BURNIN' UP TECHNICAL REPORT

PREPARED BY
Caitlin Lien
Caitlin O'Callaghan
Cherry Sun
Lauren Mangibin
Samantha Tuapen
Uma Sethuraman

OUR STORY

Earth’s climate is changing faster than ever. The emission of pollutants in the air can result in serious changes to the climate. These pollutants can be extremely dangerous with harmful effects for public health, ecosystems, and agricultural productivity. We want people to realize how serious the issue is by showing data that captures the impact of climate change around the world. We hope that our website shows a clear picture of how fast and how far the climate has changed around the world in the past decades and encourages people to take action to help the environment.

1. Project Overview

This website aims to educate people on the climate crisis of our planet, and make them aware of how quickly our home is changing. This website will allow the user to navigate from city to city, or country to country, to see how each city or country is contributing to, or has been affected by climate change.Users can also see how climate change has been affecting the world on a year by year basis. We hope to encourage our users to take action by making small changes in their lives to decrease their individual carbon footprint, such as turning off lights they aren’t using, or carpooling with others when they can.

2. User Stories

Phase I User Stories

- Look up air quality data for a city: As an environmental activist, I'd like to know the air quality of various cities around the world to make a case for reducing carbon footprint.
- Observe historical trends within global climate change: As an environmental science professor, I want to know the level of greenhouse gases globally during any given year to teach my students about historical trends in global climate change.
- Suggest related locations for comparing climate trends: As a concerned resident of an urban area, I'd like to know about the air quality level in my city and other cities in my country. For instance, when I'm looking up the air quality level in my city, I'd like

to view several related locations so I can compare my situation to residents of those areas.

- Examine the relationship between air quality and economics: As an environmental researcher, I'd like to view the air quality as well as information about the economic situation (GDP, electricity, etc.) of a given country to investigate relationships between these factors.
- Examine the relationship between air quality and demographics As a sociologist, I'd like to know the air quality and demographic information (population, etc.) in a given city to understand how these factors are related.

Phase II User Stories

- Shift location of countries table: I'm an environmental activist returning to use your website. I clicked on the link for the main page for countries and I could only see the global map. Although the map is cool, I would prefer to see the country table at the top of the page so I can find it immediately!
- Add units of measurement for categories of information: I'm a resident of an urban area and a returning user to your website. I find it slightly difficult to understand what some of the values in your tables for your models mean. Please add units of measurement on your columns to help understanding!
- Adapt website to look good on mobile devices: I'm an employee of a company that makes electrical cars, and I was trying to use your website on my phone but I found that the layout is very disorganized. I would appreciate it if you made the website adaptable to different types of devices, no matter the size or shape of the screen.
- Add spacing around elements: I'm an environmental science professor using your website and thought it would look nicer if all the elements had a bit more spacing around them (bootstrap classes m-x and p-x) rather than everything being tightly packed together (Example: city, county, and climate change filters).
- Improve Filtering Options: I'm an environmental researcher. While using your website, I thought it would be helpful if you added some kind of improvement to filtering options (ex. custom value entry, sort the timezones dropdown correctly).

Phase III User Stories

- Add links where users can donate: I'm an environmental researcher. It would be really nice if you provided a link on your website to some way that the users can donate or help (can be a link to another general site that gives them the resources needed)

- Countries page console issues: I'm an environmental science professor. While using your website, specifically the Countries model page, I found a leftover console.log which pollutes the console. It can be seen in Countries.tsx, line 47
- Model pages have difficulty loading: I'm the environmental activist! I was on your model pages which seemed to be working when I clicked on them, but after staying on each page a bit it seems the requests to your database fails. If it doesn't fail, it just takes a really long time to load.
- Map not working properly: I'm a concerned resident of an urban area. It seems like the map on the countries page is not working properly (example: Germany for us dropped a pin right in the middle of the ocean https://burninup.me/countries/id=49)
- Images on the about page: I'm the sociologist! I clicked on the about page but for some reason the pictures of the developers did not show up. It would be great to see who made this awesome website.

Phase I Customer Stories

- Make attributes more readable by adding bullet points on each energy sources page
- Center photos on the Energy Sources Total Page
- On home page, center or make photos the same dimension for neatness
- Link from Energy Sources to Fueling Stations: I would like to be able to click on each station in "Charging stations that use/sell this energy" to take me to the specific page to get more information
- Make the "Website: Econyoom.me" footer consistent: Styling should be consistent throughout the pages. It switches between a blue box and no box.

Phase II Customer Stories

- Add a search and filter bar to Vehicles/Fueling Stations: As someone who is looking for an energy-efficient car, it would be nice to be able to sort by different preferences to find different models and fuel locations. Currently, there is no available option to sort. I would like to sort by annual fuel cost, CO2 emissions, and savings.
- Modify Energy Sources page to not have a dropdown and instead have a general page with links to instance pages: I am researching vehicle energy consumption. The dropdown menu doesn't seem to be the best way to display all of the different

energy sources especially if I want to click on one way down in the list. I think it would be much easier and convenient to have a common page which gives me an overview of the types of energy sources at once.

- Embed the map on each fueling station instance page: As a user who is trying to compare the locations of different service stations near me, I think it would be nice to see the map of where each service center is. It would be better for quick visual comparisons. You could possibly make the link to the map bigger and more visible but it would be more engaging to have a map.

- For the Home page, make the purpose of the site more obvious: As a first time user, I feel a little lost as to what the website is for. I think emphasizing the page's purpose on the home page would help me understand more. This could be put possibly after the carousel or on the carousel with informational photos,

- Add dividers and links to the API's/Datasets you used: As someone who is doing research on energy-efficient vehicles, I would like to see more clearly the sources that you used to populate your data. Adding in clearer dividers as well as specific links to the APIs and datasets you use would help me get more information. This can be done in the about page, and maybe in the models general page.

Phase III Customer Stories

- Add more color to the site: As a visual learner, I would like to see more color on the website. I think it would look best if you had a colored header for the tables or other places you think would highlight the information better. The white looks nice, but it could be enhanced.
- Change connection to Fuel Stations: As a local looking for cars, I'm trying to understand the connections between the fueling stations and vehicles. Looking at the website, are the fueling stations just 5 random stations or do they relate to the car itself? I would make this clarification clear.
- Fix capitalization: As a car enthusiast with an eye for consistency, I think the "EcoNyoom ecological vehicles" label at the top should be capitalized properly or you can change it with a small logo. Also, the "Wood and wood derived fuels" should be capitalized as well and have a dash between 'wood-derived'.
- Replace -1s with a different naming: As an investor on your site, I love the sleek look. To make it better, try replacing the -1s with N/A to show that the information is not available. It makes the information nicer to look at.
- Add Loading Icon: As a researcher, I look at the page and wonder if they don't have data or if the site is taking a while to load. If you could add in a loading page or icons to all the pages to indicate that information is on its way, that would be very helpful! It would also increase website retention.

3. RESTful API

We used Postman to design our API. Our API documentation can be found here.

Countries API Endpoints

- **GET all countries**
  burninup.me/api/countries
  Returns a list of all the countries in our database with basic information on the country, such as capital city, country income, latitude and longitude, region, and the most recently recorded carbon emissions for that country.

- **GET countries by country id**
  burninup.me/api/countries/id=<id>
  Returns the information of a country given the country id, such as capital city, country income, latitude and longitude, region, and the most recently recorded carbon emissions.

Cities API Endpoints

- **GET all cities**
  burninup.me/api/cities
  Returns a list of all the cities in our database with basic information on the city, such as population, latitude and longitude, and country code, as well as climate data such as carbon monoxide level, ozone concentration, and particulate matter of 10 and 25 microns in diameter.

- **GET city by city id**
  burninup.me/api/cities/id=<id>
  Returns the information of a city given the city id, such as population, latitude and longitude, and country code, as well as climate data such as carbon monoxide level, ozone concentration, and particulate matter of 10 and 25 microns in diameter.

Years API Endpoints

- **GET all years**
  burninup.me/api/years
  Returns a list of all the years in our database. This information includes yearly statistics such as the temperature anomaly, carbon dioxide level, polar ice extent, and world population of that year.
- **GET year by year id**
  burninup.me/api/years/id=<id>
  Returns the information of a yeary given the year id. This information includes statistics such as the temperature anomaly, carbon dioxide level, polar ice extent, and world population of that year.

4. Models

Countries

- Name
- List of Cities
- Population
- Flag
- Climate (Air Quality)
- Income level
- Region
- Subregion

Cities

- Name
- Population
- Country
- Location
- Climate (Air Quality)
- Historical Weather

Global Climate Change by Year

- Year
- Global Mean surface temperature anomaly
- Global Temperature
- Carbon Dioxide Levels
- Methane

- Nitrous Oxide
- Top 10 highest temperature cities for each year
- Top 10 highest co2 emission countries for each year
- Ice Extent

Filterable/Sortable Attributes:

Countries: name, primary language, region, currency, population

Cities: name, country, time zone, location, population

Global Climate Change: carbon dioxide levels, global temperature, years, nitrous oxide levels, months

Searchable Attributes:

Countries: GDP, subregion, list of cities, flag, air quality

Cities: zip codes, elevation, location, air quality, historical weather

Global Climate Change: UV indices, world population, polar ice, decade, century, sea level

Media:

Countries: photos of flags, graphs, tables, maps

Cities: pictures, descriptions, tables, maps

Global Climate Change: tables, maps

Connections:

Country: Connects to city because countries have cities, and the air quality in cities contributes to air quality in a country. Connects to global climate change because the quality of air depends on carbon dioxide emissions globally.

City: Connects to countries because cities are in countries and contribute to the air quality of a country. Connects to global climate change because the air quality of a city depends on carbon dioxide emissions and can result in increasing climate change globally.

Global Climate Change: Connects to countries because air quality in countries affects global climate change. Connects to cities because air quality in cities affects carbon emissions globally.

5. Phase II Features

Pagination:

We implemented pagination by following a tutorial listed here. We were able to make a pagination class that divided the data and had a navbar you could click on to show the different instances. We also made 3 different 'posts' classes, one per model, that had all the data for each instance. Then using the pagination class we created, we divided the different instances into sections and displayed them in the navbar.

In Phase 3, we switched to a Material UI data table which has pagination automatically implemented for us. All we needed to do was populate the table properly. We then removed the pagination and posts classes we developed in phase 2.

Database:

We use the AWS Relational Database Service, which utilizes PostgreSQL to store our data in a database on the cloud. The open source tool we chose to connect to our PostgreSQL DB instance is pgAdmin. Through pgAdmin, we were able to access and modify our data. We specify our AWS RDS endpoint under the pgAdmin host and connect to our database by providing our username and password. In our Flask script, we provided the AWS RDS endpoint under our app configuration. We are able to populate, query, and filter our database using SQLAlchemy and provide an organized schema for all the tables inside our database using Flask_Marshmallow. These schemas allow us to format our data for our API routes.

6. Phase III Features

Sorting:

Using the Material UI data table, sorting was already a feature that could simply be turned on and off for each attribute in the table. For each attribute in the `options` section, we added a "`sort: true` " option. With this option, the column was now clickable and sortable by alphabetical order if it was of type string or by increasing/decreasing order if it was of type number.

Filtering:

Using the Material UI data table, filtering was also a feature that could be turned on and off for each attribute. The default setting groups all the unique values and displays them as a filter option, but because we wanted those unique values to be grouped, we had to add `filterOptions` to the `options` section. In here, we were able to name each group and add in logic that filtered and grouped the attributes if they satisfied the condition.

Searching:

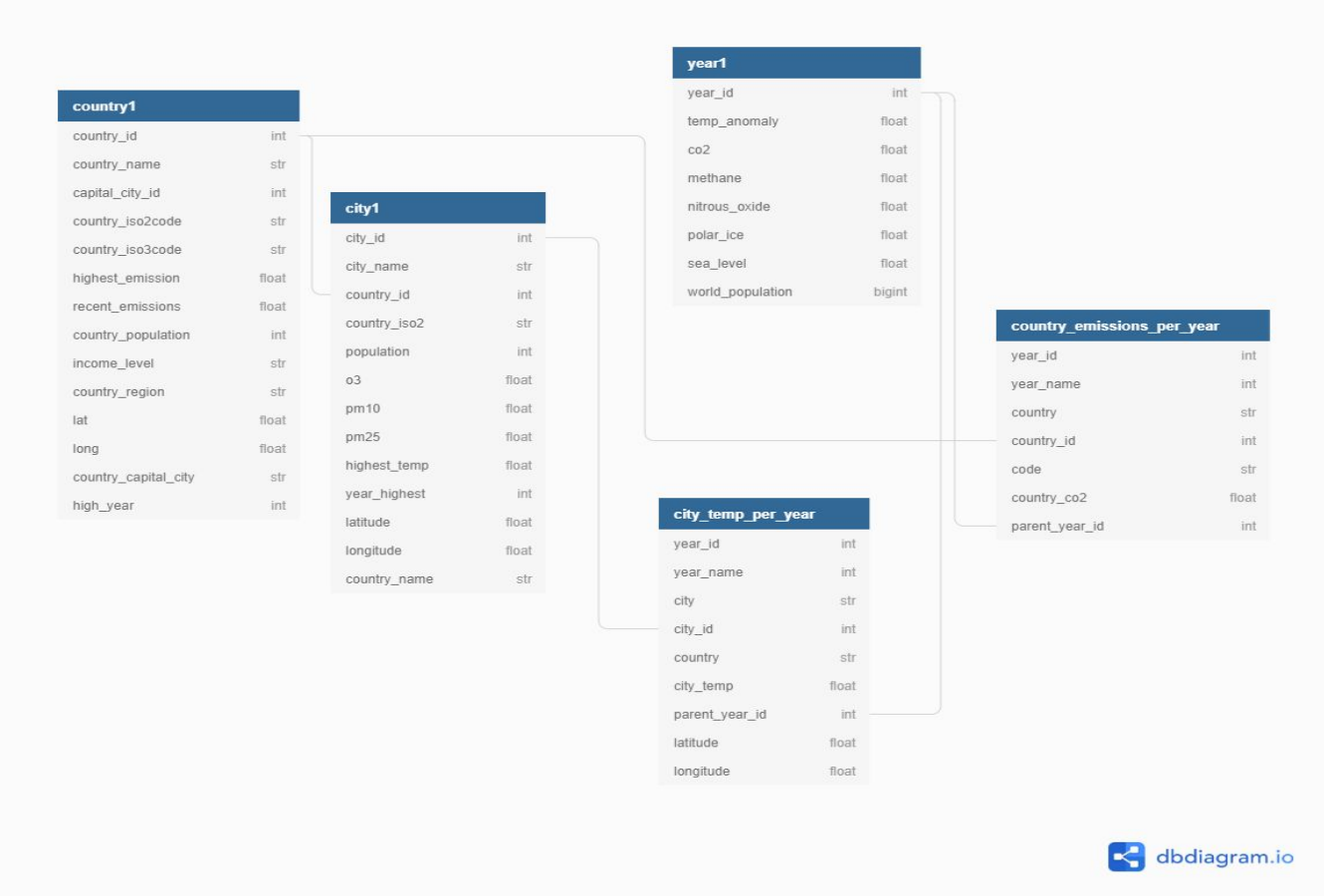
Using the Material UI data table, searching was automatically implemented. You can search anything on the data table in the model page and it will winnow the rows to match the search criteria.

To search the whole site, we used Algolia, a site search product that works well even with the free tier. You can search with multiple terms and highlight your results. In Algolia, you are able to put in the JSON responses from your API and it will automatically create 'searchable attributes' that you can enable or disable searching on.

For each model, we were able to take the JSON response from the model's API request, put it into an 'Index' (the model's responses) in Algolia, and add the list of attributes that will be used for searching on the site's UI. In the code (Search.tsx), we add in a Hit component for each model that displays and highlights the search results from the query.

The user makes a search with the search bar placed in the Navbar. We then take the query, pass it to our Search component, and display the results. We put the search results on a different search page, separated by the Countries, Cities, and Years model. If there are no results, the search page will display the Countries, Cities, and Years headers with no content beneath it.

7. Database



We designed our DB based on each of our models: Years, Countries, and Cities. For the Years model, we used the year_id, which doubled as the name of the year, as our primary key. We also have foreign keys linking our CountryEmissionsPerYear and CityTemperaturePerYear back to our Years model. CountryEmissionsPerYear acts as our link between the Countries and Years models, because for each year, we have information on the Top 10 Countries with the Highest Carbon Emissions. On the other hand, CityTemperaturesPerYear acts as our link between the Cities and Years model, because for each year, we have information on the Top 10 Cities with the Highest Temperatures. The parent_year_id foreign key links CountryEmissionsPerYear and CityTemperaturePerYear back to their respective year, and the country_id/city_id links them back to their respective country/city. For our City model, we use the city_id as our primary key, and for our Country model, we use the country_id as our primary key. To link our City table back to our Country table, we use the country_id as a foreign key. We also have a capital_city_id in our Country table that keeps track of the city_if of their respective capital city.

8. Tools

Backend

- Elastic Beanstalk: Used to host backend API.
- AWS Cloudfront: Used to deploy our website.
- AWS RDS: Used to host the tables for our models.
- Postman: Used to test and document our API.
- NameCheap: Used to get a free domain name.

Frontend

- React: Used to render website and connect the user interface to the backend.
- React Bootstrap: The CSS framework used for the website.
- Selenium: Used to test GUI of website.
- Jest: Used to test React and Typescript components and functions.
- AWS S3: Used to host the static website.
- Algolia: Used to search the whole website

Data

- Google Places API: Used to get images of the cities and countries. https://developers.google.com/places/web-service/photos
- Google Geo Location API: Used to map cities and countries. https://developers.google.com/maps/documentation/javascript/overview
- World Bank API: Used to get general information on countries. https://datahelpdesk.worldbank.org/knowledgebase/articles/898599-indicator-api-queries
- OpenDataSoft WorldCitiesPop API: Used to get the general information of cities. https://public.opendatasoft.com/explore/dataset/worldcitiespop/api/?disjunctive.country&sort=population
- Google Geocoding API: Used to get latitude and longitude of cities. https://developers.google.com/maps/documentation/geocoding/overview
- Air Quality Programmatic API: Used to get air quality data on different cities. https://aqicn.org/api/
- Global Warming API: Used to get climate data for temperature anomalies. https://global-warming.org/
- Carbon Emissions Dataset: Used to get global carbon emissions from 1880-2019. https://data.giss.nasa.gov/modelforce/ghgases/Fig1A.ext.txt ftp://aftp.cmdl.noaa.gov/products/trends/co2/co2_annmean_mlo.txt
- Methane Dataset: Used to get global methane levels from 1880-2019. ftp://aftp.cmdl.noaa.gov/products/trends/ch4/ch4_annmean_gl.txt

- Nitrous Oxide Dataset: Used to get global nitrous oxide levels from 1880-2019.
  ftp://ftp.ncdc.noaa.gov/pub/data/paleo/icecore/antarctica/law/law2006.txt
  https://www.epa.gov/sites/production/files/2016-08/ghg-concentrations_fig-3.csv
- Polar Ice Dataset: Used to get polar ice extent from 1880-2019.
  https://nsidc.org/data/g10010
- Sea Level Dataset: Used to get global sea levels from 1800-2018.
  https://www.jpl.nasa.gov/edu/teach/activity/graphing-sea-level-trends/
- World Population Dataset: Used to get the world population from 1800-2019.
  https://ourworldindata.org/world-population-growth

6. Hosting

We got the domain name from NameCheap, and then used a Custom DNS to connect our NameCheap domain to the nameservers provided to us by Route 53 on AWS. We hosted our files on AWS using a S3 bucket, and pushed our docker image to an AWS Elastic Beanstalk instance so we could run our code. We then routed our frontend to our backend using AWS CloudFront, which connects our NameCheap domain, S3 bucket, and Elastic Beanstalk instance.

7. Gitlab

https://gitlab.com/caitlinlien/cs373-sustainability