Research Doc

# Weather api

## Api1

<https://www.weatherapi.com/>

an open source api which can be used through json and xml. Also works with mobile app which is what I’ desiring

I would be using the free option which gives me access to 1mill calls to the api a month.

3 day cit and town weather both hourly and daily

It has integrated ip lookup which should tell the api where the user is coming from

Will need to ensure that the user is okay with this before hand. Also has weather history data which will have only the last 7 days which means I will be able to better average.

It also has a up time of 95.5% and a https included

Uses a restful API

Provided list of command and api and documentation <https://www.weatherapi.com/docs/>

Because I am using the free one I would want to provide a link back

Wil need to ensure that android studio is compatible with the api and a json object

## Api blog

https://rapidapi.com/blog/access-global-weather-data-with-these-weather-apis/

From a blog api the top api’s in their opinion

The top 6 where

* [OpenWeatherMap API](https://rapidapi.com/community/api/open-weather-map)
* [Weatherbit API](https://rapidapi.com/weatherbit/api/weather)
* [AccuWeather API](https://rapidapi.com/stefan.skliarov/api/AccuWeather)
* [Dark Sky API](https://rapidapi.com/darkskyapis/api/dark-sky)
* [Weather2020 API](https://rapidapi.com/Weather2020/api/weather)
* [ClimaCell API](https://rapidapi.com/ClimaCell/api/climacell)

<https://www.climacell.co/weather-api/>

USED BY UBER PRORSCHE and ford etc

Free plan offers

$0 / month

**1,000** calls/day

[Core Layer](https://www.climacell.co/data-catalog/core/)

Air Quality Layer – Free Sample\*

Pollen Layer – Free Sample \*

Tiles

Widgets

Community Support

100 calls / hour

\* Free sample of our proprietary data layers (limited to 100 calls/day)

The usp of this Is

ClimaCell uses those same traditional sources, but also new sensing technologies, including cell towers, not used by anyone else. This data has the following advantages over data from traditional weather APIs:

* **Hyperlocal observation of weather:** 500 meter radius observation vs. 2,000-4,000 meters
* **Ground-level tracking:** Precipitation as low as 30 feet off the ground vs. over 1,000 feet for radar.
* **Up-to-date data delivery:** Many weather APIs can deliver data every minute, but that data can be 5-20 minutes old (or older). ClimaCell delivers updated data every minute.
* **NowCasting:** ClimaCell provides minute-by-minute forecast data for 0-6 hours out. Most other APIs provide this data in 1-hour chunks vs. 1-minute.

<https://openweathermap.org/>

used by companies like bc facebook mircosft google amazon wb

the free scheme provides

|  |  |
| --- | --- |
| **60 calls**/minute **1,000,000 calls**/month | |
| [**Current Weather**](https://openweathermap.org/current)  [**Minute Forecast 1 hour**](https://openweathermap.org/api/one-call-api)**\***  [**Hourly Forecast 2 days**](https://openweathermap.org/api/one-call-api)**\***  [**Daily Forecast 7 days**](https://openweathermap.org/api/one-call-api)**\***  [**Government Weather  Alerts**](https://openweathermap.org/api/one-call-api)**\***  [**Historical weather 5 days**](https://openweathermap.org/api/one-call-api#history)**\*** | |
| [**Basic weather maps**](https://openweathermap.org/api/weathermaps)  Historical maps |  | |  |  |  |
| [**Weather triggers**](https://openweathermap.org/triggers) |  | |  |  |  |
| [**Weather widgets**](https://openweathermap.org/widgets-constructor) |  | |  |  |  |
| Uptime 95% |  | |  |  |  |

# Kotlin

Apparently safer code as it helps avoid null point exceptions.

Also interoperable as it allow for as much kotlin or java as necessary

It has structured concurrency which also for database updates to be simple

# Android guides

<https://developer.android.com/docs>

recommender system

<https://books.google.co.uk/books?hl=en&lr=&id=eygTJBd_U2cC&oi=fnd&pg=PR5&dq=recommender+systems&ots=mWt057CPxE&sig=VyNdgqA0Mboci4fgTR0UGtGmH_0&redir_esc=y#v=onepage&q=recommender%20systems&f=false>

<https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada>

# Recommender systems

This is a system which is a subclass of information filtering which predicts the preference of the user. These are usually things the user wants and the goal is to be able to predict these from the users habits. “RS are used primarily for individuals who lack sufficient personal experience or competence to evaluate all the choices.”

Non personal rs recommendation are typically not associated with standard rs research. The system is defined by the users preferences and constraints. The data to collect about a user can be explicit like their reviews or they can be interpreted from the users action. This can be like a the about of retention a user has on a certain site or video. So for my recommender system I need to be able to define the type of data I will collect about the user and how I will interpret that data. To do this I will create some case studies which a predictable input which should provide a mostly predictable output form the app.

The importance of recommendation systems is the amount of influence recommendation have on everyday life and routines. It streamlines the decision process and reduces the amount of choice having to be made. This can help by reducing decision fatigue. This being caused by the overwhelming about of choice which is facilitated.

It’s also improved as it often compares other users of similar taste pattern to find items which could be recommended to them. This process is called collaborative-filtering

These recommendation become more and more important in modern life as due to increased ease of access to information and items though the internet, we are faced with more and more decisions each day.

The role of a rs is to benefit the user and the owner. For the user it’s the ability to have a better selection of choices while the owner benefits as it often mean greater profit. This can be due to the user finding something they want to buy or the customer experience being improved leading to longer use of the application/site and establishing consumer loyalty.

Key point into why the service provider would want to implement a rs system would be:

* Increasing the number of items sold
* Sell more diverse items
* Increase the user satisfaction
* Increase user fidelity
* Better understand what the users want

An RS needs to be able to balance how much it benefits the user and the owner. It needs to be able to cover multiple bases at the same time.

The main focus of my recommender will be to “find some of the good items” which are provided to the app by the user. It will use this limited data set to provide the best options.

Along with this it will have to be able to “recommend a bundle” of items as the core of the app is to be able to create an outfit, so the synergy of items will play an important part in the recommendation process.

It also will use recommendations to “improve the profile” of each user using the app as it will take the input taken through the app in it’s accept or rejection of outfits to build a stronger profile.

<https://en.wikipedia.org/wiki/Recommender_system>

Francesco Ricci and Lior Rokach and Bracha Shapira, [Introduction to Recommender Systems Handbook](http://www.inf.unibz.it/~ricci/papers/intro-rec-sys-handbook.pdf), Recommender Systems Handbook, Springer, 2011, pp. 1-35

<https://link.springer.com/book/10.1007/978-0-387-85820-3#about>

https://www.healthline.com/health/decision-fatigue