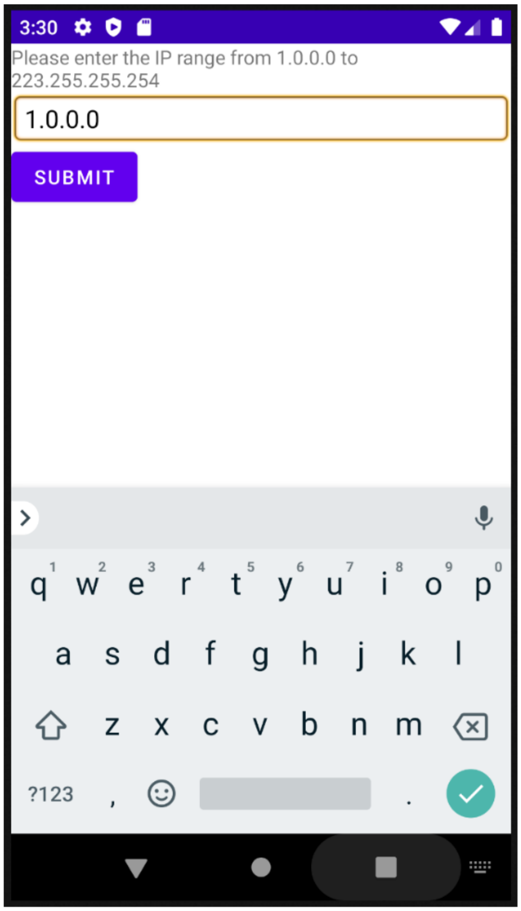
Web Service Logging and Analysis Dashboard
By Isabelle Huang

URL: [Web Service Dashboard](https://protected-scrubland-77046.herokuapp.com/dashboard)

Description:
This project involves a web service deployed on Heroku that provides logging and analysis capabilities. Users can input an IP address, which is then passed to the Heroku web service. The web service retrieves data and saves it to a MongoDB database. If the URL contains "/dashboard," the web service fetches data from MongoDB, performs calculations, and displays operational analysis and log details on the "result.jsp" page.

1.2 User Input:
Here is a screenshot of a user searching for an API using the IP address "1.0.0.0."



Please enter the IP range from 1.0.0.0 to 223.255.255.254

1.0.0.0

1.3 HTTP Request to Web Service:
The application makes an HTTP GET request in the "MyIP.java" file with the following URL:

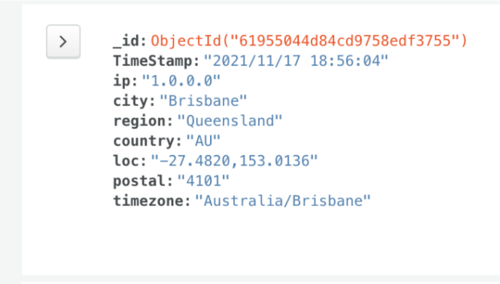
```
"https://protected-scrubland-77046.herokuapp.com/result?searchTerm=" + searchTerm
```

The "searchTerm" is the user's input. If the "searchTerm" is not null, the IP model fetches the data and saves the newly entered data into MongoDB. If the URL contains "/dashboard," the servlet fetches data from MongoDB, performs calculations, and displays operational analysis and log details on "result.jsp."

1.5 Displaying Information and Saving to MongoDB:
After the IP information is returned, it is saved to MongoDB, including fields such as "id," "TimeStamp," "IP," "city," "region," "country," "location," "postal," and "timezone."

Data in MongoDB:

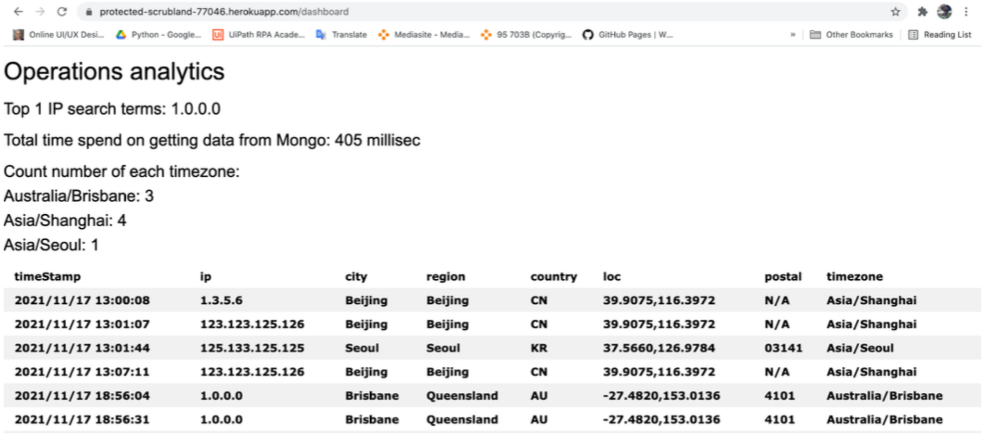
The record for "1.0.0.0" is saved in MongoDB.



```
_id: ObjectId("61955044d84cd9758edf3755")
TimeStamp: "2021/11/17 18:56:04"
ip: "1.0.0.0"
city: "Brisbane"
region: "Queensland"
country: "AU"
loc: "-27.4820,153.0136"
postal: "4101"
timezone: "Australia/Brisbane"
```

Data in Dashboard:

The record for "1.0.0.0" is displayed on the dashboard.



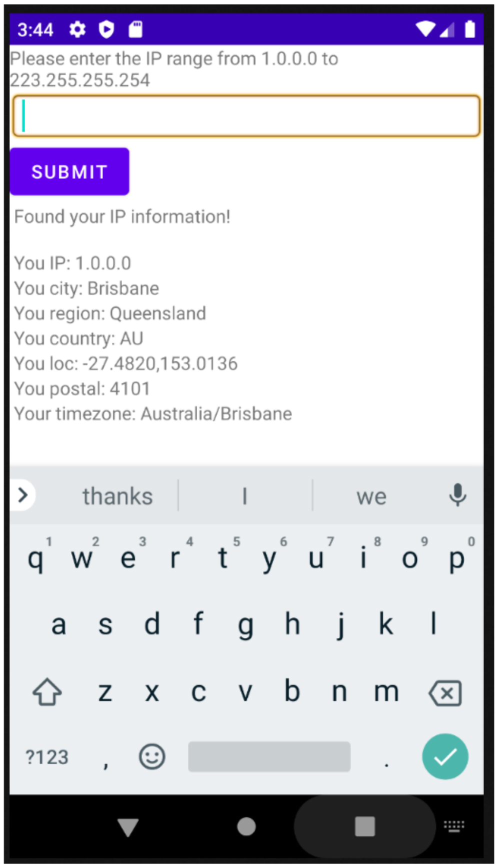
Operations analytics

Top 1 IP search terms: 1.0.0.0

Total time spend on getting data from Mongo: 405 millisec

Count number of each timezone:
Australia/Brisbane: 3
Asia/Shanghai: 4
Asia/Seoul: 1

timeStamp	ip	city	region	country	loc	postal	timezone
2021/11/17 13:00:08	1.3.5.6	Beijing	Beijing	CN	39.9075,116.3972	N/A	Asia/Shanghai
2021/11/17 13:01:07	123.123.125.126	Beijing	Beijing	CN	39.9075,116.3972	N/A	Asia/Shanghai
2021/11/17 13:01:44	125.133.125.125	Seoul	Seoul	KR	37.5660,126.9784	03141	Asia/Seoul
2021/11/17 13:07:11	123.123.125.126	Beijing	Beijing	CN	39.9075,116.3972	N/A	Asia/Shanghai
2021/11/17 18:56:04	1.0.0.0	Brisbane	Queensland	AU	-27.4820,153.0136	4101	Australia/Brisbane
2021/11/17 18:56:31	1.0.0.0	Brisbane	Queensland	AU	-27.4820,153.0136	4101	Australia/Brisbane



Please enter the IP range from 1.0.0.0 to 223.255.255.254

Found your IP information!

You IP: 1.0.0.0
You city: Brisbane
You region: Queensland
You country: AU
You loc: -27.4820,153.0136
You postal: 4101
Your timezone: Australia/Brisbane

1.6 Operation Analysis:
The operation analysis includes:
1. Top 1 IP search terms.
2. Total time spent on retrieving data from MongoDB.
3. Count of the number of each timezone.

1.7 Log:
The log includes the following information:
1. Timestamp: The time the user entered the data.
2. IP.
3. City.
4. Region.
5. Country.
6. Location.
7. Postal.
8. Timezone.

2. Implementing a Web Application Deployed on Heroku:

The URL of the web service deployed on Heroku is "https://protected-scrubland-77046.herokuapp.com." The project directory is named "MyIP."

2.1 Dashboard:
In my web app project, the structure follows the Model-View-Controller (MVC) pattern:

- Model: "IPmodel.java."
- View: "result.jsp."
- Controller: "MyIP.java."

2.2 Fetching Data from MongoDB and Displaying on the Dashboard:
When "MyIP.java" receives an HTTP GET request with a URL containing "/dashboard," it connects to MongoDB, fetches collections, and passes the data to the model. The model parses the data, performs operation analysis, and sends the results back to "MyIP.java" for display on the dashboard.

2.3 Receiving HTTP Requests from a Native Android Application and Saving Data to MongoDB:
When "MyIP.java" receives an HTTP GET request and the "searchTerm" is not null, it connects to a third-party API, retrieves JSON data, parses it, and saves the data to MongoDB.