, or as a graph consisting of edges and vertices [3]. This structure could be taken as an equivalent to the structure of JSON or XML files.

Another type of database system suitable for data visualization, could be time series database. This is a database optimized for time-stamped or time series data. Time series data are measurements that are tracked over time [4]. With the use of time-stamped data and support from the side of visualization tools, users can create impressive animated views over time.

2.3 Publishing results

We also need to understand how the data is typically processed and what happens with created visualisations. The simplest use case is to create an image of the final result, download it to local storage and share the image afterwards, e.g. via email or any messaging application or file sharing service. However, visualisations created this way has none of its interactivity and the quality could easily degrade over time if they are not shared properly.

Another way, if the application allows it, is to invite your colleagues that might be interested in your work to cooperate on the visualization. This usually requires one annoying step to do, and that is the need for registration.

The last way used to share the result and the solution to bypass the registration is, to make the display of created visualization available under specific link. Then, the link can be shared and viewed anonymously, with all of the benefits that certain application or information system provides. Views shared like this can be also easily embedded to any web page in the form of *iframe*[2].

3. Existing solutions

There is already a number of tools in this area that focus on some of the goals of this information system. A large number of APIs (Application Programming Interface) are already used for the needs of simple data visualization and presentation of results on your own website. When it comes to more complex solutions with data management, we need to look at geographic information systems, also known as GIS.

3.1 Application Programming Interface

The application programming interface (API) of Mapy.cz - Mapy API[3] or Google Maps API[4] will definitely belong among the most famous in the Czech Republic. These APIs come with one essential downside. Their user needs at least basic knowledge of programming, usually in HTML and Javascript language, to work with this kind of application interfaces to be able to create impressive results. The API is usually introduced to the web page via a loader and then, the Javascript functions are called.

3.2 Authoring systems

In addition to APIs, these systems usually come in the form of a standalone desktop application or cloud ap-

[1] https://leafletjs.com/

[2] https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe

[3] https://api.mapy.cz/

[4] https://developers.google.com/maps/documentation/javascript

plication, so they give the user the ability to import and store data on the server side. These complex systems are often an enterprise solutions for large companies. That is why many of them require to buy license first or pay monthly fees. Some of the representatives are:

1. **Elastic maps**[5] – application for exploration of location data. Lets you analyze geospatial data. Visualize multiple indices as unique layers in one view, but is strongly bound to Elasticsearch data and platform [5].
2. **ScribbleMaps**[6] – application that lets you create embeds, images, and map data. You can integrate all the features and power of Scribble Maps directly into your web-based application while they store your map data and allow you to quickly convert from one format to another [6].
3. **HERE Studio**[7] – its editor allows you to create interactive custom maps without coding or any back-end infrastructure. It supports large datasets in different formats. You can add colors, markers and other design elements for style and publish your map in minutes [7].
4. **Tableau**[8] – application that has in my opinion very easy-to-use and intuitive UI. Tableau automatically turns the location data and information you provide into interactive maps. Census-based population, income, and other standard demographic datasets are already built in [8]. See the example in Figure 1.
5. **Clever Maps**[9] – needs basics of IT understanding to configure the data sets. It geo-locates the data and sorts it into a logical multidimensional spatial data model. Then, the front-end allows you to directly perform advanced analytics on an interactive map [9].

All of them are paid but some of them offers at least basic plan that lacks some features, but you can try the application for free.

4. Design & Implementation

The whole system is divided into 3 parts as a standard of 3-tier architecture as seen in the bottom section of Figure 2. The main language of choice is JavaScript for both front-end and back-end, executed in the Node.js run-time environment and the MongoDB is used as the database management system.

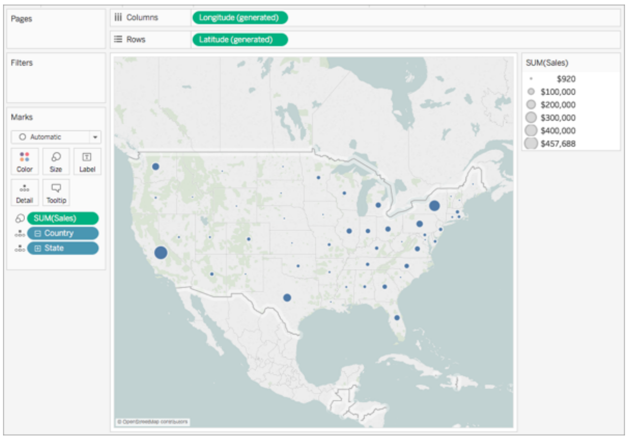

Figure 1. Tableau user interface showing data across USA.

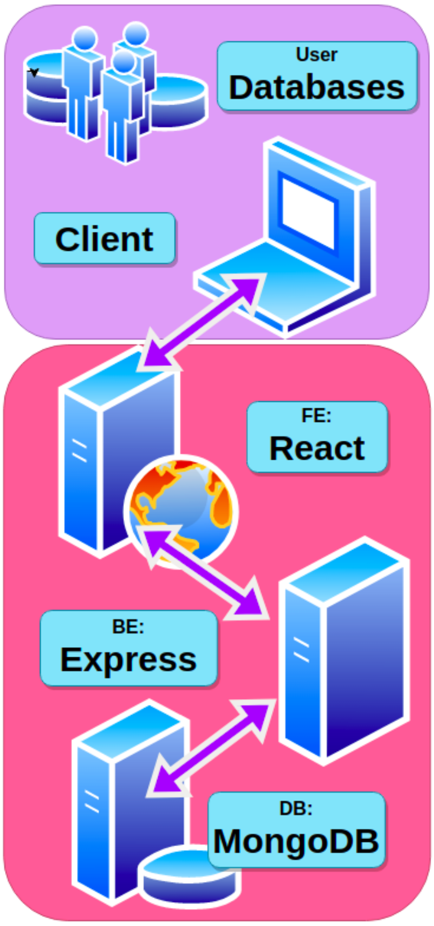

Figure 2. Basic diagram of complete system architecture. Client databases in relation with client tier (FE, front-end), server tier (BE, back-end) and database tier (DB).

4.1 Client tier

Because the original Geovisto application, which takes care of graphical visualization, is already being implemented as a **React**[10] component, the whole client tier is also written in React. The main idea of React is that every element could be developed and understood as

[5] https://www.elastic.co/maps
[6] https://www.scribblemaps.com
[7] https://www.here.com/platform/studio
[8] https://www.tableau.com/solutions/maps
[9] https://www.clevermaps.io/
[10] https://reactjs.org/

a component, which makes the system highly extensible. The face of the site is delivered by utility-first CSS framework **tailwindcss**[11], that takes advantage of the newest CSS features like the grid positioning or conditional styling for the responsive design. Responsive design is no longer a trend – it is a necessity. In 2020, the number of unique mobile internet users stood at 4.28 billion, indicating that over 90 percent of the global internet population use a mobile device to go online and mobile internet traffic accounts for more than 55 percent of total web traffic [10]. Unlike other popular CSS frameworks, there are no off-the-shelf components such as buttons and navigation bars, it is your turn to use Tailwind to build your components.

One of the advantages is the ability to create thumbnails from visualizations, that are then shown in the main roster of configurations. Another neat feature is the landing page as shown on the Figure 3, where the user is allowed to meet the visualization part for the first time, without any need for time-consuming registration. Figure 4 shows multiple previews of UI as seen on mobile devices. Another preview of the UI on Figure 5 shows the creation process of new data set.

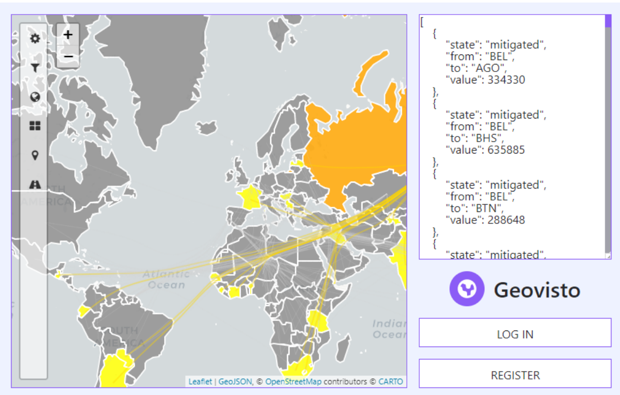

Figure 3. UI preview - Landing page with the preconfigured visualization (in original Geovisto application frame) and sample raw data on the right side.

4.2 Server tier

The server side uses the fast and minimalistic web framework for Node.js called **Express**[12]. With many HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy. The server has predefined endpoints for each entity in the system, like User, Dataset, Config etc., and communicates with the client via REST API. This type of communication expects standardized GET/POST/PUT/DELETE HTTP requests [11]. The authentication of client is solved by JSON Web Tokens.

This part of information system uses various third party database systems connectors in the form of npm packages. There are plenty of them, like porsager/postgres for PostgreSQL, oracle/node-oracledb for Oracle Database, mysqljs/mysql for MySQL and mariadb-corporation/mariadb-connector-nodejs for MariaDB for relational DBs, or NoSQL DBs connectors Automattic/mongoose, neo4j/neo4j-javascript-driver and node-influx/node-influx. These connectors handles the connection and data acquirement from foreign sources of data.

4.3 Database tier

MongoDB was chosen as the main database of the system. It is a document database, which means it stores data in JSON-like documents. As a bonus, it has its unique query language. This behaviour is highly suitable in the context of all the data stored and transferred inside the information system. It also suits the support for JSON files of Geovisto application.

To extract data from user-defined databases, server tier tries to connect with specified credentials to corresponding URI. The connector object provided for each remote dataset looks as follows:

```
connector: {
   "type": "mysql",
   "uri": "127.0.0.1",
   "port": 3306,
   "db": "geovisto",
   "username": "",
   "password": "",
   "table": "traffic",
   "query": "ip_src, ip_dst, size"
}
```

Then, the back-end tries to build the blueprint of the DB structure, depending on the type of database system. The first step is to read the names of all the tables or collections. To get the idea of its content, the system asks for table schema or queries random sample of elements, that are merged into one super-element with all the possible combinations of attributes. When the blueprint is built, the user can choose a selection of needed dimensions by clicking a form in the UI. This selection is then converted into a database query.

4.4 Demo version

The demo version had been uploaded to Heroku Application Platform[13]. Heroku lets you deploy, run and manage applications written in a large number of programming languages, where one of them is Javascript,

[11] https://tailwindcss.com/
[12] https://expressjs.com/
[13] https://www.heroku.com

and also introduces Heroku-specific build steps that are executed automatically when the deployment is in progress [12].

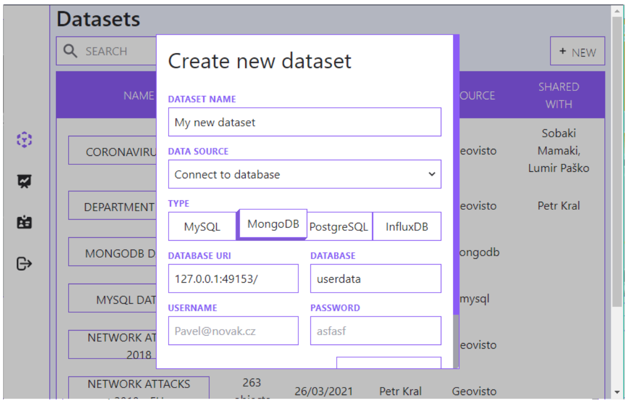

Figure 5. UI preview - Creation of new dataset by connecting to MongoDB.

5. Evaluation

In the testing environment, I was already able to recreate the basic use case of Geovisto application [1] while using databases as a source of data, instead of importing the data in form of JSON files. Then, I was able to share the result to a different account that could view and edit the current state of work. After experimenting with the colors for a while and with the help of tailwindcss, I was also able to implement the dark theme.

When the system was deployed to Heroku, some new bugs came up when interacting with it, but at least the presentation layer looks pretty stable.

6. Conclusions

This paper provides information about Geovisto information system. The information system that allows the users to easily create shareable interactive visualizations, using the already developed Geovisto application. Solves the crucial things such as import of data from various sources through responsive mobile and user-friendly interface, cross-platform cooperation between users, sharing of results, or even continuous availability of the tool and the results afterward.

The primary challenge was not to provide another information system, that just wraps a tool where you can add data and graphical entities to map. Instead, the main goal was to create a platform for cooperation of users, demonstration and storage of their visualized results that could be used across the world wide web, synchronization and backup of work, and of course real-time use of various database sources of data. The contribution of this work could be understood as, that you can have excellent tools and sources of data, but you will not be able to do much if you can not combine them together.

At the current stage, the system offers at least basic management of users, data sets and configurations – which can be understood as visualizations. The data sets can be uploaded in the form of JSON or pasted directly as plain text, or currently as proof of concept, you can enter credentials for MySQL, MongoDB, and PostgreSQL database systems to fetch the data set directly. There is still a lot of work to be done. In the future, I would like to improve the front-end part of the information system to solve all of the edge-cases that could happen when the system is in use, deliver support for other mentioned database systems, implement user management of geo-layers (that should be soon implemented in the Geovisto application) or permanent links for embedding and sharing.

Acknowledgements

I would like to express my gratitude to my supervisor Ing. Jiří Hynek for his help, advice, useful comments, and engagement during this project.

References

[1] J. Hynek, J. Kachlík, and V. Rusňák. Geovisto: A toolkit for generic geospatial data visualization. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Volume 3: IVAPP, pages 101–111, 2021.

[2] Oracle. What a relational database is. online. Available at https://www.oracle.com/database/what-is-a-relational-database/.

[3] Z. Tejada. Non-relational data and nosql. online. Available at https://docs.microsoft.com/cs-cz/azure/architecture/data-guide/big-data/non-relational-data/.

[4] InfluxData. Time series database (tsdb) explained. online. Available at https://www.influxdata.com/time-series-database/.

[5] Elasticsearch B.V. Putting geo on the map. online. Available at https://www.elastic.co/maps.

[6] 52 Stairs Studio inc. Draw on maps and make them easily. online. Available at https://www.scribblemaps.com/.

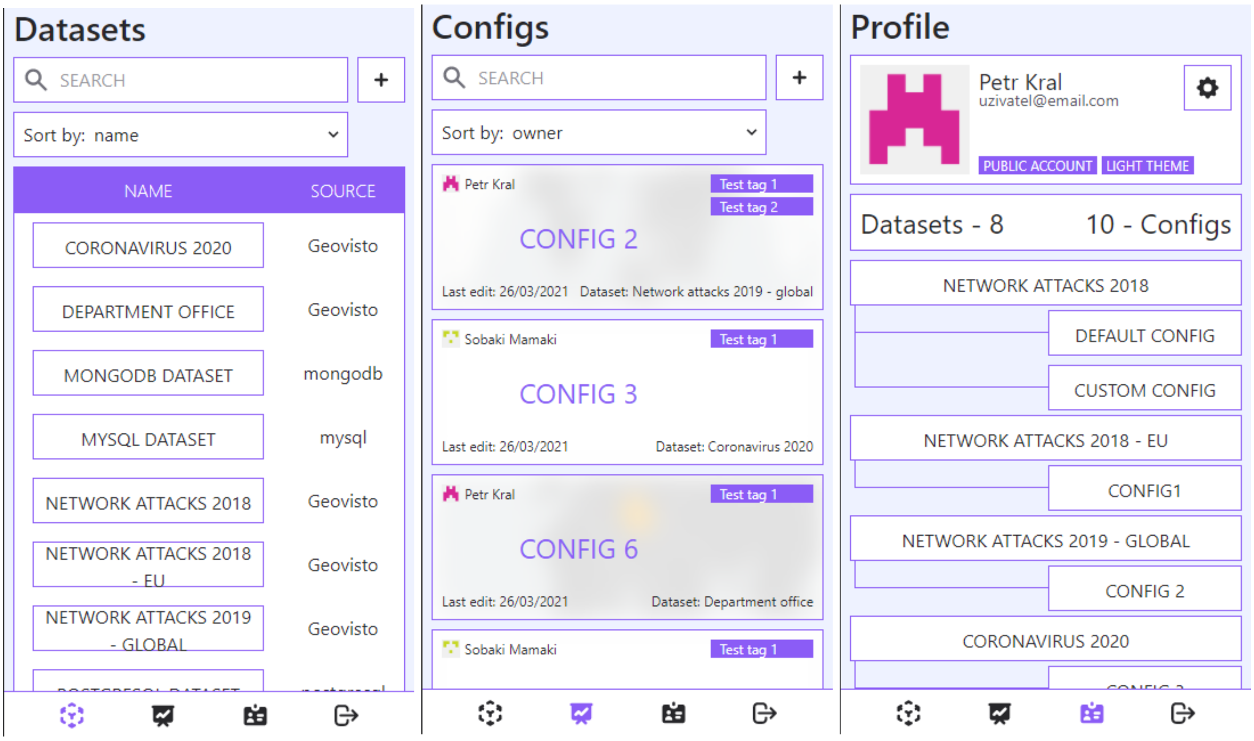

Figure 4. UI screenshots of the main 3 sections, as seen on the mobile devices.

[7] HERE. Here studio — advanced mapmaking. online. Available at https://www.here.com/platform/studio.

[8] Tableau Software & Salesforce. Maps in tableau: Analyze your geographical data. online. Available at https://www.tableau.com/solutions/mapsy.

[9] CleverMaps. Easy-to-use spatial analytics for everyone. online. Available at https://www.clevermaps.io/technology.

[10] J.Clement. Statista - mobile internet usage worldwide - statistics facts. online. Available at https://www.statista.com/topics/779/mobile-internet/.

[11] P. Lake S. Moshiri R. Hill, L. Hirsch. *Guide to cloud computing: principles and practice*. Springer Science & Business Media, 2012.

[12] Heroku & Salesforce. Learn about building, deploying, and managing your apps on heroku. online. Available at https://devcenter.heroku.com.