Architecture

We propose a distributed architecture for rendering, displaying and analyzing geospatial data. The model consists of multiple components interacting and communicating with each other via various networking protocols. This model has been designed while keeping scalability and performance in mind. The client facing web application is deployed in a Kubernetes cluster, consisting of multiple nodes to support a very high number of users. Each node has multiple pods running inside it and each pod in turn comprises of containers running different components of the service. The user connects to the service via HTTP requests and is authorized to access the service via the authorization module running on a container within the pod. After successful authorization, the user can access the various tools from service console. The service console connects to geoserver to fetch the required geospatial data from data sources providing information in a variety of formats and renders it on the webpage. The formats of data sources can be compressed or uncompressed, discrete or continuous, in LiDAR formats, web file format, 3D file format, cartographic file format, etc. and need pre processing to be displayed in the webpages. The proposed model would be able to convert this data to a user readable format.

The user can also request to run an analysis on multiple data features. For this, the service connects to High Performance Computing (HPC) machines to run the algorithms and perform intensive computation at a fast rate. The HPC cluster also provides the benefit of large-scale computing system and reduces the computational load on the application cluster. The user provides inputs on the web page such as coordinates, and feature(s) to be collected, for example, weather, elevation, census or any sensor data. This information is sent as a processing request to the HPC cluster. The data required for running the processing algorithm and analytics is collected from the external data sources present on the cloud or data sources publishing live data. The HPC sends the processed information back to the service and it is displayed/rendered on the webpage. The service has the flexibility to connect to multiple HPCs and switch between them on the fly.

The other modules of the service include the Authentication & Authorization service that needs an external cache which runs on an external VM. The service modules that store data connect to a database that also runs outside the Kubernetes cluster. The user queries can be stored anonymously in the database, which in turn generates another data sources to work and perform analytics on. This innovative approach to collect data

could help in identifying search trends among the users and could also be used for future analysis on which data is more useful and should be further improved.

The system is highly scalable as it relies on nodes that can be easily spawned or destroyed. Additional modules can also be added and features can be extended without affecting the existing performance as the service relies on containerization.

Open questions:

1. What protocol is used to connect to HPC and what should be the input payload? Will user submit job request?
   (Harshita: SSH could be used, will have to check with HPC Lab)

2. What data format is returned by HPC?

3. What type of data should be returned by geoserver?
   (Harshita: Maybe WFS, that can be utilized by the frontend)

4. Is the application which performs data processing and data analytics on HPC already created?