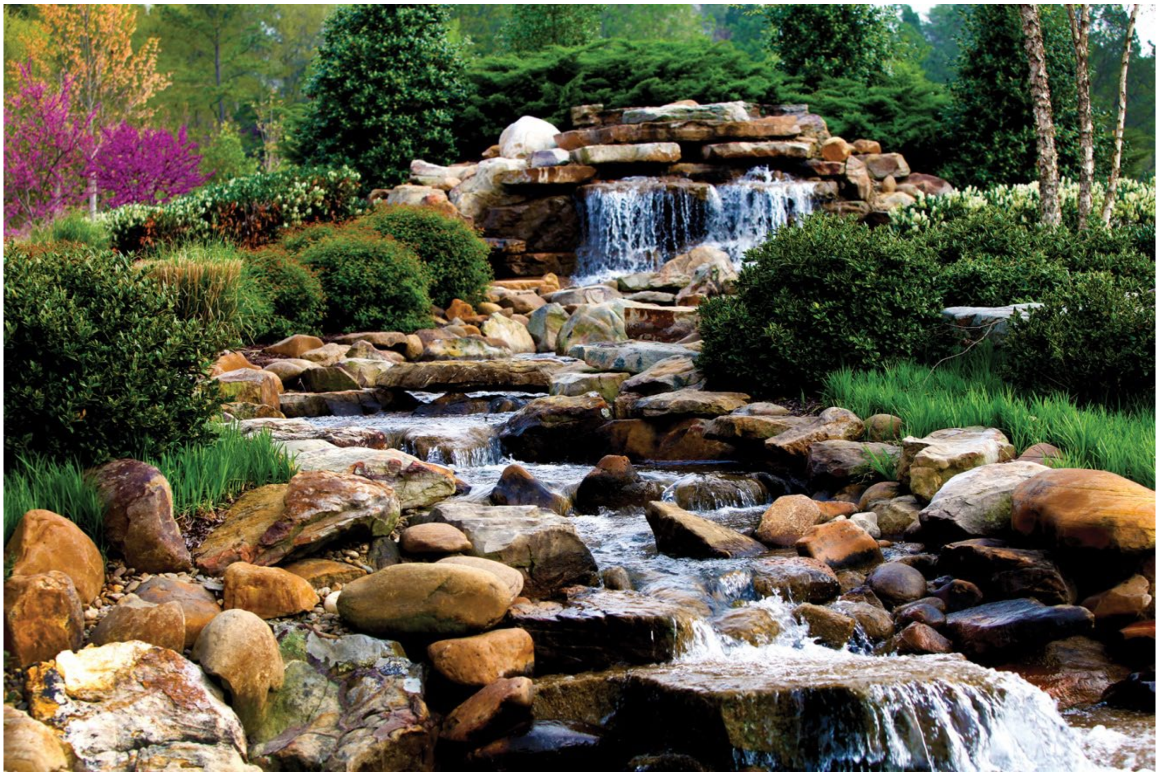
ORNL/SPR-2022/2466
PUB ID 177698

VA-Environmental Determinants of Health's Software Pipeline Framework



Hilda B. Klasky
Matthew Whitehead
Alec Hamaker
Evelyn Johnson
Anuj Kapadia

August 2023

OAK RIDGE National Laboratory

ORNL/SPR-2022/177698
PUB ID 177698

Computer Sciences & Engineering Division

VA-ENVIRONMENTAL DETERMINANTS OF HEALTH PROJECT'S SOFTWARE PIPELINE FRAMEWORK

Hilda B. Klasky
Matthew Whitehead
Alec Hamaker
Evelyn Johnson
Anuj Kapadia

August 2023

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, TN 37831
managed by
UT-BATTELLE LLC
for the
US DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

CONTENTS

1. INTRODUCTION

In order to characterize US communities in terms of social and environmental determinants of health (EDH) that affect veterans, the Social and Environmental Determinants of Health (SEDH) project, a collaboration between the Veterans Affairs (VA) Social Department and the Oak Ridge National Laboratory (ORNL), aims to identify, extract, and prepare datasets that include community variables known to be associated with health outcomes in veterans. The EDH project's responsibilities include investigating SEDH data, providing technical documentation for SEDH datasets, and automating the download, extraction, and preparation of the data through the creation of a software pipeline framework.

The Software Pipeline Framework's objective is to provide software tools to harmonize multimodal social and environmental information. The pipeline framework provides tools to facilitate the use of disparate multimodal data for analysis and modeling in an efficient manner. Since 2021 [1], the ORNL team has streamlined the environmental and socio-demographic data to support the quick creation and re-creation of SEDH datasets to help with research on how these factors may affect veterans' mental health, particularly suicide.

The pipeline framework includes tools that allow for the efficient use of diverse, multimodal data for analysis and modeling. The geographical, chronological, and input variability of the community social and environmental variables is significant (e.g., 1 km by 1 km grids of daily data, yearly zip code data, quarterly county level data, raster files, and csv files). As a result, the measurements must be standardized to the same temporal and geographic references (yearly, census tract level), which is why these software tools are so important.

The pipeline framework-generated datasets cover the entire continental United States (CONUS) at various administrative boundary resolutions, including state, county, zip code tabulation area (ZCTA), and census blocks. Depending on the source, the datasets generated by the pipeline framework cover a variety of time periods. The pipeline generates a flexible package that allows data to be exported as datasets into modeling software, queried in a database system (such as Postgres and MS SQL Server), or prepared for use in a Geographic Information System (GIS) system, such as a GeoPackage. Data can be exported into various formats, including csvs, geojsons, shapefiles, and geopackages, for use in modeling and Geographic Information System software, or queried directly in the database.

We would like to remind the reader in this document that the VA's EDH project primarily works with two types of datasets: 1) authoritative and 2) derivative. Authoritative datasets (95% of the datasets) are those that are "open source" and developed by other organizations. Derivative datasets (5% of the datasets) are those that are generated at ORNL by applying statistical models to open-source datasets available on the internet; we have implemented methods to import them into our database using the pipeline framework. Authoritative and Derivative datasets are provided to our sponsor with the addition of the Federal Information Processing Standards (FIPS) at the state or county level [1].

The job of harmonizing community factor datasets is an extension of earlier work performed in fiscal year (FY) 2021, FY21 [2]. However, during FY22 and FY23, the software architecture has been enhanced by utilizing state-of-the-art technologies that involve building a geographical data pipeline using microservices. Creating automated procedures to provide end-to-end assistance for incorporating geographical data into clinical research and prediction models is a part of this task. By building on earlier work, current ETL functions are enhanced, and new ETL functions are created to enable the pipeline to collect data from more data sources. The focus of this document is to describe the efforts performed during FY22 and FY23 regarding the software pipeline.

2. EDH’S SPATIAL DATA PIPELINE

2.1 BACKGROUND

If the reader is unfamiliar with terms like microservices and Docker, we recommend that they read the section that follows, which introduces and defines technical terms used in the pipeline. Otherwise, the reader can disregard it. The EDH's project software pipeline architecture is built around Docker and microservices. As a result, before delving into the architecture, we'll define these terms to make it easier to understand later.

Docker is a platform that allows you to run and manage distributed applications. Docker is also referred to as a platform-as-a-service (PaaS) solution since it delivers programs in containers via operating system-level virtualization [3]. Docker was founded in 2013 and is still in development [4]. Docker is available in both free and paid editions. Docker containers provide a plethora of benefits for modern program development and deployment. To begin with, they provide consistent and reproducible environments, eliminating common software dependency and requirements problems.

Containers encapsulate applications and their dependencies, making them portable and deployable in multiple environments. They promote scalability by making it possible for applications to be quickly scaled up or down in response to demand. Containers enhance security by isolating apps and preventing them from interfering with each other. They enable quicker development cycles by allowing for faster testing, deployment, and rollback. Scalability, predictability, and portability are also possible in Docker, as it allows users to go from one computer resource to another. Finally, Docker containers encourage a collaborative and modular approach to software development, which simplifies dependency management and team cooperation.

The Docker Engine is the software that runs the containers. The Docker Engine is an open-source containerization platform for developing and containerizing applications. Containerization is a software deployment method that combines the code of an application with all of the files and libraries needed to run on any infrastructure. The Docker Engine has a client-server architecture that includes the following components: 1) In multitasking computer operating systems, a server with a persistent daemon process (i.e., the Docker daemon). A daemon process is a background-running computer program [5]. 2) Application Program Interfaces (API), which define interfaces through which applications can communicate with and instruct the Docker daemon. 3) A client for the command-line interface (CLI).

A container is a software unit or software layer that encapsulates code and all of its dependencies. This wrapping enables the program to be moved quickly and reliably from one computing environment to another. Each container in the microservice architecture handles its own portion of the overall process. Furthermore, each container separates changes to other software containers. A Docker container image is a small, self-contained software package that includes everything needed to run an application, such as code, runtime, system tools, system libraries, and settings [4]. Containerized applications, such as the one depicted in Figure 1, are software applications that incorporate containers into their software architecture.

Microservices architecture, which structures applications as a collection of small, loosely coupled services, provides numerous advantages. Its modular design allows for flexibility, scalability, and ease of maintenance, permitting each service to be developed and deployed independently. Microservices enable faster development cycles because individual services can be updated without affecting the entire system. Furthermore, because failures in one service do not necessarily affect the entire application, microservices facilitate fault isolation. This architecture also encourages agility by allowing teams to use various technologies for various services. Overall, microservices architecture improves the efficiency, resilience, and adaptability of software development, making it a valuable choice for modern, complex applications.

A microservice architecture consists of a collection of services that can be deployed independently, are loosely connected, are organized around business capabilities, and are easily maintainable and tested. This architecture enables the timely, frequent, and dependable delivery of large and sophisticated applications [6].

A collector is a segmented container that will communicate with its original data location.

Using Docker, containers are created to handle a single task or a collection of related activities (implemented in microservices). Depending on the inputs, one or more containers will be launched to conduct extract, transform, and load (ETL) processes.

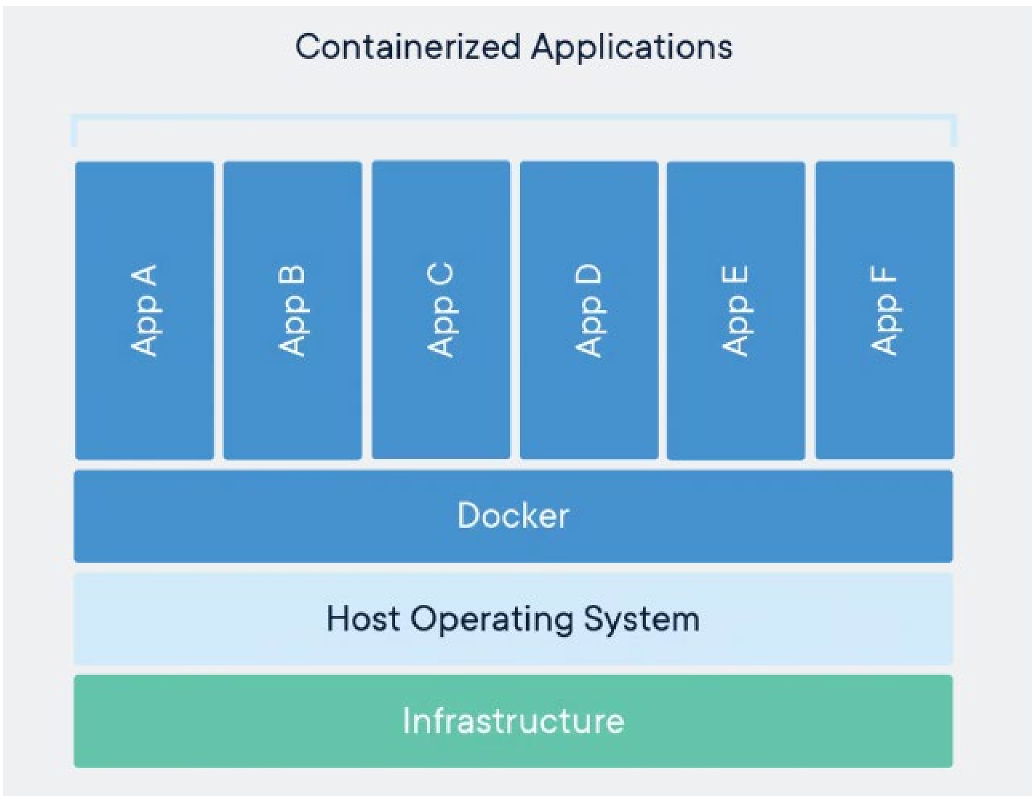


Figure 1. Generic Software Architecture of Containerized Applications, image from https://www.docker.com/resources/what-container/

2.2 SOFTWARE ARCHITECTURE

We developed procedures and code for extracting, transforming, and loading (ETL) data from reliable sources during Fiscal Year 22' (FY22). These sources provide open-source data in various formats, which we standardized and incorporated using our pipeline framework. We created software tools to ensure that the importing procedures were followed consistently. We developed ETL software that collects and retrieves open-source datasets on social and environmental determinants of health from well-known sources. A collector container in this context is a piece of software that interacts with authoritative sources to collect data. Data sources include the Centers for Medicare and Medicaid Services, the

Substance Abuse and Mental Health Data Archive, and the United States Census Bureau of Labor Statistics, among many other ones.

A key advantage of a containerized microservices-based architecture for developing and deploying ETL functionality is the ability to maintain reproducibility when dealing with collections from different sources as well as multiple collections from a single source. Docker is used to deploy a series of containers to perform ETL functions, store data, provide API access to the data, and serve a web application and services in the case of this project.

Figure 2 depicts the software architecture of the spatial data pipeline. The components are described in detail in the following section: Container Descriptions and Functions. The red containers are in development or are about to be developed. The colors of the arrows indicate whether the process occurs over HTTP/web connections (green), custom command line tools (purple), or internal Docker container communication (black). The pipeline's ETL functions are invoked in three ways: 1) by a pipeline operator using command line tools, 2) by adding data to the designated 'watcher' folder, or 3) by using the graphical user interface (GUI) that leverages the API. Because this is a multitask architecture, the containers can be run in different forms by another user or the same user.

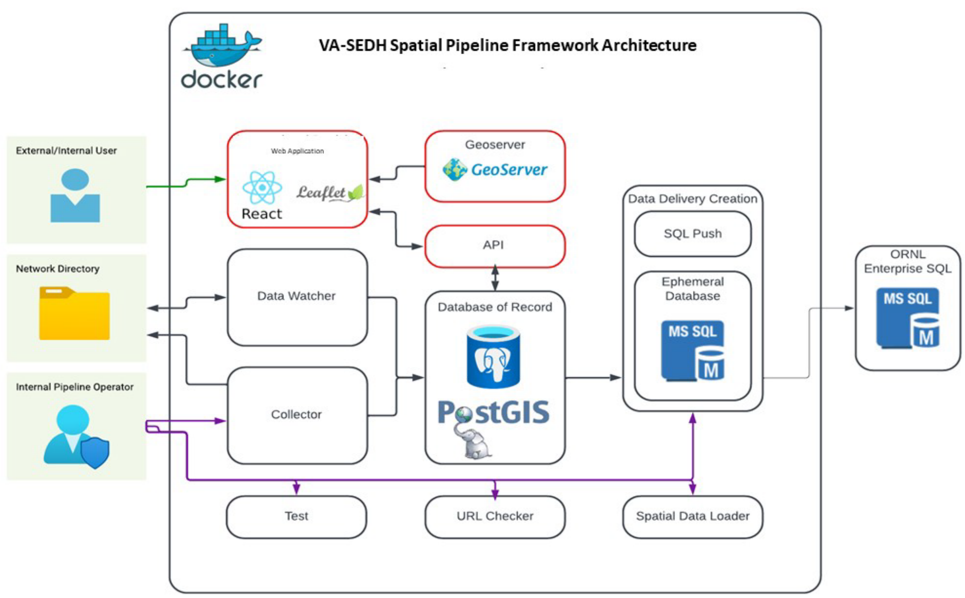


Figure 2. VA EDH project's software architecture of the spatial data pipeline framework.

To allow for version control, all development, including system architecture and ETL development, is stored in GitLab (see https://code.ornl.gov/ctsa_2020/spatial_data_pipeline). Along with the code, the git repository contains detailed README files that contain detailed instructions for running the spatial data pipeline and deploying the pipeline containers. This ensures that all users have clear data generation instructions, reduces reliance on specific personnel, and ensures that the pipeline can be quickly deployed on a new system if the need arises.

2.3 CONTAINERS DESCRIPTIONS AND FUNCTIONS

- **Database of record**: The Database container is the centralized database responsible for storing data and serves as the system's backbone. This persistence database container holds all processed data, whether from authoritative sources or derivative datasets. PostgreSQL with the PostGIS extension is installed in the container.
- **Collector**: The collector container is in charge of gathering authoritative data from a variety of sources. Before storing the output in a network directory on the actual hardware that it is running on, any (extraction, transformation, and loading) transformations to the data are applied, and Quality Assurance and Quality Control are performed.
- **Data Watcher**: The load stage is managed by Data Watcher. The Data Watcher container scans the network directory for new data files that have been manually copied from derivative data or authoritative data from the Collector container. When new data is detected, it is automatically added to the Database container and a record is added to the data catalog table. The data catalog table can be queried to obtain a high-level overview of the data in the database.
- **Geoserver**: The web mapping services (WMS) and web feature services (WFS) are generated by the Geoserver container. This is a critical component that generates the services required for data to be displayed in the Graphic User Interface. A new WMS is created when data is added to the Database container. When new data is added to the Database container, a new WMS is created automatically. (Currently under development, adding support for new datasets). At the moment, we only support NICS data.
- **Web Application**: React Leaflet is the web mapping framework used. This front-end application allows users to query the data and visualize it in a web map. Users can also view the data in a table view, filter it, and export it in a variety of formats, including csv. (In the future, it will also support shapefiles and GeoPackages.)
- **API Container**: The API container facilitates communication between the database, the front-end application, and end users. The API container can query the database and pass the results to the Frontend.
- **URL Checker**: The URL Checker checks the URLs used by the Collector container to retrieve data on a regular basis to see if the endpoints are still accessible. If a website is no longer accessible, the internal pipeline will write a text message via log file.
- **SQLsvr Push and MSSQL database**: These containers collaborate to generate deliverable outputs from Postgres to be imported to the MSSQL database outside the ORNL's Knowledge Discovery Infrastructure (KDI) enclave. When it's time for a quarterly delivery, the internal pipeline operator activates this container. The database container is ephemeral in the sense that it only runs to create the deliverable before being stopped and removed.
- **Spatial Data Loader:** The spatial data loader is a lightweight container for loading boundary files from known sources into a spatial database. This is only run when a new database is created (for example, when migrating the spatial data pipeline to new hardware). We manage spatial resolutions such as county, state, and tract.

The following two scripts have been developed to aid in the automation of time-consuming tasks. The docker.sh script starts different containers on different conditions for example: “docker.sh main” will run all the main containers:

```
usage:
sh docker.sh [--help | -h] [--deployed | -d]
             [--build | -b] [--attached | -a]
             [--datasource=<data source>]
        <container name(s) | alias>

Examples:
        ./bin/docker.sh db


        ./bin/docker.sh --build -a datawatcher


        ./bin/docker.sh main


        ./bin/docker.sh -b -a --datasource=nics collector


Avaiable Containers:
        - db, datawatcher, collector, urlchecker, test, sqlsvr_push,
         dataloader

Avaiable Aliases:
        - main
```

Figure 3. docker.sh command options and examples.

```
./bin/create_collector.sh [OPTIONS] LOADER_NAME
        --help | -h
                Print this help message

        --doc-dump=* | --documentation-dump=*
                Define the documentation dump to add to the urls.json file

        --description=*
                Define the description of the data set to add to the urls.json file

        --resolution=* | -r=*
                Define the resolution of the data set to add to the urls.json file
                Options are 'state', 'county', and 'tract'

        --url=* | -u=*
                Define the url to add to the urls.json file.
                Where to pull data from

        --file-name=* | --filename=* | --file=* | -f=*
                Define the name of the file to pull to add to the urls.json file.

        --url-type=* | --urltype=*
                Define the url type.
                This should be something like: curl or Additional

        --commonname=* | --common-name=* | -c=* | --c-name=*
                Define the common name of the dataset for the urls.json file
```

Figure 4. create_collector.sh command options

The list of software and its versions used in the EDH pipeline framework as of FY22 and FY23 is found in Table 1.

Table 1. list of software utilized in the EDH pipeline framework.

Software	Version	Comments
Python	3.9	
Docker	24.0	In the VA-EDHDATA virtual machine
Geoserver	2.20.4	
Postgres	13.4	
PostGIS	3.1	
SQL Server Management Studio	19.0	
React Leaflet	4.0	Uses React's context API to make some Leaflet elements available to children's elements.
VA Spatial Data Pipeline	0.1.3	Load package
Typescript	4.6.3	Programming language.
React	18	Open-source front-end JavaScript library
React Router	6.3	JavaScript framework used to handle client and server-side routing in React applications
MUI	5.5	Multilingual User Interface
Jest	27.4	JavaScript testing framework
Bootstrap	4.5.3	Framework for development using HTML, CSS, and JavaScript
HTML	5	
CSS	3	

3. PRELIMINARY WEB MAPPING APPLICATION

As the software matures, preliminary efforts have been made to develop a web mapping application that will eventually become the geospatial decision support system. An evaluation of existing web mapping applications, as well as a technology review, were part of the early stages of development. The decision on the front-end framework for the geospatial decision support system was made at the end of this review. The web mapping application is written in React, a front-end JavaScript library for creating user interfaces that also uses TypeScript, HTML, and CSS. React Leaflet is used for map elements. React Leaflet was chosen based on our evaluation because of its mature code base, widespread usage, and robust documentation. It also includes a third-party React library and Leaflet wrapper, as well as support for the use of TypeScript. Geo-server is the technology that is providing the WMS and WFS that are displayed in the application.

While the graphic user interface (UI)'s of the web mapping application is still in its early stages, some UI elements and functionality have been created. To provide the user with spatial context, a map-base layer has been added. A software layer has been added to the map by leveraging Geoserver services. Users can toggle between layers for visualization by using the toggle functionality. When a user selects a layer, the active layer displays a legend to provide context for the visualization. There is also an on-click function, which allows a user to click the map and get a popup window with data related to the selected feature. The data is also available in a table format. By clicking on a separate page, users can view the data in tabular form.

A separate page allows users to view the data in tabular form by clicking on the table name displayed in the 'Available Tables' section. In the future, we plan to develop functionality that will allow users query and filter the data and export the data as a csv.

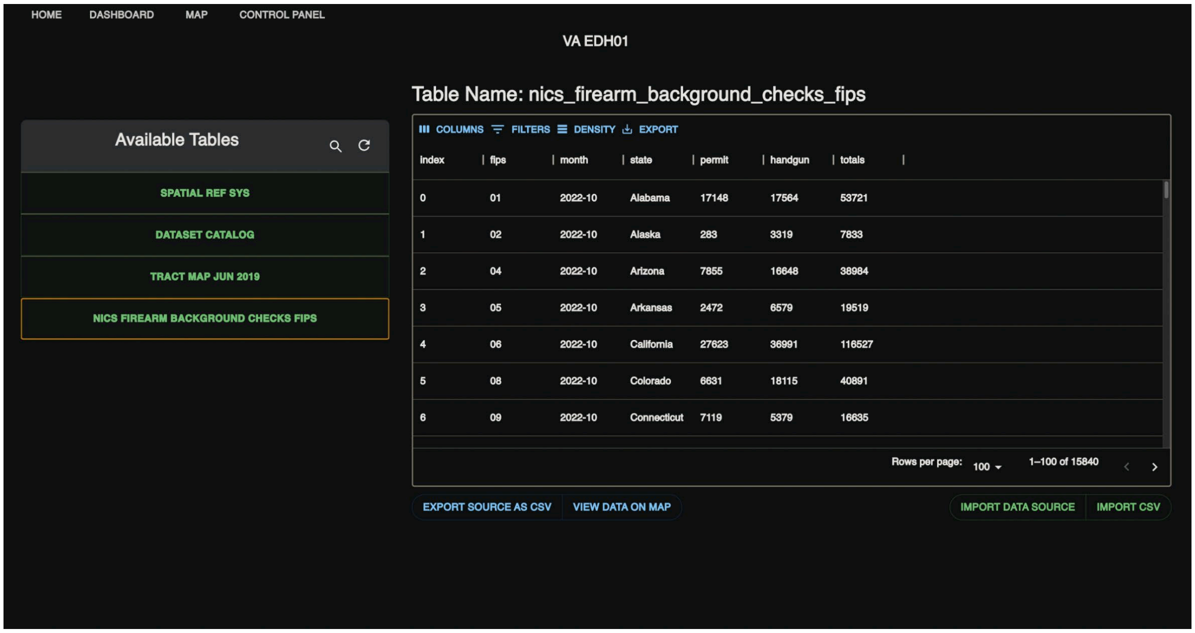


Figure 5. Spatial Data Pipeline (a)

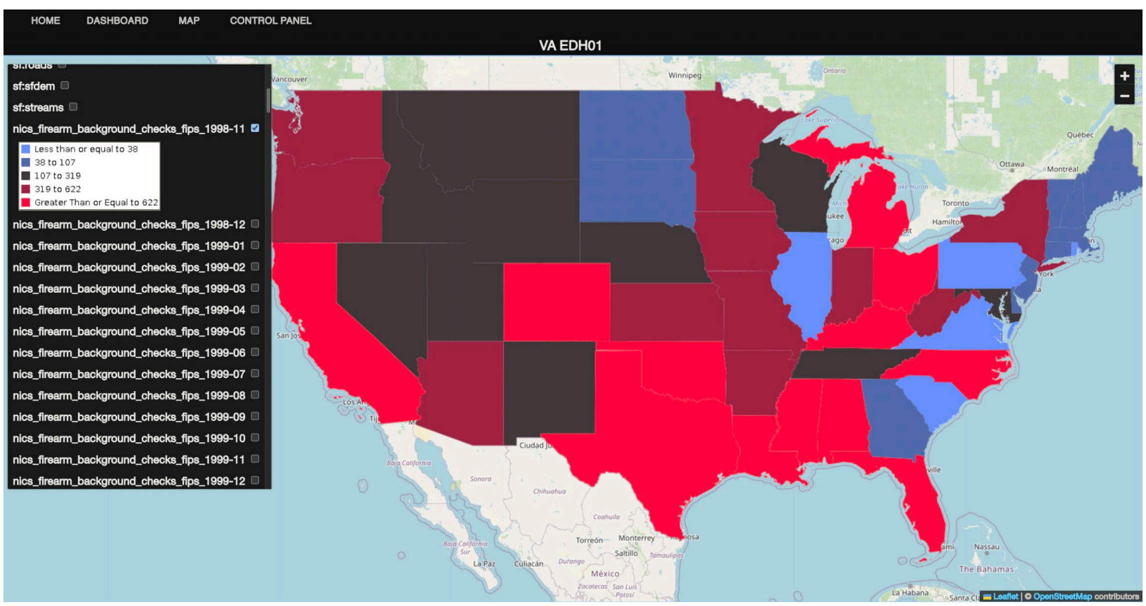


Figure 6. Spatial Data Pipeline (b).

Figure 6 and Figure 7 show the web mapping application. This is NICS firearm data at the state level. Behind the scenes in our pipeline framework, a container runs the GEO server. This GEO server is in charge of producing online mapping services. Our implementation follows the standard Open Geospatial Consortium (OGC)-compliant manner of transmitting geographic data over the web. We believe that this approach will function in the GEO-BISL application, which is part of the ESRI base system.

4. AUTOMATED DATASETS RETRIEVAL AVAILABLE

Through the development of a datasets catalog our team has automated the acquisition of the datasets in in our quarterly reports to our VA sponsor [2, 7-13]

	common_name text	table_name text	table_schema text	file_name text	access_date date	description text	urls text	doc_dump text	push boolean
1	Individual-Oriented Social Vulnerability Index Census Block Groups	iosvi_bg_2019	public	iosvi_bg_20...	2022-02-15	Our team at O...	None	None	false
2	Rural Urban Continuum Codes County Level	usda_rucc_2013	public	USDA_RUC...	2022-02-15	None	https...	None	false
3	Evictionlab Data All Cities County Level	evictionlab_all_sites_monthly_2020_2021	public	evictionlab_...	2022-02-15	We hope that ...	https...	None	false
4	Evictionlab Data All States County Level	evictionlab_all_states_monthly_2020_2021	public	evictionlab_...	2022-02-15	We hope that ...	https...	None	false
5	Household Income Quintile Upper Limits County Level	census_household_income_quintile_uppe...	public	census_hou...	2022-02-15	None	https...	None	false
6	Mean Household Income of Quintiles County Level	census_mean_household_income_of_qui...	public	census_me...	2022-02-15	None	https...	None	false
7	Share of Aggregate Household Income by Quintile County Level	census_share_aggregate_household_inco...	public	census_sha...	2022-02-15	None	https...	None	false
8	GINI Index of Income Inequality County Level	census_gini_index_of_income_inequality_...	public	census_gini...	2022-02-15	None	https...	None	false
9	Ratio of Income to Poverty Level in the Past 12 Months County Level	census_ratio_income_to_poverty_lvl_past...	public	census_rati...	2022-02-15	None	https...	None	false
10	Census Income Inequality	census_income_inequality	public	census_inc...	2022-02-15	None	None	None	false
11	Health Profession Shortage Area Primary Care County Level	hrsa_hpsa_fct_primary_care	public	HRSA_HPS...	2022-02-15	None	https...	None	false
12	Health Profession Shortage Area Dental Health County Level	hrsa_hpsa_fct_dental_health	public	HRSA_HPS...	2022-02-15	None	https...	None	false
13	Health Profession Shortage Area Mental Health County Level	hrsa_hpsa_fct_mental_health	public	HRSA_HPS...	2022-02-15	None	https...	None	false
14	Medically Underserved Areas County Level	hrsa_medically_underserved_areas	public	HRSA_MEDI...	2022-02-15	None	https...	None	false
15	Facebook Social Connectedness Index County Level	humdata_county_county_fb_soc_conn_oc...	public	humdata_c...	2022-02-15	Details on the...	https...	None	false
16	Population Report State Level	cms_us_state_lvl_pct_chron_2018_june_2...	public	CMS_US_ST...	2022-02-15	None	https...	None	false
17	Population Report County Level	cms_us_county_lvl_pct_chron_2018_june...	public	CMS_US_C...	2022-02-15	None	https...	None	false
18	Medical Disparities by Hospital County Level	cms_med_dispar_by_hosp_2021	public	/tmp/data/...	2022-02-15	None	https...	None	false
19	Market Saturation and Utilization State Level	cms_market_sat_and_util_state_level_2021	public	CMS_MARK...	2022-02-15	None	https...	None	false
20	Market Saturation and Utilization County Level	cms_market_sat_and_util_county_level_2...	public	CMS_MARK...	2022-02-15	None	https...	None	false
21	Evictionlab All Counties County Level	evictionlab_all_counties	public	evictionlab_...	2022-03-09	None	https...	None	false
22	Firearm Background Checks State Level	nics_firearm_background_checks_fips	public	NICS_FIREA...	2022-03-09	None	https...	None	false

Figure 7. Data Catalog of the EDH software pipeline framework as of FY22Q2.

Figure 7 depicts a list of datasets stored in the developer server's Postgres database. We deployed 41 datasets to the VA's Corporate Data Warehouse (CDW), but only the ones shown in Figure 7 are automated (22). Each table represents a single dataset delivered. Future plans include developing capabilities to allow users to determine what is in the dataset by clicking on a table in Figure 7. Each table will include standard filtering and sorting options, as well as the ability to export to a csv file and view data on a map. Users will also be able to import a new data source from this page. This website will include a module for the user to populate to add raw data, and just add a csv as well.

5. LIMITIATIONS AND FUTURE WORK

The development Spatial Data Pipeline is an ongoing project. Data curation automation, ETL processes and workflows will evolve further in the next FY. This will include both improving and expanding the Collector container's functionality. To retrieve new datasets from authoritative sources, new ETL scripts will be developed. We are developing additional Quality Assurance, Quality Control and validation functionality that can be added to the Collector container to improve data quality in a reproducible manner.

The following key features will be developed or improved:

- Automate the retrieve and regenerations of all datasets deployed to CDW.
- An interactive layer selection tool that will allow users to toggle web mapping layers on and off, filter the active layer, choose a single year or multiple years (with the option to sum or average the data over multiple years), and display a legend for the active layer.

- The ability to query by clicking web mapping features or drawing a bounding box around multiple features is one example of rich selection and querying capabilities we will implement in the future.
- The ability to select and query a state, as well as auto-pan and zoom to that location, will be added.
- The print feature allows the user to save the current map view as a PDF.

Other features and functionality will be identified through ongoing interactions with the VA in order to continuously improve the front-end web application. wIn addition, we will develop a decision support GIS system and plan to make the source code available to our sponsors.

6. REFERENCES

[1] H. B. Klasky, Hanson, H., Sparks, K., Whitehead, M., Blair, C., Trafton, J.A., and Kapadia, A.,, "Dataset Repository for Investigating Suicide Risk Using Social and Environmental Determinants of Health - Manuscript under editorial consideration," 2022.

[2] B. Christian *et al.*, "VA EDH Data Curation Documentation FY22-Q1, Rev. 2, ORNL/SPR-2022/2316," Oak Ridge National Laboratory, United States, 12 2021. [Online]. Available: https://www.osti.gov/biblio/1854460-va-edh-data-curation-documentation-fy22-q1-rev

[3] M. O'Gara, "Ben Golub, Who Sold Gluster to Red Hat, Now Running dotCloud," *SYS-CON Media,* vol. 26, 2013.

[4] "Docker." https://www.docker.com/ (accessed November, 2022.

[5] C. Negus, *Linux Bible Tenth Edition*.

[6] "Microservice Architecture." https://microservices.io/ (accessed November, 2022.

[7] B. Christian *et al.*, "VA EDH Data Curation Documentation – FY21, Rev. 2, ORNL/SPR-2021/2366," Oak Ridge National Laboratory, United States, 2022. [Online]. Available: https://www.osti.gov/biblio/1854468-va-edh-data-curation-documentation-fy21-rev

[8] B. Christian *et al.*, "VA EDH Data Curation Documentation FY22-Q2, Rev. 2, ORNL/SPR-2022/2391," Oak Ridge National Laboratory, United States, 3 2022. [Online]. Available: https://www.osti.gov/biblio/1862127-va-edh-data-curation-documentation-fy22-q2-rev

[9] H. Klasky, K. Sparks, and J. Logan, Tuccillo, Joe, Whitehead, Matthew, Hamaker, Alec, Hanson, Heidi, Watson, Rochelle, and Kapadia, Anuj., "VA EDH Data Curation Documentation - FY22-Q3, ORNL/SPR-2022/2487," Oak Ridge National Laboratory, United States, 2022. [Online]. Available: https://www.osti.gov/biblio/1876283-va-edh-data-curation-documentation-fy22-q3

[10] H. B. Klasky *et al.*, "VA EDH Data Curation Documentation FY22-Q4, ORNL/SPR-2022/2587," Oak Ridge National Laboratory (ORNL), Oak Ridge, TN United States, 2022.

[11] H. Klasky, Sparks, K., Logan, J., Hamaker, A., Whitehead, M., Peluso, A., Hanson, H., Watson, R., & Kapadia, A.,, "VA EDH Data Curation Documentation FY23-Q1, ORNL/SPR-2022/2694," Oak Ridge National Laboratory, United States, 2022. [Online]. Available: https://doi.org/10.2172/1909101

[12] H. Klasky *et al.*, "VA EDH Data Curation Documentation FY23-Q2 ORNL/SPR-2023/2857," Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), United States, 2023.

[13] H. Klasky, Sparks, K., Peluso, A., K., Logan, J., Hamaker, A., McGee, M., VanDerslice, J., Hanson, H., Watson, R., and Kapadia, A.,, "VA EDH Data Curation Documentation - FY23-Q3 ORNL/SPR-2023/2930 PUB ID 195499," Oak Ridge National Laboratory, United States, 2023.