



Clear Lake Website and Data Visualizations **User Guide**

Cesar Ayuso
Valentina Lai
Alison Chen
Kenneth Lieu

Preface	4
Readme	4
Intended Audience	4
Related Documentation	4
Overview of the product	4
Background	4
Technical Specifications	5
Features	5
Installation or distribution	6
Site Farm Clear Lake Main Website	6
React Clear Lake Data Visualization Website	6
Functionalities	7
Charts	7
Map	12
Weather Widget	12
Data Archive Page for Users	13
Data Upload Page for Client	13
SiteFarm Site	14
Troubleshooting	14
Frequently Asked Questions	16
Glossary	16
Contact Information	17
Appendix	17
Technology Survey	17
Frontend	17
Database	18
Data Visualization	19
System Architecture Overview	21
Overview of our data processing architecture.	21
External Data Collection	23
User Stories	24
Prototyping Code	25
Technologies Employed	25
Cost Analysis	25
Legal and Social Aspect	25

Preface

Readme

Clear Lake, the largest lake in California and one of the oldest lakes in North America, is subject to eutrophication as a consequence of the high concentrations of nutrients. These nutrients stimulate high concentrations of algae and cyanobacteria and consequently present severe challenges to Lake County, the lake users, and stakeholders. UC Davis is engaged in a multi-year research study to understand the dominant processes in the Clear Lake watershed and in the lake itself that are negatively impacting the restoration of the lake water quality and the ecosystem health. We are currently collecting a wide variety of data to form the basis of a long-term monitoring strategy to measure the status and trends in the future in Clear Lake.

Intended Audience

This is intended to guide current and future engineers and TERC team researchers to update and maintain the data visualization graphs, front and backend functionalities, and any installations and deployments of the Clear Lake websites. As our project entails two websites, one made under UC Davis's SiteFarm and another made from scratch using React.js, HighCharts, and AWS, this user guide will also help the audience with little or no experience in using SiteFarm and to gain some basic knowledge in using GitHub to access, troubleshoot, and run our Clear Lake Data website.

Related Documentation

React.js Documentation: <https://reactjs.org/docs/getting-started.html>

SiteFarm Training: <https://sitefarm.ucdavis.edu/training/all>

HighCharts Documentation: <https://www.highcharts.com/docs/index>

HighCharts Demos: <https://www.highcharts.com/demo>

Overview of the product

Background

The UC Davis Tahoe Environmental Research Center (TERC) team currently owns a non-official UC Davis Wix website to share their blogs, data, data visualization, and publications. This brings several concerns, one being the static use of data in the current data visualization of their website. Additionally, the big danger in this static data is the location in which the data is currently stored, which is a former web developer's personal Github repository and being called

for in the data visualizations. This brings instability and insecurity, as the data is controlled and managed by a single former employee.

Moreover, the TERC team is looking to have more interactive and dynamic data visualizations, allowing lake users and stakeholders to view specific time windows of data, generate CSV files to their local computer, and filter different lake and stream variables within a specific data chart. We are innovating this problem to create more visually appealing and user-friendly data visualizations. We plan to increase the user journey and experience through this new site, which will encourage more lake holders and stakeholders to be part of the research and the findings of the TERC team.

Technical Specifications

- Frontend
 - React.js: front-end JavaScript library for building user interfaces based on UI components
 - HighCharts: JavaScript based charting library meant to enhance web applications by adding interactive charting capability
- Database:
 - Amazon S3: enables client to upload clean data CSV files to be uploaded in the database tables
 - Amazon Lambda Functions: processes the uploaded clean data files to the database tables and connect API endpoints to retrieve data from the database tables
 - Amazon API Gateway: creates API endpoints to retrieve data from database tables
 - Amazon RDS: managed services that makes it simple to set up, operate, and scale databases in the cloud
 - Amazon Event CloudWatch: to schedule events, such as automating data scraping from external websites
 - MySQL: relational database management system
- Site Farm: allows client to easily manage and edit the main Clear Lake website under UC Davis domain
- Other tools: Github to hold frontend code and host the React website


Features

- Dynamic and interactive data visualizations for Streams, Meteorology, Lake Profile, and Lake Mooring
 - Allow users to query graphs on different variables and time intervals
 - Allows zooming features, tooltips for data point information, variable toggling to view trends between different variables

- Frequently Asked Questions for any user concerns, such as the link to metadata and how to use the graphs
- Map of the Clear Lake Watershed Boundary and data collection locations
- Provide a weather widget to show current and forecasted weather at Clear Lake
 - Includes a brief description of the weather, wind, and humidity, and icons to illustrate the weather
- Downloadable CSV feature for users to download clean and real-time data
 - Allows users to query different variables and time frames to download
 - Allows other downloadable types, such as an image of the chart in PNG, JPEG, PDF, SVG forms
- Uploadable CSV feature for client to upload clean data
 - Connect any uploaded data to MySQL with the data visualization graphs and automatically update them
- Easy to manage the site for clients to update blogs, news, photos, and more
 - Used SiteFarm as it is easy to learn to use and has many training videos and UC Davis SiteFarm managers for more complex guidance

Installation or distribution

Site Farm Clear Lake Main Website

1. Login at <https://clearlakerehabilitation.ucdavis.edu/login> using UC Davis credentials
2. To access login information, please contact Alicia Cortes at alicortes@ucdavis.edu
3. Some documentation:  SiteFarm Documentation

React Clear Lake Data Visualization Website

Prerequisites:

- [Github](#) account
- [git](#)
- [Node.js](#) for npm

Note: Insert the text after \$ (excluding \$) into the terminal.

1. Copy the repository to your local machine:


```
$git clone
https://github.com/tercdev/Clear_Lake_Website_Data_Visualization.git
```
2. Navigate to the newly created folder:


```
$cd Clear_Lake_Website_Data_Visualization
```
3. Navigate to the folder that holds the code for the React app:


```
$cd clear-lake-restoration-app
```
4. To run the app locally for development:


```
$npm start
```
5. The site will be found at http://localhost:3000/Clear_Lake_Website_Data_Visualization/

6. For changes made to show up on the live site at
https://tercdev.github.io/Clear_Lake_Website_Data_Visualization/
`$npm run deploy`

Functionalities

Charts

- All Charts
 - Interactive synchronized zoom-in graph windows
 - Download data displayed in charts in various formats
 - Date pickers for users to select desired time frames.
 - Click on the name of the series in the legend to toggle graph lines.
 - Included instructions on how to use the graphs and answers to some FAQs.
 - Tooltips that show the data in the chart the cursor is hovering over
- Streams
 - Flow data from
<https://cdec.water.ca.gov/dynamicapp/QueryF?s=KCK&d=17-Apr-2022+24:00&span=1days>
 - Stations: KCK: Kelsey, MCU: Middle, SCS: Scotts
 - Fetched every 15 mins
 - Rain data from
<http://cdec4gov.water.ca.gov/dynamicapp/QueryF?s=LYO&d=01-Nov-2018+23:00&span=31days>
 - Stations: KCK: KTI, MCU: LYO, SCS: Scotts
 - Fetched every hour
 - Turbidity and Water Temperature usually has real-time sensors that record every 10 mins.
 - Fahrenheit or Celsius option for water temperature.
 - The research team cleans and updates any data.
 - The dashed line indicated real-time data
 - Able to fetch 3+ years data
 - 1-week window is initially shown
 - PST Time. Charts show 7 hours before the start date. This is because data fetched from the endpoint is in UTC.

Note: These data are provisional and not error checked!

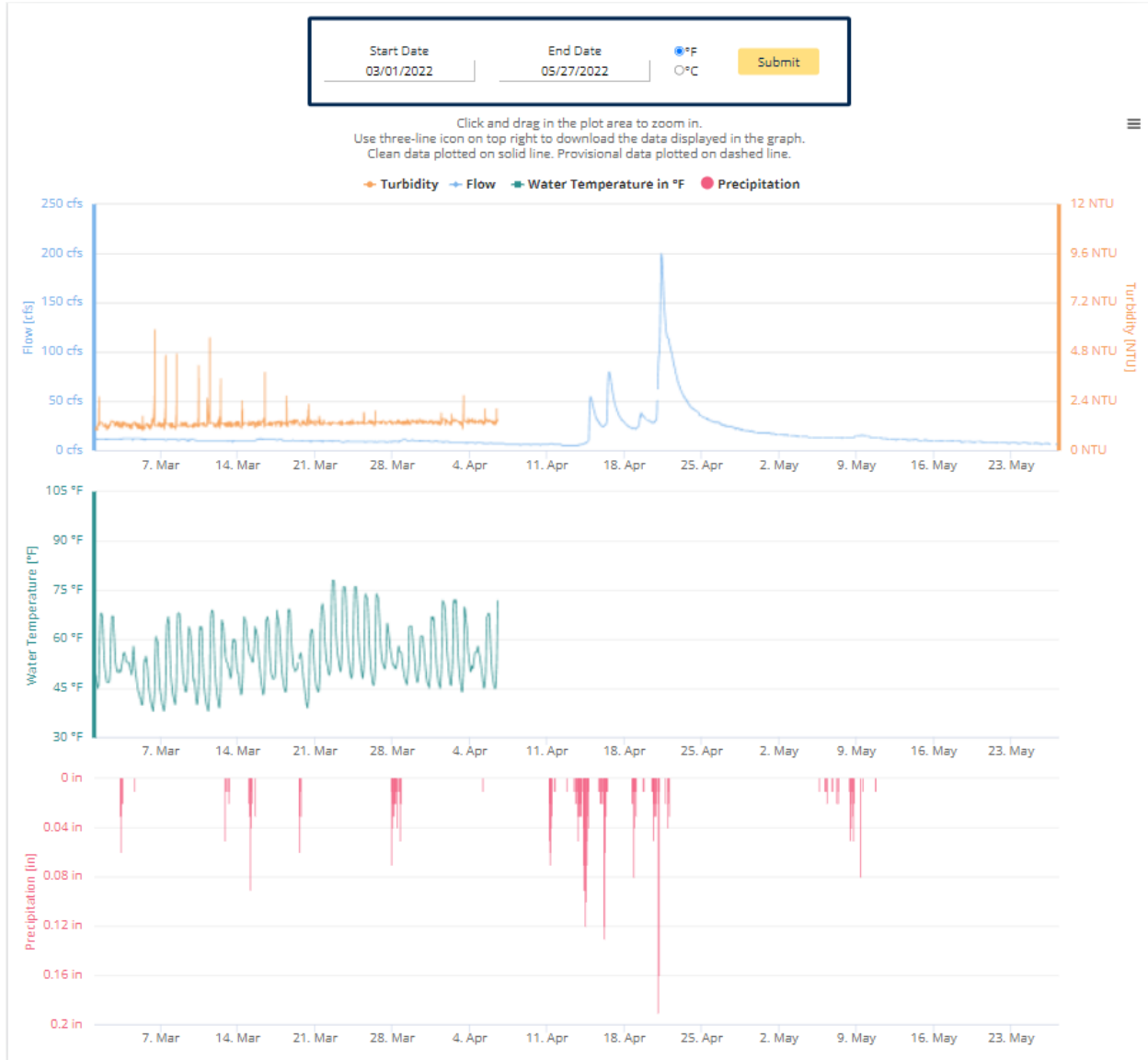
These data were collected and are currently being processed and analyzed by the UC Davis Tahoe Environmental Research Center (TERC). They are considered preliminary. Do not use or distribute without written permission from TERC.

For all questions please contact Dr. Shohei Watanabe (swatanabe@ucdavis.edu) or Dr. Alicia Cortes (alicortes@ucdavis.edu)

How to use the graphs and see the data below?

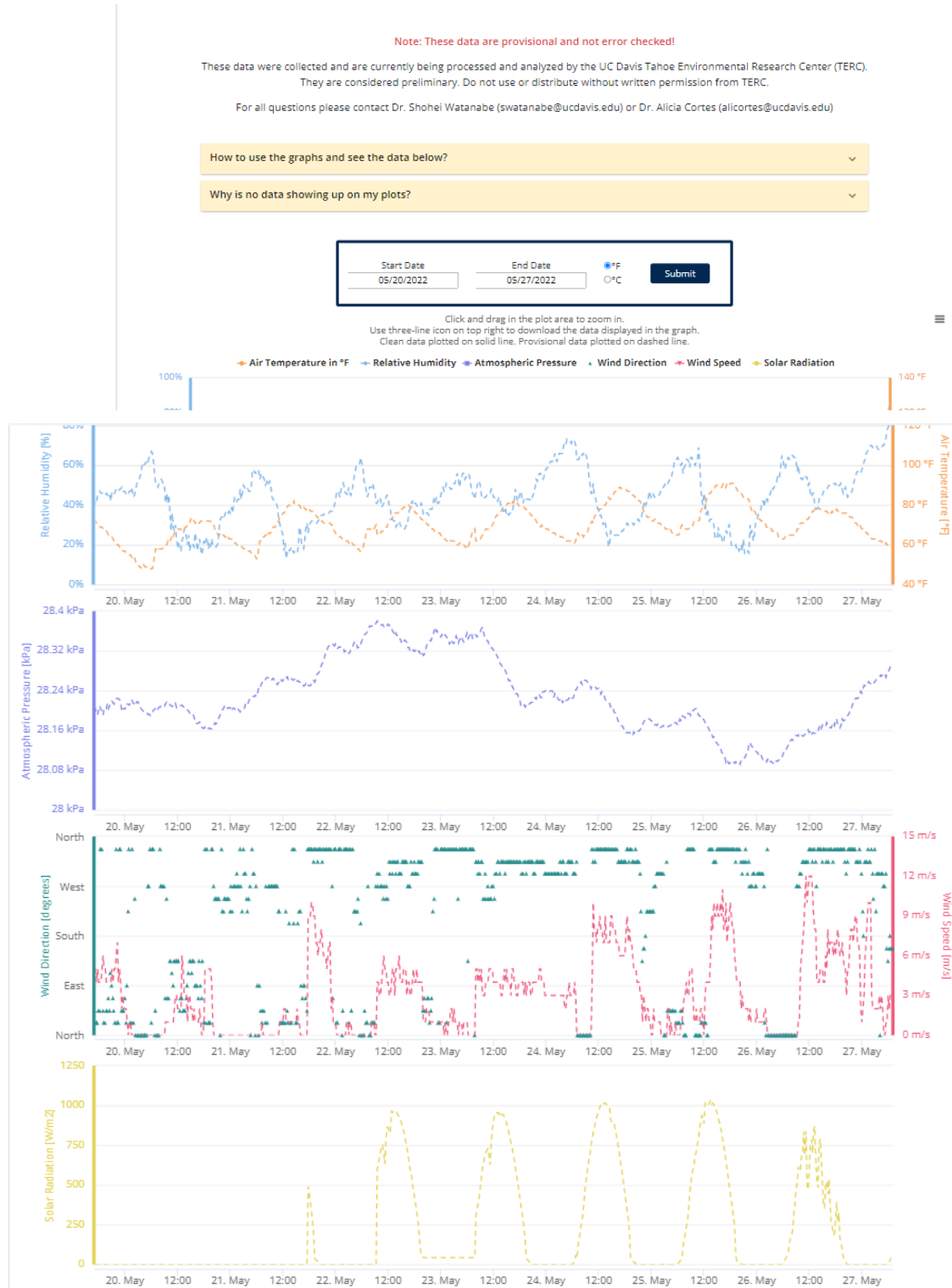
Why is no data showing up on my plots?

Where is the data collected?

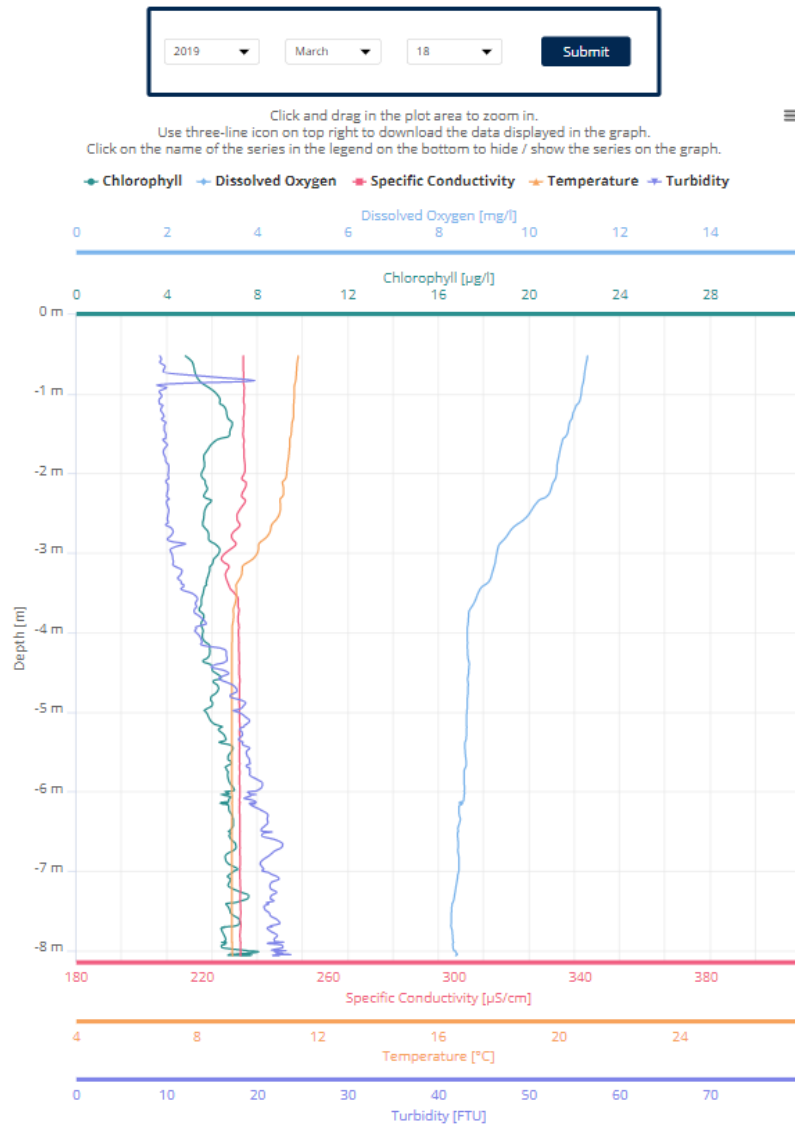


- Meteorology
 - Fahrenheit or Celsius option for air temperature.
 - The dashed line indicated real-time data

- Can be fetched at 3+ years. (Chart performance might lag to a large about of data being fetched)
- 1-week window is initially shown when page loads
- PST Time. Charts show 7 hours before the start date. This is because data fetched from the endpoint is



- Lake Profiles
 - Select only specific dates where data was collected



- Lake Moorings
 - 1-year window is initially shown
 - Displays temperature and dissolved oxygen heatmaps
 - White dots on the left indicate where the instrument that gathers the data is located.
 - The black line indicates the depth of the water column on the oxygen heatmap since oxygen data is not collected throughout the water column.
 - Interpolation to fill any gaps between data
 - PST Time. Charts show 7 hours before the start date. This is because data fetched from the endpoint is in UTC

Note: These data are provisional and not error checked!

These data were collected and are currently being processed and analyzed by the UC Davis Tahoe Environmental Research Center (TERC). They are considered preliminary. Do not use or distribute without written permission from TERC.

For all questions please contact Dr. Shohei Watanabe (swatanabe@ucdavis.edu) or Dr. Alicia Cortes (alicortes@ucdavis.edu)

How to use the graphs and see the data below?

Why is no data showing up on my plots?

Start Date

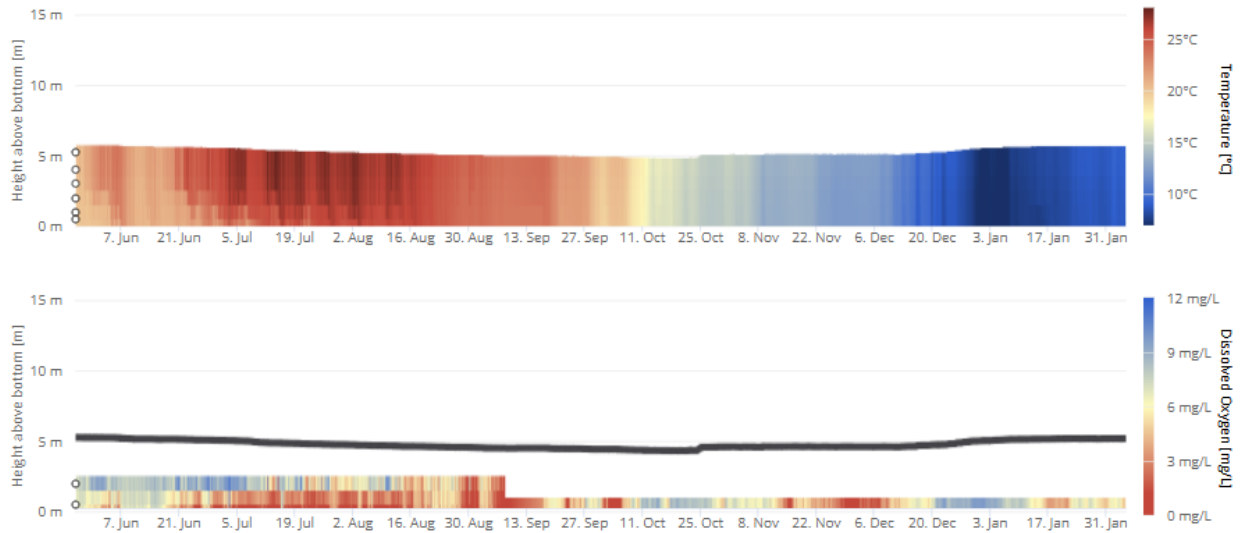
05/27/2021

End Date

05/27/2022

Submit

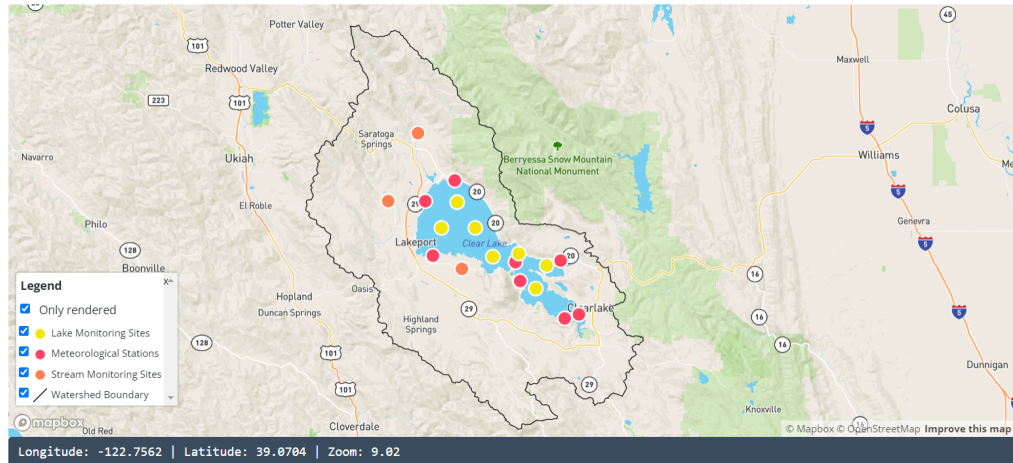
Click and drag in the plot area to zoom in.
Use three-line icon on top right to download the data displayed in the graph.
White Dots represent depth of the loggers. Black line is the depth of the water column



Map

- Shows the Clear Lake Watershed Boundary and data collection locations.


Click on the markers to see the name of the location. Click on the link in the pop up to be go to a page with the corresponding visualizations.



- Click on the markers to see the name of the location and a link to the data visualizations.


Weather Widget

- Shows the current weather at Clear Lake




Clear Lake Data

Tahoe Environmental Research Center

51°F 


[Home](#) [Stream](#) [Meteorology](#) [Lake Mooring](#) [Lake Profile](#) [Wind Animations](#) [Data Archive](#) [Main Clear Lake Site](#)

- By clicking on the widget, forecasted weather for the next four days will also be displayed.



Clear Lake Data

Tahoe Environmental Research Center

51°F


Clear Lake


Fri 27 May





51 °F

overcast clouds

Wind: 3 miles/h

Humidity: 58 %



<p>Sat 28 May</p>  <p>overcast clouds</p> <p>68 / 47 °F</p>	<p>Sun 29 May</p>  <p>clear sky</p> <p>73 / 41 °F</p>	<p>Mon 30 May</p>  <p>clear sky</p> <p>77 / 40 °F</p>	<p>Tue 31 May</p>  <p>clear sky</p> <p>86 / 50 °F</p>
--	--	--	--

[Home](#)
[Stream](#)
[Meteorology](#)
[Lake Mooring](#)
[Lake Profile](#)
[Wind Animations](#)
[Data Archive](#)
[Main Clear Lake Site](#)

Data Archive Page for Users

- Download data collected by the Clear Lake Research Team as CSV files.
- Download a corresponding readme for each data set.
- Select specific date ranges and variables and locations.

Stream Data
Meterology Data
CTD Data
TChain Data

Clean Data
Real-Time Data

Maximum 365 days at a time.

Location

Kelsey Creek

Station_ID

DateTime.UTC

Turb

Temp

Select

Start Date

05/20/2022

End Date

05/27/2022

Submit

There is no clean stream data from Fri May 20 2022 to Fri May 27 2022.

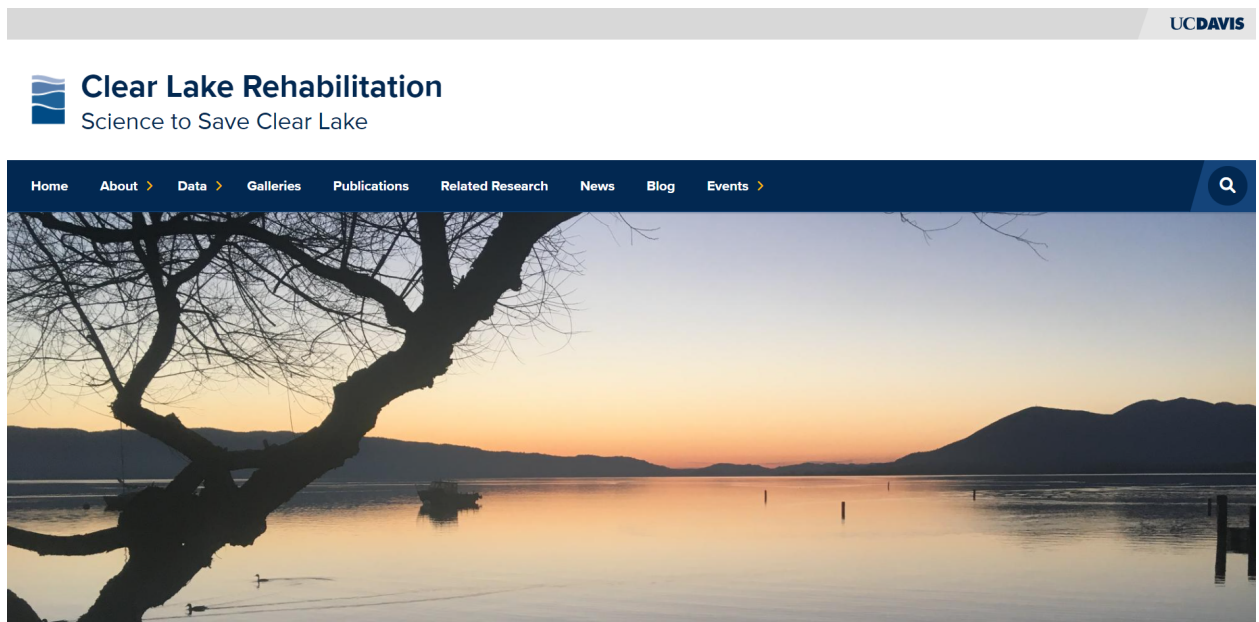
Data Upload Page for Client

- Allows the client to upload clean CSV data files to AWS S3 and MySQL WorkBench

- Allows up to 5 files to be uploaded at a time through an easy-to-use dropzone UI component
- Client follows naming convention to ensure the files are being added into the correct S3 bucket folders and MySQL tables
- Updates data from MySQL tables when user uploads data with the same date and time for a site

SiteFarm Site

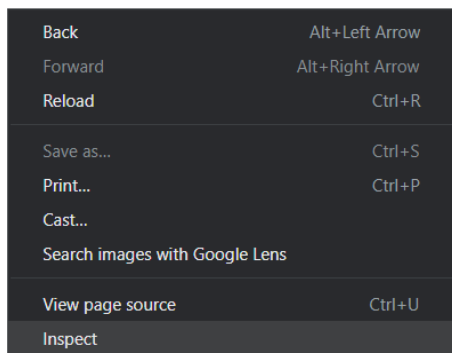
- Main Clear Lake website where information about the Clear Lake Research team can be found as well as photos, publications, and blogs.

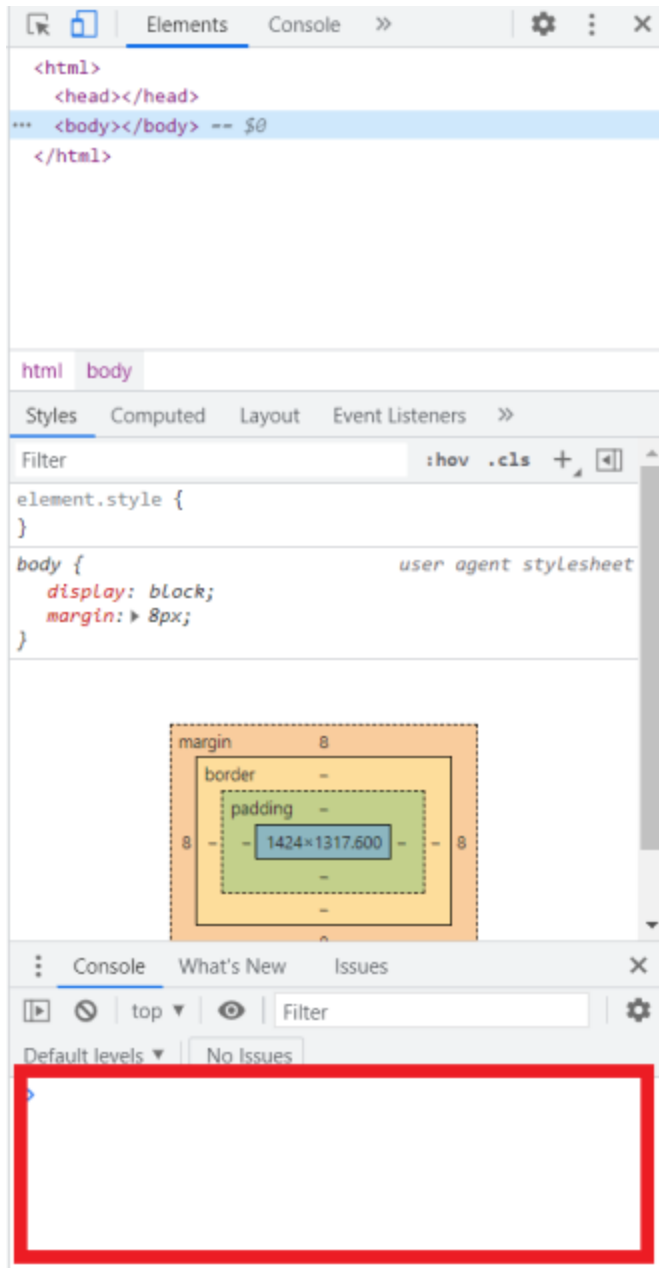


Troubleshooting

Q: There is nothing on the screen.

A: Right-click on the page and select the developer tools (Inspect). Read the errors in the console window circled in red in the image below.





Q: Module not found: Error: Can't resolve 'name-of-module' in '/path/to/file'

A: First try stopping and starting the local development server again. (Ctrl+C in the terminal to stop and `npm start` to start.) Otherwise, you might need to install a module. Google search: `npm name-of-module`. Typically it is something like: `npm install name-of-module`.

Q: "Internal Server Error" from API endpoints.

A: CORS is usually the issue. Troubleshooting is done through [enabling cors on aws api-gateway](#)

Frequently Asked Questions

Q: Where do I access the login for Site Farm?

A: To access login information, please contact Alicia Cortes at alicortes@ucdavis.edu

Q: Where do I access the login for the data upload page and MySQL tables?

A: To access login information, please contact Alicia Cortes at alicortes@ucdavis.edu

Q: Why are there two Clear Lake websites?

A: Since the SiteFarm site is easy to manage and edit, that is the main site that may hold more current news from the client. The React Data App is only used to visualize the data and allows for more interactive and dynamic abilities for the graphs.

Q: Where is the repository for the code?

A: This is the link for the React Clear Lake Data App, which also holds clean CSV files.
https://github.com/tercdev/Clear_Lake_Website_Data_Visualization (Location might change due to transfer of ownership with TERC)

For the backend code, please contact Alicia Cortes at alicortes@ucdavis.edu as it holds team confidential keys.

Q: Where is the server?

A: No backend server was developed. AWS services were used to make our application serverless. This was to make technical overhead simpler. This means data endpoints are accessible to the public, this should be fine since endpoints only fetch data.

Glossary

API: application programming interface allows two programs to talk to each other

AWS: Amazon Web Services (AWS) is the primary profit driver for Amazon. AWS provides servers, storage, networking, remote computing, email, mobile development, and security

Backend: Back end is the operations part of a business, where customers and members of the public rarely see or hear.

CSV file: comma-separated values file is a text file with a specific format

Endpoint: Access to data, usually in a link form. All endpoints are accessed through AWS API Gateway.

(<https://tepfsail50.execute-api.us-west-2.amazonaws.com/v1/report/metweatherlink?id=5&rptdate=20220518&rptend=20220525>)

Frontend: The frontend of a software program or website is everything with which the user interacts.

geoJSON: data format based on JSON format but for geographic data

Git: version control system

Github: website where code is shared between teammates

Highcharts: library used to create data visualizations

Lambda Function: [AWS service](#) that allows us to take advantage of serverless feature. No cost for server. Also allows useful things like scheduling

Mapbox: provides customizable maps for React projects

npm: package manager for Javascript

Javascript: programming language used to build most websites

JSON: javascript object notation is a data format

OpenWeatherMap: provides weather data

React: frontend JavaScript library for building what the user sees in the web browser

Contact Information

Alicia Cortes [TERC Clear Lake client]: alicortes@ucdavis.edu

Cesar Ayuso [Fullstack developer]: csayuso@ucdavis.edu

Valentina Lai [Fullstack developer]: vtlai@ucdavis.edu

Alison Chen [Frontend developer]: alychen@ucdavis.edu

Kenneth Lieu [Frontend developer]: kenlieu@ucdavis.edu

Appendix

Technology Survey

Frontend

SiteFarm	
Pros	Cons
<ul style="list-style-type: none">• Official UC Davis website<ul style="list-style-type: none">◦ Can include UC Davis departmental logos and contact information• Free to use• Service and hosting included	<ul style="list-style-type: none">• Embedded visualizations on page is not guaranteed mobile ready<ul style="list-style-type: none">◦ For example, on https://campusready.ucdavis.edu/testing-response/dashboards the Tableau table not mobile ready

<ul style="list-style-type: none"> • Includes different pre-configured page templates (basic pages, events, articles) • Easy to use and maintain for future developers (includes training and documentation) • Mobile ready • Includes level 1 (public) and level 2 (internal) information security • Client has other research members that are using SiteFarm and would like to keep the frontend consistent with team 	<ul style="list-style-type: none"> • Limited to SiteFarm page template design <ul style="list-style-type: none"> ◦ Not much room for creativity or self-creating features
---	--

React	
Pros	Cons
<ul style="list-style-type: none"> • More room for creativity and self-design • Can create more dynamic features using the state and lifecycle (i.e. carousel) 	<ul style="list-style-type: none"> • Need to manually do hostings <ul style="list-style-type: none"> ◦ To host site, there may a price depending on influx of users visiting site • Higher learning steep for future developers if wanted to redesign the site <ul style="list-style-type: none"> ◦ Not as maintainable • Would need to design using Figma • Data is owned by COE, restrictions on data access if we use a non UC Davis domain

Conclusion: Use SiteFarm as the general website for Clear Lake that is easy to update and React for displaying data visualizations.

Database

- **SQL Database**
 - Pros
 - Easy to learn and understand
 - Quick retrieval of large amounts of data
 - Can be ported to various other devices
 - Client already using SQL database to store data

- Cons
 - Not given complete control to the database
 - SQL Interface can be complex for some users
- **Amazon S3**
 - Pros
 - Easy to do backups
 - Easy to host static resources
 - Easy to transition ownership and organize
 - Cons
 - Can be complex to set up
- **Amazon Lambda**
 - Pros
 - Reduce the cost of execution
 - Only pay for the activity of code
 - Has improved application resilience
 - Allows packages to be added for code
 - Cons
 - Lack of control over the environment used
 - Complex call patterns
- **Amazon API Gateway**
 - Pros:
 - creates API endpoints to retrieve data from database tables
 - Flexible
 - Already used within TERC and CoE team
- **Amazon RDS**
 - Pros:
 - managed services that makes it simple to set up, operate, and scale databases in the cloud
 - Already used within TERC and CoE team
- **Amazon Event CloudWatch**
 - Pros:
 - to schedule events, such as automating data scraping from external websites

Conclusion: We will be using both SQL, Amazon S3, Amazon Lambda, Amazon API Gateway, Amazon RDS, and Amazon Event Cloud Watch for storing and managing the data, since the client is already using both of these services.

Data Visualization

Pros	Cons
<ul style="list-style-type: none"> • Brings data to life using HTML, CSS, SVG • Can read in data from .csv, .json, .tsv, .xml files • has a massive community of developers who improve the library regularly and it has a massive library of projects to learn it <ul style="list-style-type: none"> ◦ High active usage • Many graph types: https://www.d3-graph-gallery.com/index.html • Has brushing, tooltip, button, and zooming features 	<ul style="list-style-type: none"> • Steep learning curve → not maintainable for future developers when we finish the course • Old browsers not supported

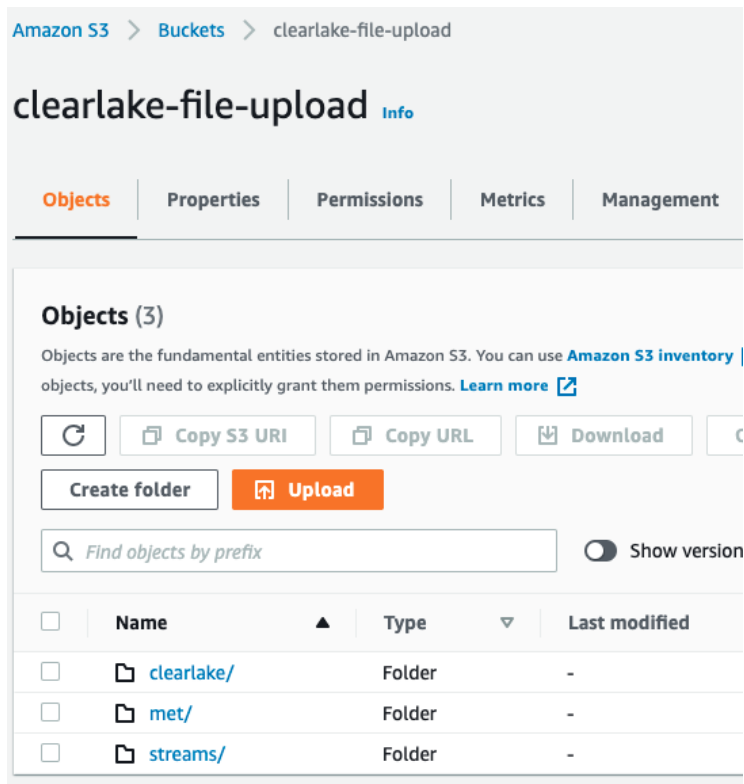
Highcharts	
Pros	Cons
<ul style="list-style-type: none"> • Lots of examples • Capability of handling large datasets • Easy configuration • Low learning curve and powerful tool 	<ul style="list-style-type: none"> • Not as easy to integrate

Conclusion: Use Highcharts for data visualizations so we won't have to build everything from scratch and Highcharts supports the line graphs and heatmaps we want to make.

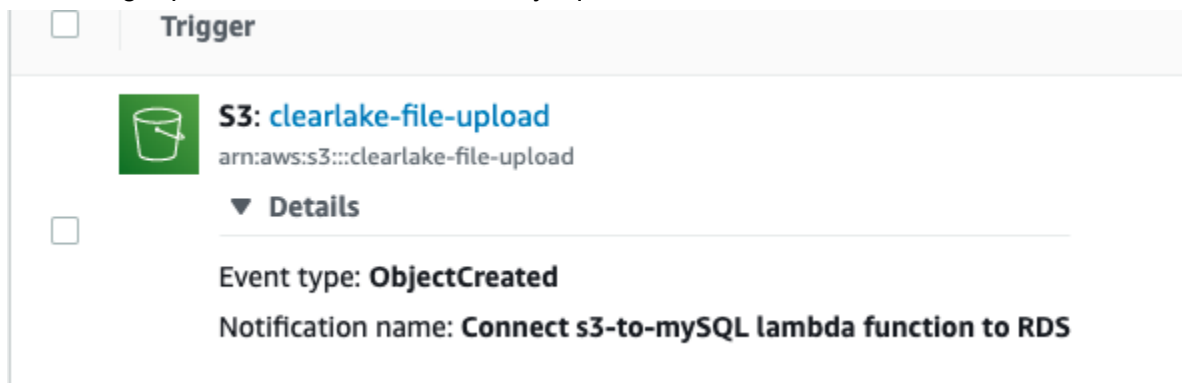
System Architecture Overview

Overview of our data processing architecture.

1. User uploads “clean” data to appropriate s3 bucket2.




2. [Lambda function that processes file contents to database](#) is triggered on file upload. File contents get placed into RDS instance MySQL database.



- Once file contents are in the database, Lambda functions to retrieve data have been set up. These lambda functions are triggered by API gateway.

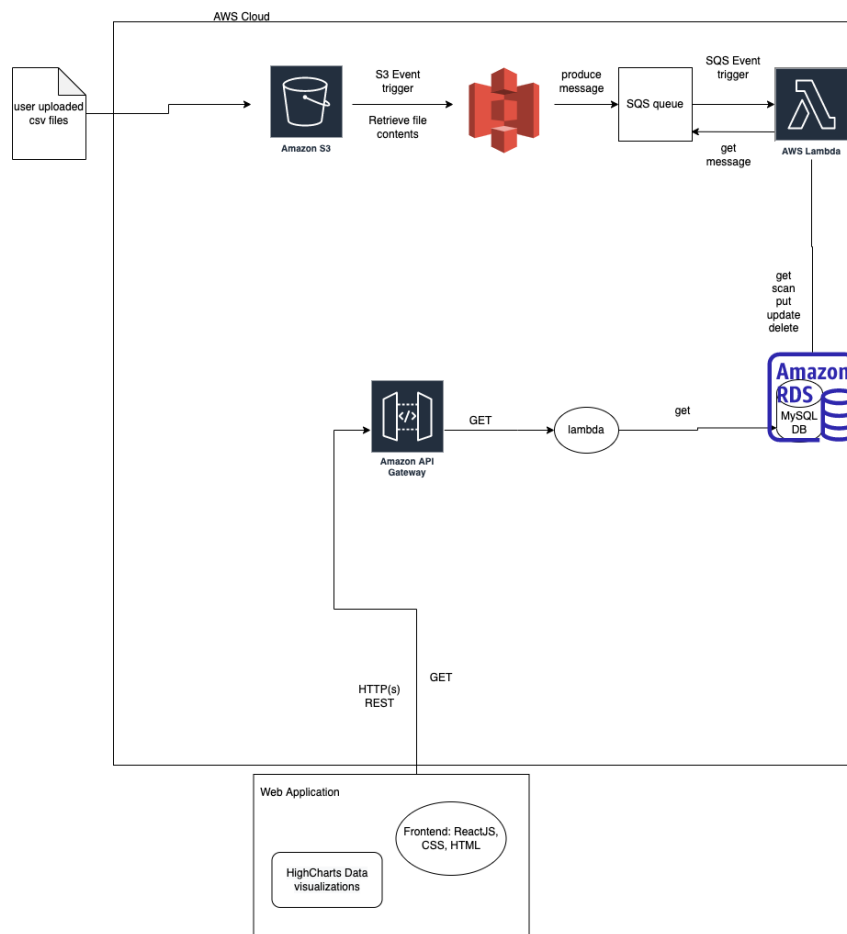
☐ Trigger

 **API Gateway:** [clearlake-streams-API](#)
arn:aws:execute-api:us-west-2:978477868465:b8xms0pkrf/*/*/*clearlake-streams
API endpoint: <https://b8xms0pkrf.execute-api.us-west-2.amazonaws.com/default/clearlake-streams>

▼ Details

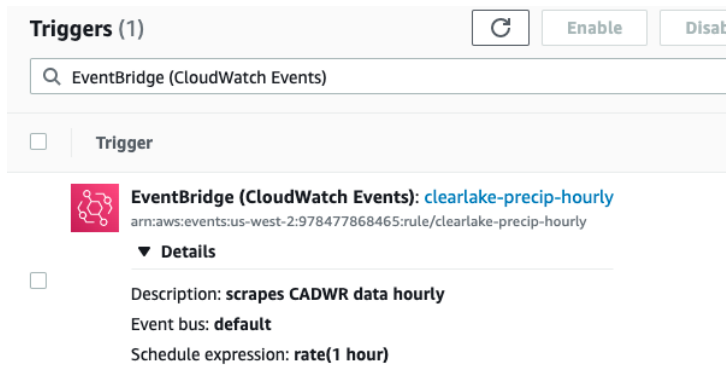
☐ API type: **REST**
Authorization: **NONE**
Method: **ANY**
Resource path: **/clearlake-streams**
Stage: **default**

- In the frontend code, we call the api gateway functions.
https://github.com/tercdev/Clear_Lake_Website_Data_Visualization/blob/a011d2ce6b06408ff60703975725f83c826cd8db/clear-lake-restoration-app/src/Components/pages/met/Met.js#L379
- The data is returned in JSON format. Processing is done to achieve data visualizations.



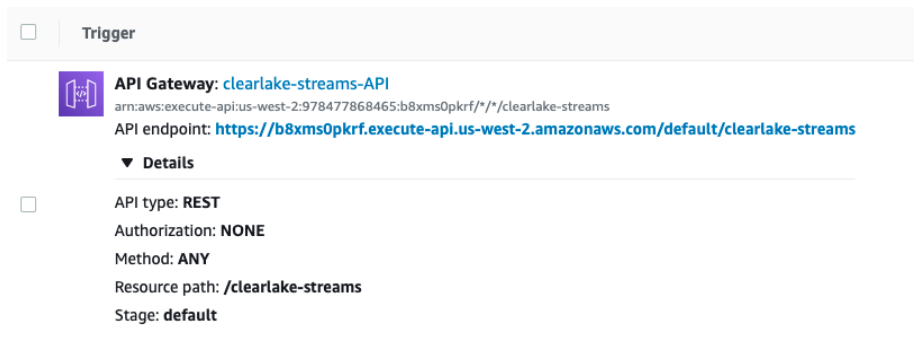
External Data Collection

1. External data is collected and scraped with pandas.html. This was because it has a handy [readhtml](#) function that processes any tables found on an html page.
2. Data is only added to a table if it is before the current time. The span of hours we scrape is 24. This is to account if the site goes down, data is changed.
3. Data is placed into the corresponding table in Database.
4. Lambda function is triggered with EventBridge (CloudWatch Events)



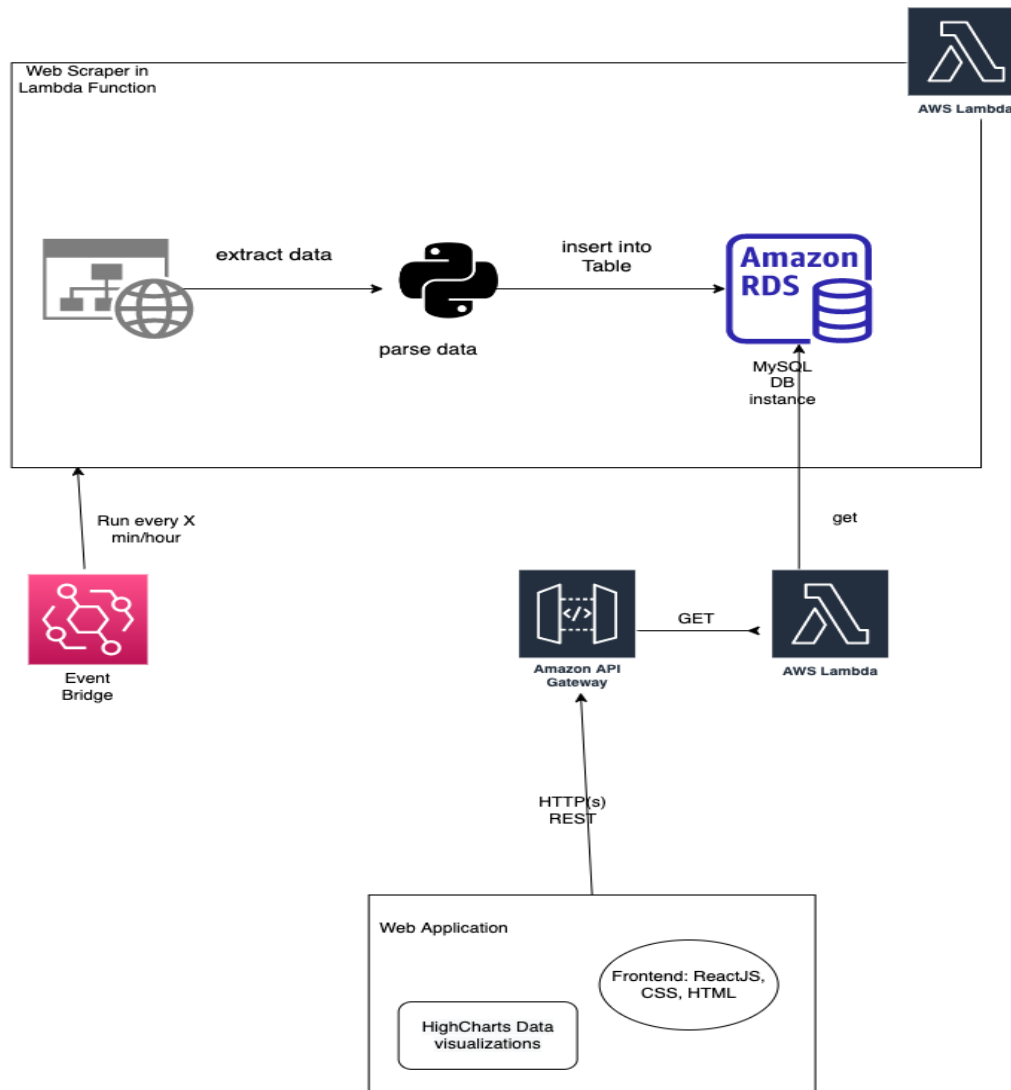
The screenshot shows the AWS EventBridge console. At the top, there's a section titled 'Triggers (1)' with a refresh icon, 'Enable', and 'Disable' buttons. Below this is a search bar containing 'EventBridge (CloudWatch Events)'. A table lists triggers, with one entry selected: 'EventBridge (CloudWatch Events): clearlake-precip-hourly'. The details for this trigger are expanded, showing: 'Description: scrapes CADWR data hourly', 'Event bus: default', and 'Schedule expression: rate(1 hour)'.

5. Once file contents are in the database, Lambda functions to retrieve data have been set up. These lambda functions are triggered by API gateway.



The screenshot shows the AWS API Gateway console. A table lists APIs, with one entry selected: 'API Gateway: clearlake-streams-API'. The details for this API are expanded, showing: 'API type: REST', 'Authorization: NONE', 'Method: ANY', 'Resource path: /clearlake-streams', and 'Stage: default'.

6. In the frontend code, we call the api gateway functions.
https://github.com/tercdev/Clear_Lake_Website_Data_Visualization/blob/a011d2ce6b06408ff60703975725f83c826cd8db/clear-lake-restoration-app/src/Components/pages/met/Met.js#L379
7. The data is returned in JSON format. Processing is done to achieve data visualizations.



User Stories

- As a local lake user, I want to be able to see useful data visualizations about the current state of the lake such as wind speed.
- As a web developer, I want to be able to understand how the data visualizations were done through well-documented code, so I can add more if needed.
- As a local/state government official, I want to know the current health and status of the lake and use the information to make decisions on the management of the lake.
- As a (web) site manager, I want to update the information on the website with recent publications, events, and blog posts.
- As a researcher, I want to be able to view long-term and short-term data through a flexible and interactive chart time selector to help me plan other potential parameters I will need to record and measure on the lake.

- As a lake user, I want to be able to know the oxygen levels of the lake during each season and relate that data to the health of the aquatic ecosystem, like algal blooms (HABs), the release of nutrients accumulated in the sediments, and the fish habitat
- As an environmentalist, I want to be able to know how to improve the long-term health of Clear Lake.
- As a data manager user, I want to see that the data used for data visualizations is safe and being accessed through well-documented APIs.
- As a lake stakeholder, I want to be able to download .csv files of the requested data to further study the data trends on my local computer.
- As a frontend developer, I want to know what extra features to implement to the website so I can best organize how they are displayed.

Prototyping Code

https://github.com/tercdev/Clear_Lake_Website_Data_Visualization

Technologies Employed

React, Highcharts, Amazon S3, Amazon Lambda Functions, Amazon API Gateway, Amazon RDS, Amazon Event CloudWatch, MySQL, SiteFarm, Github

Cost Analysis

SiteFarm is free to use. In addition, free tier Mapbox API and OpenWeatherMap API are used. Currently using the Amazon free-tier for RDS, Lambda functions, and S3. This is transferred over to the CoE department. The storage of all these tools is low, so there is little to no cost.

Legal and Social Aspect

The sites have a UC Davis domain, so this makes it clear that any data and information on the sites are owned by UC Davis. This helps protect their data and owns the copyright to their data.