# Funcionament de l'aplicació.

Aquesta aplicació permet a l'usuari saber quin temps farà en una ciutat en els cinc dies vinents, en intervals de tres hores.

A l'introduir el nom d'una ciutat i al prémer el botó de buscar, l'aplicació realitza una petició a l'API d'OpenWeatherMap, processa el resultat i el mostra per pantalla si tot ha anat bé.

Hi ha un apartat de configuració molt bàsic on l'usuari pot escollir la unitat de temperatura, el format de descàrrega de l'API, i entre dos tipus de layout per a mostrar la informació.

Les dades descarregades s'emmagatzemen en una base de dades SQLite. D'aquesta manera es pot recuperar la informació que hi hagi, sempre que sigui recent, i evitar realitzar una petició que poques diferències pot aportar.

Vídeo que mostra el funcionament de l'aplicació:
https://mega.nz/#!61AFzKob!uq62wTib1Uth8yV6jRJ_FTc_uDohE0-7erMtL3-B7NI
. (El rellotge del mòbil està mal configurat i mostra un valor incorrecte.)

## Amb més detall.

Quan l'usuari cerca les dades d'una ciutat, es comprova en primer lloc si es troben emmagatzemades a la base de dades. Si és així i a més són suficientment recents (hem establert un marge de 30 minuts), es recuperen i s'envien per a presentar-les per pantalla.

Si no hi són presents o no són del tot recents (cas en què s'eliminen de la base de dades), es realitza una petició de descàrrega a l'API d'OpenWeatherMap.

Aquesta permet descarregar la informació en format XML o JSON.
A l'hora de descarregar i parsejar la informació, mirem als settings quin format ha escollit l'usuari (per defecte JSON), i es descarreguen les dades que toquen.

Després es parsejen amb la classe adequada: ParsejadorBlocXML o ParsejadorBlocJSON, que implementen la interfície ParsejadorBloc. Si en un futur apareguessin més formats de descàrrega, es podria crear una altra classe que la implementés.

Aquests parsejadors extreuen la informació que ens interessa:

- Data d'inici, de la qual ens interessen només el dia, el mes, l'hora i els minuts.
- Nom de la ciutat.
- Temperatura en graus Kelvin.
- El codi de la imatge del fenomen atmosfèric més rellevant.

Aquestes dades s'emmagatzemen a la base de dades i s'envien per a presentar-les per pantalla posteriorment.

Si el nom de la ciutat és incorrecte, s'informa l'usuari.

En qualsevol moment es pot buidar la base de dades mitjançant una opció al menú de l'activitat principal.

A l'hora de presentar les dades per pantalla, es determina la unitat de temperatura que hagi escollit l'usuari (per defecte Celsius; les altres opcions són Kelvin i Fahrenheit) i el tipus de layout que es farà servir. L'usuari en té per a escollir dos, que únicament es diferencien en la manera de presentar la mateixa informació.

Un cop tot preparat, es comencen a presentar les dades. Es parseja la data per a mostrar-la en un format més llegible i amb el més essencial, es passa la temperatura a les unitats desitjades (si cal), i es descarrega la imatge de manera asíncrona a partir del seu codi.

El dia només es mostra quan cal, és a dir, quan en comença un de nou (a les 00:00 h). El primer element sempre l'inclou.

Tot això es veu com una llista en un RecyclerView, ja que d'aquesta manera el sistema pot gestionar millor els recursos. Es troba en un fragment dinàmic acabat de crear/carregar.

Un cop presentades les dades, l'aplicació ja no fa res més. Si es torna a buscar una ciutat, es repeteix tot el procés, fent les mateixes comprovacions.

# Dificultats.

## Imatges de fenòmens atmosfèrics.

L'API d'OpenWeatherMap proporciona el codi de les icones de temps que identifiquen visualment quins fenòmens atmosfèrics són els més rellevants.
Per a obtenir la imatge actual s'ha de fer una petició a una URL, on hi figura el codi, i assignar-la a un ImageView (en el nostre cas).

Inicialment vam crear una classe que es deia Descarregador (o algo així) que heretava d'AsyncTask. Amb un codi semblant al de fer una petició a l'API, descarregava la imatge, en creava un Bitmap i el posava a l'ImageView adequat.

Malgrat que el seu funcionament era correcte, al fer scroll al RecyclerView les imatges dels elements que apareixien no eren les correctes. Al cap d'uns segons, però, i després de canviar unes quantes vegades, es mostraven les que havien de sortir.

Com que això no hauria de passar, vam investigar una mica i de seguida vam trobar una manera de solucionar-ho: fent servir la llibreria Picasso. Amb un parell de crides a Picasso vam substituir tota la classe Descarregador, i aquest cop les imatges apareixien ràpidament on tocava.

Suposem que fent servir la nostra classe Descarregador les imatges no es guardaven en cap *cache* o alguna cosa d'aquest estil, i al fer scroll al RecyclerView tot el procés d'enviar la petició i situar-les al seu lloc era massa lent i es notava.

Picasso es preocupa de guardar les imatges en un *cache* i recuperar-les quan cal.

# Gestió del projecte.

La manera que hem fet servir per a gestionar les tasques ha estat anar creant issues al GitHub i anar deixant TODOs al codi. D'aquesta manera, la feina es pot repartir de manera més flexible i hi ha més llibertat a l'hora d'escollir.

L'únic que s'ha de fer és avisar a la resta de l'equip quan s'escull una tasca, per a evitar malgastar temps treballant de manera separada en una mateixa tasca.

Es poden deixar comentaris a les issues per a discutir possibles solucions i plantejar els problemes que podrien comportar i, un cop completades, eliminar-les.

Enllaç al repositori de GitHub: https://github.com/Elieroz/PT16 .

# Innovacions o coses extra.

## Mostrar el dia quan és necessari.

Mostrar el dia a cada CardView és innecessari i fa que hi hagi massa coses a la pantalla. És per això que només el mostrem quan comença un dia nou (és a dir, quan l'hora és 00:00) i quan l'element és el primer de la llista.

## Recuperació de dades al recrear-se l'activitat.

Al buscar exemples de com mantenir/restaurar l'estat d'una activitat al ser recreada pel sistema, vam llegir que afegint una línia al manifest es podia aconseguir més fàcilment el mateix que es podria fer amb un objecte Bundle.

S'ha d'afegir això com a atribut de l'activitat definida al manifest:

```
android:configChanges="orientation|screenSize"
```

.

(No és res espectacular però és per dir alguna cosa.)