COMP90024 2024 Semester 1

Assignment 2 Report

Team 1

Zhuoyang Hao (1255309)
Haoyi Li (1237964)
Zilin Su (1155122)
Angela Yifei Yuan (1269549)

Github Repository Link
Youtube Video Link

Table of Contents

Introduction

This report provides an overview of a cloud-based project hosted on Melbourne Research Cloud (MRC), focusing on its architecture, functionalities, and insights. The project is motivated by weather and nature related discussions on social media, and investigates the relationship between bushfires, air quality, and respiratory disease admissions. We focus primarily on Australia, with a specific emphasis on the New South Wales (NSW) region, leveraging data spanning from 2016 to the present. Additionally, this report details our experiences throughout the project, highlighting the advantages and disadvantages of the technical tools used during the deployment process, our teamwork dynamics, and our approaches to error handling.

Cloud System Architecture and System Design

System Architecture Diagram

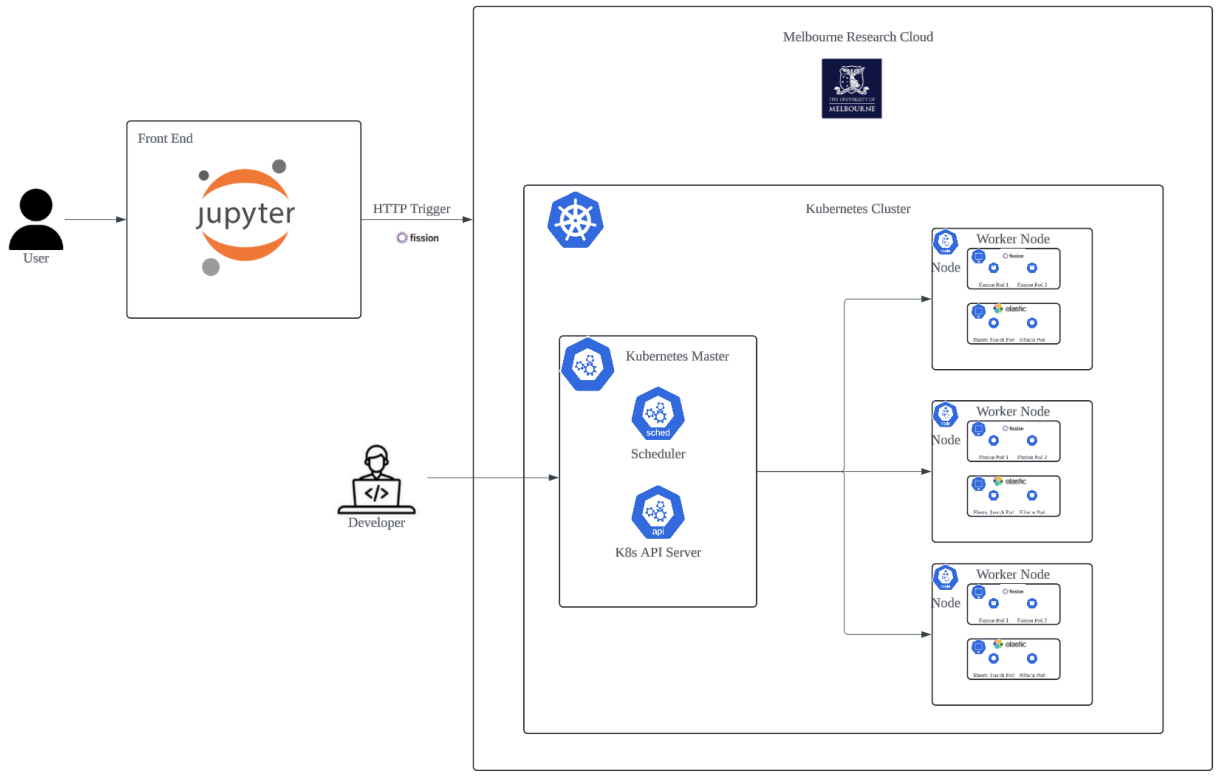

Figure 1: System Architecture Design

This design diagram describes the overall architecture of this project's cloud system, which is hosted on Melbourne Research Cloud. The system comprises several key components working

together to facilitate the collection, processing and storage of data, as well as providing a user interface for interaction.

Resources Allocation

In resource allocation, we follow the default setup as introduced in the workshop.

Figure 2: MRC Resource Allocation

Components of System

Front-End (Jupyter Notebook)

The front-end of the system is a Jupyter Notebook environment, which allows users to interact with the system through triggering HTTP requests via Fission functions. The interface is used for running basic data analysis and visualizations.

Fission Functions

Fission is a serverless function framework running on Kubernetes. It allows for the creation of functions that can be triggered by HTTP requests or Message Queue Trigger. HTTP triggers are used for data collection and pulling data from Elasticsearch. These triggers can be invoked via HTTP requests, making them suitable for real-time data interactions and integrations. They can also be used to fetch data from Elasticsearch and display it within a Jupyter Notebook interface. This enables interactive data analysis and visualization within the notebook environment.

Message queue triggers are used specifically for harvesting data from Mastodon. These triggers respond to messages in a queue, allowing for asynchronous data processing and handling. By using message queue triggers, data from Mastodon can be harvested efficiently, ensuring that the data processing tasks are decoupled from direct HTTP requests and can handle higher loads and latency.

By port forwarding `9090` on a local machine to port `80` of the 'router' service in Kubernetes, developers can access the Fission HTTP triggers directly from the local environment. When using Jupyter Notebook, developers can easily pull data from Elasticsearch by making HTTP requests to the Fission functions through `http://localhost:9090`, facilitating seamless data retrieval and analysis.

Kubernetes Cluster

The whole system is running on a Kubernetes cluster, which is responsible for orchestrating the deployment, scaling and management of containerized applications. The master node manages the cluster, including scheduling workloads, maintaining cluster state, and managing API requests. The worker nodes execute the workloads (applications and services). Having multiple worker nodes ensures that if one node fails, the workloads can be rescheduled to other nodes, maintaining application availability. With three worker nodes, the workloads can be distributed across multiple nodes, preventing any single node from becoming a bottleneck. This enhances performance and ensures efficient resource utilization. Also, additional worker nodes allow the cluster to handle more workloads and scale out as the demand increases. It provides the flexibility to add more nodes based on the resource requirements. In terms of fault tolerance, In case of a node failure, Kubernetes can automatically reschedule the workloads to the remaining healthy nodes. This fault-tolerant design ensures that the applications remain available and resilient to node failures.

Elasticsearch

Elasticsearch is used for storing and querying large volumes of data. Each worker node hosts Elasticsearch pods, which work together to form a distributed search and analytics engine. Elasticsearch is deployed as stateful sets across multiple worker nodes to ensure high availability and reliability. By using the command `kubectl port-forward service/elasticsearch-master -n elastic 9200:9200`, developers and administrators can forward local port 9200 to the port 9200 on the `elasticsearch-master` service within the cluster. This setup allows for direct interaction with the Elasticsearch API from the local machine.

There are two nodes in the elasticsearch cluster currently, which can be increased in the future depending on the demand. Each index is created with 3 primary shards and 1 replica (a total of 6 shards). The replica supports better failure tolerance, as they can be elected as a primary shard when the node hosting the primary shard fails. At present, some primary shards are located on the same node, which is efficient enough for the current workload. However, as we add more nodes to the cluster to accommodate increased workloads, the primary shards can be distributed across different nodes. This approach supports dynamic scaling of the application as demand grows.

Developer Interface

Developers can interact with the Kubernetes API server to manage and deploy applications within the cluster. This interface is essential for maintaining and updating the system as needed. Secure access to the Kubernetes API server is achieved through the use of a bastion host, which acts as a gateway for developers.

Testing

The focus of our testing is the RESTful API. Using Python's unittest framework, we conducted end-to-end testing to examine the application's entire workflow from start to finish, ensuring the system behaves as expected. These tests can be executed automatically by running the Python source code, significantly reducing the time and effort needed for manual testing after each development or code refactoring. This approach enhances the efficiency of software delivery.

The testings confirms the following aspects of the project:

- Existence of the fission routes and http triggers
- The right functions are called upon a http trigger
- Data are retrieved from the correct index
- Accuracy of retrieved data
- The streamlined air quality data collection from government website and ingress to elastic search

Several indexes, including those for bushfires and hospital admissions, were downloaded from various sources like SUDO, and uploaded to Elasticsearch. Since these indexes are static and not expected to receive new data, testing is conducted using the development database. The deployed functions aim to retrieve or merge data satisfying certain criterias, specified as fission parameters. We use the designed and deployed RESTful APIs during testing to examine the behavior of routes and triggers, ensuring they function correctly and return the intended data.

The air quality index data, sourced from the NSW government website via an API, has ingression triggered by HTTP requests, and is anticipated to grow dynamically as the application is used. To ensure a smooth data collection and ingestion process, we created a testing database. This database is cleared during the setup phase of testing. During testing, we call the HTTP trigger to collect data from a specific site for a designated year and month, then verify that the intended data is successfully captured in the database.

Motivation

Since the year 2000, bushfires in the peri-urban areas of Southeastern Australia have resulted in the loss of more than 200 lives and necessitated emergency assistance for nearly 18,000 individuals [1]. As people's awareness of fire prevention increases, the mortality rate in fires has been greatly reduced. However, the long-term effects of bushfires on the air quality should not be ignored, thus we raised two questions:

1. As an area frequently affected by bushfires, are Australian residents concerned about weather and air quality?
2. Is their respiratory health impacted by the inferior air quality caused by bushfires?

Furthermore, with the sudden decrease in temperature recently, there are many complaints about the bad weather, leading us to pay attention to weather changes. At the same time, we observed that there was a surge in posts on social media about the rare occurrence of aurora lately. This made us realize that people might also frequently discuss weather changes on social media, just as they do in everyday conversations. We are hence keen to find out the significance of weather and climate in people's lives, which may be indicated by whether individuals are inclined to make related discussions on social media. To find the answer, the team scraped Mastodon post data from the aus.social server, which collects posts from Australia. Since our analysis area was limited to Australia, focusing on Australian social media posts made sense.

Mastodon posts have been collected from May 9th to today. The requesting API was called periodically, retrieving the 40 latest posts each time. After working for a week, we compiled a substantial dataset ready for analysis. We first defined a set of keywords related to weather and air quality to measure the degree of concern Australians have for weather information. Air quality can be considered a highly related factor to weather; a sunny day usually indicates better air quality, while a hazy day often signifies heavy pollution.

According to Table 1, we can see the set of words defined for these two subtopics.

Topics	Words
Weather	'weather', 'temperature', 'rain', 'snow', 'storm', 'wind', 'cloud', 'sunny', 'humid', 'humidity', 'forecast', 'climate', 'thunder', 'lightning', 'cold', 'hot', 'warm', 'cool', 'freezing', 'breeze', 'hail', 'flood', 'drought', 'aurora'
Air Quality	'air', 'quality', 'pollution', 'pm10', 'pm2.5', 'ozone', 'aqi', 'smog', 'haze', 'clean', 'dirty', 'particles', 'particulate', 'emission', 'carbon', 'co2', 'monoxide', 'dioxide', 'so2', 'sulfur', 'methane', 'nh3', 'ammonia', 'nox', 'no2', 'nitrogen'

Table 1: Mastodon Bag of Words for Topic Oriented Post Search

By analyzing the collected posts according to different topics, we found that posts related to weather and air quality were more prevalent than we anticipated. In Figure 3, posts related to weather occupy 12% of all recent posts, while posts related to air quality are less frequent but still remain a major topic. Although many topics are discussed on open social media, people still choose to talk about the weather, indicating its importance in people’s lives and representing the maintenance of a popular topic. Therefore, focusing on a topic in this area is valuable, and the insights gained from this analysis could be helpful in many industries, such as social science.

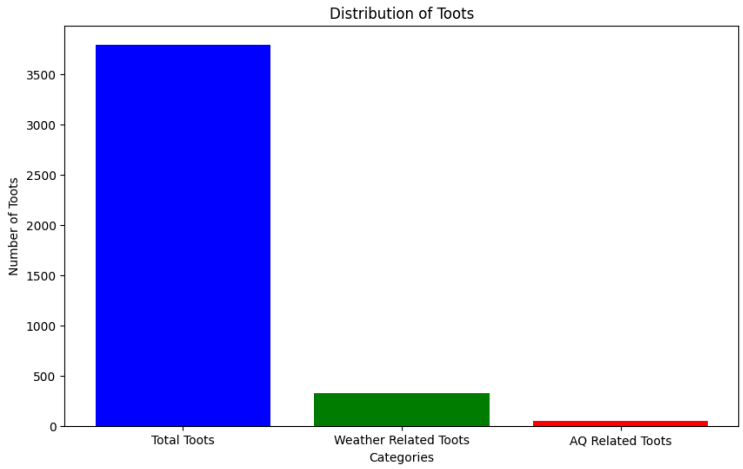

Figure 3: Distribution of Toots

Due to widespread concern about the weather, we plan to discuss a subtopic that aims to analyze whether there is a relationship between bushfires, air quality, and respiratory disease admissions. For this research topic, a bushfire can be defined as a fire in scrub or a forest, especially one that spreads rapidly. Respiratory disease can be defined as a type of disease that affects the lungs and other parts of the respiratory system. Since Australia frequently experiences bushfires, which can sometimes be a major source of air pollution, discussing their impacts is a choice adapted to local conditions.

Although these three items may not initially seem related, the intuition becomes apparent after combination. We consider air quality which is an important factor in weather, where frequent bushfires could be one of the main causes of poor air quality. At the same time, if bushfires occur frequently and lead to heavy pollution, the likelihood of people developing respiratory diseases increases. Furthermore, if the number of respiratory disease admissions rises, it may also be an indicator of poor air quality in the area.

Based on these potential relationships, our team analyzes and discovers valuable information using different data sources. The table below summarizes the data used in the analysis.

Datasets (source)	Description
nsw_admission (nsw health)	Monthly hospital admission data for respiratory diseases from 2016 to 2022 in NSW.
pha_admission (SUDO)	Yearly hospital admissions data for respiratory diseases from 2014 to 2018 in Australia. The data is aggregated by public health areas (pha).
bushfire	Bushfire information, including location, area, etc., in NSW from 2016 to 2021

Table 2: External Data Summary

Several issues exist in the retrieved data, including outdated data, missing data, and different data formats. These issues were particularly prevalent in SUDO data and health data retrieved from the Australian health dataset. Therefore, the area and time zone were carefully selected. In this report, we will mainly focus on data from 2016-2021 in New South Wales (NSW), one of the financial centers in Australia. This selection is justified by the availability of adequate medical resources, the high frequency of bushfires in the state especially Blue Mountains area (Figure 4), and the dense network of air quality observatories, all of which provide an excellent data base for further analysis.

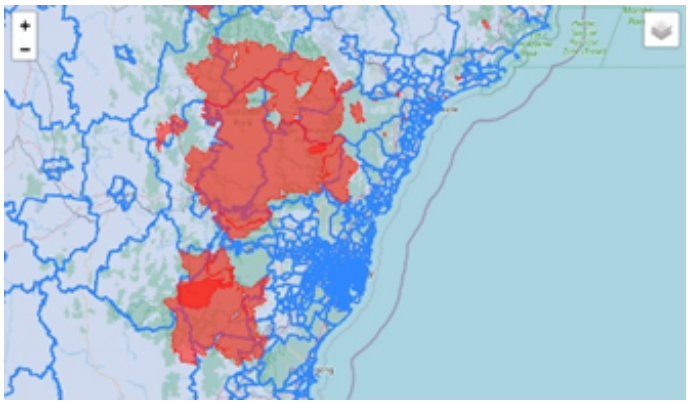

Figure 4: Map for the Occurrence of Bushfire in NSW

In the next sections, we will describe the application functionalities supporting and discuss the analysis results. Considering the outstanding findings, a summary for special discoveries will also be involved.

Functionalities

In this section, functionalities will be discussed in two aspects: data uploading and data retrieving.

Upload Data

As an important part of big data analysis, uploading data to a database is one of the most significant steps before analysis. For our scenario, we designed three different methods for data uploading, tailored to the different types of datasets collected from external sources.

CSV/JSON data files

The easiest way to collect data is to download a table of data from the internet, such as from ABS or a research platform like SUDO. The data downloaded from these sources is usually in a

CSV or JSON file, which cannot be easily shared by the team when stored on a local hard disk. Therefore, uploading these files to our chosen database, ElasticSearch, is necessary. To achieve this, we connect to ElasticSearch with the port URL https://127.0.0.1:9200 and then use the written Python scripts to create new indexes for data storage on ElasticSearch, mapping the values and keys in the dataset to the created index.

This procedure is completed to upload bushfire data and respiratory disease admissions data to ElasticSearch.

Real time stream data

As we are going to use real-time posts from Mastodon, our scenario should support periodic uploading of new data. For this purpose, we first harvest data from Mastodon using their API key and publish it to Kafka, which is a real-time streaming data pipeline. In case of security issues, we store the API key as an environment variable on Kubernetes. We deploy the Mastodon Harvester as a pod and deploy it with Kubernetes. Each time a new post is made, the information of that post, including content, timestamp, and tootid, will be read from the Kafka topic and added to the created ElasticSearch index for Mastodon data by mapping values for each key. To avoid errors during the uploading process, we log errors for problematic toots and let the program continue running.

URL collected data

For data that is not saved in a CSV or JSON file and is also not streaming data, we can collect it using a URL and directly upload it to ElasticSearch by deploying a fission function. In our scenario, the air quality dataset is an example. First, we harvest data from an api provided by the NSW government [2], and temporarily store the data as a JSON file for data aggregation. Since the air quality is measured several times a day, we intend to calculate the average value for different air quality indices. Once the data is well-prepared, we use the same methods as in the previous code to create a new ElasticSearch index and map values for each key.

Pull Data

To provide front-end availability for data collection on ElasticSearch, our team designed ReSTful APIs that use HTTP methods to pull pre-processed data from ElasticSearch pods. In the following sections, the API design will be introduced according to the different types of datasets.

ReSTful API design

- **Air quality**

  Air quality data is well-organized and detailed to the hour. Therefore, the first group of HTTP triggers is designed to return air quality data for a specific year or a specific month. For each index, the average value for all observations within a defined time range (year or month) is calculated and stored as a line of data. The second group of

functions creates HTTP triggers to retrieve data according to each observatory, which includes the monthly average air quality recorded by different observatories across all years and for every year.

- **Bushfire**

  As the content of bushfire data is not huge and well-formatted, only two types of ReSTful APIs are designed. The first one is used to retrieve all bushfire data, which will be used in analyzing the relationship between admissions, bushfire, and air quality for the long year period from 2016 to 2021. The second API will mainly be used for retrieving specific types of fires for a specific year. This pre-processed data will be used in both analytics regarding bushfire itself but also used for relationship predictions.

- **Admissions**

  The admission data is retrieved from many data sources, including SUDO data for respiratory disease and health data from ABS. Before designing an API to directly retrieve the data from ElasticSearch, we manage query templates to preprocess different sources of data, which include conditional selection, common join, as well as filling missing values. The preprocessing steps are designed in a created Fission function for the admission dataset and develop an HTTP trigger with a GET method accordingly. For this type of data, we design APIs in two ways. The first type of API will mainly focus on scraping data for all PHA areas for different types of respiratory diseases. However, for specific diseases, we face a serious problem of missing data. Therefore, we only tend to use all data with the tag ‘all respiratory disease’. The second type of APIs is for NSW admission data, which can retrieve data from a specific year, month, and disease. For consistency, all respiratory diseases will be chosen for analysis.

- **Mastodon**

  Since we preprocess the Mastodon toots while collecting by limit the collection content only involves content, timestamp, and ID. Thus, no further preprocess is needed. For this reason, only a simple HTTP get method trigger will be used for getting all collected toots.

Analysis

Using the indexes stored on ElasticSearch, as well as the RESTful APIs deployed on Fission, we extracted meaningful data to conduct analysis on relationships between bushfire, air quality, and hospital admissions of respiratory disease. There are two scenarios: the impact of bushfires on air quality and the impact of air quality on respiratory diseases.

Bushfire vs Air Quality

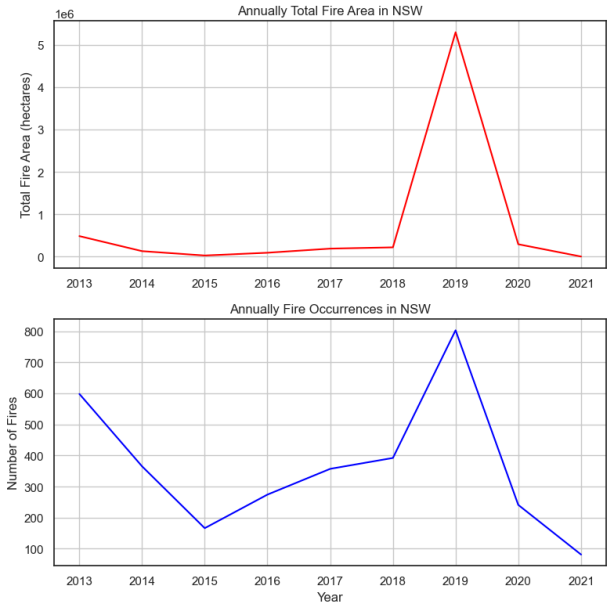

Figure 5: Annually Total Fire Area in NSW vs. Annually Fire Occurrences in NSW

Figure 5 shows that 2019 was the most severe year for bushfires, with the highest occurrences and largest total areas. After 2019, the number and area of fires decreased significantly. This may be due to the reduction in outdoor human activities and industrial activities caused by COVID-19. In addition, the increasing number of sites yearly in Figure 6 reflects people's emphasis on fire monitoring. This may also be the reason for the decrease in bushfires.

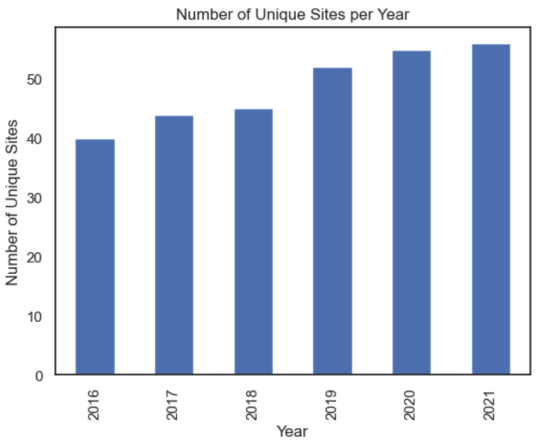

Figure 6: Number of Unique Sites per Year

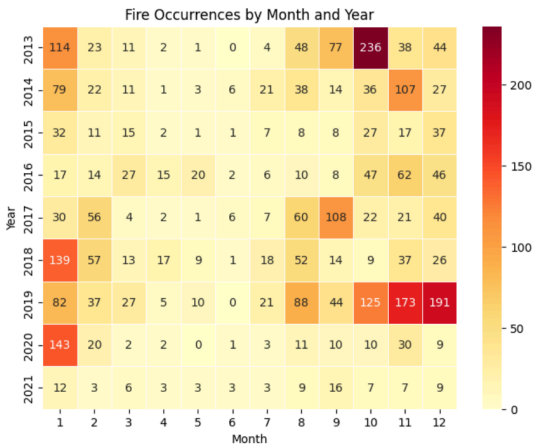

Figure 7: Fire Occurrence by Month and Year

Figure 7 illustrates that summer has the highest incidence of fires (from September to December and January), while there are few bushfires in winter (from April to July). Notably, November, December of 2019, and October of 2013 are the worst bushfire periods with the most bushfire occurrences. However, Figure 8 shows that although the frequency of fires was

high in 2013, they were small and spread out, whereas there were more large fires in 2019, which were more concentrated.

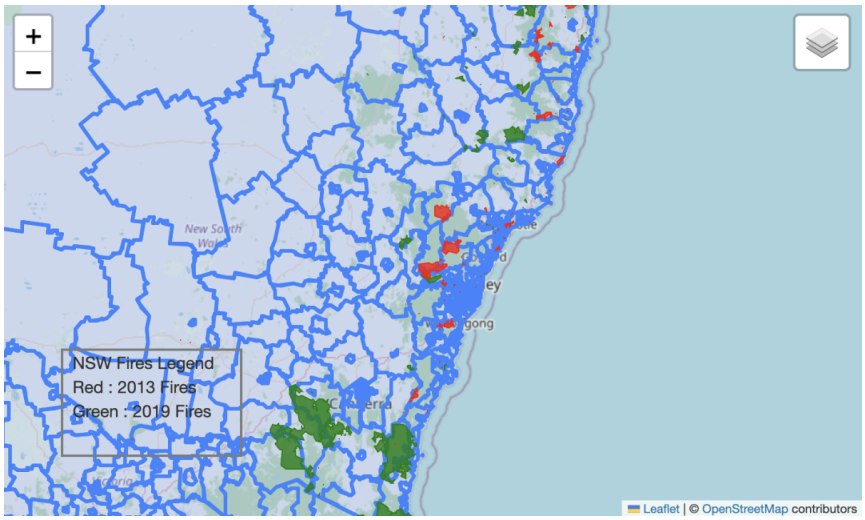

Figure 8: Fire area in NSW 2013 vs. 2019

Research shows that ambient ozone concentrations may be associated with sinusitis and hay fever, while particulate matter may be associated with more serious respiratory diseases [3]. Therefore, although we have statistics about ozone, PM2.5, and PM10, we will mainly focus on PM2.5 and PM10. The two pivot tables in Figure 9 show that the air quality in November and December of 2019 was inferior, with the highest concentrations of PM2.5 and PM10, while the concentrations in October of 2013 was far less than that in 2019. Therefore, the impact of bushfires on air quality varies according to the intensity of the fire.

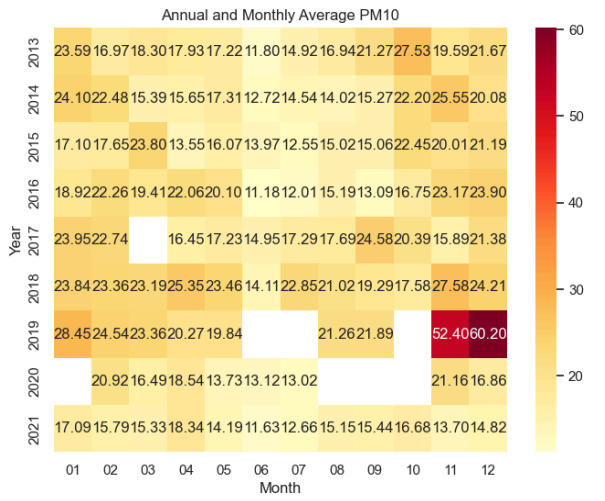

Figure 9: Annual and Monthly Average PM10

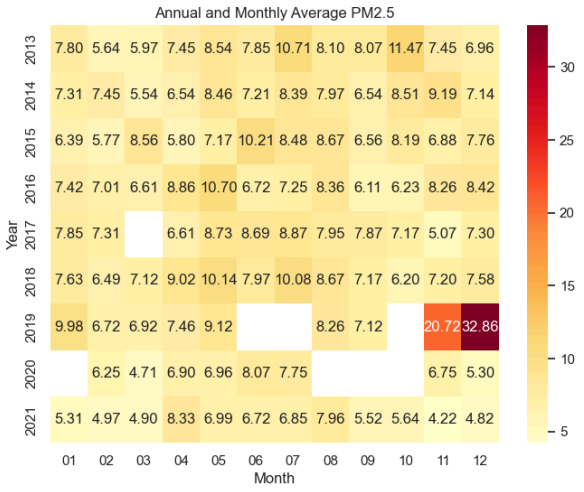

Figure 10: Annual and Monthly Average PM2.5

In this scenario, we believe that the impact of a large fire should be significant and affect the entire New South Wales, while if it is a small fire, its impact should be insignificant and only affect the surrounding air quality. Since the data obtained on admission is from 2016 to 2021,

our bushfire data will also be selected from 2016 to 2021. 2019 was the worst year for fires, and 2016 was a relatively normal year for fires in the selected range, if not considering the year 2020 and 2021 attacked by COVID-19). Therefore, we will focus on the data of 2016 and 2019.

The most intuitive way to judge the impact of fire on air quality is to observe the concentration of PM2.5 and PM10 at the nearest and farthest sites. By iteratively calculating the distance between each site and the bushfire area, we got the closest and farthest sites and their corresponding records. The data about PM10 is more complete so that we will compare the concentration distribution of PM10.

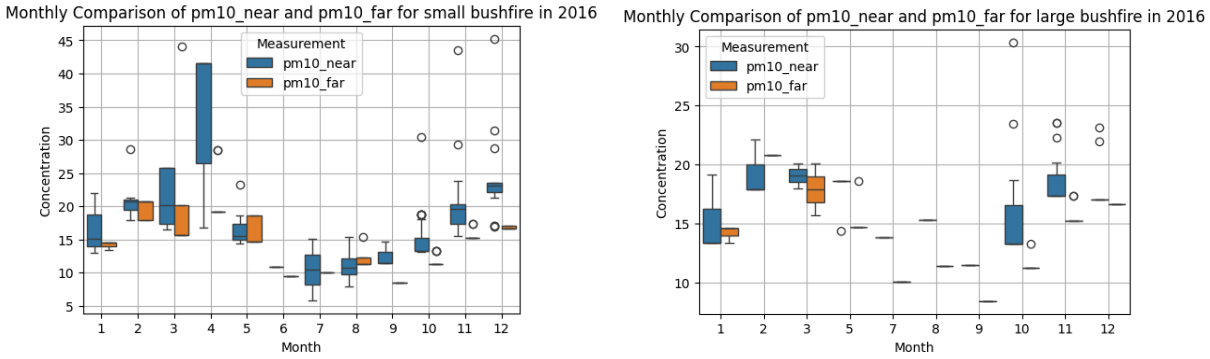

Figure 11: PM10 Concentration of the Nearest and Farthest sites in 2016

The boxplots in Figure 11 represent that there were more small fires than large fires in 2016. This is the reason why the air quality in 2016 was favorable. In addition, from June to September, when there are fewer fires, the PM10 concentration measured at the nearest site is even lower than that at the farthest site in other months.

Figure 12 shows that for 2019, the overall air quality is much worse than that in 2016, which is reflected in up to twice the concentration of PM10. This is because there were many large fires in 2019. What is more, the difference between the nearest and farthest concentration is slight for large fires. As we envisioned, the impact of large fires on the entire NSW cannot be underestimated.

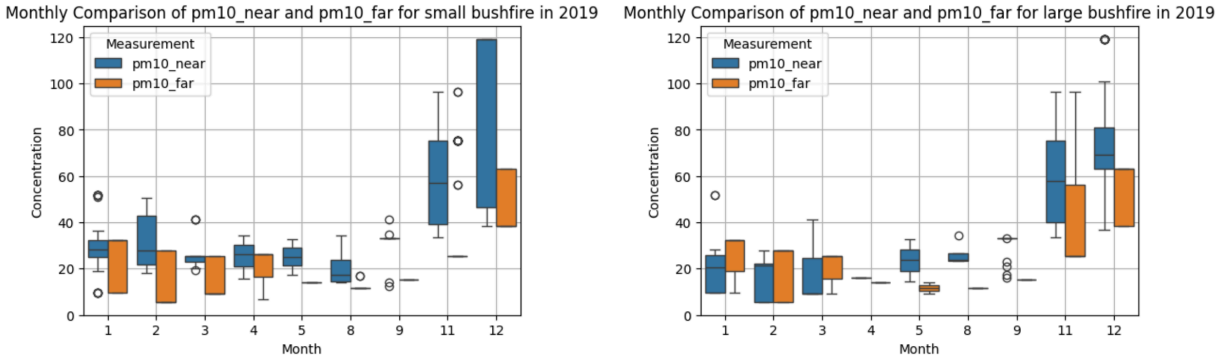

Figure 12: PM10 Concentration of the Nearest and Farthest sites in 2019

Admissions, Air Quality, and Bushfire

To provide a comprehensive introduction to the dataset, it is necessary to consider the last factor: admissions. In this scenario, respiratory disease admissions are chosen as a standard for assessing respiratory health in Australia. Since many respiratory diseases can be immediately fatal, the direct relationship between hospitalization and morbidity makes admissions a suitable measurement for respiratory health.

To offer an overview of respiratory health, the SUDO data for disease admissions has been selected for analysis. This dataset counts the number of admissions in each Population Health Area (PHA). PHA represents a type of regional division used in Australia based on Statistical Areas Level 2 (SA2). SA2 is designed to represent a community ranging in size from about 3,000 to 25,000 usual residents. The purpose of using PHA is to ensure that the number of cases in some population health datasets is sufficient for mapping, particularly in non-metropolitan areas. Therefore, using PHA is a feasible choice since mapping from PHA to the commonly used regional division code SA2 is available.

Although SUDO is an easy-to-use data collection platform with a long timeline, the dataset collected still cannot ignore issues such as unformatted or missing values. To provide an overview of recent insights into the distribution of admissions, we choose to use the “hospital_admissions_pha” dataset. This is a dataset that can be downloaded for SUDO.

As shown in Figure 13, it compares the number of admissions in Australia in 2017 for all genders. It illustrates that a high number of admissions are mainly concentrated in financial centers in Australia, such as Melbourne, Sydney, and Perth. This may be due to the centralization of hospitals in those areas and the high population density in those areas.

Moreover, before selecting specific research regions, it is important to determine whether there is a difference in the morbidity of respiratory disease between males and females. By using the same dataset, a comparison has been done, with the output shown in Figure 13. This graph clearly shows that the number of male and female respiratory admissions is similar in each PHA area. Therefore, it can be concluded that gender will not be an influencing factor for the number of admissions. Because of this discovery, for the following analysis, we will not consider gender as an important factor for respiratory admissions. Furthermore, to identify feasible potential relationships, it has been decided to conduct research in New South Wales (NSW), as this area includes adequate data for all factors we intend to use.

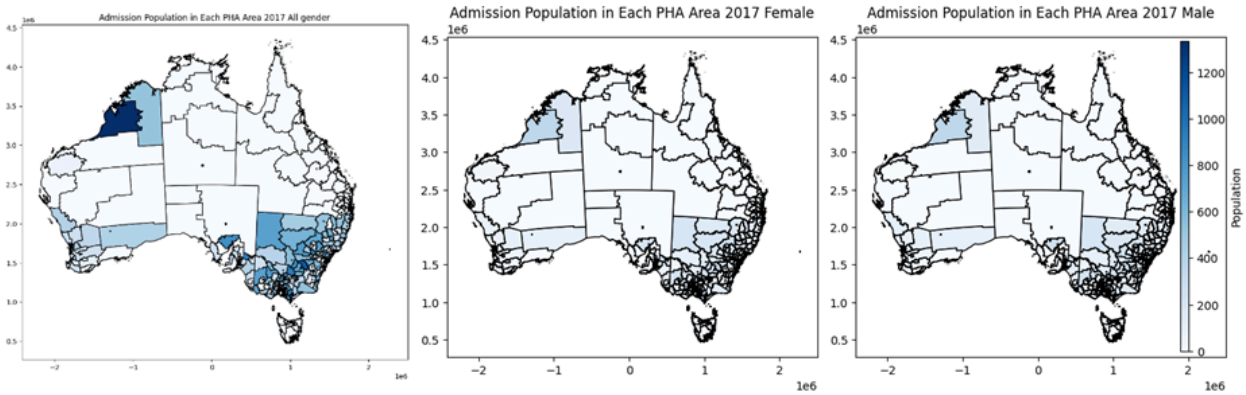

Figure 13: Australia Admissions by PHA All Gender vs. Male vs. Female

While focusing on bushfires and air quality in NSW, it is impossible to ignore the relationship between admissions for respiratory diseases and these factors. The direct connection between air pollution and respiratory diseases in humans is well-established.

Therefore, we analyzed the distribution of respiratory diseases in NSW from 2016 to 2021. Although we found the dataset includes age information, in the preprocessing steps we notice that not all types of respiratory disease are related to age. Some of the respiratory diseases are easy to occur for people at all age groups, therefore at this time we do not consider age as an important factor for the number of respiratory disease admissions.

According to this pivot heatmap below, compared with 2016-2019, the number of respiratory admissions in 2020 and 2021 shows a significant decrease. This may be due to the lockdown during the COVID-19 period, as fewer people chose to go to the hospital. Additionally, COVID-19 was not included as a respiratory disease at that time. Furthermore, the graph shows that the number of admissions usually increases in winter and decreases in summer. This indicates that people are more likely to get respiratory diseases in winter.

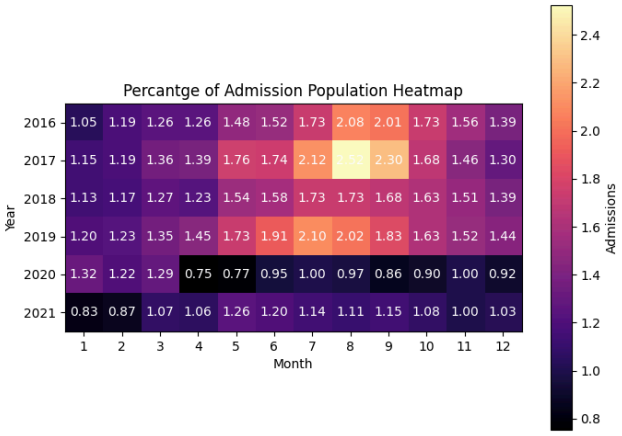

Figure 14: Percentage of Admission Population Heatmap

To show the relationship between the number of admissions and the frequency of bushfires, we analyzed the data in two aspects:

- Compared the percentage of admissions and the percentage of bushfires each year from 2016 to 2021.
- Compared the percentage of admissions and the percentage of bushfires each month from 2016 to 2021.

We chose to use percentages to scale the number of bushfires and the number of admissions to the same numeric level.

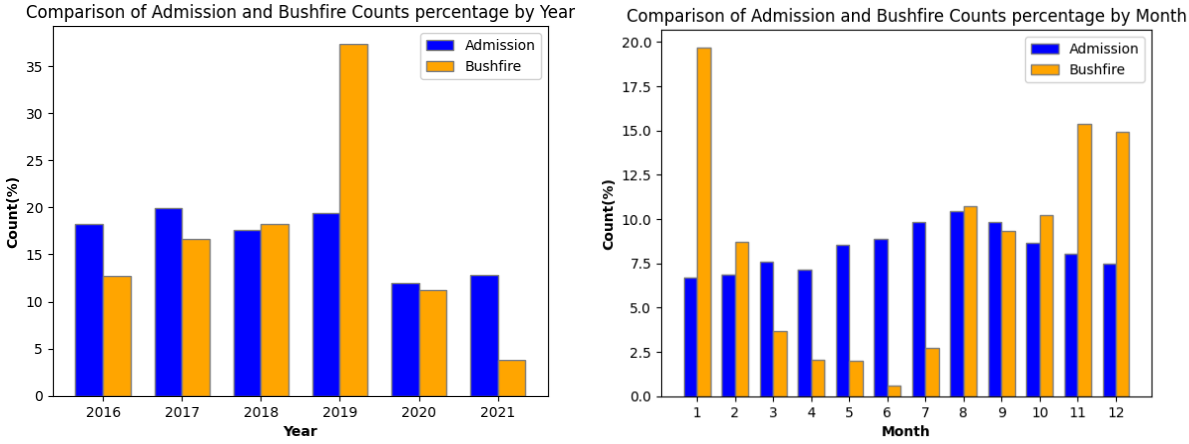

Figure 15: Comparison of Admissions and Bushfires Counts Percentage by Year vs. by Month

The left graph above shows the relationship between the percentage of admissions and the percentage of bushfires yearly. Considering that COVID-19 will impact admission rates as a latent variable after 2019, we only observe data from 2019 and before. The graph shows that when fires are widespread, the increase in admission rates is not very significant. The right graph above shows the relationship between the percentage of admissions and the percentage of bushfires monthly from 2016 to 2021. It illustrates that bushfires usually occur in summer, while the number of admissions typically increases in winter. This indicates that within a year, the percentage of admissions is inversely proportional to the percentage of bushfires and the admission rate may be more related to the seasons (temperatures).

In conclusion, our results suggest that bushfires affect air quality and that large bushfires have a much more comprehensive range of impacts that can cover the entire NSW. However, although air quality affects the respiratory system, the development of respiratory disease is not solely determined by air quality.

Discussions on Utilized Tools

Throughout the application deployment process, we utilized a variety of platforms and tools to facilitate image creation and deployment, such as Melbourne Research Cloud, Kubernetes, and Fission. We performed an extensive evaluation, considering perspectives including ease of use, supported technologies, scalability, security, and monitoring and management capabilities,

across these platforms and tools. Some key benefits and drawbacks are outlined in table 3, followed by detailed discussions, both in general, and specific to this project.

Benefits	Drawbacks
● MRC's intuitive user interface ● Fission's abstraction of containerization and orchestration ● Free access of MRC ● Open-source frameworks ● Secure ● Kubernetes' orchestration capabilities	● MRC has limited external training ● Difficulty in using Windows Subsystem for Linux (WSL) with VPN ● MRC lacks advanced services ● MRC has resource limitations

Table 3. Key benefits and drawbacks in MRC, Fission, and Kubernetes in image creation and deployment.

Ease of Use

MRC provides an intuitive dashboard for project management, allowing users to easily monitor the current resource consumptions, and efficiently perform tasks such as image, instance, and volume management. Online documentation and tutorial on using MRC is also made available by University of Melbourne [4] to ease the learning process.

Function as a Service functionalities are delivered through the open-source Fission framework. It enables simple and quick function deployment by abstracting the complexities of containerization and orchestration. Users are thus relieved from the necessity to build Docker images and use Docker Registries.

Some drawbacks of the selected tools include the limited external training and certification programs for MRC, compared to commercial cloud platforms such as Microsoft Azure [5]. Additionally, learning OpenStack's command line interface, Kubernetes, and Fission, may pose a steep learning curve.

Supported Operating Systems and Technologies

While MRC supports a wide range of operating systems, the usage of WSL for Windows operating systems imposed a substantial challenge in this project, especially when VPN is required for connection to the kubernetes cluster hosted on MRC.

Furthermore, while MRC provides fundamental infrastructure services and general-purpose cloud capabilities, adequate for this project's requirement, it falls short in providing specialized services found in commercial offerings. MRC runs on OpenStack and only offers a subset of OpenStack services, which already has less services than what's offered by commercial cloud platforms. For example, Amazon SageMaker [6] and Azure Machine learning [7], which empower developers to rapidly build, train, and deploy machine learning models quickly.

Moreover, in this project we used Jupyter Notebook for our frontend development. It offers a high level of interactivity and provides strong visualization capabilities. However it is difficult to deploy the Jupyter Notebook on Fission.

Scalability

MRC allows users to have granular control over the infrastructure, tailoring resources specifically to their needs. Each team is granted access to 11 virtual CPUs and 700 GB of volume storage on MRC, a sufficient capacity for a project of our scale. Nonetheless, the limitations on computational power and storage may impact projects with significantly higher demands.

Cost efficiency

MRC offers free access for University of Melbourne researchers and graduate students. This not only reduces costs but also eliminates the necessity to navigate and compare intricate pricing models often found with commercial cloud providers.

Security

MRC ensures secure user authentication and access control, by mandating login with university accounts, and instance access via SSH key pairs. In this project the instance is deployed on private networks accessible only from campus or via VPN, adding additional security. However, this network requirement poses challenges for setting up continuous integration (CI) and continuous deployment (CD) pipelines using tools such as GitHub Action. In such cases, more complex approaches such as deploying a Jenkins server within the kubernetes cluster might be necessary if one intends to facilitate CI processes.

Monitoring and Management Tools

Kubernetes provides powerful orchestration capabilities, enabling efficient management of containerized applications with features like automatic scaling, self-healing, and load balancing, etc. This abstraction further simplified the deployment process.

Error Handling

Melbourne Research Cloud

The issues and challenges our team faced while using MRC predominantly affected Windows users. They consistently encountered timeout errors when attempting to establish an SSH

connection to the Melbourne Research Cloud. Upon investigation, we discovered that WSL fails to make an internet connection when the VPN is active, which is necessary for connecting to MRC. This imposed a significant challenge, as the varying configurations and environments of different devices meant that many online solutions were ineffective. Two team members who faced this issue eventually implemented different workarounds, as described below.

1. Create a file named .wslconfig in the directory C:\Users\%USERPROFILE%\ and added the following lines:

   [experimental]
   autoProxy=true
   dnsTunneling=true
   networkingMode=mirrored

   This configuration was necessary because domain name system calls to the Windows host are blocked [8].
2. Directly modify the resolv.conf as follows:

   # This file was automatically generated by WSL. To stop automatic generation of this file, add the following entry to /etc/wsl.conf:
   # [network]
   # generateResolvConf = false
   nameserver 127.0.0.42
   search unimelb.edu.au unimelb.net.au its.unimelb.edu.au

Fission

Similar to the challenges we encountered with MRC, WSL users were unable to deploy Fission functions following the same procedure that worked on macOS devices. They encountered an error indicating that the required Python environment did not exist, and attempts to create the Python environment were rejected with an error message stating that it already existed. Given the time constraints and the potential complexity of resolving this issue, we decided to assign the function deployment tasks to team members using macOS devices.

Moreover, in order to make the front-end presentation more readable, we tried to deploy the function calculating the distance between each detection site and fire area in Fission. However, this function uses geopandas, which depends on several underlying libraries like GDAL. This complicates deployment in containerized environments (like Docker) or cloud platforms like MRC. A solution is to use pure Python alternatives like pandas and math libraries. After the trade-off, we decided to retain the function at the front end to avoid reduction in efficiency brought by pure Python.

Language processing

The language processing in our scenario mainly focuses on processing Mastodon post data. Initially, the team intended to use Twitter data, however, after conducting some research, the Twitter API was found not to be available in 2021. Therefore, stream data from Mastodon has been chosen. During this process, three different types of issues occur.

The first issue is about the location. Mastodon has many servers around the world, but our research needs to be limited in Australia. To address this problem, aus.social is chosen as a data collection base area. This is because aus.social is a server located in Australia, and most of the posts there are from Australia. Since the topic only focuses on weather concerns, a very specific location for each post is not necessary. Therefore, the choice of aus.social and analyzing the content posted there is reasonable for the scenario.

Secondly, while analyzing how much people care about weather, we wanted to search all posts under weather-related topics. It is initially found that there is a topic tag on Mastodon; however, weather-related tags are not commonly used. This does not mean that weather is not a hot topic because, when wenting through many new posts, it can be found that people still prefer to talk about weather-related topics but do not use a tag. Therefore, a designed bag of words for weather and air quality-related topics and attempted to count the number of posts with relevant topics.

Finally, it was noticed that Mastodon data lacks accurate timestamps for each post, making it difficult to determine the latest one based on IDs alone. To address this issue, a timestamp was added while collecting each new post. As a result, posts are now ordered correctly.

Data preparation

While analyzing the external data, especially the admission data, we frequently encountered missing values and unformatted data columns. Data downloaded from SUDO include a large part of missing data for PHA areas. After comparing it with the official shape data for PHA in all of Australia, we discovered that not only some PHA areas lack admission population data, but also some PHAs have impossible populations of respiratory disease admissions. Considering realistic factors and common sense, we realized that the PHAs without admission data are likely uninhabited. Most of these PHAs are centralized in the center of Australia, which is desert. Therefore, filling all the missing values with an admission population of 0 is the final decision. Furthermore, for those impossible values, we found that most of them were formatted as 3-digit divisions. This may be because the population of admissions in those PHAs is greater than a thousand. Thus, the solution could be to convert the data into a reasonable float.

Moreover, while combining data using a fission function, different issues occur. Since the unmatch between two datasets we intend to concat there may be null value existing in the dataset. In our scenario, we prefer to leave it “None” as the result will not affect the front end analytics based on that.

Teamwork

Work allocation

Team Member	Roles
Zhuoyang Hao	● Gathering data on air quality ● Deploying fission functions and triggers for importing and retrieving air quality data ● Retrieving Mastodon data via API, and deploying corresponding fission functions and triggers ● Data analysis with respect to Mastodon data collected ● Report writing (Cloud system architecture and system design) ● Presentation preparation
Haoyi Li	● Gathering data on hospital admissions ● Analysis on hospital admissions data and its relationship with bushfire and air quality ● Adjusting the fission functions and triggers for hospital admissions data ● Report writing (motivation, functionalities, analysis, and error handling) ● Video filming for the hospital admissions related analysis demonstration ● Presentation preparation
Zilin Su	● Gathering data on bushfire ● Deploying fission functions and triggers related to bushfire data - analysis on bushfire data and its relationship with air quality ● Report writing (analysis, error handling) ● Video filming for the bushfire related analysis demonstration ● Presentation preparation
Angela Yuan	● Gathering data on health admissions ● Deploying fission functions and triggers related to health admissions data ● End to end testing ● Github ReadMe file and adjust github repository structure ● Report writing (introduction, testing, discussions on utilized tools, and error handling) ● Presentation preparation
We only have four team members because our fifth member dropped the subject in the initial stage of the project and thus made no contribution.	

Table 4: Team Work Allocation Detail Table

Team Collaboration

One challenge we faced was that a team member dropped the subject during the initial stages of the project when we were focusing on data collection and ingesting data into Elasticsearch. This team member was responsible for collecting Twitter data and investigating whether location information could be extracted for our project. Zhuoyang took over this task and shifted the focus to collecting Mastodon posts instead. With one less team member, the remaining team members had increased workload, especially given our time constraints.

Despite this setback, our team collaborated well throughout the project. At the beginning, we outlined the tasks necessary for project completion and divided the workload by week, aligned with the project deadlines. During weekly meetings, we shared our individual progress and discussed work allocations for the upcoming week. Throughout the week, we also stayed active online to communicate our progress and collectively solve any issues that arose. Major decisions were made collaboratively, ensuring that every team member was informed and in agreement.

References

1. https://link.springer.com/article/10.1007/s00267-015-0525-x
2. https://data.airquality.nsw.gov.au/api/Data/get_Observations
3. https://www.sciencedirect.com/science/article/pii/016604629090019Y
4. https://docs.cloud.unimelb.edu.au/
5. https://azure.microsoft.com/en-us/resources/training-and-certifications
6. https://aws.amazon.com/sagemaker/
7. https://azure.microsoft.com/en-au/products/machine-learning
8. https://github.com/MicrosoftDocs/WSL/blob/main/WSL/troubleshooting.md#networking-considerations-with-dns-tunneling