**1. INTRODUCTION**

**1.1. OVERVIEW**

Twitter Data Visualization is an application which is used to depict trending hashtags across the globe in a map interface on a mobile application. Data visualization helps to understand the trends easily in any field of application.

**1.2. AIM OF THE PROJECT**

The main aim of our project is to visualize the trending hashtags.Twitter produces loads and loads of data in form of tweets. The abundant availability of the social media data offers the end users to understand different social dynamics. The exponential growth in the volume, velocity and variability prevents the people from fully utilizing the data where in Visual Analytics comes into picture to understand large scale structured and unstructured social media data. In this project we will be dealing the current ongoing issues and popular trends to analyze and visualize their outcome in a map interface.

**1.3. ORGANISATION OF REPORT**

The organization of the report is as follows:

Chapter 1 deals with the Introduction of the project and gives the details about the project in an abstract view.

Chapter 2 deals with the information about React-Native, Firebase and tweepy module and its utilization details in brief.

Chapter 3 deals with the Software Requirements Specifications which is a specification of the project software and hardware requirements.

Chapter 4 deals with the Implementation part which includes the tools and software that are used.

Chapter 5 deals with the Testing of the project and screenshots of the project

Chapter 6 explains the Conclusion and further scope of the project.

**2.TECHNOLOGIES**

**2.1 REACT NATIVE**

At the React.js conference in 2015 Facebook introduced their new framework React Native, a framework they thought would revolutionise the way mobile applications are created. When React Native was released, there was only support for iOS but since then the support for Android has been added and is still expanding. Facebook have started to become more opensource and is the approach they have chosen for React Native. Even though the source is not completely open yet, Facebook attempt to achieve this and contemplates that the community will contribute to improve the framework.

The main purpose of React Native is simple, a developer should not require the knowledge or need to spend superfluous time in order to create a mobile application since at least two applications need to be developed in order to support both iOS and Android. Since different platforms have different looks, feels, and capabilities, there can not be an application which is homogeneous on all operating systems. However since it is the graphical interface that differs, the development could base on the same language but have the graphics be rendered differently depending on the targeted platform and be real native components. Facebook call this approach "learn once, write anywhere" which describes what React Native is all about.

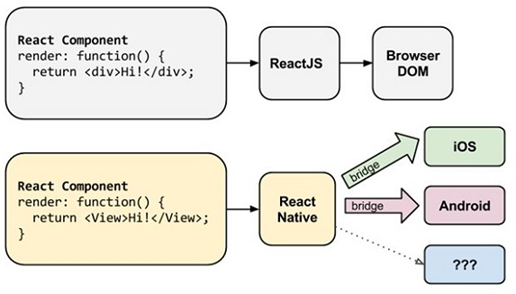
****

Fig 2.1 React Native Architecture

React Native lets you build mobile apps using only JavaScript. It uses the same design as React, letting you compose a rich mobile UI from declarative components. With React Native, you don't build a mobile web app, an HTML5 app, or a hybrid app; you build a real mobile app that's indistinguishable from an app built using Objective-C or Java. React Native uses the same fundamental UI building blocks as regular iOS and Android apps. You just put those building blocks together using JavaScript and React.

**2.1.1 The core of React Native**

As earlier stated, React Native is able to render the React Native components to real native Views for Android or UI Views for iOS. This is possible due to the abstraction layer known as the "bridge" which enables React Native to invoke the rendering APIs in Java for Android or Objective-C for iOS. Furthermore, the framework reveals the interface of JavaScript, allowing the application to access platform-specific features as the battery or the location. Section 2.4 mentioned the different threads used in React Native and there are actually three main threads on which React Native is based on: the shadow queue, where the the layout is handled; the main thread, where the UI rendering is performed; the JavaScript thread, where the scripts are running.

**2.1.2 Features**

A few features of React Native have been briefly introduced earlier in the paper but there are several features of React Native which makes it even more attractive to the development community.

First of all, React Native has support for asynchronous execution of operations between the JavaScript code in the application and the native platform but also permits threading in the native modules. This allows many different operations to be performed in the background while not blocking the user interface. It also allows the developer to debug the code while running the application by for example using the Chrome Developer Tools.

Secondly, React Native handles screen interaction by implementing a system which handles touches in a complex view with high level features. Since the gesture recognition is more advanced on a mobile device than on the web, there are many different actions which can be interpreted by the touch such as scrolling, tapping, sliding and so forth. Moreover, there can be multiple touches being performed concurrently. The users notice the difference between a web app and a native since every touch should display what will happen when its released and the user should be able to cancel the action by dragging their finger away. React Native has solved this problem by adding an abstract Touchable and TouchableHighlight implementation that assimilates properly with scrolls and other elements without requiring additional configuration.

**2.1.3 Environment Setup**

In order to install React Native, Homebrew is required to be installed and is then used to install Node.js. By using the node package manager (npm) which is a supplement to Node, the developer can install React Native with the command line. In order to run a React Native application in Android, Java JDK and Android SDK together with Android SDK Build-Tools 23.0.1 need to be installed on the computer. With React Native installed, a new React Native application can be created and be run by yet again entering a new command.

a) Installing react native :

npm install -g react-native

b) Create Project:

react-native init MyReactNative

c) Start React Native:

npm start

d) Running Project:

react-native run-android

The newly created project TestProject contains:

• package.json - A file containing the relevant metadata for the project or project dependencies.

• node\_modules/ - Contains the dependencies and the CLI tool, node\_modules/reactnative/local-cli/cli.js, which controls the project. This script executes another helper script node\_modules/react-native/init.sh that assembles the boiler plate code.

• index.android.js - The React Native main file of the project.

• android/ - Contains android specific code and is executed together with the code from index.android.js, resulting in a React Native application.

• iOS/ and index.ios.js - The iOS version of the two aforementioned file and folder.

As stated, the file index.android.js is the main file of the code and contains basic functionality from the beginning. First of all, the imports are listed. This states that the application is React Native and lists the imports needed as components, stylesheets, images etc. These imports can then be used to create components which can be rendered in the application. If the developer requires more imports, these can be added to the import object.

import React,

{

AppRegistry,

Component,

StyleSheet,

Image,

Text,

View

} from ’react-native’;

Figure : Example of an import object used in React Native

**2.1.4 Components**

Anything you see on the screen is some sort of component. A component can be pretty simple - the only thing that's required is a render function which returns some JSX to render.React Native follows a “component concept”. A component consists of a state with some properties and a lifecycle. During the development of the application, the developer can use the components multiple times, and this usually really helps to speed up the development of the applications. Furthermore, React Native also introduces to a new system called Virtual document object model (Virtual DOM). This system is a virtual representation, and simplified copy of the HTML DOM6 . The Virtual DOM allows React Native to do all the computational work within the virtual representation, to skip DOM operations, often slow and browser-specific. More precisely, when an event occurs, the page reacts to it modifying the Virtual DOM elements of the page.

Unlike the Ionic framework, React Native does not use WebViews, and this is a positive aspect because the WebView usually requires a substantial amount resources that lower the applications performances compared to the ones developed with native UI. React Native maps every component into the native interface of the Android and iOS platforms. Thanks to this mapping, the mobile applications developed with React native provide performances that are close to platform-specific mobile applications performances.

**2.1.5 Challenges**

React Native applications also consist of some challenges that developers must take into consideration:

• Limited cross-platform styling of components. Developers must style components on their own or use third-party libraries. This styling process can take a considerable amount of time.

• Only a basic set of pre-built components. The solution to this problem is to build components from scratch or use a third party library to speed up the building process. Usually, it is common to use the second solution due to the number of open sources third-party libraries available on the Internet.

• No responsiveness out of the box. Developers must create responsive designs on their own, defining sizes in the stylesheets or using external packages (e.g., react-native-responsive-ui)

Besides having some restrictions, React Native is a framework that is constantly growing, and a new version is available practically every month; therefore, bugs sometimes still appear, and this issue is also due to the high dependency on third-party libraries React Native also includes powerful tools that help to overcome the challenges presented above. The development process might become hard, but the final results consist of applications with deep integration into the operating system and excellent performance close to native apps.

**2.2 PACKAGES USED IN REACT-NATIVE**

**2.2.1 react-native-router-flux: React Native Routing**

Mobile apps are rarely made up of a single screen. Managing the presentation of, and transition between, multiple screens is typically handled by what is known as a router. For routing between components of the app we used react-native-router-flux module

Install Router:

npm i react-native-router-flux --save

Since we want our router to handle the entire application, we will add it in index.android.js. Now we will create the Routes component inside the components folder.A core component of any app is navigating between different routes/screens/scenes. It will return Router with several scenes. Each scene will need key, component and title. Router uses the key property to switch between scenes, component will be rendered on screen and the title will be shown in the navigation bar. We can also set the initial property to the scene that is to be rendered initially.

**2.2.2 react-native-maps: Map Interface**

To build the map interface for our app we used react-native-maps API

a) Installation

npm install react-native-maps --save

b) Import

import MapView from 'react-native-maps';

c) Components

<MapView />

<Marker />

<Callout />

<Polygon />

<Polyline />

<Circle />

<Overlay />

d) Mapview

It is the map component that uses Apple Maps or Google Maps on iOS and Google Maps on Android. All the map configurations and the objects to be visible on the map are defined in the mapview component

import MapView, { PROVIDER\_GOOGLE } from 'react-native-maps';

<MapView

ref={ref => this.refMap = ref}

provider={PROVIDER\_GOOGLE}

style={Styles.map}

customMapStyle={MapStyles}

initialRegion={{

latitude: this.state.origin.latitude,

longitude: this.state.origin.longitude,

}}

showsUserLocation={true}

followUserLocation={true}

>

</MapView>

e) Markers

Markers are the objects which are used to point particular locations on the map. In our app we used them to point the locations of the tweets.

import { Marker } from 'react-native-maps';

{this.state.markers.map(marker => (

<Marker

coordinate={marker.latlng} //Object containing latitude and longitude of the location

//of the tweet

title={marker.title}

description={marker.description}

/>

))}

**2.3 FIREBASE**

Firebase is considered as web application platform. It helps developers‟ builds high-quality apps. It stores the data in JavaScript Object Notation (JSON) format which doesn‟t use query for inserting, updating, deleting or adding data to it. It is the backend of a system that is used as a database for storing data.

**2.3.1 Services:**

a)Firebase Analytics

It provides insight into app usage. It is a paid app measurement solution that also provides user engagement. This unique feature enables the application developer to understand how users are using the application. The SDK has the feature of capturing events and properties on its own and also allows getting custom data.

b) Firebase Cloud Messaging (FCM)

It is formerly known as Google Clouds Messaging (GCM), FCM is a paid service which is a cross-platform solution for messages and notifications for Android, Web Applications, and IOS.

c) Firebase Auth

Firebase Auth supports social login provider like Facebook, Google GitHub, and Twitter. It is a service that can authenticate users using only client-side code and it is a paid service. It also includes a user management system whereby developers can enable user authentication with email and password login stored with Firebase.

d) Real-time Database

Firebase provides services like a real-time database and backend. An API is provided to the application developer which allows application data to be synchronized across clients and stored on Firebase's cloud. The client libraries are provided by the company which enables integration with Android, IOS, and JavaScript applications.

e) Firebase Storage

It facilitates easy and secure file transfer regardless of network quality for the Firebase apps. It is backed by Google Cloud Storage which is cost-effective object storage service. The developer can use it to store images, audio, video, or other user-generated content.

f) Firebase Test Lab for Android

It provides cloud-based infrastructure for testing Android apps. With one operation, developers can initiate testing of their apps across a wide variety of devices and device configurations. The various test results like screenshots, videos and logs are available in the Firebase console.. Even if a developer hasn't written any test code for their app, Test Lab can exercise the app automatically, looking for crashes.

g) Firebase Crash Reporting

The detailed reports of the errors are created in the app. The errors are grouped into clusters of similar stack traces and triaged by the severity. The other features are: the developer can log custom events to help capture the steps leading up to a crash.

h) Firebase Notifications

It enables targeted user notifications for mobile app developers and the services are freely available.

**2.3.2 Steps to add firebase to android**

Firebase can be added to any project intended for Android 2.3 (Gingerbread) or newer version and have Google Play services 9.6.1 or newer. Following steps are to be followed next:

a)A project has to be created in firebase console. Enter project name and the location. Project name under firebase console can be anything and can also be different from the application name given.

b)Click on Add Firebase to the Android app and follow steps mentioned there.

c) The user will be prompted with a page asking for the package name and optional SHA-1 certificate. SHA-1 code is available at android studio.

d)At the end google-services.json file will be downloaded once the above steps are completed. The file can be downloaded again at anytime.

e) The downloaded file has to be copied into the project‟s app or module folder. Firebase is added to the project

**2.3.3 Database**

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.Store and sync data with our NoSQL cloud database. Data is synced across all clients in realtime, and remains available when your app goes offline.

Add the Realtime Database to your app:

* Add the dependency for Realtime Database to your app-level build.gradle file: implementation 'com.google.firebase:firebase-database:16.0.6'
* Configure Firebase Database Rules

The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to.By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up authentication, you can configure your roles for public access. This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

**2.3.4 react-native-firebase: Reading from database to react-native**

readUserData() {

firebase.database().ref(node).once('value', function (snapshot) {

console.log(snapshot.val())

});

}

It will read data once from the node object and print it on console. If we want to get

data whenever there is any change in it, we can use on instead of once

**2.3.5 Firebase\_admin module: Writing to database from python**

Retrieve an instance of your database using getInstance() and reference the location you want to write to.

FirebaseDatabase database = FirebaseDatabase.getInstance();

DatabaseReference myRef = database.getReference("message");

myRef.setValue("Hello, World!");

**2.4 TWEEPY MODULE**

Twitter is a popular social network where users share messages called tweets. Twitter allows us to mine the data of any user using Twitter API or Tweepy. The data will be tweets extracted from the user. Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API.The first thing to do is get the consumer key, consumer secret, access key and access secret from twitter developer available easily for each user. These keys will help the API for authentication.

a)Installation

pip install tweepy

b)Steps to obtain keys:

– Login to twitter developer section

– Go to “Create an App”

– Fill the details of the application.

– Click on Create your Twitter Application

– Details of your new app will be shown along with consumer key and consumer secret.

– For access token, click ” Create my access token”. The page will refresh and generate access token.

c)Authentication

import tweepy

# Consumer keys and access tokens, used for OAuth

consumer\_key = ‘’

consumer\_secret = ‘’

access\_token = ‘’

access\_token\_secret = ‘’

# OAuth process, using the keys and tokens

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_token, access\_token\_secret)

# Creation of the actual interface, using authentication

api = tweepy.API(auth)

d) Getting tweets and location of particular hashtag

geopy makes it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.

Nominatin - Nominatim geocoder for OpenStreetMap data

class geopy.geocoders.Nominatim(format\_string=None, view\_box=None, bounded=False, country\_bias=None, timeout=DEFAULT\_SENTINEL, proxies=DEFAULT\_SENTINEL, domain='nominatim.openstreetmap.org', scheme=None, user\_agent=None, ssl\_context=DEFAULT\_SENTINEL)

Geocode - Return a location point by address.

geocode(query, exactly\_one=True, timeout=DEFAULT\_SENTINEL, limit=None, addressdetails=False, language=False, geometry=None, extratags=False)

geolocator = Nominatim()

for tweet in tweepy.Cursor(api.search, q='#'+hashtag).items(maximum\_tweets\_to\_be\_extracted):

location = geolocator.geocode(tweet.user.location)

**3 SOFTWARE REQUIREMENTS**

**3.1 INTRODUCTION**

The requirements specification is a technical specification of requirements for the

software products. It is the first step in the requirements analysis process it lists the

requirements of a particular software system including functional, performance and

security requirements. The requirements also provide usage scenarios from a user, an

operational and an administrative perspective. The purpose of software requirements

specification is to provide a detailed overview of the software project, its parameters and

goals. This describes the project target audience and its user interface, hardware and

software requirements. It defines how the client, team and audience see the project and its

functionality.

**3.2 PURPOSE OF DOCUMENT**

This software requirement specification describes all the requirements elicited for “TWITTER DATA VISUALIZATION” and is intended to be used by the members examining the project and

implementing and verifying the application. Unless otherwise noted all requirements are

of high priority and are committed.

**3.3 SOFTWARE AND HARDWARE REQUIREMENTS**

|  |  |
| --- | --- |
| Criterion | Description |
| OS version | Microsoft® Windows® 7/8/10 (32-bit or 64-bit) |
| RAM | 3 GB RAM minimum, 8 GB RAM recommended |
| Python | Version 3.4 |
| Packages | React-Native: react-native-router-flux  react-native-maps  react-native-firebase  Python : tweepy  geopy  firebase\_admin |

**4. IMPLEMENTATION**

**4.1 WORKING STAGES OF OUR PLATFORM**

Twitter is a social media platform where issues across the world are discussed.Our application visualizes trending hashtags in a map interface.The following are the working stages:

* Getting Twitter API access
* Collecting Trending hashtags and their geo locations
* Creating Firebase database
* Connecting Python to firebase
* Developing mobile application

Implementation process:

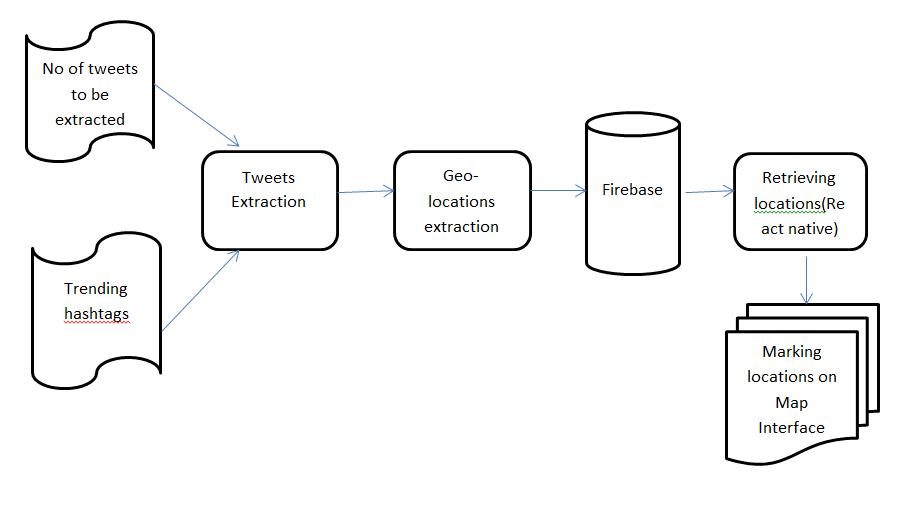


Fig 4.1 Project Flow

**4.1.1 Getting Twitter API access:**

1. Visit the Twitter Developers Site:

The first thing you need to do is head on down to [dev.twitter.com](http://dev.twitter.com/). In order to create an account, all you need to do is click on the “Sign In” link at the top right.

2. Sign in with your Twitter Account:

Next, sign in with the Twitter account you want to associate with your app.

3. Go to “My Applications” :

Once you’re logged in, click on the downwards arrow to the right of your Twitter image and select “My Applications”. This is where all your registered Twitter apps will appear.

4. Create a New Application:

If you are new to the Developers site you won’t see any applications registered. Either way, it’s time to create our first application. To do this, click on the big “Create a new application” button.

5. Fill in your Application Details:

Put **your website i**n the website field- don’t worry that it isn’t (as Twitter ask) your application’s publicly accessible home page. However, this website will be where your app is hosted.

For now ignore the Callback URL field. If you are allowing users to log into your app to authenticate themselves, you’d enter the URL where they would be returned after they’ve given permission to Twitter to use your app.

Once you’ve done this, make sure you’ve read the “Developer Rules Of The Road” blurb, check the “Yes, I agree” box, fill in the CAPTCHA (don’t you just love them) and click the “create your Twitter Application” button.

6. Create Your Access Token:

You’ll then be presented with lots of information, but we’re not quite done yet. We now need to authorize the Twitter app for your Twitter account. To do this, click the “Create my access token” button. This takes a few seconds, so if you don’t see the access tokens on the next screen, you may have to refresh the page a few times.

If you change the app permissions at any future point a new auth token will need to be generated to reflect the permissions change.

7. Choose what Access Type You Need:

Change application type to ‘Read & Write’ to give the application permission to follow other accounts on your behalf. This will require that you verify your mobile phone within your twitter account.

8. Make a note of your OAuth Settings:

Once you’ve done this, make a note of your OAuth settings.

* Consumer Key
* Consumer Secret
* OAuth Access Token
* OAuth Access Token Secret

It goes without saying that you should keep these secret. If anyone was to get these keys, they could effectively access your Twitter account.

**4.1.2 Collecting Trending hashtags and their geo locations:**

We have a deeply rooted need to share our ideas and experiences, which gives us the ability to connect with other people, to be heard, and to feel a sense of worth and importance. We are curious about the world around us and how to organize and manipulate it, and we use communication to share our observations, ask questions, and engage with other people in meaningful dialogues about our quandaries.Which is where twitter comes in use to make note of all the trending topics and visualizing them on a mapping interface so that we will have generalizations of the trends going on across the world in a nutshell.

First steps include Collecting Tweets according to Trending Hashtags:

* We already know as twitter helps us to access the tweets using tweepy module and as we have the access tokens and access keys we can extract the tweets
* We used geopy module :

geopy is a Python 2 and 3 client for several popular geocoding web services.geopy makes it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.geopy includes geocoder classes for the [OpenStreetMap Nominatim](https://wiki.openstreetmap.org/wiki/Nominatim), [Google Geocoding API (V3)](https://developers.google.com/maps/documentation/geocoding/), and many other geocoding services.

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

api = tweepy.API(auth)

maximum\_number\_of\_tweets\_to\_be\_extracted = \

int(input('Enter the number of tweets that you want to extract- '))

# Mention the hashtag that you want to look out for

hashtag = input('Enter the hashtag you want to scrape- ')

geolocator = Nominatim()

for tweet in tweepy.Cursor(api.search, q='#' + hashtag, lang='en',

rpp=100).items(maximum\_number\_of\_tweets\_to\_be\_extracted):

location = geolocator.geocode(tweet.user.location)

As we can see we have used some functions like :

* OAuthHandler():Tweepy supports oauth authentication. Authentication is handled by the tweepy.AuthHandler class.

The next step is creating an OAuthHandler instance. Into this we pass our consumer token and secret which was given to us in the previous paragraph:

auth = tweepy.OAuthHandler(consumer\_token, consumer\_secret)

If you have a web application and are using a callback URL that needs to be supplied dynamically you would pass it in like so:

auth = tweepy.OAuthHandler(consumer\_token, consumer\_secret,callback\_url)

* geolocator.geocode(): geopy allows us to use the geocode function to locate the latitudes and langitudes of the trends across the world .

**4.1.3 Creating Firebase database:**

We already came to know about database creation in firebase.we will be getting a key.json which plays a major role in all the connections to make with the mobile application.

Steps involved:

a)A project has to be created in firebase console

b)Click on Add Firebase to the Android app and follow steps mentioned there.

c) The user will be prompted with a page asking for the package name and optional SHA-1 certificate. SHA-1 code is available at android studio.

d)At the end google-services.json file will be downloaded once the above steps are completed. The file can be downloaded again at anytime.

e) The downloaded file has to be copied into the project‟s app or module folder. Firebase is added to the project

**4.1.4 Connecting Python to firebase**

cred = credentials.Certificate('key.json')

firebase\_admin.initialize\_app(cred, { 'databaseURL' : '<https://locations-ccc0b.firebaseio.com>'})

root = db.reference('/users1/')

if location is not None:

ref= root.child(str(i))

ref.set({

i:{

'latitude':(location.latitude),

'longitude':(location.longitude)

}

})

i=i+1

**4.1.5 Developing mobile application;**

For the map-interface we used react-native-maps which uses Google API. So we need to create a Google maps API key to access the map.

a) Creating API key

1. Go to the [Google Cloud Platform Console](https://cloud.google.com/console/google/maps-apis/overview).
2. From the Project drop-down menu, select or create the project for which you want to add an API key.
3. From the Navigation menu, select APIs & Services > Credentials.
4. On the Credentials page, click Create credentials > API key.  
    The API key created dialog displays your newly created API key.

b) Setting up react-native-maps

In your android/app/build.gradle file make sure you have this dependency:

dependencies {

compile project(':react-native-maps')

}

In your android/app/build.gradle file make sure you have this dependency:

dependencies {

compile project(':react-native-maps')

}

And finally in your android/app/src/main/AndroidManifest.xml file you will need to insert your API key inside the <application>.

<application>

<meta-data

android:name="com.google.android.geo.API\_KEY"

android:value="INSERT GOOGLE MAPS API KEY HERE!!!!!!!"/>

</application>

include a <MapView> in your index.android.js file.

<MapView

ref={ref => this.refMap = ref}

provider={PROVIDER\_GOOGLE}

style={Styles.map}

customMapStyle={MapStyles}

initialRegion={{

latitude: this.state.origin.latitude,

longitude: this.state.origin.longitude,

latitudeDelta: 0.0922,

longitudeDelta: 0.0421,

}}

showsUserLocation={true}

followUserLocation={true}

>

{this.state.makArray}

</MapView>

c) Retrievng locations from firebase and add markers: The data in firebase looks like this

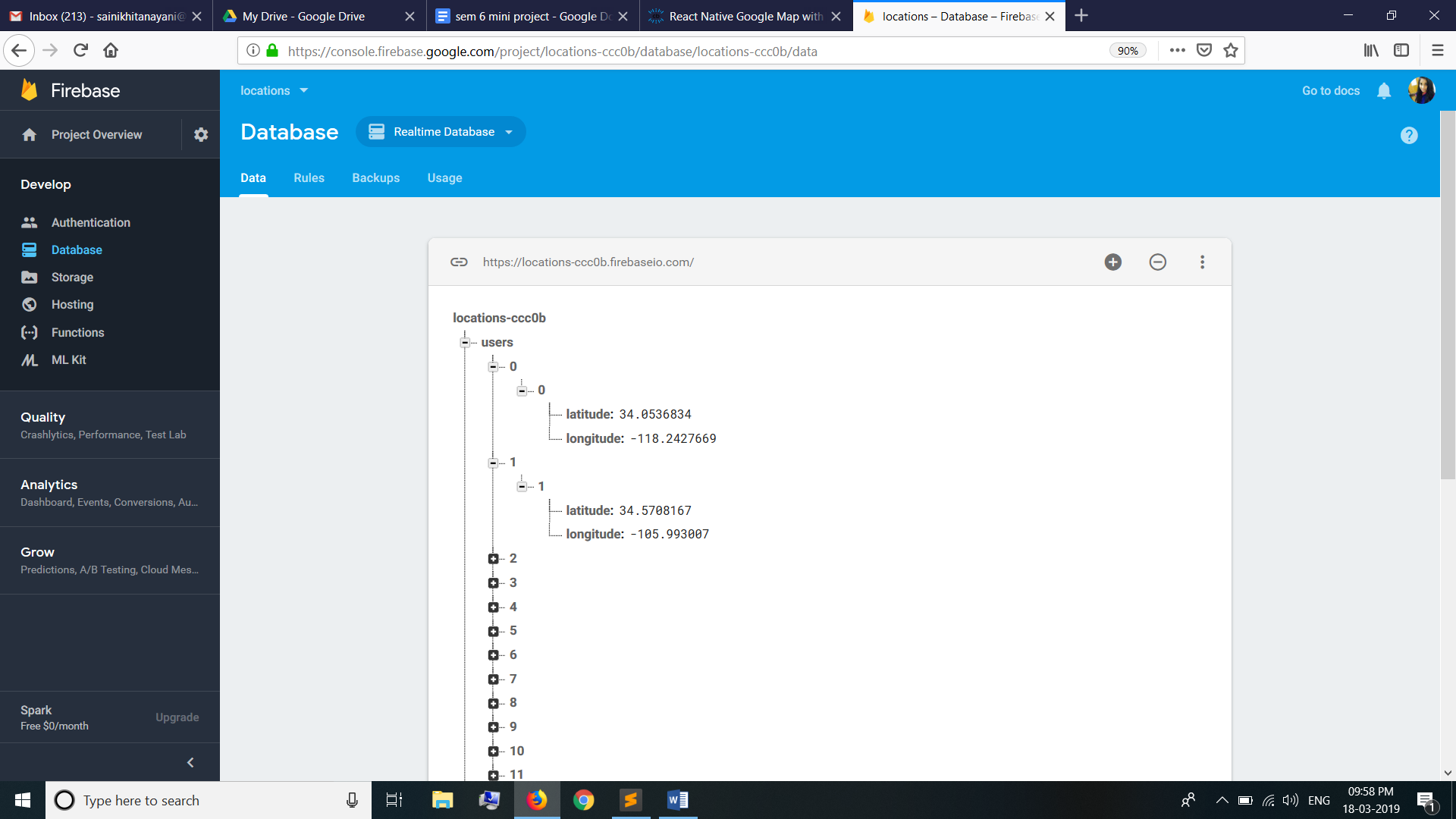


Fig 4.2 Firebase database

To retrieve data the following function is used

renderMarkers(){

var markArray = new Array();

c = 0;

var data = firebase.database().ref('/users/').once('value', (snapshot) => {

var stores = snapshot.val();

for(var i=0;i<n;i++){

loc=stores[i][i]; //retrieving location

markArray.push(<Marker coordinate = {loc}/>); //creating markers for location

}

})

}

**4.2 FLOWCHART:**

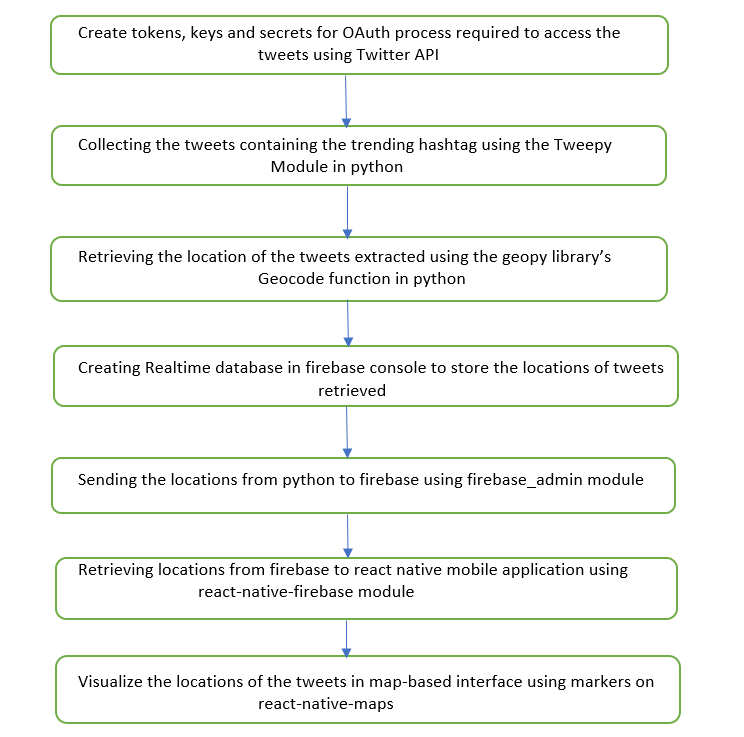


Fig 4.3 Flowchart

**5.RESULTS**

**5.1 INTRODUCTION**

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive. A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

**5.2 TESTING OBJECTIVES**

The main objective of performance testing is designed to test whether display is as expected and whether the source code is functioning properly or not. As the test results are gathered and evaluated they begin to give a qualitative indication of the reliability of the code. If proper output is not obtained, the overall quality of the code is questioned. If, on the other hand, all the results which are not successful, are encountered, and are easily modifiable, then the following conclusion can be made: The tests are inadequate as the requirements mentioned are not compatible.

The testing includes:

• Checking whether the information is displayed or not.

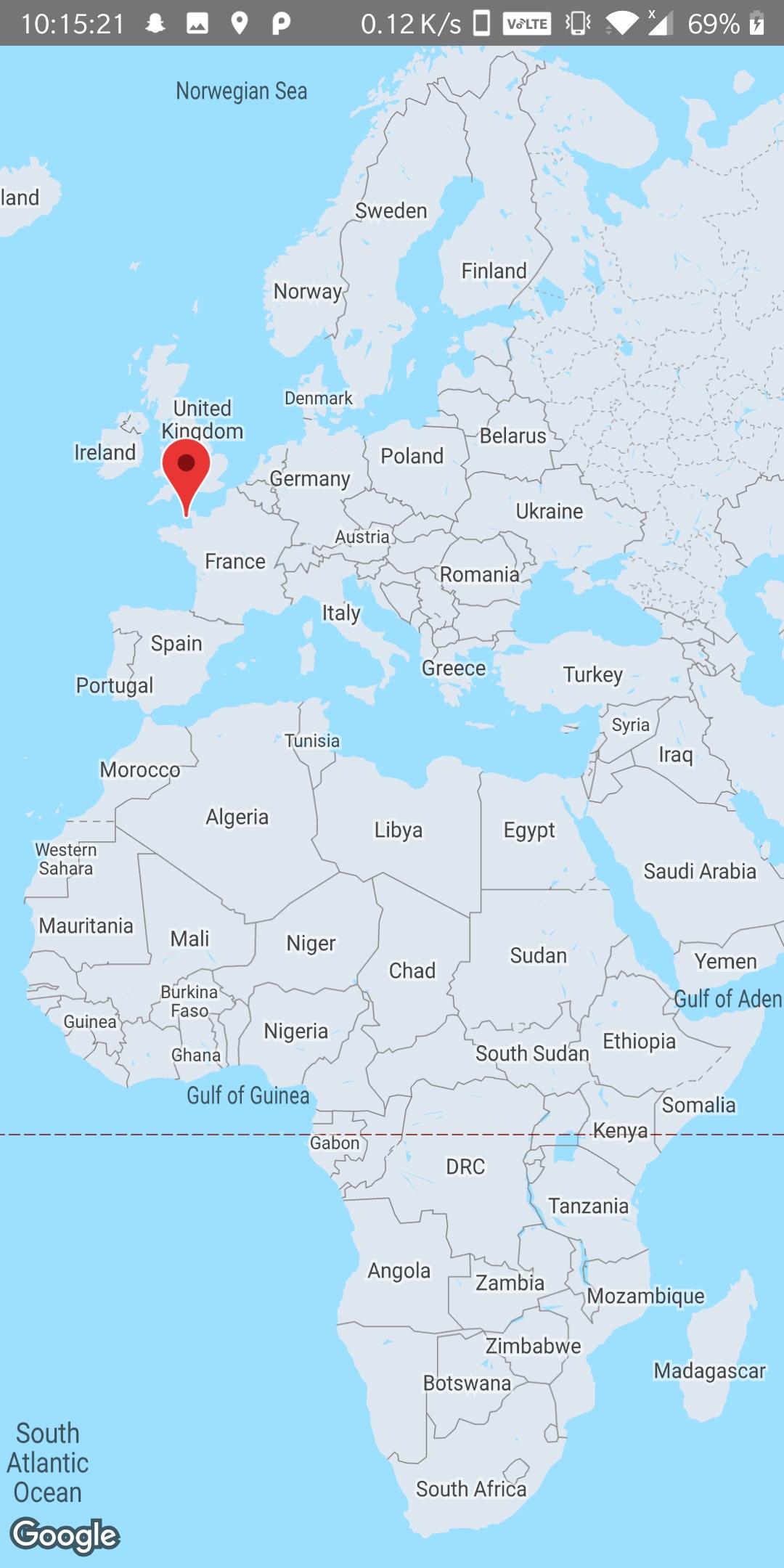
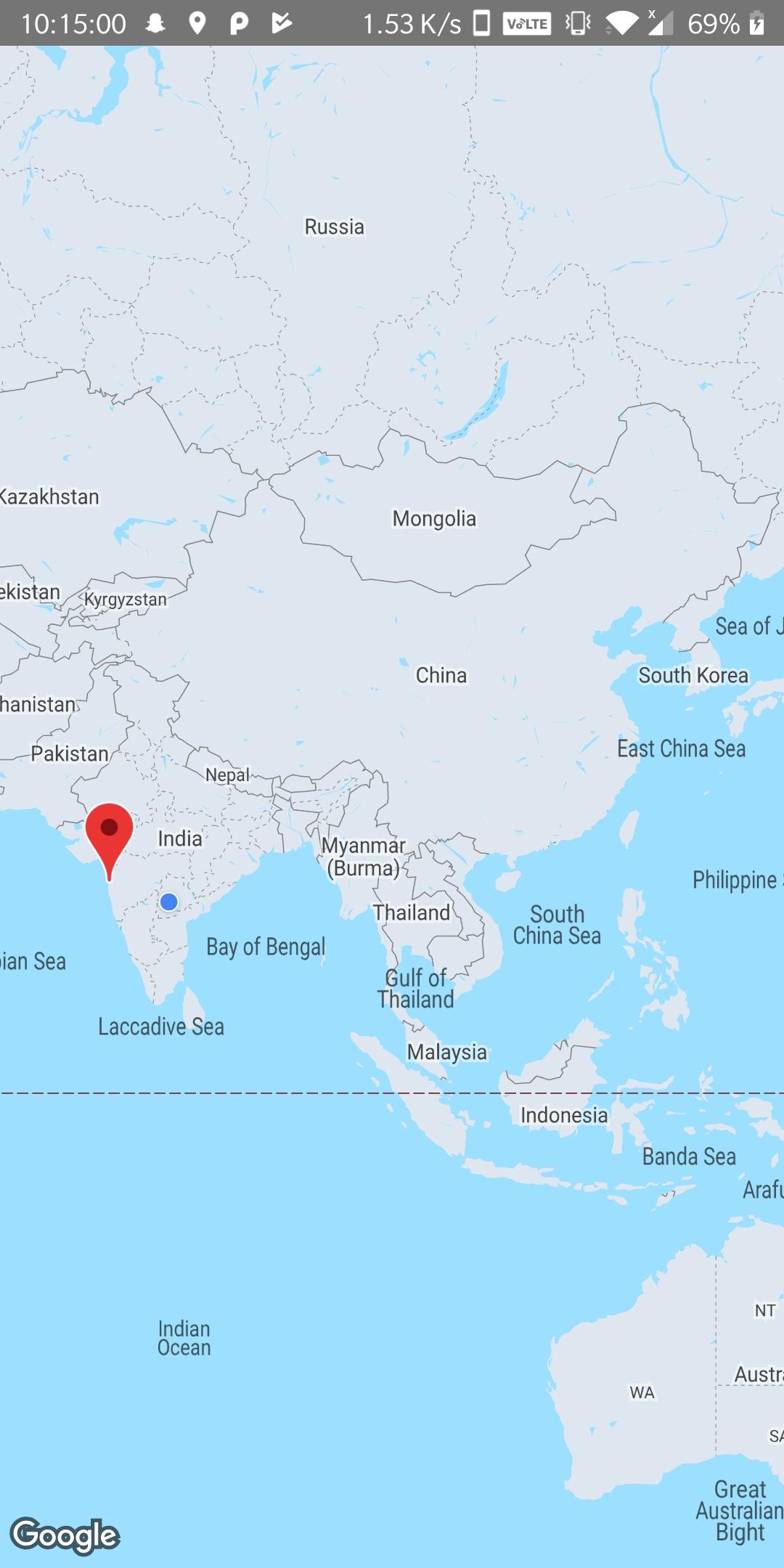
• Checking whether files are correctly linked and opened.

• Checking whether the required data is displayed correctly or not by checking through android emulator or by mobile phones

**5.3 OUTPUT SCREENS**

Results are obtained based on the hashtag and their geo locations where they are trending.

The result shown below depict the #metoo movement which became an trending issue regarding women harassment.The issue gained world wide recognition and was discussed mostly in areas of North America but the issue started in India.

****

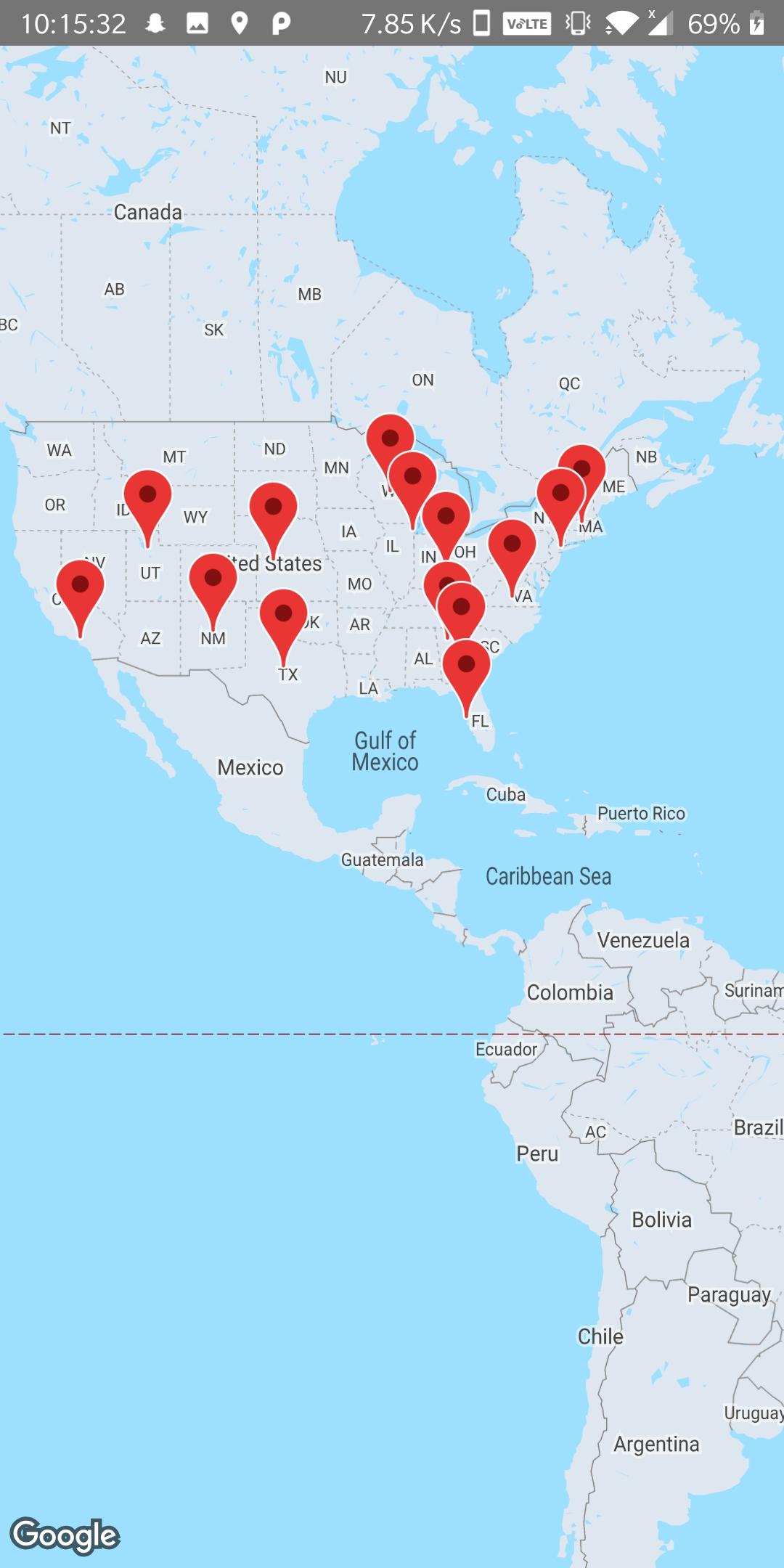
****

Fig 5.1 Results

The tweets regarding trending hash tags are extracted along with their geo-locations are stored in the database which is created using firebase.So python to firebase connection is made and the tweet locations are stored.Then fire base to mobile application connection is made to map the markers on the mapper interface.

**6 CONCLUSION AND FUTURE SCOPE**

In this project we will be dealt with the current ongoing issues and popular trends to analyze and visualize their outcomes. For example, after the presidential election in US there was a difference in opinion among people about Mr. Donald Trump regarding many issues like granting VISA, not allowing part time jobs for non locals, not allowing immigrants etc. We can analyze and visualize the opinion of people on the decisions taken using the tweets of people about him around the world.

Future scope is that Google Trends recently released an interactive map that shows where searches for #MeToo are trending i.e. areas that have the highest interest in the movement. The map also highlights the top five cities wherein the #MeToo movement has been searched the maximum on any given day on the map’s homepage, allowing users to click on the city and read complete coverage of the movement dating back to last fall. It gained importance only because it was visualized effectively. Otherwise no one would bat an eye.

**7. BIBILOGRAPHY**

[1] S.  Malik, A.  Smith, P. Papadatos,  J. Li. TopicFlow: Visualizing Topic Alignment of Twitter Data over Time.Proceedings of the 2013 IEEE/ACM International Conference  on Advances in Social Networks Analysis and Mining.

[2] S.  Bhulai, P.  Kampstra, L.  Kooiman, G. Koole, M. Deurloo, B. Kok. Trend Visualization on Twitter: What's Hot and What's Not?. DATA ANALYTICS 2012, The First International Conference on Data Analytics.

[3] J. Bradley, N.  Fung, I. Julien, M. Malu, M. Mauriello. ViralViz: Visualizing Temporal Content Flow in Social Networks.

[4] M. Hao, C. Rohrdantz, H. Janetzko, U. Dayal, D. Keim, L.  Haug, M. C. Hsu. Visual Sentiment Analysis on Twitter Data Streams. In Visual Analytics Science and Technology  (VAST), 2011 IEEE Conference on (pp.277-278). IEEE.

[5] D. M. Blei, A. Ng, M. I. Jordan. Latent dirichlet allocation. The Journal of Machine Learning Research,3, 993-1022.

[6] L. Byron, M. Wattenberg. Stacked Graphs-Geometry &Aesthetics.  IEEE Trans. Vis. Comput. Graph., 14(6),1245-1252.