UNIVERSITY OF EXETER

Computer Science Department

M.Sci. Computer Science
Computer Science Department

Exploring the World Through Dynamic Web Maps: Natural Disasters

Candidate

Pedro Miguel Catarino da Silva Pereira

Student ID 700044334

Supervisor

Dr. Chunbo Luo

University of Exeter

Academic Year
2022/2023

Abstract

Climate change is increasing the frequency and severity of natural disasters all around the world, leading to an urgent need for access to reliable information about these hazards. It is an issue that is impacting a growing amount of people. This project aims to create a dynamic website that provides key information about real-time and past disasters. It will require proficiency in web development technologies, in particular web mapping, database theory and design and visual style of a web page. The goal is to develop a website that is user-friendly, visually beautiful, and instructive, and that shows geographic data regarding natural catastrophes in a secure and accessible manner so that it can aid future humanitarian and catastrophe relief operations.

	Yes	No
I certify that all material in this dissertation which is not my own work has been identified.	☑	☐
I give the permission to the Department of Computer Science of the University of Exeter to include this manuscript in the institutional repository, exclusively for academic purposes.	☑	☐

Contents

List of Figures

List of Tables

List of Algorithms

List of Code Snippets

List of Acronyms

HTML HyperText Markup Language

CSS Cascading Style Sheets

JS JavaScript

PHP Hypertext Preprocessor

SQL Structured Query Language

CSV Comma Separated Values

GIS Geographic Information System

PC Personal Computer

GPS Global Positioning System

JSON JavaScript Object Notation

API Application Programming interface

AJAX Asynchronous JavaScript And XML

XAMPP Cross-platform, Apache, MySQL, PHP and Perl

HTTP Hypertext Transfer Protocol

CRED Centre for Research on the Epidemiology of Disaster

OCHA United Nations Office for the Coordination of Humanitarian Affairs

EM-DAT Emergency Events Database

WHO World Health Organization

UN United Nations

MySQL My Structured Query Language

1 Introduction

1.1 Background and Motivation

Over the course of human history, maps have been an essential tool in the advancement of our of society. They provide a visual depiction of geographic information that can be paired with anything from maritime routes to economic data to produce a simple yet informative educational tool.

Web maps first emerged as specialised software. These Geographic Information System (GIS) based applications required expertise to create, manage, analyse and maintain thus making them a tool exclusive to geographers and cartographers. A modern example of one of these applications is arcGIS, which is commonly utilised in university degrees such as geography [9].

Recent breakthroughs in mapping technology have led to a widespread use of maps both by cartography experts as well as the general public [16]. In particular, web maps have become a common tool in everyday life with applications such as Google Maps [12] and Flight Radar. Google Maps, a cutting-edge platform with real-time Global Positioning System (GPS) navigation and coverage of more than 220 nations and territories, is one of the most innovative web maps used by more than 154 million people. A significant advantage of these modern web maps is their interactive feature which allows users to, zoom in and out, pan across different areas, and, most importantly, click on individual features to access additional information or data.

Climate-related disasters, including but not limited to earthquakes, floods, droughts, heat waves and storms have recorded a global increase in not only frequency, but also intensity. [14]. Numerous studies suggest these events will continue to intensify under rising global temperatures and thus cause substantial repercussions for people, infrastructure, and natural ecosystems [7]. As sea levels continue to rise as a direct consequence of climate change, more water is expected to be pushed inland during hurricane seasons and hence increase risks of coastal flooding and erosion [10]. In addition to environmental damage, food and water scarcity, and other social and economic effects, these catastrophes are also causing displacement. In

order to mitigate the effects of climatic disasters and ensure the resilience of communities and ecosystems in the face of future difficulties, addressing climate change and lowering greenhouse gas emissions is essential to a sustainable future.

Modern-day climate change and the inevitable increase in natural disasters [2] places significant implications on technological advancements such as web maps as a means of visualizing disasters on a regional and global scale. Web maps can aid in increasing awareness and organising public support for disaster relief initiatives by providing real-time information on the magnitude of a disaster. This could encourage people to take action, whether it be through volunteering, donating resources, or advocating for policy change.

Since the 1960s, the number of natural disasters in the world has multiplied by ten, making it imperative to enable global access to disaster information for both educational and risk-reduction purposes. [6]. This access would advance the efforts of climate action which has, considering its urgency and relevance, been at the centerpiece of contemporary debates [9]. This further emphasises the importance of a simple and informative web maps on natural disasters.

1.2 Project Aims and objectives

The aim of this project is to convene the extent and severity of natural disasters through a dynamic web map that utilises interactive features. It displays geographic, demographic and economic information on both a real-time and past time scale.

The importance of the project centers on displaying this information in the form of a web map. Its simplicity and easy readability will allow users to become aware of patterns and trends of natural disasters. This may include where disasters affect populations the most and where disaster relief actions are needed. Furthermore, it will help direct people towards how they can support those affected by natural hazards be it through donating, volunteering, or other means. The availability of a web map like this to any internet-enabled device ensures that it reaches a global populace as well as organizations that may choose to use the data in their own natural disaster relief operations.

2

Project Specification

In this section, the Functional and Non-Functional Requirements of the project will be described, followed by the evaluation criteria and the ethical issues.

2.1 Functional Requirements

The functional requirements are the objectives the project has to achieve to consider it a successful finished product.

1. The website must work in both computers and mobile devices, be responsive.
2. The website must retrieve data about natural disasters from the *reliefweb* Application Programming interface (API) every 90 seconds. This time is picked because the reliefweb API allows for 1000 calls day, meaning a maximum of 1 call every 86.4 seconds, so given this limitation and the objective of relaying as up to date information to the user as possible 90 seconds was chosen.
3. Data is retrieved in an asynchronous manner through the use of Asynchronous JavaScript And XML (AJAX) or Hypertext Preprocessor (PHP), wherever each one is applicable.
4. The web map must display the entire world and the border between countries.
5. The web map must allow the user to zoom in and out on the map.
6. The web map will be able to display information in 4 different categories of which multiple parameters can be displayed. *

   Demographic information:

   - Occurrences of natural disasters.
   - Death toll of natural disasters.
   - Number of affected by natural disasters.

   Economic information:

   - The amount of damage caused by natural disasters.

   Geographic information:

- Natural disasters by country.

Time span information:

- Select span of years to visualise natural disasters.
- Select real time visualisation of natural disasters.

Disasters information:

- Select type of natural disasters.
- Select sub type of natural disasters.

7. The user has to be able to change the information displayed as per these previous categories and parameters.

*These parameters may be expanded during development if more parameters reveal themselves to be useful.

2.2 Non-Functional Requirements

The non-functional requirements are the features that would be positive additions to the project but not exactly "must haves".

1. The website is visually pleasing.
2. The website loads within 10 seconds.
3. The website sends an email to users who want alerts about new disasters.
4. The web map has low latency in its interactions.
5. The website allows the user to pick what colours to paint the countries with.
6. The web map shows the exact point the natural disaster happened.
7. The web map has an icon with a small image depicting a type of disaster.

2.3 Evaluation Criteria

Functional Requirements

- To test requirement number 1 Chrome's developer tools will be used since it has a feature that allows to see how your page would look when being accessed from another device.
- To test requirements number 2 and 6 at first the Postman software will be used to test out the *relief web* API to test it out before implementing it in AJAX.
- To test requirements 4, 5 and 7 the web map will be first developed on its own and its features rigorously tested following *leaflet* documentation before using the natural disasters data. Only after this is achieved will the natural disasters data be combined with the web map.

Non-Functional Requirements

- To test requirement number 1 I will conduct a small survey amongst people I know about their opinion on the attractiveness of the website along its development.
- To test requirements number 2 and 4 the website will be tested in different locations with different internet speeds and the website load times and latency will be noted to see if it would work for most internet users.
- To test requirement number 3 my personal email will be used to track if the alert subscription is working.
- To test requirement number 7 verify icon matches disaster.

Alongside the aforementioned numbered points, general evaluation strategies for the web development process as a whole will be used, such as ensuring that the font size is large enough for good readability, ensuring that the contrast between the background and font colors works well, and so on [4].

3

Development Life Cycle

3.1 Design

It is vital to employ both front-end and back-end technologies when creating a dynamic website with a web map. The front-end is especially significant in this project since it deals with the client-side of a web page, which is where the user interacts with it. This interactivity is the foundation of the web map [8]. This section will first go through front-end development, followed by back-end development, and finally web mapping technologies.

This section will now discuss, the resources utilised in this project and how they will come together to build the dynamic website with a web map.

HTML

The only technology required to create a simple website is HTML. HyperText Markup Language (HTML), as its name suggests, is the most fundamental web development technology, allowing web pages to be structured using features like paragraphs, links, images, and many more.

CSS

An HTML page is given a visually appealing appearance using CSS. Cascading Style Sheets (CSS), as its name implies, are in charge of the majority of aspects of how a web page appears. It has the ability to alter the font and color of text as well as the arrangement of HTML elements. This CSS feature is what will make the website responsive, meaning it works on Personal Computer (PC) and mobile devices. Due to the visual nature of the web map CSS is of the upmost importance for this project.

JavaScript

To generate anything dynamic, meaning something that changes with user interaction, on a web page, JS is required. It is a scripting language that allows for interaction between the user and the website. Let's say, for example that we wanted to have a button on a web page that changed the background colour of the website, with HTML we'd create the button, with CSS we'd style the button and with JS we program what happens when the button is clicked.

PHP

PHP is the web development language. It first appeared in 1993 and its first version was released in 1995. PHP is not a very easy language to learn and this can be very off-putting for many people at first, in particular when compared to other currently used languages like Python, yet when the time is taken to learn it it proves to extremely useful and powerful. It can do things such as link a website to a database and send and retrieve things in Hypertext Transfer Protocol (HTTP) Requests. In 2021, it was used on 79.1 % of websites [15], including big names like Facebook.com and Wikipedia.org. This constitutes a big selling point for learning PHP as it is highly likely to come across it in web development.

Web mapping

From requiring specialized software to view and analyse data, restricting those capable of using said data to experts such as cartographers to having it displayed in light web services with great interactive capabilities such as *openstreetmaps*, web mapping has evolved greatly over the last few decades [13]. There are numerous ways to do web mapping, and ,in this section, *Leaflet* will be described in detail as, in this project, it is the tool picked for creating a web map. Firstly, a definition on what GeoJSON is.

GeoJSON

GeoJSON is a data standard based on the JavaScript Object Notation (JSON) format that can be accessed and used by anyone. It started being worked on in 2007 and in 2008 it was finalized. GeoJSON is a simple format standard that defines a simple geographic feature, such as coordinates of a point in a map and its non-spatial attributes [3].

Leafleft

Created in 2011, Leaflet is an open-source JS library which focuses on building mobile-friendly interactive web maps. It is very lightweight, with approximate weight being 42kb of JS, and it's made so that knowledge in GIS is not necessary at all. It works alongside the essential web development technologies, HTML and CSS to create a tiled map, which is a map made up of very small images put together, rather than one large image. As such it allows for interactive overlays to be placed on it to place markers and other popups. Furthermore, Leaflet can load feature data which is an object with a geographic location and other properties, in the form

of GeoJSON. Implementing Leaflet does not require a substantial amount of code and is fairly straightforward when the goal is to develop a simple map. However, if more than its standard functionality is required, it can lead to more complex code [5].

Visual design of web map

Now that we have a deeper understanding of the required resources to develop a web map, we may divert our focus to how this project envisions the website. The website will be filled by the web map completely, with only a few overlays to describe key required information. To display detailed information about a certain natural disaster, there will be a modal that is displayed when the user clicks on its country.

A modal is a type of popup or text box that is only displayed upon user interaction with an element of a web page, typically a button or a link. While a modal is open, the user is generally unable to interact with the rest of the website until the it is closed, typically by clicking anywhere outside it or by completing any action it requires. This is due to the fact that the modal takes precedence over other interface components and is intended to direct the user's attention to the information at hand.

The decision to display natural disaster information in a modal was made as it allows users to access detailed data while also not displaying too much information and cluttering the web map.

The past disasters map will work differently. It will have a small settings section where the user can change the metrics to see on the map, for example, number of disasters and it will show the data for a country in the span of certain years for the data, for example, number of disasters from 1900 to 2022.

In both the real-time natural disasters map and the past disasters map the main feature will be the colouring of countries based on statistics, this is known as a choropleth map [1]. For the ongoing hazards map, there will be 3 categories each with its own colour: ongoing disaster (red), on alert for possible disaster (amber), and no ongoing or on alert disaster (green). For the past disasters map, there will be more colours and these will be in a colour palette, for example, different shades of red, to demonstrate if a country has more or less of a certain statistic.

This design has the goal of achieving what is specified in the project requirements while remaining simple. With the use of Leaflet and its lightweight nature combined with simple, compartmentalized code that focuses on keeping things simple, the web map should be responsive, enabling access from every any type of internet capable device.

3.2 Data Collection

A crucial part for this project is the data it will utilise to convey the information on natural disasters. To fulfil this need, I procured online databases that met 3 requirements: A trustworthy source which will ensure the web map is not spreading false information (Reliability), a source which employs an API to retrieve its data in a dynamic recurrent manner (Retrievability), and

a source which provides a lot of information about the natural disasters (Detailed). I divided the desired key information into geographic information which includes the region, country, and when available location inside a country, information about when the disaster happened which includes the year it started and ended, and demographic information which includes the number of dead, injured, and affected.

These needs are best met by two databases:

- EM-DAT: The Emergency Events Database (EM-DAT) created by the Centre for Research on the Epidemiology of Disaster (CRED) with the support of the World Health Organization (WHO) that has around 17,000 natural disasters occurrences from 1900 to 2023 (current time) which have structured data about them that answers all the types of information I look for. However, it does not give real-time data and only provides data in an excel spreadsheet with no API to retrieve it.
- ReliefWeb: A humanitarian information service provided by the United Nations Office for the Coordination of Humanitarian Affairs (OCHA) that was first launched in 1996. Its database has 3393 disasters (as of the moment of writing this report) and they're from around 4 decades ago to current time. It has developed its own API that can return JSON data about the latest updates on disasters in a structured way, ideal for the real-time aspect of the web map. However, it only provides very basic information on each disaster such as its type and country and not detailed information such as number of people affected.

Ideally, the project would utilise one data source for both the current disasters and past disasters maps, but as we can see through the descriptions above, neither of these two data sources are complete. Both score highly on Reliability with both having the support of a reputable government or an international United Nations (UN) organization, EM-DAT scores highly on being Detailed with its plethora of statistics on disasters but low on Retrievability, while ReliefWeb is the opposite, scoring highly on Retrievability with its API, but low on being Detailed.

Despite none of them meeting all three requirements, each data set solves a part of the website. The reliefWeb API suits the ongoing disasters map, due to its real-time feature and the EM-DAT solves the past disasters map, due to its detailed statistics on disasters. For this reason, both these data sources will be utilised .

3.3 Development

In web development, all parts of a website are strongly linked and interconnected. A seemingly independent change in the CSS file that styles the page may affect some of its JS components to stop working or break the layout of the page and more. For this reason, a Waterfall model type of approach to the development process was considered because it is known for only progressing to a further stage when the current one has been thoroughly completed.

However, this model was not adopted as it lacks in one very important aspect: Adaptability. This means that if a design idea is altered it is difficult to go back and perform this change. A more fitting development approach would be the Agile model due to its focus on completing

small scale software which can then be checked, reviewed and, if necessary, changed before integrating it into the overall product [11].

This project aims to be visually pleasing as its main feature is a map, a visual tool. This is how Agile development helps the most as it allows for change in visual elements, such as colour and layout in a late stage of development. In this case, when the general overview of the web page has been produced, it allows for improvements on how the information is displayed, for example.

This section will now discuss the main features developed as part of this project, why they were developed and how.

Data Retrieval

The first part of the project to be developed was Data Retrieval. This focused on getting the data from its source into a JS object. For the relief web API utilised in ongoing disasters map, this was performing HTTP requests first, with all the data and then trimming it so that only the necessary information is asked for

```
let parameters = {
   limit: 1000, // Number of results to return. Can be between 0 and 1000
   sort: ["date.event:desc"], //Sorts the result by specified field. Can be asc or desc
   fields: { //Specifies fields to include in the response. Can have an include or exclude field that can contain lists of fields to include or exclude, respectively
        "include": [
             "country",
             "date.event",
             "description",
             "status",
             "name",
             "primary_type.name",
             "url"
         ]
   }, //Additional parameters exist but are not used due to not being useful or necessary
   query:{
      "value": '\"ongoing\"\"alert\"',
      "fields": ["status"],
      "operator": "OR" // Default value is or. Can be AND or OR depending on how space in value field should be interpreted
   },
}
```

Code 3.1: HTTP Request Parameters

In the code snippet above it is possible to see the use of the parameter "query" to retrieve only the ongoing and on alert natural disasters. It is also possible to see that instead of retrieving all the fields, only some were retrieved, those deemed necessary. This could have been done

on the local level with the full data set but reducing the data in the HTTP request allows for a lower load time for the web map.

```
let xhttp = new XMLHttpRequest();
let paramJson = JSON.stringify(parameters)
let url = "https://api.reliefweb.int/v1/disasters"
xhttp.open("POST", url, true);
xhttp.send(paramJson);
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        let res = JSON.parse(this.responseText);
        let listDisasters = res["data"];
        mapCreate(listDisasters);
    }
};
```

Code 3.2: HTTP Request

In the code snippet we can see that AJAX was used to execute the HTTP request. The code achieves the goal of retrieving the response in the form of a JSON that is easily converted to a JS object that can now be used in other parts of the code.

Web Map

Parallel to the data the web map was also developed. The focus was to see how to create a map with Leaflet and after achieving that how to add GeoJSON to it.

```
map = L.map('map', {
    zoomControl: false, // Remove Zoom button from map
    center: [20,0], // Define center of the map. LatLng
    zoom: 3, // Define map zoom
})
var labels = map.createPane('labels');
labels.style.zIndex = 650;
labels.style.pointerEvents = 'none';
var tiles = L.tileLayer('http://{s}.basemaps.cartocdn.com/light_nolabels/{z}/{x}/{y}.png', {
    subdomains: 'abcd',
    attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors, &copy; <a href="http://cartodb.com/attributions">CartoDB</a>'
}).addTo(map);
var positronLabels = L.tileLayer('https://{s}.basemaps.cartocdn.com/light_only_labels/{z}/{x}/{y}.png', {
    pane: 'labels'
}).addTo(map);
```

Code 3.3: Creation of map with leaflet

In the code snippet above, it is possible to see all the code that is necessary to create a map with leaflet. This part of development proved Leaflet was the right choice to use for the web

map as the time it takes for the website to load is negligible, it seems instant. The code is inside a function, so maps is declared outside of it and only defined in it so it is a global variable, this is due to it being used in many other sections of the code.

After the data and web map sections of the web page had been fully developed it was time to combine the two to create the web map that displays ongoing disasters. It is here that GeoJSON comes in, to combine the geographic data of the map with the natural disaster data of the API. The intention was to assign each country information of disaster occurring in it, if such existed.

```
1    geojsonLayer = L.geoJSON(countriesGeoJSON, {
2        // Applies styles by cycling through features
3        style: function(feature){
4            for (var i = 0; i < listDisasters.length; i++){
5                // Cycle through countries in disasters
6                for(var j = 0; j < listDisasters[i].fields.country.length; j++){
7                    if (feature.properties.ISO_A3.toLowerCase() == listDisasters[i].
    fields.country[j].iso3){
8                        feature.properties.date = listDisasters[i].fields.date.event
    ;
9                        feature.properties.description = listDisasters[i].fields.
    description;
10                       feature.properties.title = listDisasters[i].fields.name;
11                       feature.properties.type = listDisasters[i].fields.
    primary_type.name;
12                       feature.properties.url = listDisasters[i].fields.url;
13                       feature.properties.activeDisaster = "ongoing";
14                       let colour = "#EB0734";
15                       if (listDisasters[i].fields.status == "alert"){
16                           feature.properties.activeDisaster = "alert";
17                           colour = "#FFA500";
18                       }
19                       return {
20                           color: colour,
21                           fillOpacity: 0.8,
22                           weight: 2
23                       }
24                   }
25               }
26           }
27           // Style for all others
28           return{
29               fillColor: "green",
30               color: "#7bba78",
31               weight: 2
32           }
33       },
34       onEachFeature: function(feature, layer) {
35           if (feature.properties.activeDisaster == "ongoing" || feature.properties
    .activeDisaster == "alert"){
36               layer.on({
```

```
                    mouseover: highlightAndTitle,
                    mouseout: resetHighlight,
                    click: displayDisasterInfo
                });
            }
        }
    }).addTo(map);
```

Code 3.4: Creation of GeoJSON layer with key disaster data

In the code snippet above, a GeoJSON layer is being created which will sit on top of the previously created basemap and give colour and interactivity to the web map in general. In the "style" function the disaster information is being added country by country and at the same time colouring said country based on its disaster status. In the "onEachFeature" each country is given its interactive properties, when a mouse hovers over it, when a mouse stops hovering over it and when it's clicked. The disaster information was placed in the GeoJSON object so that every time a country's disaster data needs to be accessed it's not necessary to loop through the response from the API, by having everything on the same object it greatly simplifies the code.

Modal

```
function displayDisasterInfo(e){
    openModal();
    let feature = e.target.feature;
    document.getElementById("disasterInfo").innerHTML = '<b>' + feature.properties.title + '</b><br>';
    document.getElementById("disasterInfo").innerHTML += '<p> Date of start: ' + formatDate(feature.properties.date) + '</p>';
    document.getElementById("disasterInfo").innerHTML += '<p> Disaster type: ' + feature.properties.type + '</p>';
    document.getElementById("disasterInfo").innerHTML += '<p> Country: ' + feature.properties.ADMIN + '</p>';
    document.getElementById("disasterInfo").innerHTML += '<p> Disaster description: ' + feature.properties.description + '</p>';
    document.getElementById("disasterInfo").innerHTML += '<p> Full UN article : <a target="_blank" href=' + feature.properties.url + '>' + feature.properties.title + '</a></p>';
}
```

Code 3.5: Creation of modal

In the code snippet above, it is possible to see how the contents of a modal are dynamically created. This function is called when a country is clicked which gives us the information of what GeoJSON feature was clicked: "e.target.feature", from which we can get all the available disaster information. The bottom element created is a link to a page on the reliefWeb website with the full profile of a hazard so that the user can further inquire into how they can help with disaster relief efforts and others way of supporting.

At this point the development of ongoing disasters map was concluded and developments progressed to the past disasters map.

It was easier to build the past disasters map as many elements of the ongoing disasters map could be reutilized in the past disasters one. This was the benefit of developing code in small blocks that work by themselves and working on parts of the website from start to end. The web map for the past disasters web map is the same as in the ongoing disasters, the difference between the two maps is the data handling and the geoJSON layer.

Data Handling

The data handling part was the most different between the two maps. As discussed before, the data source chosen for past disasters is EM-DAT which does not have an API, it only allows to download the data as an excel spreadsheet. The idea because of this was to upload this excel spreadsheet into a My Structured Query Language (MySQL) Database. For this it was necessary to utilised a locally hosted web server to had database capabilities. For this purpose Cross-platform, Apache, MySQL, PHP and Perl (XAMPP) was utilised as it has a MySQL database with an easy to use interface and most importantly, the capability to create tables from external files, such as from a Comma Separated Values (CSV) file. This is useful since excel is capable of converting files to CSV files so the initial idea was to do just that. However, the EM-DAT did not work this way as it had spaces in its column names and other not so clear reasons. After much time and effort, I was able to figure out that the database accepted Open Document Spreadsheets and Excel was able to convert the file into this format. This solution is good since it is easily reproduced, only involving two steps and no manual change of column names.

With the data from 1900 to 2022 in the database, it was possible to retrieve it using PHP due to its ability to make Structured Query Language (SQL).

```php
<?php
        // Get records from table AND relevant fields (back ticks for column with space)
    $listDisastersByCountry = array();
    $res = $conn->query("
      SELECT
        `Country`,`ISO`, COUNT(`Disaster No`) AS `No_Disasters`,SUM(`Total Deaths`) AS `Total_Deaths`,SUM(`Total Affected`) AS `Total_Affected`,SUM(`Total Damages Adjusted (000 US)`) AS `Total_Damages_(000_US)`
      FROM
        disasters
      GROUP BY
        `ISO`
      ");
    if ($res->num_rows > 0){
      while($row = $res->fetch_assoc()) {
        array_push($listDisastersByCountry,$row);
      }
    }
?php>
```

```javascript
  <script type="text/javascript">
    var disByCountry = <?php echo json_encode($listDisastersByCountry) ; ?>;
```

```
3  </script>
```

Code 3.6: Import of disaster statistics to JS

In the code snippet above, the "GROUP BY" clause combined with the "COUNT" and "SUM" function adds up the statistics of all the rows with the same country "ISO". This provides a total for each country from the year 1900 to 2022. It is possible four metrics were selected as they were considered to be the most important, since it wasn't feasible to include all of the over fifteen statistics available. Each country's natural disaster information is then stored as an object in an array whose value is passed onto a JS array. Following this, the data is finally able to be passed to the GeoJSON layer.

Web Map

```
1   geojsonLayer = L.geoJSON(countriesGeoJSON, {
2       style: function(feature, layer) {
3           for(var i = 0; i < disByCountry.length; i++){
4               if (disByCountry[i].ISO == feature.properties.ISO_A3){
5                   feature.properties.disInfo = disByCountry[i];
6               }
7           }
8           if(feature.properties.hasOwnProperty("disInfo") == false){
9               return{
10                  color: "#999999",
11                  fillOpacity: 1,
12                  weight: 0.5
13              }
14          }
15          let grades = getGrades("No_Disasters");
16          colour = getColor(feature.properties.disInfo.No_Disasters, grades);
17          return{
18              fillColor: colour,
19              fillOpacity: 1,
20              weight: 1
21          }
22      },
23  }).addTo(map);
```

Code 3.7: Creation of past disasters map GeoJSON layer

In the code snippet above, it is visible that, while different, this section is very similar for both web maps, following the same logic. In here, the number of disasters metric is shown as default when the page loads as seen in line 17.

```
1  function getGrades(metric){
2      return metric == "No_Disasters"  ? [0, 50, 120, 250, 600, 1200]
3              : metric == "Total_Deaths" ? [0, 50000, 500000, 2000000, 520000,
   13000000]
4              : metric == "Total_Affected" ? [0, 1000000, 4000000, 1200000,
   2100000000, 3500000000]
```

```
            : metric == "Total_Damages_(000_US)" ? [0, 120000000, 240000000,
    480000000, 960000000, 2400000000]
            : [];
}
function getColor(metric,grades){
    return metric < grades[1]  ?  "#bdd7e7"
            : metric < grades[2] ? "#6baed6"
            : metric < grades[3] ? "#3182bd"
            : metric < grades[4]  ?  "#08519c": "#042649";
}
```

Code 3.8: Defining the map legend

In the code snippet above, this project highlights two functions that are vital for this past disaster web map. Firstly, in the "getGrades" function we can see that for each metric the minimum, maximum and the interval between values are different. This had to be done manually in a way that suit each metric on the span of years from 1900 to 2022 best as to give convene information in the best way possible. If this wasn't done and an equal interval scale was picked, many of the metrics would colour a small number of countries the maximum colour and the rest of the world the minimum, giving the user very little idea of the global impact of natural disasters. Lastly, in the "getColor" function we can see the colour palette that was defined from lightest to darkest (top-down) for each range of values. For this web map, the colour chosen was blue so that the user can easily distinguish between the two web maps.

After defining these functions, it was possible to develop the last part of the past disasters web map, its "Settings" section. This is a form where the user can select what type of disasters to see, the span of years to see them, and what metric to see displayed. Unfortunately, the first two settings had to be abandoned as after extensive development time were dedicated to them they couldn't be accomplished, meaning the user can only change the metric they see, for example change from number of disaster to number of people affected, and the year span and type of disasters are locked at from 1900 to 2022 and all natural disasters, respectively.

Finally, after the two web maps were completed a home page with the goal of being simply and bringing the two together. In it the users can click one of two buttons, each linking to one of the maps and a similar button were added in the corner of both web maps to take the user back to the home page.

3.4 Testing

This project revolves around web development technologies which due to their differences from standard programming languages can not be tested in the same way. For this reason, in this project the main method of evaluating the website was trough user testing.

Throughout the development of the web maps, users were asked to experiment with the functionalities and try and find any way to "break" the web page. They were also asked to provide feedback on the colour scheme and adjustments that they would make to it.

4 Final Product

This project will now review the final product and evaluate whether it met the requirements set out for it in the Project Specification.

4.1 Description of Final Product

The final product is a website consisting of 3 different web pages. The first is a home page that contains a background image with two centered buttons, the top one links to the ongoing disasters web page and the bottom one to the past disasters web page.

The other two are the ongoing disasters web page and the past disasters web page. The ongoing disasters consists of web map that takes up the whole page. This web map displays countries borders each country has a colour based on its current disaster status: ongoing disaster (red), on alert for possible disaster (amber), and no ongoing or on alert disaster (green). It also has three small overlays, one in the bottom right corner displaying the map legend, one in the top left corner displaying the title of a disaster which changes to the country currently hovered over with the cursor and one in the top right corner consisting of a clickable text link which returns the user to the home page. This web map also updates itself every 90 seconds to ensure it is showing the most up to date data.

The last web page, again, consists of a web map that fills up the whole page. It represents data for different countries through progressively darker shades of blue. This data consists of four different statistics about natural disasters from 1900 to 2022 per country, these are: Number of disasters, Number of Deaths, Number of People Affected, Total Damages. The web map has a sidebar located on the top left of the web page which can be toggled on and off by the user, in it the user can change the metric into one of the four previously mentioned to be displayed in the map. It also has two overlays, one in the bottom right corner with the map legend which changes based on the currently displayed statistic and one in the top right corner which does the same as the one in the ongoing disasters web page.

In summary, this project developed a website with two dynamic web maps, one that displays

real-time information about natural disasters and one that displays information about past natural disasters in a user-friendly and visually pleasing way.

4.2 Evaluation of the Final Product

In this section, this project will discuss how the final product has fulfilled its functional, non-functional requirements and whether it reached its aims.

4.2.1 Functional Requirements

The final product achieves most of the functional requirements. It delivers a responsive website which works across all devices, it retrieves data in asynchronous manner with the use of AJAX every 90 seconds and with PHP. In relation the the web map, it displays the world with borders between countries and users can zoom in and out as they wish, it also displays all the demographic, economic and geographic information and allows the user to change the statistics displayed. Unfortunately, the time span information was wasn't fully met as only real time visualisation of natural disaster is available. The selection of a span of years is locked at 1900 to 2022 and the user isn't able to change it because with the time given to develop this project this wasn't achievable. The disaster information wasn't fulfilled because for both data sources the sub type of natural disaster wasn't available for most disasters.

4.2.2 Non-Functional Requirements

The final product fulfills most of the non-functional requirements. It delivers a visually pleasing web map, according to test users feedback, which takes less than 10 seconds to load and has low latency in its user interactions. The rest of the non-functional requirements were not implemented as they weren't a priority and upon user feedback the decision was made that they didn't add to the final product.

4.2.3 Project aim evaluation

Overall, the final product delivers on its aim to convene the extent and severity of natural disasters through a dynamic web map that utilises interactive features. It displays the essential geographic, demographic and economic information on both a real-time and past time scale.

5 Conclusion

5.1 Critical Assessment of the Project as a Whole

This section will reflect on analysing what methodologies worked and which ones would be changed. The development decision that reduced development time and difficulty was to compartmentalize the coding into data retrieval and web map. This allowed for the data to be fully understood and the web map to be fully functional, had development been simultaneous the errors would be harder to catch. Keeping the website simple was a sensible design decision since it guarantees that the site is accessible and understandable to all users, regardless of technical ability. This choice was especially significant considering the website's objective of disseminating essential information concerning natural disasters.

The main aspect I would change is to go improve the planning as it was only late in the project that I realised that writing down an idea and thinking it through, would have cut down on a lot of development time when I was too quick to start coding and then had to go back and redo it.

Overall, the main design and development decisions were good, yet with further planning all the functional requirements could have been achieved.

5.2 Future Work

This project could be expanded upon in different ways. One way would be, if a new data source for natural disasters is created that suits this project better, the project could have more statistics about natural disasters and other information users would enjoy knowing. It could also be expanded by completing the functional requirements this project was not able to. Lastly, due to the project not requiring the website to be publicly available, safety was not addressed in great detail. This would definitely be an area to develop for the final product to be published online.

5.3 Conclusion

In summary, the aim of the project was to develop a dynamic web map that utilises interactive features to show the extent and severity of natural disasters, displaying geographic, demographic and economic information on both a real-time and past time scale. I'm happy to say I believe the project met this objective through its simple design, reproducible development and a visually pleasing final product. It has fulfilled its objective of building awareness about natural disasters as both myself and test users were surprised by the amount currently happening in the world and the damage they have caused in the last 100 years.

This report has greatly expanded my ability to research scientific papers and has helped me learn about many web development technologies I was previously unfamiliar with. It has improved my English academic writing.

References

[1] Gennady Andrienko, Natalia Andrienko, and Alexandr Savinov. "Choropleth maps: classification revisited". In: *Proceedings ica*. 2001, pp. 1209–1219.

[2] Sandra Banholzer, James Kossin, and Simon Donner. "The impact of climate change on natural disasters". In: *Reducing disaster: Early warning systems for climate change* (2014), pp. 21–49.

[3] Howard Butler et al. "GeoJSON". In: *Electronic. URL: http://geojson. org* (2014). URL: https://www.ietf.org/proceedings/92/slides/slides-92-dispatch-3.pdf.

[4] Wen-Chih Chiou, Chin-Chao Lin, and Chyuan Perng. "A strategic framework for website evaluation based on a review of the literature from 1995–2006". In: *Information & Management* 47.5 (2010), pp. 282–290. ISSN: 0378-7206. DOI: https://doi.org/10.1016/j.im.2010.06.002. URL: https://www.sciencedirect.com/science/article/pii/S0378720610000510.

[5] P. Crickard. *Leaflet.js Essentials*. Community experience distilled. Packt Publishing, 2014. ISBN: 9781783554829. URL: https://books.google.co.uk/books?id=A65OBAAAQBAJ.

[6] Institute for Economics & Peace. "Ecological Threat Register 2020: Understanding Ecological Threats, Resilience and Peace". In: (2020). URL: http://visionofhumanity.org/reports.

[7] National Centers for Environmental Information (NCEI). "U.S. Billion-dollar Weather and Climate Disasters, 1980 - present (NCEI Accession 0209268)". In: *U.S. Billion-dollar Weather and Climate Disasters, 1980 - present (NCEI Accession 0209268)* (). URL: https://www.doi.org/10.25921/stkw-7w73.

[8] Kamal Lamsal. "Designing and Developing a dynamic website using PHP". In: (2020).

[9] Anthony J McMichael and Elisabet Lindgren. "Climate change: present and future risks to health, and necessary responses". In: *Journal of internal medicine* 270.5 (2011), pp. 401–413.

[10] Robert J. Nicholls and Anny Cazenave. "Sea-Level Rise and Its Impact on Coastal Zones". In: *Science* 328.5985 (2010), pp. 1517–1520. DOI: 10.1126/science.1185782. eprint: https://www.science.org/doi/pdf/10.1126/science.1185782. URL: https://www.science.org/doi/abs/10.1126/science.1185782.

[11] Kai Petersen, Claes Wohlin, and Dejan Baca. "The waterfall model in large-scale development". In: *Product-Focused Software Process Improvement: 10th International Conference, PROFES 2009, Oulu, Finland, June 15-17, 2009. Proceedings 10*. Springer. 2009, pp. 386–400.

[12] Gabriel Svennerberg. *Beginning google maps API 3*. Apress, 2010, pp. 1–3.

[13] *The history and importance of web mapping*. URL: https://www.e-education.psu.edu/geog585/node/643.

[14] Vinod Thomas and Ramón López. "Global Increase in Climate-Related Disasters". In: *Physical Geography eJournal* (2015).

[15] *Usage statistics of PHP for websites*. 2021. URL: https://w3techs.com/technologies/details/pl-php.

[16] Bert Veenendaal, Maria Antonia Brovelli, and Songnian Li. "Review of Web Mapping: Eras, Trends and Directions". In: *ISPRS International Journal of Geo-Information* 6.10 (2017). ISSN: 2220-9964. DOI: 10.3390/ijgi6100317. URL: https://www.mdpi.com/2220-9964/6/10/317.

Acknowledgments

I want to thank my family and friends, without whose support the completion of this project would not have been possible.