Universitat Pompeu Fabra Barcelona

IRWA FINAL PROJECT

USER INTERFACE AND WEB ANALYTICS

GROUP 6

Jordi Guillén: u198641
Anira Besora: u189647
Ana Cereto: u199767
Marina Castellano: u188311

INDEX

Github/ Tag: https://github.com/Marinagrup2/IRWA.git

1. User Interface

The web page where our user will search for their queries and find the tweets that have the most relation with it.

1.1. Data reading:

We have used the already filtered .json "tweets-data-who.json". This .json has, for each tweet, the following structure: {"id", "date", "url", "likeCount", "retweetCount", "content", "hashtags"}. We changed the definition *_load_corpus_as_dataframe(path)* so it reads the .json, transforms it into a DataFrame, and then changes the name of the rows so it's more clear when transforming it into the object Document. The final structure is {"Id", "Date", "Url", "Likes", "Retweets", "Tweet", "Hashtags"}. One thing we have done and decided to keep this way, contrary to what we had done previously, is to not mix the "Tweet" with the "Hashtags". We have concluded that the content of the tweet could be more interesting rather than only reading the hashtags. For instance, a tweet could only contain hashtags and the user probably would not be satisfied if they wanted to read about people's opinions.

1.2. Ranking algorithms:

We have implemented the 4 algorithms of the 3rd part of this practice. Those algorithms are:

- TF-IDF (Term FRequency - Inverse Document Frequency) + cosine similarity: This method calculates the product of the term frequency (TF) of a term in a document and the inverse document frequency (IDF), the inverse of the frequency of a term across all documents in the collection, giving less weight to common terms, so taking into account the rarity of the term. Then we compute the cosine difference between them to see if they are more or less related.
- Our score (count of likes has retweets) + cosine similarity: In this algorithm, we make a score based on 3 different parameters. This ranking method prioritizes the similarity between the query and document content, placing a higher weight on this parameter, with cosine similarity while also incorporating tweet popularity within each document. Popularity, in this context, is determined by the number of likes and retweets.

- BM25: BM25 is a technique that uses the IDF of the term but introduces two parameters, k1 to control the term frequency, so that high term frequency terms don't increase the score too much, and b for document length regulation, so there is no bias toward longer docs.
- Word2Vec + cosine similarity: For this method, we create embeddings for the tweets in our documents. For each tweet and query, we position them in the n-dimensional space and compute the cosine similarity to determine how close, and thus how similar, they are. However, this is by far the worst method of the four. Not only is it way slower, but the results are way off, and mort time has nothing to do with the query.

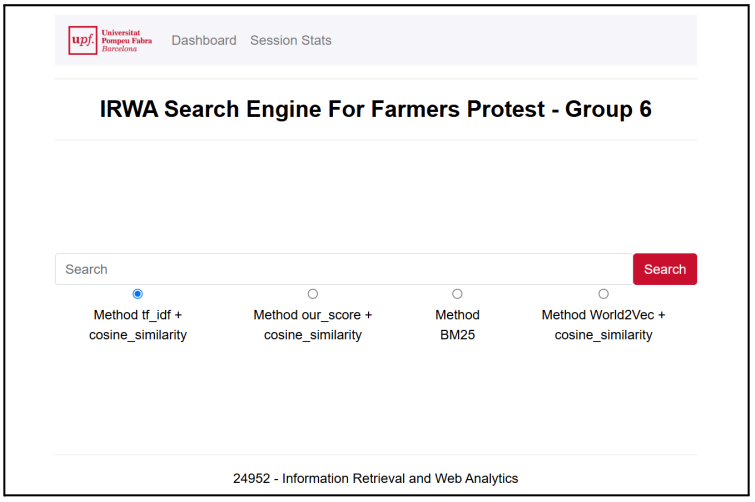
1.3. Executing the page:

The main flow in which our page is: Main page with the search bar ⟶ Page with the documents for the result ⟶ The content of the page ⟶The original tweet (or other).

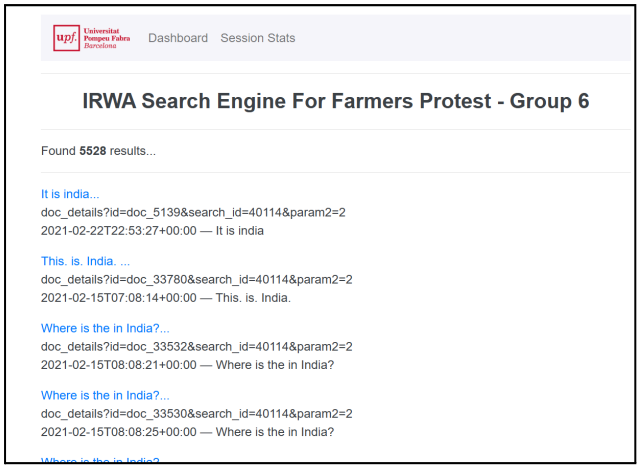
Main page:

On this page, you do the search for your query. Before searching for your query, you must select one of the methods under the search bar.

The first three (method tf_idf + cosine similarity, method our_score + cosine similarity, and method BM25) will show you good results. However, the last one (method Word2Vec + cosine_similarity) will not only be slower but also show results that have almost nothing to do with the query (we do not recommend using it).



You can also see the Dashboard and the Session Stats by clicking them directly here (the explanation for what is on both pages is done in the Web Analytics part).



Results page:

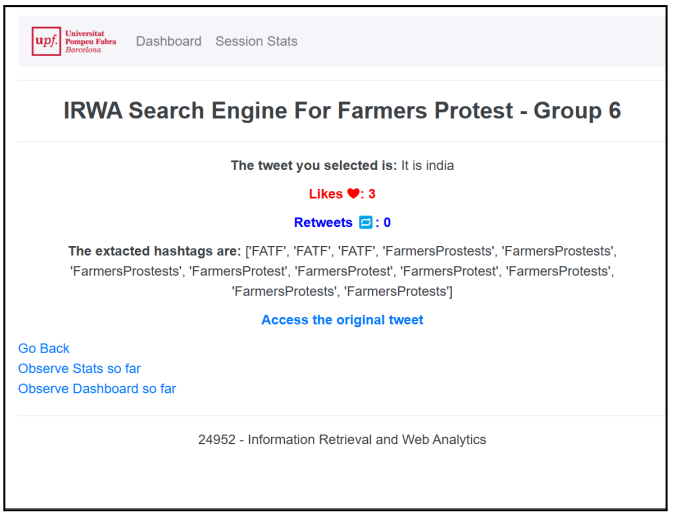
Once you have searched your query (for this example we have used “india”) a list of results will show. It will tell you the number of total documents it found relation with, but will only show you the top 20. This is because if we loaded all of them the web would freeze and crash.

Each result will show a brief title of the tweet, some document details, the date of the creation, and the tweet itself.

Document information:

Once you click the title, the link will redirect you to the information on the document. This information will tell you, in a more clean and structured way, the tweet, the likes, the retweets, and the hashtags and will let you access the original tweet by clicking on “Access the original tweet”.

It also will let you go back, observe the statistics so far or observe the dashboard so far by clicking each line



Warning: This database is quite old, so the chances of accessing an original tweet and not finding it are quite high.

2. Web Analytics

To track and analyze how the users navigate our search app, we decided to track the following aspects:

1. Clicked documents.
2. Session start and end time.
3. Searched queries.
4. Browser, OS and IP address of visitor.

In analytics Data, we initialize it with different variables so we can store the clicks, queries, clicked docs and sessions. There we have different functions such as save query terms, save click data, etc so we can append the clicked data on the variables we first defined.

For the analytics we store the data in a csv file so we could access the information as if it was a database but in an easier way. In the file analytics_datatase.py, we initialize the csv and if it does not exist, it creates it and it writes the first row with the headers of the info we wanted to keep: session_id, ip_address, browser, operating_system, timestamp, query, doc_id, title and description. In that file we have the following functions:

- append_to_csv: to write a new row
- save_session: with the session id, ip, user agent and start time we get the browser and operating system of the user and store it as a new row
- save_query: similar to the previous function, we store the query of the user and when it was written
- save_click: here instead of the click we save the document that the user clicked, the title and the description

Once the csv is initialized, when loading the home page we wanted to check the user's session id, firstly by checking if it existed. If it did, we checked the last timestamp we have stored in the csv to see since when there was inactivity in that session. If the difference was less than 2 hours, the session is still in place, but if longer time has passed, a new session is created (like if there didn't exist any session) and we use save session for our analytics_csv. We also have a function called generate_realistic_ip that randomly assigns an ip address to the session with ranges from europe, asia and america so we could display more variety in our dashboard, as we are all from spain.

Finally, to get the query or docs clicked data, we get them from the search page and doc details page respectively, by using the save query and save click functions.

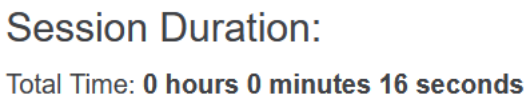
We displayed this data in the following 3 pages:

2.1. Session Stats

In this page we analyze the data from the session of the user:

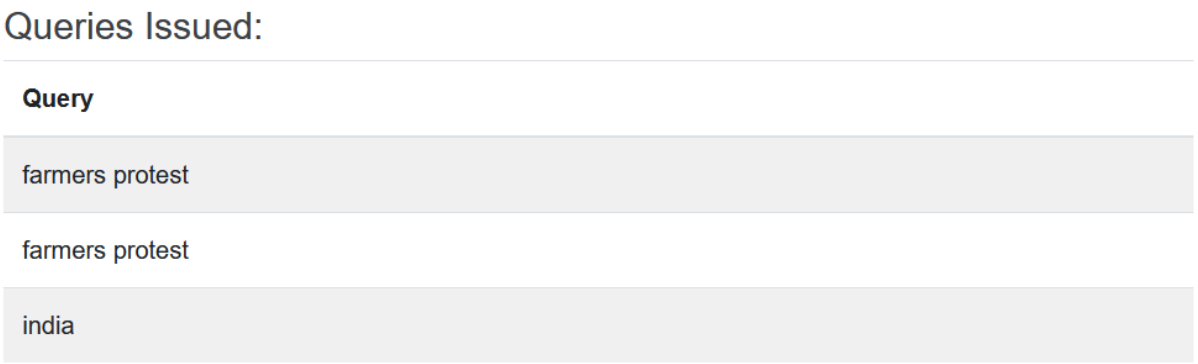
- **Session Duration**:

The session statistics display the total time of a user session (duration from the start of interaction to the current time). This helps analyze how long users spend exploring the website in a single "sit-down" session. For example, this session lasted only 16 seconds (as it was created only for display reasons) but a real one would go for longer.



- **Queries During the Session**:

All queries submitted by the user in the session are recorded and displayed. This allows tracking of search patterns and the relevance of clicked documents to submitted queries.



- **List of Clicked Documents**:

Documents clicked during the session are displayed, including details such as the document title, description and timestamp. There is also an additional column where we show the time difference of the click regarding the previously visited document, so we can see for example, if the document was

useful for the user's information needs whether they spent minutes or seconds before clicking another document.

The list is ordered by the timestamp of clicks, showing the sequence in which the documents were accessed.

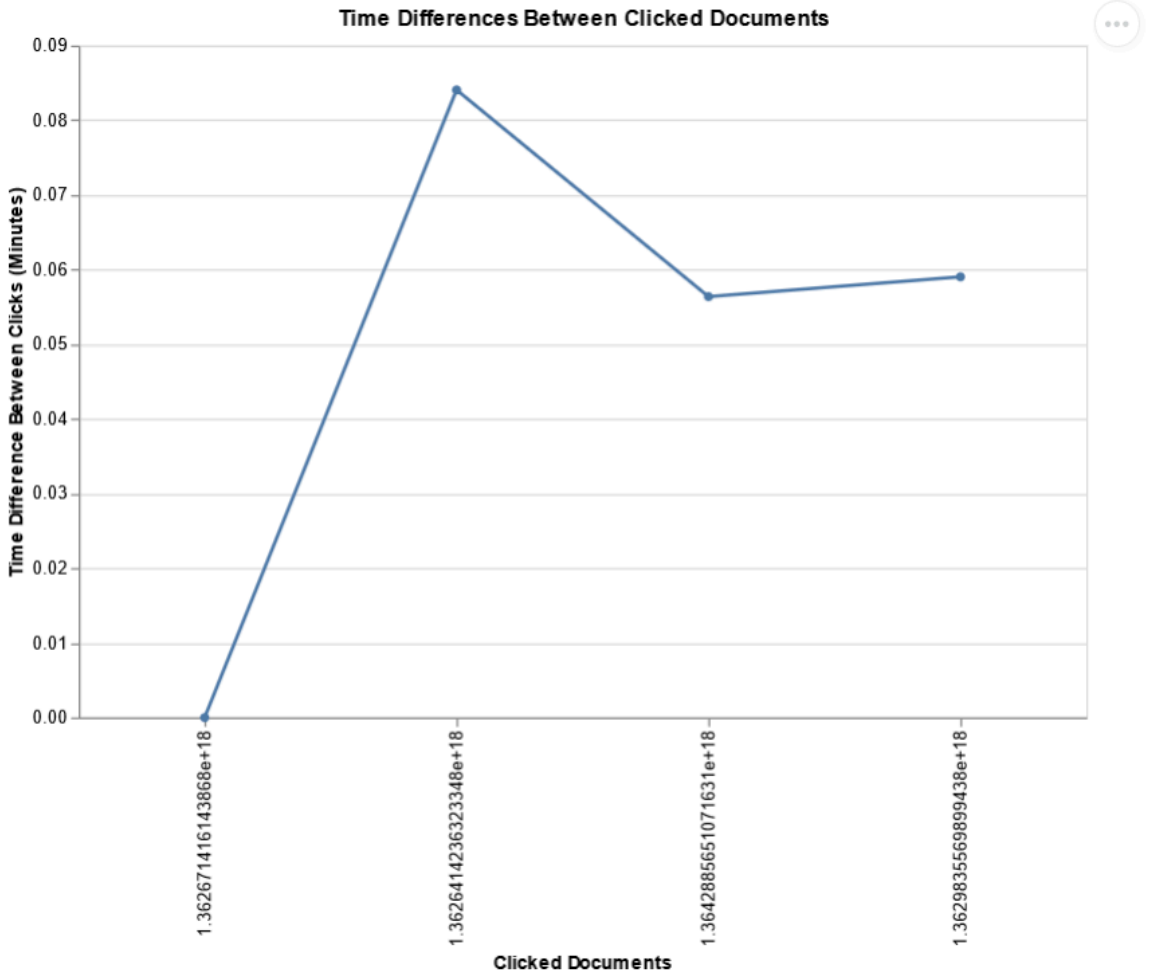
Clicked Documents:

Document ID	Title	Description	Timestamp	Time Difference (min)
1.362671416143868e+18	Hi @UniofOxford can you please update in dictionary "BJP = Bigotry" #bjp #bjplies #bjpfails #Petro	Hi @UniofOxford can you please update in dictionary "BJP = Bigotry" #bjp #bjplies #bjpfails #PetrolDieselPriceHike #ModiGlobalDisaster #FarmersProtest #PetrolPrice https://t.co/OnObycsEol	2024-11-30 15:50:10	0.0
1.3626414236323348e+18	@Harpree98409095 #ReleaseDetainedFarmers #FarmersProtest https://t.co/0W66GMmJa5	@Harpree98409095 #ReleaseDetainedFarmers #FarmersProtest https://t.co/0W66GMmJa5	2024-11-30 15:50:15	0.08406883333333333
1.3642885651071631e+18	True #FarmersProtest #Pagdi_Sambhal_Jatta https://t.co/oW3XWsgncH	True #FarmersProtest #Pagdi_Sambhal_Jatta https://t.co/oW3XWsgncH	2024-11-30 15:50:18	0.0563783
1.3629835569899438e+18	See the way BJP speaks. #FarmersProtest https://t.co/IIAWwkGuPf	See the way BJP speaks. #FarmersProtest https://t.co/IIAWwkGuPf	2024-11-30 15:50:22	0.059027750000000004

- **Graph of Timestamp Differences**:

Finally, a graph plots the time difference between consecutive document clicks during the session. This provides a visual representation of the user's click activity, allowing insights into their engagement level and showing peaks for delays between clicks, indicating when the user spent more time on a document or navigating other site features.

Timeline of Document Clicks:



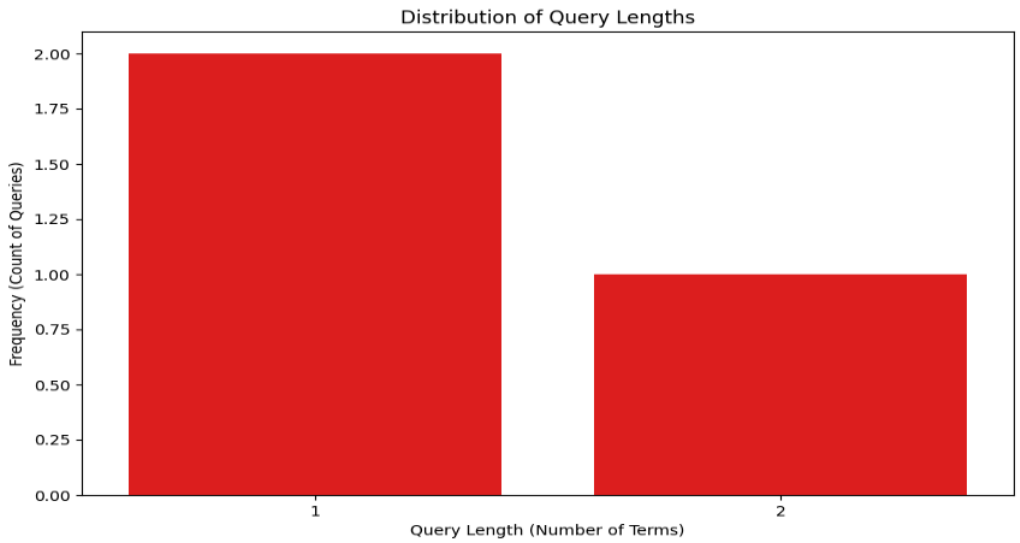
2.2. Dashboard

The dashboard provides aggregated analytics based on all user interactions with the site, giving a comprehensive overview of search and visitor behavior.

- **Query Analytics**:
    a. **Histogram of Query Lengths**

A histogram shows the distribution of the number of terms in user queries. It highlights the complexity of user searches (e.g., short queries vs. detailed multi-term searches).

Histogram of Query Lengths:



b. **Word Cloud of Search Terms**

A word cloud visualizes the most frequent terms in all user queries. Words are sized based on their frequency, making it easy to identify popular topics and keywords.

Word Cloud:

- **Clicked Documents Ranking**:

A ranked list of the most-clicked documents is provided, highlighting popular content by displaying the top documents in our app.

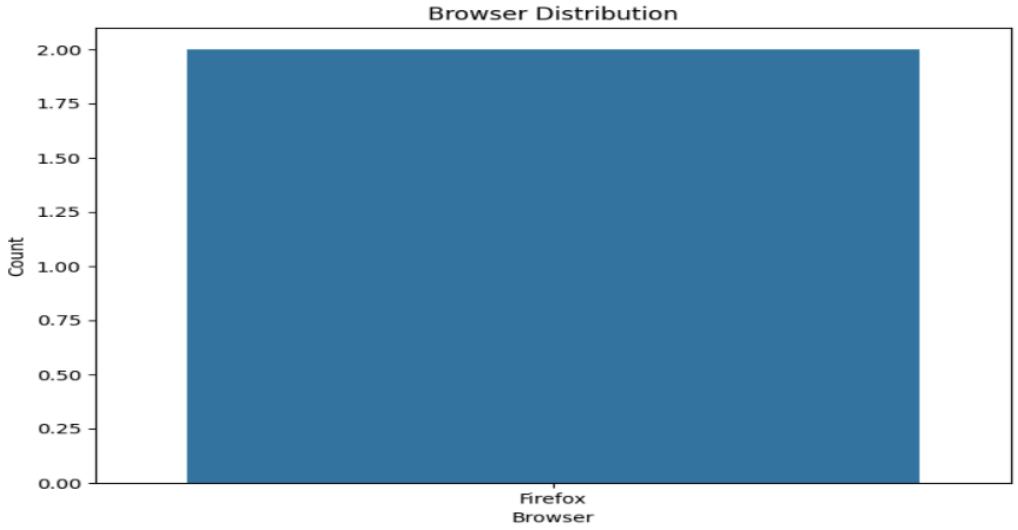
Visited Documents:

Document ID	Description	Click Count
1,36020966338862E+018	#FarmersProtest #FreeNodeepKaur #ReleaseNodeepKaur #ReleaseDetainedFarmers #FarmersAboveReligiousHate #blessedfarmers #Tractor2Twitter #IStandWithFarmers #indiastandswithfarmers #worldstandswithfarmers #namonomo #preetpatialvi #Lfreshthelion #mirrah https://t.co/6wmCzB6NfW	2
1,36271013347881E+018	#ReleaseDetainedFarmers #FarmersProtest #i_stand_with_farmers #KisaanMajdoorEktaZindabad #FarmBills2020 #NoRepealNoGharWapsi #NoFarmersNoFood #Stop_Killing_Farmers #ReleaseDishaRavi	1
1,36426317943863E+018	#modi_rojgaar_दो #FarmersProtest	1
1360644216372690944	#FarmersProtest in #Berlin https://t.co/ViyetfMFzC https://t.co/zF42ewgX6I	1
1363366953327480837	#FarmersProtest #Narendermodi #GodiMediaStopMisleading #KisanMajdoorEktaZindabaad https://t.co/APw8oYCAP5	1

- **Visitor Device Statistics**:
    a. **Browser Distribution**

A histogram displays the most-used browsers, such as Chrome, Firefox, and Safari. Helps identify browser preferences for optimizing website compatibility.

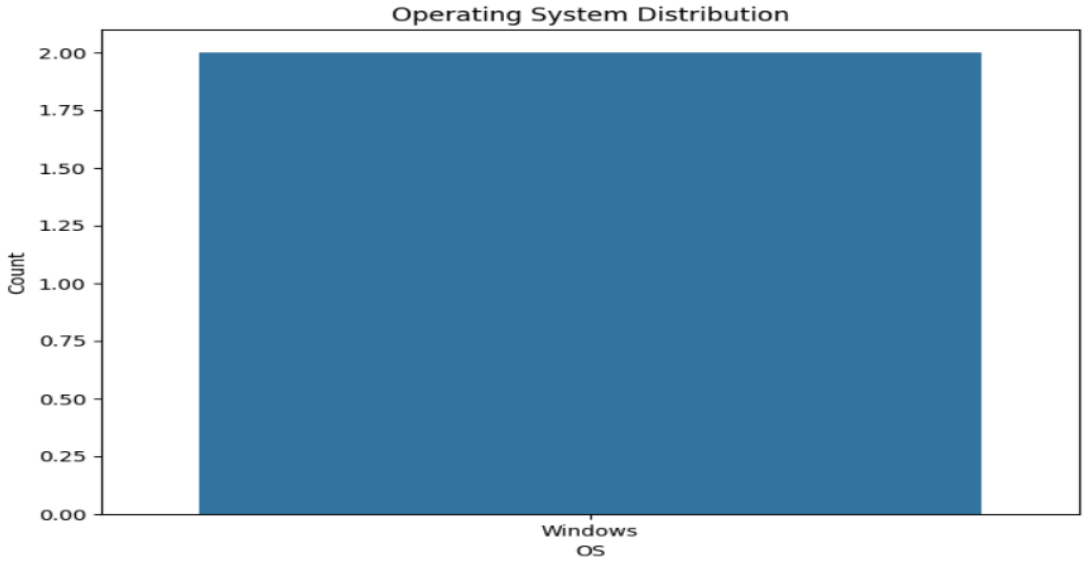
Browser Distribution:



b. **Operating System (OS) Distribution**

Another histogram shows the most common operating systems (e.g., Windows, macOS, Linux, Android).

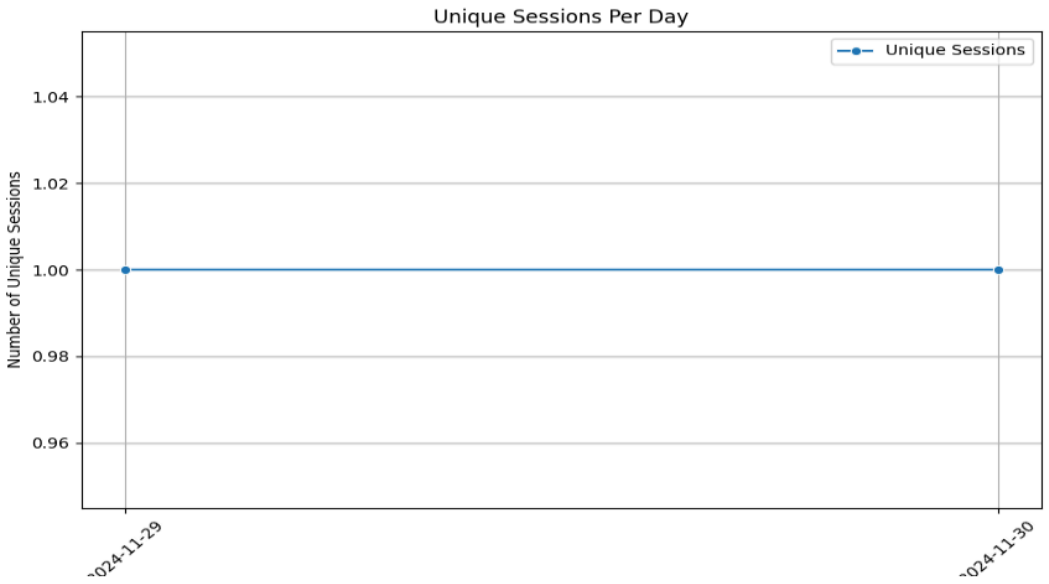
Operating System Distribution:



- **Daily Visitor**:

A time-series plot shows the number of visitors per day, indicating site traffic trends so we can monitor the usage of our app. Here we can see that we had two sessions on the days 29 and 30 of November.

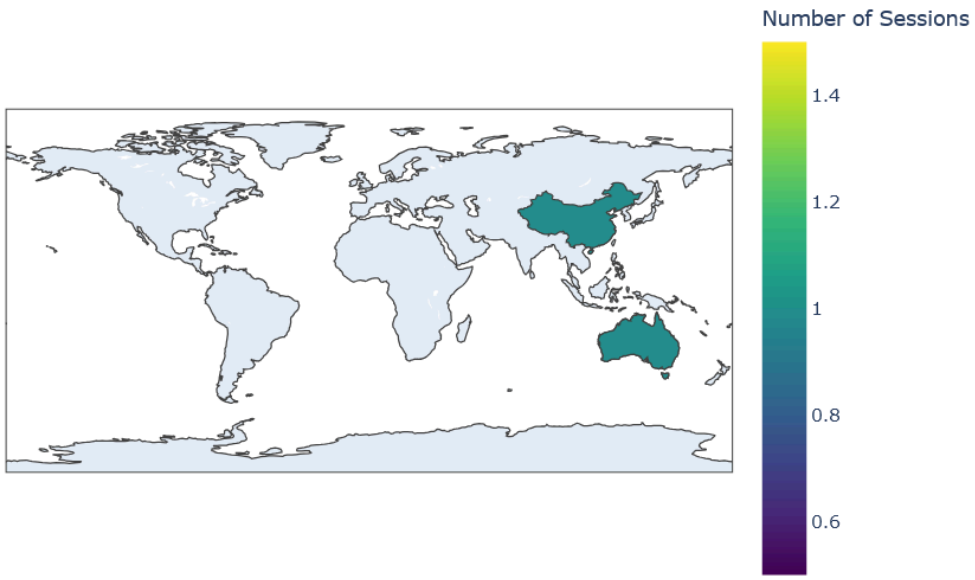
Unique Sessions Per Day:



- **Visitor Country Data**:

Visitor IP addresses are mapped to countries in a map. To achieve this we used the database from MaxMind called GeoLite2-City.mmdb where we convert our previously created ip addresses to the countries those are from. Finally, with Plotly we display them in a map, where each country that has visited our site is colored according to the sessions we get from there.

Sessions by Country



We also added a table to display the ip_address of the sessions and their corresponding country, which would be displayed as Unknown if we can't merge our session data with the database of ip addresses from MaxMind.

IP Addresses and Corresponding Countries

ip_address	country
49.75.18.79	China
138.217.237.206	Australia

3. Video

Here is a video of the web. How to execute the code is indicated in the README file. The page should work as this:
https://drive.google.com/file/d/1bjSwYxXVXPQHnuqcSKl-tgxjjemY6raf/view?usp=sharing