FREC 4044

02/07/2025

Capstone Proposal - Draft 1

Throughout the semester, we’ll be developing a shiny app for the International Soil Radiocarbon Database (ISRaD). This database holds data collected from various sites, but it also allows other teams and organizations to contribute their own data. The data in ISRaD is organized into a hierarchical structure, with different categories that are all connected to one another. Each data entry belongs to a specific category, and the way it’s formatted will determine which categories it gets linked to. So, depending on the format of the data, it might be linked to one or more categories within the hierarchy.

The shiny app should always include the metadata, site, and profile tables as a default. These three tables will be displayed with all of their columns, and no matter what other tables the user selects, these will always be included. Users can choose from additional tables like layer, flux, interstitial, fraction, and incubation. The app needs to handle different combinations of tables, as some are linked to others. For example, flux and interstitial tables connect directly to the profile table, while selecting fraction also requires merging the layer table in the background. Similarly, the incubation table depends on either fraction or layer data, so the app should ask the user how they want to incorporate incubation data.

Once the tables are selected, the app will merge them into one flattened file. The flattened file will combine the metadata, site, and profile tables first, and then add any other selected tables, like layer or flux, in the right order. This merging process will be done behind the scenes.

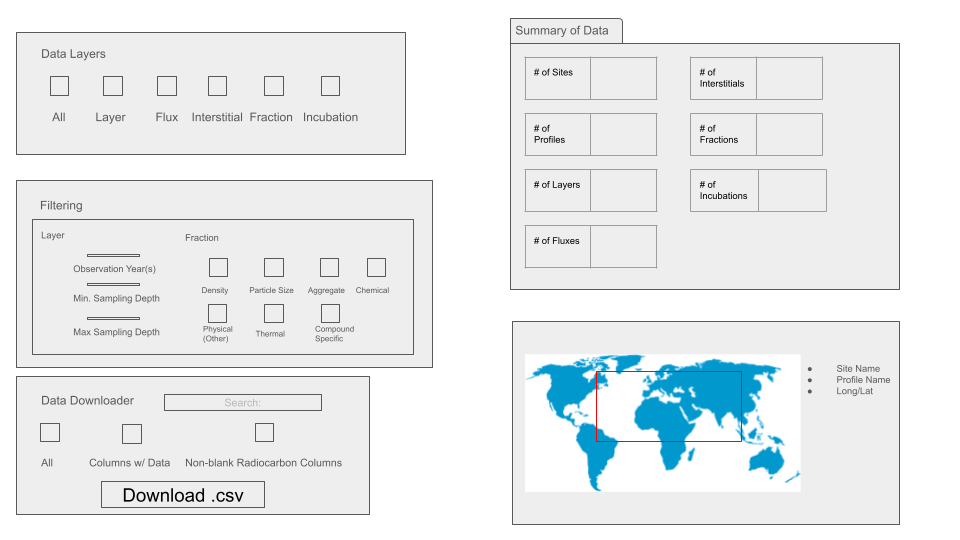
Users will also be able to filter the data based on what they’ve selected. If they choose the layer table, they can use sliders to filter by observation year and sampling depth. For sampling depth, there will be two sliders: one for the minimum depth and one for the maximum. The app will make sure to include all data points that fall within the chosen depth range, even if they’re part of overlapping ranges. If the fraction table is selected, the app will let users pick from different fractionation methods using checkboxes, so they can select terms like density, particle size, thermal, and other fractionation methods.

There will also be a map feature, allowing users to visualize the data. Users can click on points on the map to see the site name, pro name, pro lat, and pro long for that location. Plus, they’ll be able to draw a square on the map to highlight a region of interest.

To help users understand their data, the app will show a summary of basic statistics, like the number of sites, profiles, layers, fluxes, interstitials, fractions, and incubations in the dataset. These stats will give a quick overview of the data they've selected.

Before downloading, users will have a few options for how they want their data exported. They can choose to download all columns, including those with no data, just the non empty columns, or only rows that have data in the radiocarbon columns. There will also be a search function so users can easily find specific terms in the data.

Finally, it would be great to offer an optional video tutorial to guide users through the app. This tutorial would explain how to use the features, select data, filter it, and download results, making it easier for users to get started, but this will only be included if we have enough time after finishing the shiny app.



*Fig. 1 - Mockup of UI for proposed app.*

As described above, the app serves to display data and determine which aspects will be compiled and downloaded in a .csv file. In the top right corner, there is a section for data layers and checkboxes to determine which columns of data from each entry you wish to download. The section below gives additional options on subsections of data based on the layer and fraction options, if they are selected. Lastly, the bottom section on the left side of the app is the data downloader itself. There are also several additional options for filtering data, allowing users to filter by keywords or by the contents of certain columns. The top right section of the app gives summaries of the filtered data. The bottom right section has an interactive map that will show characteristics based on what is selected on the map. This includes site name, profile name, and long/lat when an individual entry is selected. Alternatively, users can draw a rectangle or polygon on the map to return summary statistics of the sites within the demarcated area.

There are a few existing apps that currently possess similar functionality to what we are trying to achieve here with our app. The app of choice for this representation was the Utah Lake Water Quality Profile Dashboard. This app was used by the researcher to give a good template and place to start from. There is quite a significant amount of usable information from this site that can be applied to our app. The key aspect that stands out is the functionality around site selection. If a region is selected on the map a whole readout is given and from there, data may be downloaded. This is an element that we are looking to integrate into our app. For us, our app will do a similar function where the user may enter a filter or region that they wish to choose from and from there the respective information will be displayed in table format. Subsequently, a download will be available where a CSV file will be provided as done in the example site.

Some possible errors or areas of trouble that we expect to encounter during app creation occur in a few key regions. In no particular order, the first area of trouble will appear with data importation. Our goal for this project is to have an adaptable upload for new data pulled directly from a repository so new information is always available however with this some issues come to mind. Once the app has been created a strict “template” of data must be created for consistency. This could come to be an issue where new rows are created or areas of reference are changed or destroyed. A key area of issues/troubles will be the filters. This will arguably be the toughest point of creation as many filters are looking to be applied, where each comes with a complex hierarchical structure that must be followed and “orders of operations”.

A more front-ended issue that will be a repeated event is the constant breaking of the app with new additions being made. To combat this it is important to write extremely clear and segmented code to separate each section for easy diagnosis when display issues occur.