An Exploration of Liveability in Melbourne

COMP90024 Cluster and Cloud Computing

Assignment 2 - Team 20

University of Melbourne

Cenxi Si
1052447 China
cenxi@student.unimelb.edu.au

Yipei Liu
1067990 China
yipeil@student.unimelb.edu.au

Jingdan Zhang
1054101 China
jingdan@student.unimelb.edu.au

Chengyan Dai
1054219 Melbourne
chengyand@student.unimelb.edu.au

Ruimin Sun
1052182 China
ruimins@student.unimelb.edu.au

May 15, 2022

Abstract

Melbourne is one of the main cities in Australia which has repeatedly appeared in kinds of lists of most livable cities around the world, and the group assignment is dedicated to exploring Melbourne's livability. We selected two different topics, house price and income, in Melbourne and compared with the same theme of Sydney city from collected tweets expressing emotions in a positive rate.

The overall software systems are built on a multi-node of virtual machines through Melbourne Research Cloud with Ansible and OpenStack, and harvesting tweets from the Twitter APIs. Our group also download house price and income data from the AURIN platform and associated CouchDB database with the AURIN data and preprocessed Twitter data. After presenting the analysis the data from database, we demonstrate the plots and the results on the frontend web application.

At the end, all these could lead us to a conclusion that the livability of Melbourne and Sydney is basically similar, and there is no obvious difference could be seen from the analytical diagrams. However, judging from the positive rate of emotion in tweets, Melbourne still performs a little better compared to Sydney and we could say that it is a liveable city.

Contents

1 Introduction

1.1 Background Information

Life in Australia changes rapidly since the breakout of the pandemic. To conquer this worldwide obstacle, people have put in a large amount of effort. As the huge transition in lives happened, this report would use Twitter to examine liveability after COVID-19 in Melbourne. There are 7 categories of liveability according to the AARP website, which is separately housing, neighbourhood, transportation, environment, health, engagement and opportunity. To corporate with the enormous economic impact, we chose 2 factors - housing and opportunity together in the further exploration. To collect statistical data to compare with the public's view, data from AURIN and tweets from Twitter are the first preference in this research.

Twitter is one of the most popular social media platforms which has around 436 million[8] monthly active users containing various discussions. Through this platform, tweets including every emotion would be posted and accessed around the globe. Thus, we can use tweets to analyse the sentiment towards housing and income from the public in a specific geographic region. In the process of data collection, we firstly created lists of keywords around house prices and personal income to represent housing and opportunity factors. The searching approach is that if one tweet contains one of the keywords, this twitter would be saved to be analysed its sentiment. These results would be used in this report to compare the attitudes of the public in Melbourne and Sydney towards house prices and personal income. To explore the relationship between house prices and the public's yearly income, a division between them is executed. It represents a rough time interval that one person living in the corresponding area should work to purchase one residence. Furthermore, the feasibility that the use of sentiment analysis in social media platforms is also examined in this report.

1.2 System Functionality

This report follows an organised structure to describe the functionalities of our work in details. At the beginning of this report, a general user guide is introduced. In this section (Section 2), the accesses to GitHub, website demonstration on YouTube are included.

In order to achieve the purpose of this project, a set of applications are created on the Melbourne Research Cloud (MRC) which is provided by the University of Melbourne. For the deployment of these applications, Docker and Ansible are used throughout the execution. A more detailed description of the system design and architecture would be provided in Section 4.

In the data collection part, high volumes of data from tweets are harvested and stored using Twitter API and CouchDB. To examine the feasibility of using Twitter to analyse sentiments, actual statistics are investigated from AURIN. Section 5 and Section 6 would present a more specific introduction from the algorithm that used in Twitter harvesting to exact storage in CouchDB.

The observation of the data analysis for both tweets and AURIN would be demonstrated in Section 7. Furthermore, the presentation and visualisation of web applications would be introduced in Section 8. It is coded in JavaSrcipt for both server and client interface and a template language called Handlebars is used for the 'views' in HTML.

2 User Guide

GitHub Repository: https://github.com/JingdanZHANG888/CCC2022-Team20

Web Application Link: http://172.26.131.132:3000

YouTube Demonstration Link:

- Ansible Deployment: https://youtu.be/zxmNIVBY1gQ
- Web Page: https://youtu.be/MjdD0YRZgrk

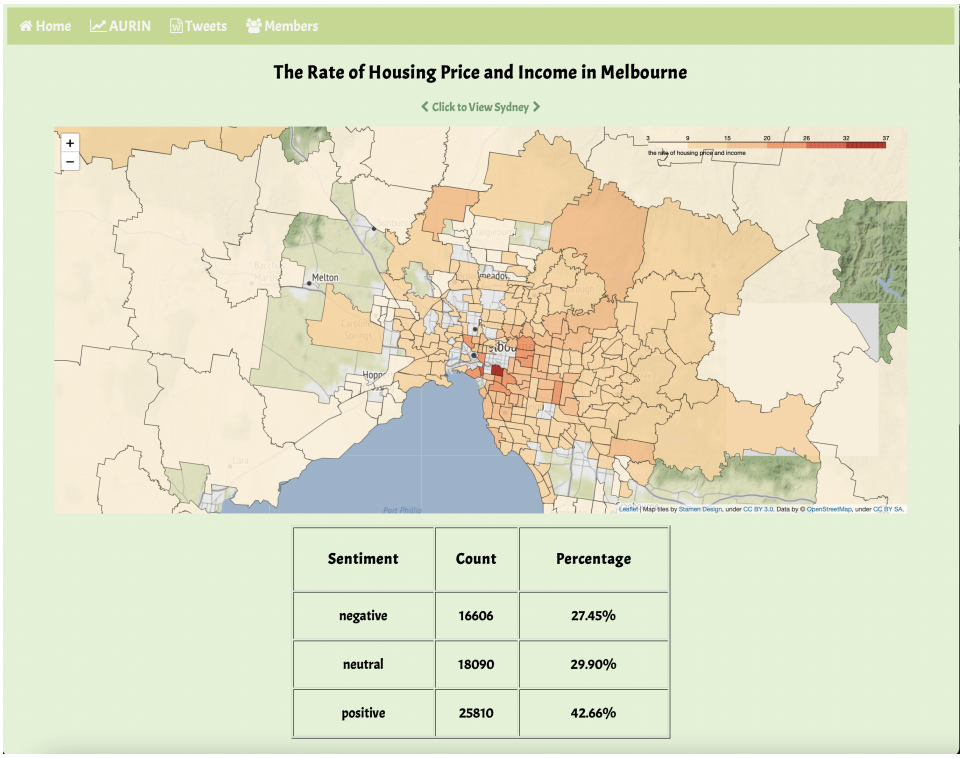
2.1 System Deployment

To run the whole project,

1. The user should install Ansible.
2. Clone the GitHub repository from the GitHub link to the Ansible.
3. Create instance by typing "./run" in terminal (In Mac) or smd (In Windows) and enter the password.
4. Typing "./run" again to run the rested code and enter the password.
5. The next step is to connect instance 4 by typing "ssh -i [route] ubuntu@172.26.131.132".
6. Users can access to the web page by the web demonstration link (http://172.26.131.132:3000), but should type the next three line code in terminal (In Mac) or smd (In Windows) first.
    - sudo apt update
    - sudo apt install -y nginx
    - sudo systemctl status nginx.service

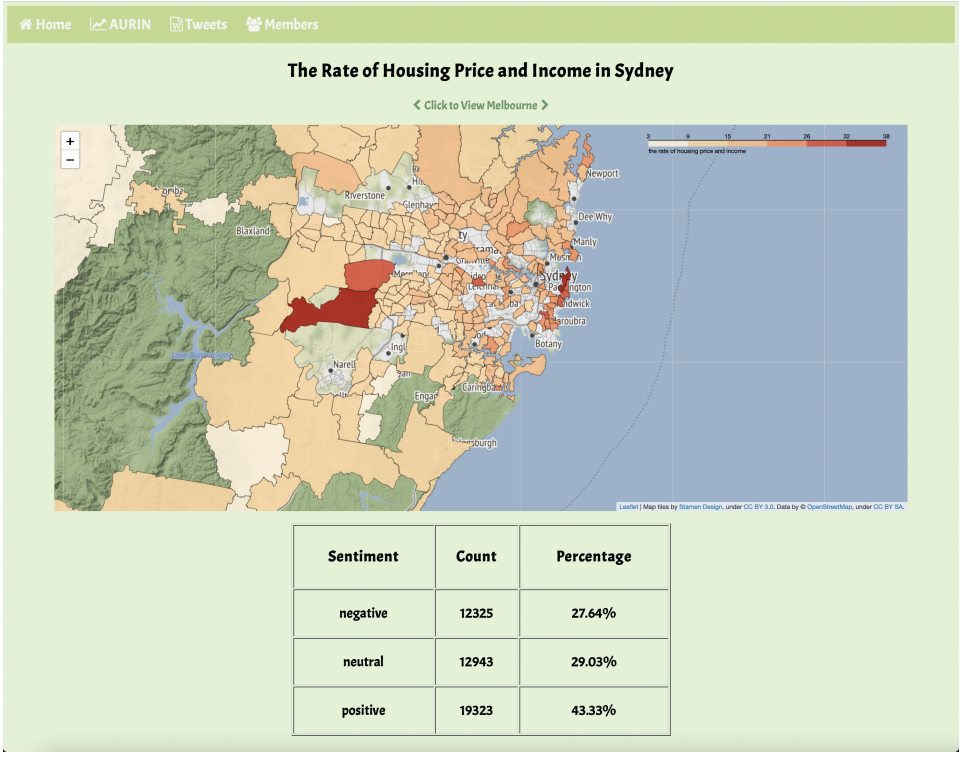
2.2 Web Guide

The Web demonstration link will lead the user to the index page (shown as Figure 1). This page display the Melbourne map and a table of twitters collection from Melbourne. We calculated the house price and income ratio of different regions in Melbourne and displayed it on the map. It can be seen from the label that the darker the color is, the higher the house price and income ratio is; on the contrary, the lighter the color is, the lower the house price and income ratio is. Users can clearly see the regional differences on the map. You can also zoom in and out of the map to see Melbourne as a whole or a specific area by using the plus and minus signs in the upper left corner. The table below the map shows the number and percentages of positive, neutral and negative sentiment about house price and income expressed in tweets in Melbourne.



Sentiment	Count	Percentage
negative	16606	27.45%
neutral	18090	29.90%
positive	25810	42.66%

Figure 1: Index Page (Melbourne Map and Data Collection)



Sentiment	Count	Percentage
negative	12325	27.64%
neutral	12943	29.03%
positive	19323	43.33%

Figure 2: Index Page (Sydney Map and Data Collection)

The "Click to View Sydney" button at the top of the map can directly preview the map of Sydney (shown as Figure 2). Similarly, we also mark the house price to income ratio of different regions on the map. Users can view the house price and income ratio in Sydney and sentiment for tweets in the same way they browse Melbourne. "Click to View Melbourne" button will bring the users back to the map of Melbourne.

Figure 3: House Price and Income Ratio Analysis

Users can also access AURIN, Twitter and Members via the Top Navigation bar at the top of the web page. On the AURIN page (shown as Figure 3), users can see the Analysis of the house price and income ratio of Melbourne and Sydney. The first column is the bar chart of the areas with the top 10 highest house price to income ratio in Melbourne and Sydney. In addition, users can also see the comparison chart of the average house price and income ratio of Melbourne and Sydney.

The page of Twitter shows the emotional difference of users in different cities about house price or income (shown as Figure 4). Users can see the high frequency vocabulary that appear in tweets which from Melbourne and Sydney and the topic is about house prices, incomes or emotions about the previous two, we also compared the two cities sentiment positive, neutral and negative count and proportions, So users can clearly see the difference between the two cities. Our team member information are shown in the member page. Each page has a "Return" button that allows the user to return to the home page (Melbourne map page Figure 1).

Figure 4: High Frequency Vocabulary from Tweets and Sentiments Analysis

See the YouTube demonstration link for details on how to use the web page.

3 Scenario Overview

To examine the liveability, we set our scenario to be: What are people feeling regarding the house price and their income situation in terms of different types of sentiments?

To have a deeper exploration of this scenario, three supporting aspects are discussed in this report:

- Twitter will be a suitable platform for examining lives after the global pandemic. Due to the impact of COVID-19, a large number of searches might be affected and suspended. However, social media platforms have not to be interrupted and people continually communicated through them without a pause.
- Sentiment analysis about tweets is correspondingly true to the statistics from AURIN. We separate emotions into three categories: positive, neutral and negative. A keyword library is created for pairing words in tweets to classify them into 3 categories for each category. Compared to the AURIN data that we collected, they roughly relate to each other.
- A detailed comparison of which suburbs are more liveable in Melbourne is conducted and the macro comparison between Melbourne and Sydney is followed. The detailed analysis and comparison would be seen in Section 7.

4 System Design and Architecture

The system of this program is built on the Melbourne Research Cloud (MRC) and could be set up automatically by Ansible and Docker.

4.1 Melbourne Research Cloud

Melbourne Research Cloud (MRC) is a private cloud that provides free on-demand configurable computing resources for researchers in University of Melbourne to implement some projects. It offers Infrastructure-as-a-Service (IaaS) and users can create virtual machines with appropriate configuration by MRC. Although MRC is powerful, it still needs to be used on a case-by-case basis and it has both advantages and drawbacks.

4.1.1 Advantages of MRC

- Scaling up computational power: When the computer performance is insufficient for a specific project or development of an application, MRC can probably provide enough computational power without extra cost.
- More Trust: MRC is a private cloud led by University of Melbourne and compared to public cloud, it is more secure.
- Sharing projects: Team members can develop collaborative projects through MRC, and researchers can easily view the work done by others and share the results.

4.1.2 Drawbacks of MRC

- Limited resources for each researcher: Since the computing resources is limited and MRC provides its service to all researchers at the University of Melbourne, there are limited computing resources could be used for researchers on each projects. Besides, it might be some difficulties in using computing resources on MRC.
- High management and maintenance cost: It is expensive to build and manage a private cloud.

4.2 Ansible

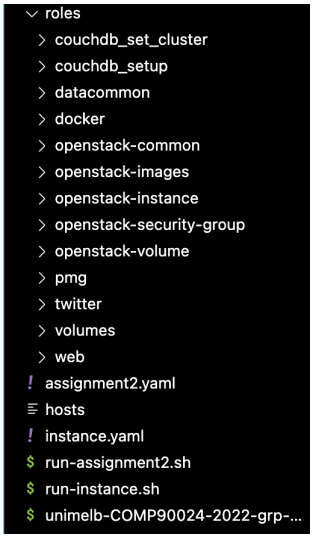


Figure 5: Ansible Roles

Ansible is a python-based automated operation and maintenance tool. It provides a framework for deploying modules in bulk and can specify the host to operate on, which is applied in this assignment. Two Ansible playbooks are created to implement dynamic deployment. There are 13 roles in these two playbooks and the overview is shown in the figure. The first playbook is used to create four instances with image "NeCTAR Ubuntu 18.04 LTS (Bionic) amd64 (with Docker)". Each instance has 50 GB volume and one cpu with four cores due to only four CPU can be set in this project. The second playbook can set all other works and build the system for Assignment 2. When the second playbook is run, several packages including Python and Docker are installed into three instances named "database". Then CouchDB is set up across these three instances. The source code of Twitter Harvester is installed on the master node which is one of the three "database" instances and run by a Docker container. The results of Twitter Harvester is stored into CouchDB directly. Source codes of data analysis and website building are installed on the last instance named "webnode". The name of hosts of instances should be allocated in the "host" file as shown in the figure.

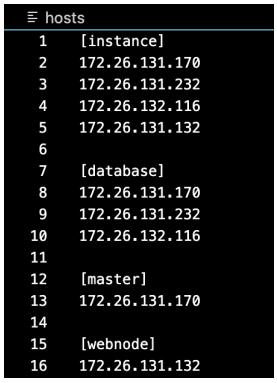


Figure 6: Host List

4.3 Docker

Docker is a software container platform and users can package dependencies and applications into a Docker image. Then a Docker container can be created through the Docker image. Therefore, the process of development is accelerated through Docker. In this project, CouchDB and some source codes of program are developed through Docker container. For file of source code of each program, there is a Dockerfile that contains information about the Docker image to be built, including the required dependencies, source code, and so on. The files are cloned by an Ansible task. Then it will build a Docker image and run a Docker container through Ansible commands automatically.

In addition, Docker has a good community called Docker Hub, which can provide a lot of resources and examples to study how to use it. Hence, it is an appropriate tool for us to finish this assignment.

4.4 System Design

Four instances are created on MRC and the first three of instances are named "database". CouchDB are set up across these three instances by Docker as shown in the figure. Twitter harvester is run by a Docker container, connecting with Twitter through API on instance 1. Analysis program and web application are built on instance 4 named "webnode" with PYTHON, Docker and NGINX.

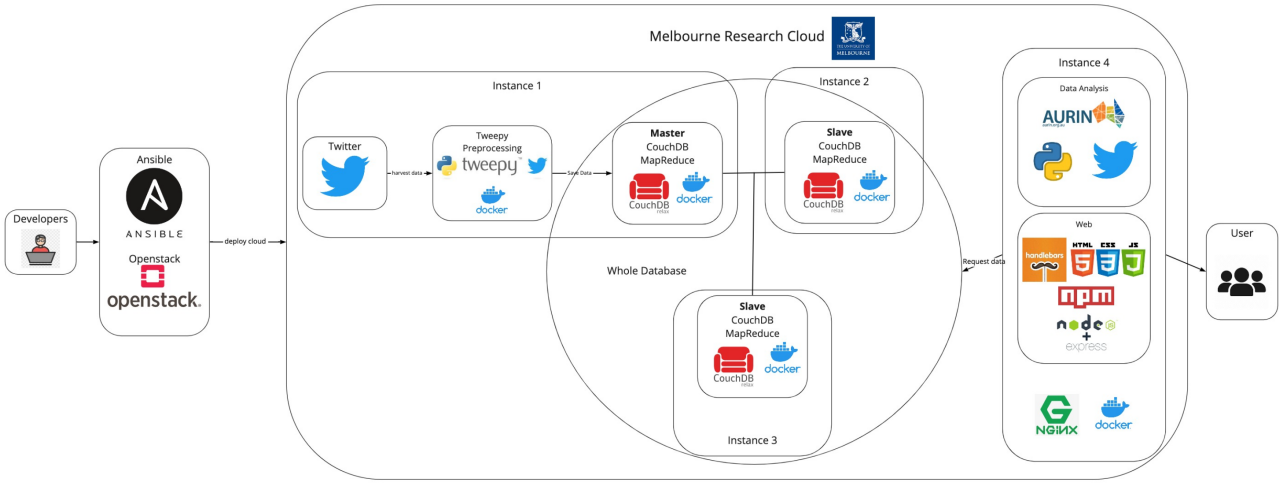


Figure 7: System Architecture Design

4.5 Error Handling

In the usage of Ansible, the first error we met is related to permission. Since we connect instances through MRC instead of administrator, when we build Docker images and Docker containers, we don't have enough permissions to run the commands. The solution is to add "sudo" in front of shell commands and "become:yes". The second error is related to using Docker. An error occurs when creating a Docker container with the same name as the one created earlier. The solution is to delete the previous Docker container with the same name while it exists.

5 Data Collection

During system architecture, twitter harvester is running on one instance, AURIN data is downloaded from AURIN website and imported to CouchDB.

5.1 Twitter Harvester

Twitter data harvester aims to collect tweets information using twitter API. Our group applied four evaluated twitter API accounts which enable the twitter harvester to collect data with less waiting time. The prepared JSON format data for twitter harvester are a group of twitter keys and a set of related keywords including Melbourne, housing and income for searching relevant tweets. Also, sentiment of each tweets is analyzed before storing to CouchDB database. Since the twitter harvester needs to continue collecting data on instance and switching to appropriate twitter API key automatically, it is necessary to design an algorithm to deal with all possible situations while the twitter API key's limit is reaching.

The main package used for twitter harvester is tweepy with version 4.8.0. The package Sentiment Intensity Analyzer imported from Natural Language Toolkit (nltk) is used for analyzing the sentiment category from each tweet's content. CouchDB package is used for connecting CouchDB database and storing tweets into correspond data sets. Before running the main part of twitter harvester, it is necessary to initialize the states of twitter API keys. The field "last_used" records the latest date time of using the twitter API key. The default state of API keys is "last_used = 2022-04-10 00:00:00".

5.1.1 Algorithm of Twitter Harvester

1. The twitter harvester needs to select a most appropriate valid API key from the current API key list depending on the length of rest time on each key.

2. Since every tweepy cursor requires a set of words to query, the program could randomly pick 20 keywords from the existed json file for every round of API keys and the keywords are connected by "OR".

3. Tweepy cursor plays the role of querying relevant tweets and uses geocode to filter the region in Melbourne or Sydney. During iteration of tweets, required information of each tweet are stored into a dictionary including id, content, coordinate, place, user's location, created date and sentiment. Tweet's sentiment is analyzed by Natural Language Toolkit and classified into three categories which are "positive", "negative", and "neutral". Then, the information could be stored into CouchDB databases. After successfully saving tweet to CouchDB, the last used date time of API key will be updated.

4. When there is an tweepy error appearing, it means that the rate limit reached. The corresponding API key needs to sleep and update its latest used date time. From now on, a new round of tweets collection will be started.

5.1.2 Twitter Harvester Error Handling

1. The twitter API keys have limitations on number of querying per 15 minutes. The system needs to deal with the problem of choosing a valid key for each round. The valid keys are defined as the duration of current date time and last used date time is greater than 900

seconds. If there is a valid API key existing, the program stores its duration and position in the JSON file into the duration list. The best valid key is selected according to the longest rest time. Otherwise, all API keys require extra sleeping time for the next round. In order to make sure the waiting time be minimum, the waiting time is calculated by the difference between 900 and longest rest time period. Furthermore, when the API key is selected, the authorized twitter API could be acquired from tweepy. Meanwhile, once the program gets an available API key, it is important to update the selected twitter API key's state in the json file for the next round.

2. To avoiding duplication of tweets, every tweet's id is unique and the tweet will be saved in CouchDB only if its id did not appear in the corresponding dataset before.

5.2 AURIN

The data of housing and income in Melbourne and Sydney are retrieved from AURIN. AURIN is a collaborative national network provided by leading researchers and data providers in different areas including academic, government and private sectors. It is convenient to access thousands of multidisciplinary data sets from more than 100 data sources[1].

Housing dataset is attributed by Australian Property Monitors. It is a time series dataset and is aggregated to the 2016 Statistical Area Level 2 (SA2) boundaries. This dataset describes the monthly sold, rent and sale house price on various statistical levels from 01/03/1994 to 31/12/2021. An brief overview of dataset is revealed by Figure 8.

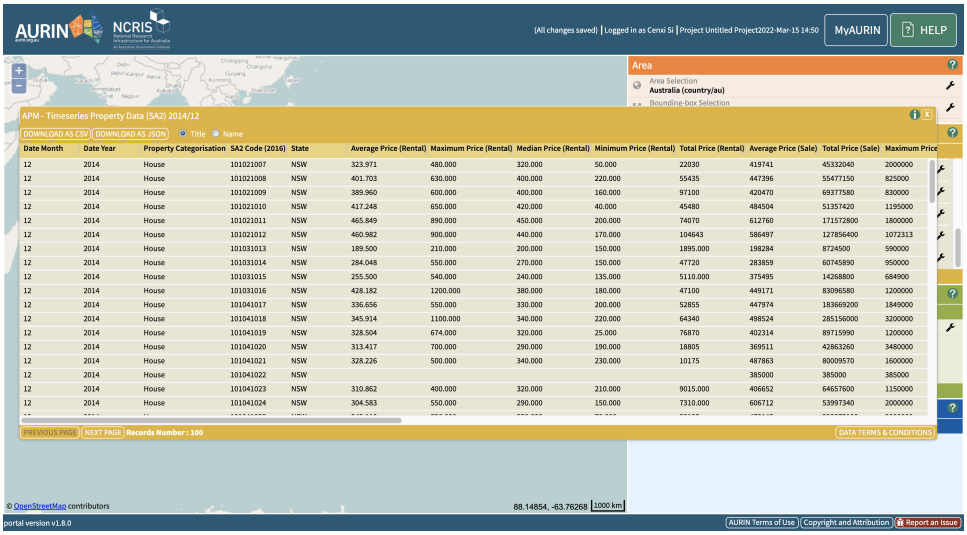


Figure 8: AURIN-housing dataset

Income dataset is provided by AU_Govt_ABS which is the Government of the Commonwealth of Australia - Australian Bureau of Statistics. The dataset comprises the income of employee from 2014 to 2015 and is aggregated to the 2016 Statistical Area Level 2 (SA2) boundaries. It includes 8 attributes: SA2 Code 2016, Median Income, SA2 Name 2016, SA2 5 Digit Code 2016, Mean Income, Total Income, Income Earners (persons) and Main source (of earners). Figure 9 shows the structure and partial data.

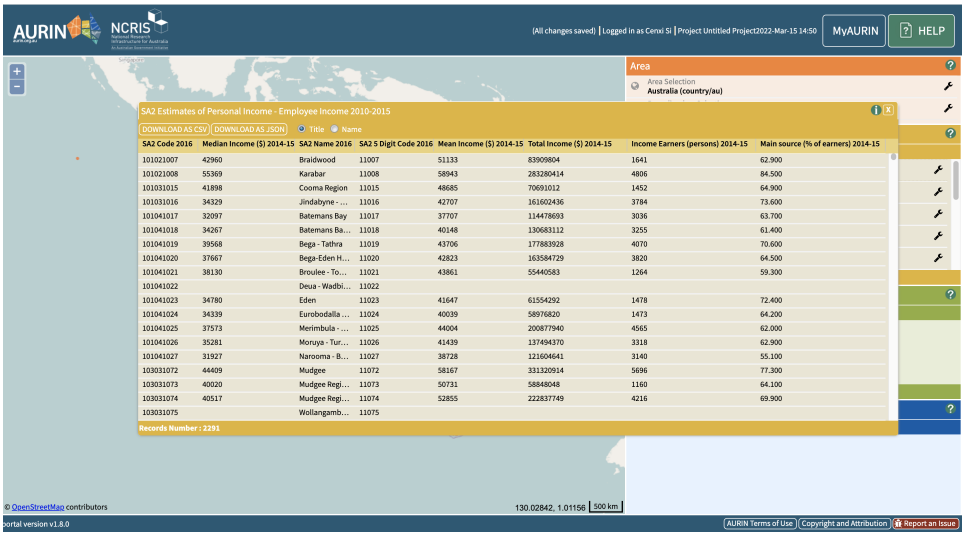


Figure 9: AURIN-income dataset

6 Data Storage

CouchDB is one of the document-oriented Database Management Systems which has an HTTP-based ReST API that makes communicating with the database, it stores data as JSON or XML, and supports computing capacity distribution across multiple computers [2].

6.1 CouchDB Setup and Finish Setup

During the system architecture, there are three instances setting up CouchDB. Before creating docker containers,the system needs to make sure there is no existing containers running. While creating, ibmcom/couchdb3 docker image with CouchDB version 3.2.0 is chosen. Apache CouchDB 3 image is maintained by IBM cloudant team. The container is mainly used with Operator for Apache CouchDB and compatible with CouchDB Helm Chart. It also works independently and follows the same instructions as community-maintained Apache CouchDB container [3]. In addition, the required ports are 5984 which is for CouchDB database server [5], 4369 for Erlang Port Mapper Daemon [Erlang] [4] and 9100-9200 for PDL Data Stream [6] and all API calls over HTTP [7]. Then, the system sets the proper environment and setup the CouchDB. Furthermore, the steps of setting CouchDB cluster could be divided into enabling cluster, adding nodes to CouchDB cluster and getting master node. Once the three nodes added on CouchDB, the system needs to finish CouchDB cluster setup.

6.2 CouchDB Databases

There are four dataset stored in CouchDB which are housing and income datasets obtained from AURIN, tweets with prepossessed information collected from twitter harvester located in Melbourne and Sydney. Figure 10 illustrates the specific information about the four datasets in CouchDB. “housing_2014” represents the housing price in 2014 dataset from AURIN and “income_2014” is the dataset of average income in 2014 from AURIN. “twitter_sentiment” stores the tweets from Melbourne and “twitter_sentiment_syd” stores the tweets from Sydney.

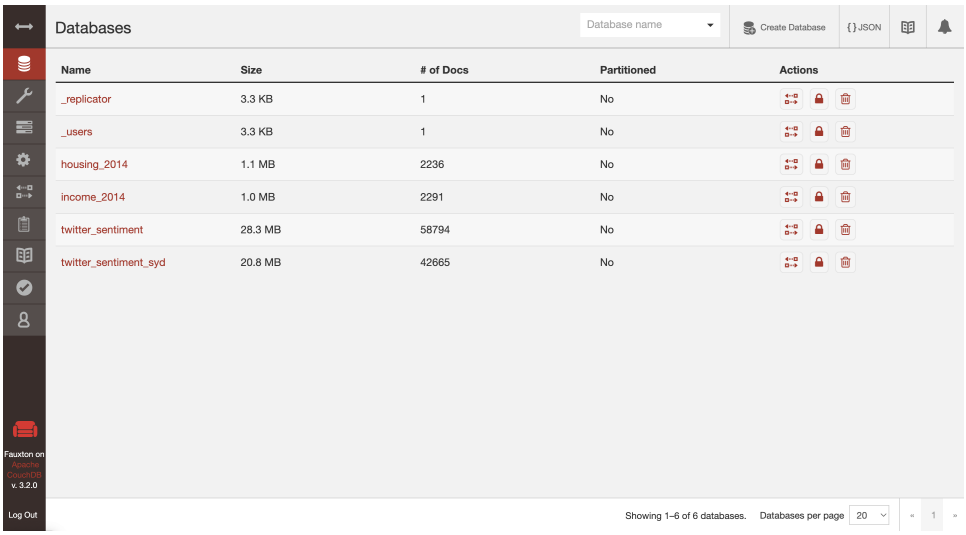


Figure 10: CouchDB datasets

7 Data Prepossessing and Analysis

7.1 Twitter Data

7.1.1 MapReduce

Based on the harvested tweets stored in CouchDB, MapReduce is useful to primarily analyze the sentiments distribution among tweets. To achieve the parallelism on a large data set, we use MapReduce. Firstly, distribute data across machines to "Map", and then summarize when the result is obtained hierarchically which is "Reduce". This also can help reduce the network traffic when moving the process to data. Since the twitter harvester has already categorised sentiments of each tweets while collecting, MapReduce (Figure 11)is used for counting the number of each type of sentiments in Melbourne and Sydney and returning the result. The statistical result (Figure 12)is obtained by iterating rows of view in CouchDB and stored into a predefined dataframe including sentiment type, count number and its proportion.

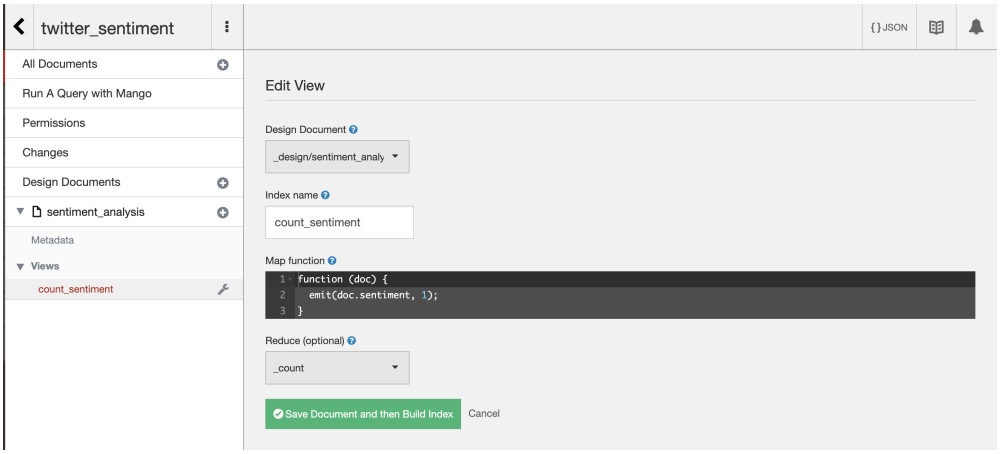


Figure 11: MapReduce Function

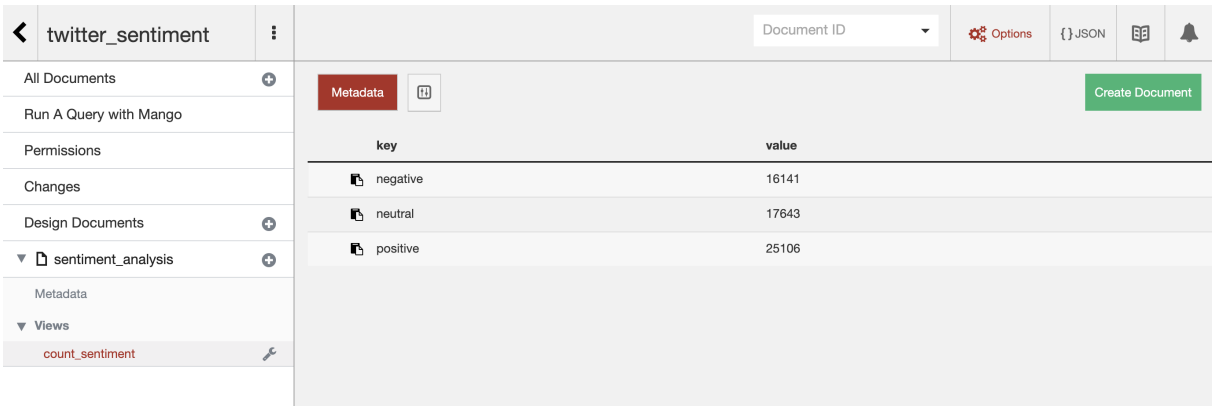


Figure 12: MapReduce Result of Melbourne

7.1.2 Sentiment Analysis on Twitter Data

The pie charts (Figure 13) reveal the percentages of each kind of sentiments distributed in tweets from Melbourne and Sydney. It shows that the proportion of negative sentiments in Melbourne is approximately 27.0% which is slightly lower than 28.0% in Sydney. Although the rate of positive sentiments in Melbourne is 0.8% less than Sydney, the integration of negative and positive sentiment tweets illustrates people in Melbourne response a slightly more positive reaction on house price and income.

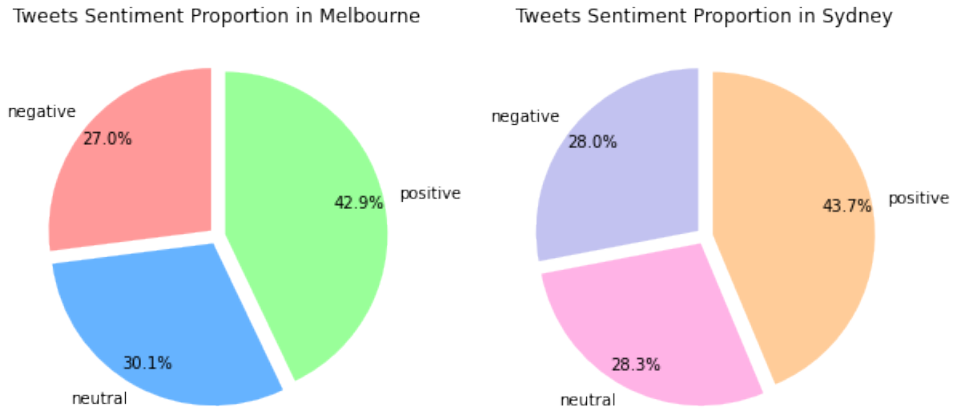


Figure 13: Sentiment frequency of tweets in Melbourne and Sydney

7.1.3 Key Word Analysis on Twitter Data

Furthermore, the word cloud could reveal the most frequent words appeared in tweets from Melbourne and Sydney. The prepossessing of tweets' texts includes removing punctuation, RT, stop-words, stemming and tokenization. Then, the word clouds are created based on the cleaned texts. Figure 14 shows the word clouds based on 3000 words from tweets in Melbourne and Sydney about housing

price and income. Figure 15 summarizes the top 10 frequent words from tweets in Melbourne and Sydney. It could analyze from these two kinds of graphs that people in Sydney might be more anxious about tax, payments and profit which are relevant to income quality. Thus, in general, people in Melbourne are more positive to the quality of income and level of house price which reveals a higher level of liveability.

Figure 14: Word Clouds of Melbourne and Sydney

Melbourne	
home	4116
pay	3631
interest	1614
get	1469
cash	1236
hous	1214
go	1189
amp	1167
like	1104
one	1027

Sydney	
pay	3436
home	2066
interest	1192
amp	1056
get	974
tax	817
rate	784
cash	767
like	744
profit	743

Figure 15: Top 10 Frequent Words of Melbourne and Sydney

7.1.4 Limitation on Twitter Data

Due to the limited speed and collecting time of twitter harvester, the total amount of tweets collected is around 100000 which might be not enough for the sentiment analysis of housing and income on tweets. Also, since the tweet searching tool is collecting the real-time data within 7-day period, the time span of twitter data is not wide enough to support the comprehensive analysis on sentiments from tweets and assist the analysis on AURIN data.

7.2 AURIN Data

7.2.1 Analysis of AURIN Data and Graphs

Due to the large amount of dataset size, we chose to treat data for December 2014 as a sample to represent the housing dataset. In the dataset, the SA2 5-digit code 2016 version is used to distinguish each geographical region. We first calculated the average income and house prices by each region. In order to find the rate between average house price and personal income, a division was conducted between them. By comparing the rates of Sydney (SA2 code:11298-11444) and Melbourne (SA2 code:21105-21385), from the bar chart 16, it is observed that Melbourne has a lower rate which means one person could be able to purchase a house in a smaller number of working years ideally.

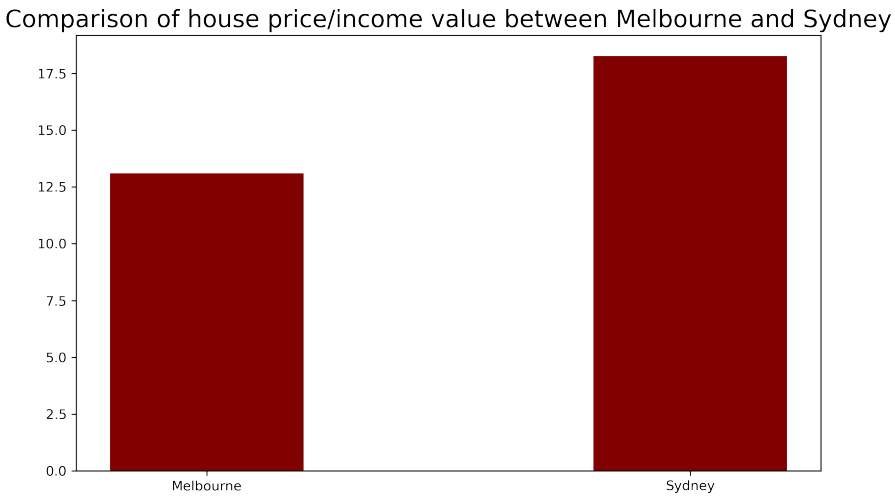


Figure 16: Comparison of House Price/Income between Melbourne and Sydney

Turning to the top 10 suburbs in Melbourne which have the smallest rate, they separately are Melton West, Wyndham Vale, Melton, Melton South, Hoppers Crossing - South, Sunbury - South, Wallan, Truganina, Cranbourne West, Seabrook. Melton West ranks the first with 6.42 and Wyndham ranks the second with 6.49. Leumeah - Minto Heights takes first place in Sydney with the rate of 8.09. The detailed top 10 suburbs list 19 in Sydney can be found in the picture along with the bar chart.

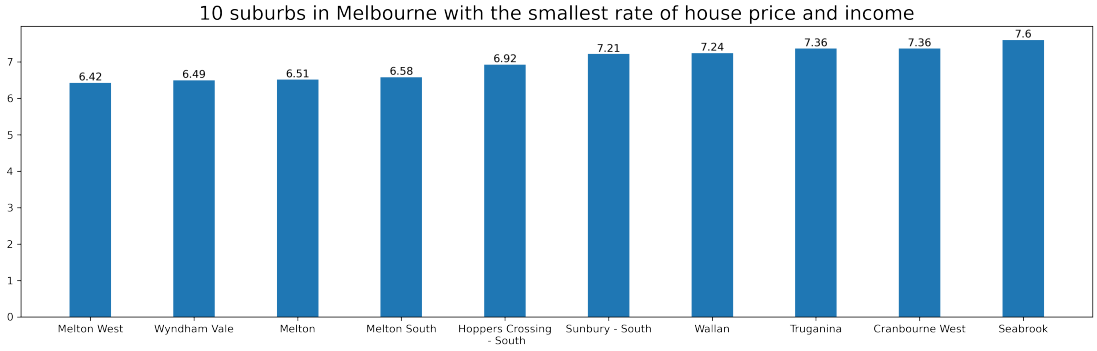


Figure 17: 10 Suburbs in Melbourne with the Smallest Rate of House price and Income

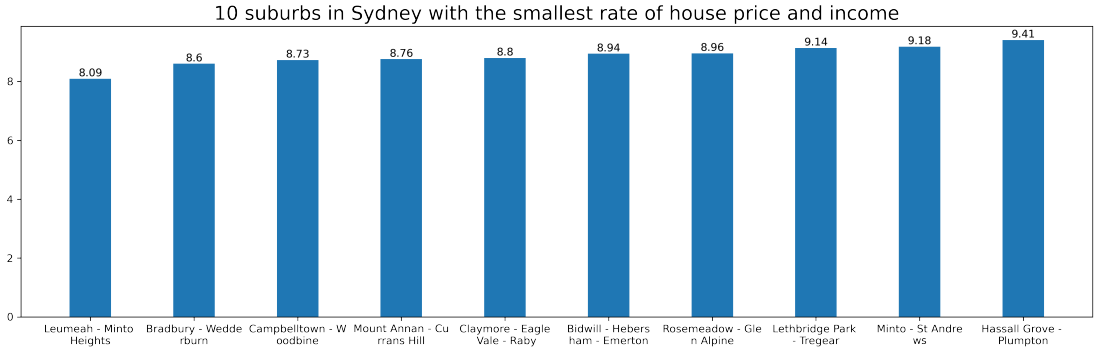


Figure 18: 10 Suburbs in Sydney with the Smallest Rate of House Price and Income

```
458          Leumeah - Minto Heights
454            Bradbury - Wedderburn
455          Campbelltown - Woodbine
453       Mount Annan - Currans Hill
456     Claymore - Eagle Vale - Raby
322    Bidwill - Hebersham - Emerton
461         Rosemeadow - Glen Alpine
325        Lethbridge Park - Tregear
460               Minto - St Andrews
324         Hassall Grove - Plumpton
```

Figure 19: Names of Top 10 Suburbs in Sydney

It is obvious to find that Melbourne has an average smaller rate between house prices and personal income from both citywide and suburb-wide bar charts. The maps 20&21 created on HTML share the same outcomes. The overall tone of colours in Melbourne is lighter than that of Sydney. In addition, Melbourne has a smaller number of zones colouring mahogany compared to Sydney.

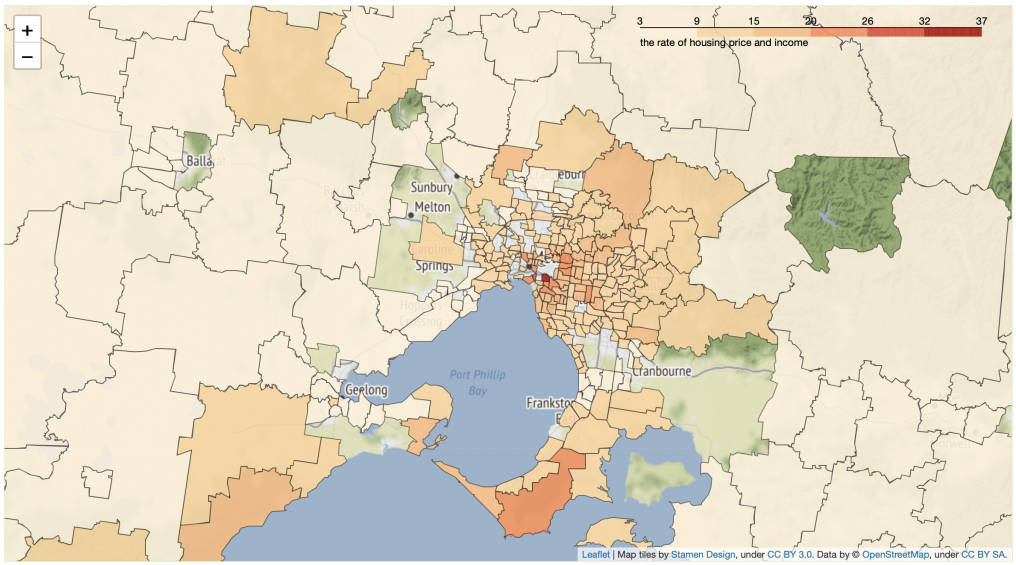


Figure 20: Map of Melbourne

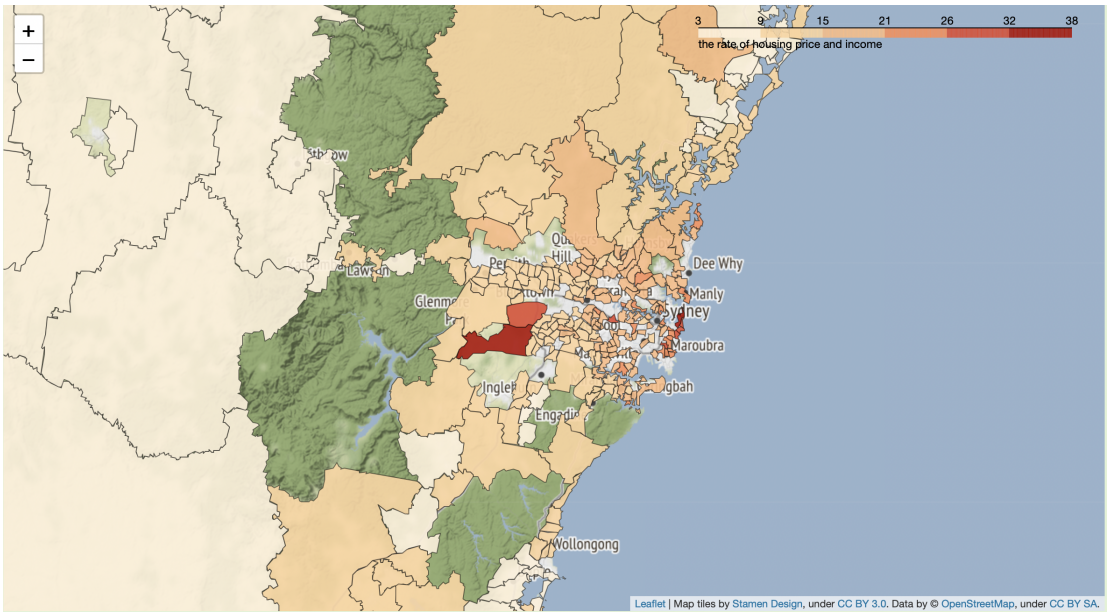


Figure 21: Map of Sydney

7.2.2 Limitations of AURIN Data

The drawbacks of our results may be divided into several aspects. The first main disadvantage is that the sample of data has a short time span, which would increase the variance between the calculated result and the actual condition. In the other words, it would affect the credibility of our assumption. The small size of the sample would follow to be the second weakness. Both of these factors would influence the conclusion that we obtain from this report.

8 Web Application

8.1 Overview

Users of the system can operate on the web page to get updated information, which is the main web-based data visualisation platform. By sending queries through different routers, our web could display analytical information from the designed database system. The server is responsible for fetching and processing data from the CouchDB database, while the client end is responsible for showing the data analysis output in a web-based application built by the web template Handlebars and several libraries. These utilised tools (a ReST API server and client frontend) help us construct the application and now users could view current results.

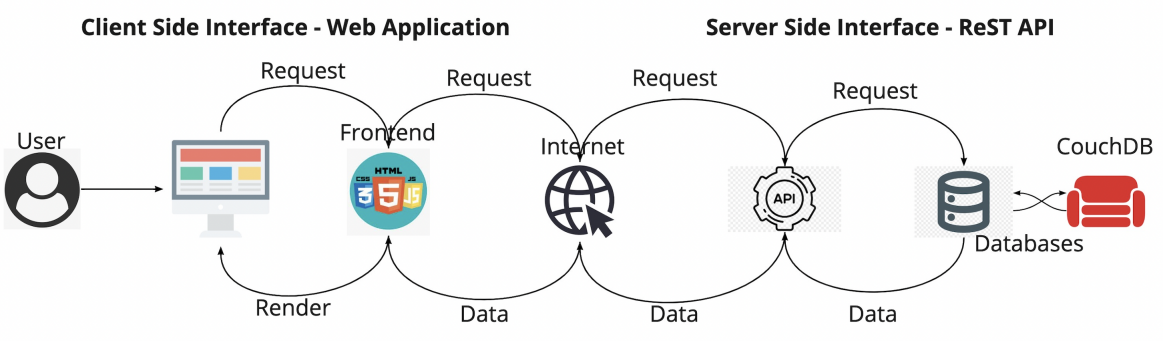


Figure 22: General Process for Building the Web Application

8.2 Server Side Interface

The server side is deployed from the main App.js making retrieving results from databases on CouchDB easier. The coding language used is JavaScript, and the web development framework has the ability to deliver useful stuff through a Representational State Transfer (ReST) API that allows the frontend website to quickly access specified datasets from wanted databases. In general, App.js is in charge of receiving requests from the user side and returning the required analysis, which is obtained from the CouchDB database. The server works by calling an HTTP server from the command line and then visiting the localhost which is suitable for deployment.

We are focusing on the analysis of two cities and correspondingly store relevant information and filtered data in separate databases on CouchDB. The database housing_2014 is storing the data which we set the particular query of house price, while the database income_2014 is processing data regarding individual's income. The databases twitter_sentiment and twitter_sentiment_syd contain valuable data that contributes to the web's table analysis. These databases are connected so that the server could gather all the information and then pass it through routers to the user side, and then be summarized. The work needs MapReduce functionality to obtain the counts of tweets under the three sentiment labels and the calculated percentage. The connection to information is detailed by App.js as introduced before. Regarding the implementation of ReST API for data visualisation, it works as the data transport between the user/client-side and the remote resource from CouchDB. We package the result data into a DataFrame to the front after retrieving the data as described in section 7.1.1 and then implement GET request.

8.3 Client Side Interface

Our group have the web page implemented by Handlebars which is one kind of view engine of express, as it looks for pages in the "views" folder in the source directory where we could put HTML code into the ".hbs" extension file. In addition, it is from the express-handlebars package and then can be applied to create reusable web templates. The templates are a combination of HTML, text, and expressions. We have configured the main application file to make use of the template engine, then create the necessary directories and the basic layout file to write any Handlebars views. At the concept level, a general Node.js sets up the environment for running JavaScript, npm is its default package at first. Furthermore, express is one of the frameworks in Node.js and then express-handlebars serve as a view engine.

Handlebars have become an open-source that can be used to create single-page or mobile applications, rendering the client side is one of the most significant operations in our design. For each time user refreshes the web, our application would be updated to render the written Handlebars views, passing each view the data that is wanted to be displayed on prescribed pages. In each of the HTML pages being rendered, data has been captured by assigning a variable to each endpoint in JavaScript. This application allows us to display the results from data analysis and users could fresh the page to fetch updated data from wanted databases. With the implementation of several handlebars files and the use of CSS files to display the design, the front-end website is able to demonstrate our findings through diagrams and tables. More precisely, users can enter Index Page, AURIN Page and Tweets Page to observe the outcomes of data in the form of map graphs, bar charts, pie charts and word cloud diagrams.

8.4 Web Pages

Our application is comprised of four pages:

- **Home** Demonstrates the choropleth visualisation of cities and provides summary on the amounts of tweets for different sentiment types through the table
- **AURIN** Presents an overview on the house price and income ratio in terms of bar charts
- **Tweets** Designed to show visualisations of tweet in the form of word clouds and pie charts
- **Members** Where we provide the basic information of each of the team members

8.5 Issues & Challenges

The most challenging task we are facing when designing the web application is the connection to CouchDB. The generated maps are rendering tweets from two cities in the format of HTML, and we plan to obtain an updated summary which is calculated automatically from the MapReduce function on CouchDB. This is also one of the critical missions of this project, thus we put a lot of effort into designing a balanced application between the frontend and backend.

Initially, we focused on developing a backend server using Flask, which is a small web hosting framework that lacks the scalability and flexibility required for large data projects. More crucially, we had begun producing HTML with Handlebars, making it harder for a Python backend to link to the design. Express Handlebars has already structured the system easily and enables us to code the frontend statically.

We gave up on creating a clear divide seen between frontend and backend in our work. We reconsidered and now have a combination of the server/user interface. This approach overcame the challenges of using separate libraries for different programming languages while keeping the application consistent.

9 Team Collaboration

This assignment was completed by a well-integrated team and each team member was assigned an area of responsibility, which is detailed in the table below. Throughout accomplishing our assignment design, Zoom virtual meetings were frequently hosted to catch up with the timeline.

Team Members	**Tasks Performed**
Cenxi Si	Worked on CouchDB setup with Yipei Liu, tweet collection, tweet searching, analysis and making MapReduce functions on the CouchDB
Yipei Liu	Worked on MRC instances and Ansible scripts, CouchDB cluster deployment, Docker containers and AURIN data analysis on map
Ruimin Sun	Worked on tweet streaming, AURIN data extraction, and ran scenario analysis together with Yipei Liu and Cenxi Si
Jingdan Zhang	Worked on the Web server/user construction, HTML rendering and up-to-date connection to the CouchDB
Chengyan Dai	Worked on the design of the Web-based data visualisation platform together with Jingdan Zhang

Python was chosen to be the primary programming language for most of the sections since it was familiar to all members of the team. However, we are unfamiliar with the front-end web application for visualising datasets and chosen scenarios. As a result, we started following online tutorials and picked JavaScript and HTML to achieve that.

10 Conclusion

Throughout the project, we experienced multiple hurdles, but we were able to overcome them and plan strategically for identifying and analysing our issues. This report provides some details concerning error handling. This assignment taught us that increasing the volume of data collected and involving several collection occurrences throughout time may be beneficial in the future for better capturing and analysing sentiment and modelling particular themes.

The initial task is to successfully develop a Cloud-based solution that exploits a multitude of virtual machines (VMs) across the Melbourne Research Cloud (MRC). Harvesting tweets using Search API interfaces is additionally pointed out by the assignment instruction, and then we are asked to store our collected tweets in CouchDB databases. Following that, we conducted more research using MapReduce which is a coding function for analysing massive amounts of data, and it works by utilising a suitable filter that can sort datasets and contributes to our research case.

In the final step, we used Handlebars and JavaScript frameworks to create a web-based data visualisation platform that comprised all our findings as described in previous sections. Each HTML page is rendered using a combination of tools mentioned in the introduction session on the frontend architecture. And at last, our Web application included four pages: the Cholopleth visualisation and table of sentiment analysis performed (Home), key characteristics about house price and income of our collected data (AURIN), display of analytical outputs (Tweets), and profiles of each of the team (Members).

The research result of our scenario helps us summarize that Melbourne is a liveable city. Though the rate of positive sentiments in Melbourne is lower than that of Sydney, our integration of negative and positive types still tells that Melbourne residents response more positively towards the two factors. In addition, another part of key word study shows same result that people in Melbourne are more positive to income quality and housing. Furthermore, AURIN data shows people in Melbourne are more affordable housing than Sydney. To be concluded, the analysis difference between the two cities could be ignored and Melbourne performs better in this liveability comparison with Sydney.

Throughout the project, we experienced multiple hurdles, but we were able to overcome them and plan strategically for identifying and analysing our issues. This report provides some details concerning error handling. This assignment taught us that increasing the volume of data collected and involving several collection occurrences throughout time may be beneficial in the future for better capturing and analysing sentiment and modelling particular themes.

References

[1] " ARUIN " AUSTRALIA'S SPATIAL INTELLIGENCE NETWORK. Accessed May 1, 2022. https://aurin.org.au/.

[2] " lecture 13-14 " Big Data and CouchDB. COMP90024. University of Melbourne.

[3] " ibmcom/couchdb3 " Docker Hub. Accessed May 4, 2022. https://hub.docker.com/r/ibmcom/couchdb3.

[4] " Port 4369 Details " Speed Guide. Accessed May 4, 2022. https://www.speedguide.net/port.php?port=4369.

[5] " Port 5984 Details " Speed Guide. Accessed May 4, 2022. https://www.speedguide.net/port.php?port=5984.

[6] " Port 9100 Details " Speed Guide. Accessed May 4, 2022. https://www.speedguide.net/port.php?port=9100.

[7] " What are ports 9200 and 9300 used for? " Elastic. Accessed May 4, 2022. https://discuss.elastic.co/t/what-are-ports-9200-and-9300-used-for/238578.

[8] "Most used social media 2021" Statista. Accessed May 11, 2022. https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/

[9] "AARP livability index - great neighborhoods for all ages". AARP. Accessed May 11, 2022. https://livabilityindex.aarp.org/scoring#categories