

# Weather APP <sup>2 0 2 5</sup>

## **Fet per:**

Iolanda Martínez

Iman Idrissi

Joel Espinosa

<b>1. INTRODUCCIÓ.....</b>	<b>3</b>
1.1 Descripció general del projecte.....	3
1.2 Objectiu de l'aplicació.....	3
1.3 Problema que resol.....	4
1.4 Públic objectiu.....	5
<b>2. TECNOLOGIES UTILITZADES.....</b>	<b>5</b>
2.1 Llenguatges i frameworks.....	5
2.2 Llenguatges i frameworks.....	6
2.3 Llenguatges i frameworks.....	6
PETITA CONCLUSIÓ.....	7
<b>3. REQUISITS DEL SISTEMA.....</b>	<b>7</b>
3.1 Sistemes operatius compatibles.....	7
3.2 Dependències i llibreries necessàries.....	7
3.3 Espai requerit i requisits mínims del dispositiu.....	8
CONSIDERACIONS ESPECIALS.....	9
<b>4. INSTAL·LACIÓ I CONFIGURACIÓ INICIAL.....</b>	<b>9</b>
4.1 Descàrrega del projecte.....	9
4.2 Configuració de l'entorn (Flutter i Android Studio).....	10
4.3 Execució inicial de l'aplicació.....	10
4.4 Configuració del fitxer pubspec.yaml.....	11
4.5 Permisos necessaris.....	11
Altres consells de configuració:.....	11
<b>5. INSTRUCCIONS D'ÚS.....</b>	<b>12</b>
5.1 Pantalla de benvinguda i SplashScreen.....	12
5.2 Configuració inicial de ciutats (MainCitiesScreen).....	12
5.3 Pantalla principal – Home (HomeScreen).....	13
5.4 Pantalla de ciutat – Detalls (CityScreen).....	13
5.5 Pantalla astronòmica (AstronomyScreen).....	14
5.6 Pantalla del radar meteorològic (RadarScreen).....	14
5.7 Canvi d'unitats (°C / °F).....	15
5.8 Afegir i eliminar ciutats.....	15
5.9 Navegació entre pantalles.....	15
<b>6. Flux de navegació.....</b>	<b>16</b>
6.1 Diagrama general de navegació.....	16
6.2 Transicions entre pantalles.....	16
6.3 Comportament especial del flux.....	17
<b>7. ESTRUCTURA DEL CODI I EXPLICACIÓ DELS FITXERS .DART.....</b>	<b>17</b>
7.1 Punt d'entrada de l'aplicació (main.dart).....	17
7.2 Pantalles (screens/).....	17
7.3 Components visuals (widgets/).....	19
7.4 Model de dades (models/weather_model.dart).....	19
7.5 Servei d'API(services/weather_service.dart).....	20
7.6 Experiència visual de l'usuari.....	20

<b>8. MANEIG DE DADES I PERSISTÈNCIA.....</b>	<b>20</b>
8.1 Accés i ús de SharedPreferences.....	20
8.2 Gestió de preferències i ciutats.....	20
8.3 Tractament d'errors i dades sensibles.....	21
<b>9. ÚS DE LES APIS EXTERNES.....</b>	<b>21</b>
9.1 OpenWeatherMap: endpoints i exemples.....	21
9.2 IPGeolocation: dades astronòmiques.....	21
9.3 Gestió de respostes i dades.....	22
<b>10. ARQUITECTURA DEL SISTEMA I FLUX GENERAL.....</b>	<b>22</b>
10.1 Interacció entre pantalles i serveis.....	22
10.2 Flux de dades entre components.....	23
<b>11. CONCLUSIONS I MILLORES FUTURES.....</b>	<b>23</b>
11.2 Reflexions del desenvolupament.....	23
11.2 Funcionalitats pendents o suggerides.....	24
11.3 Possibilitats d'escalat o comercialització.....	24
CONCLUSIÓ FINAL:.....	24

# 1.INTRODUCCIÓ

## 1.1 Descripció general del projecte

WeatherApp és una aplicació mòbil multiplataforma desenvolupada amb Flutter, pensada per oferir informació meteorològica detallada, fiable i visualment atractiva a usuaris de qualsevol nivell tècnic. A través d'una interfície moderna, neta i intuïtiva, l'aplicació permet consultar el temps actual, la previsió horària i diària, la situació astronòmica, i accedir a un radar meteorològic interactiu en temps real.

Els usuaris poden personalitzar la seva experiència afegint fins a 4 ciutats favorites, així com utilitzar la seva ubicació actual per obtenir informació precisa del temps en temps real. L'aplicació gestiona les dades a través de la integració amb l'API OpenWeatherMap, i utilitza SharedPreferences per conservar la configuració personalitzada i les preferències de l'usuari.

A més, WeatherApp ofereix funcionalitats avançades que la diferencien d'altres aplicacions del mercat, com ara: animacions visuals atractives, previsió astronòmica en base a coordenades reals, i compatibilitat amb múltiples idiomes dins el radar meteorològic.

## 1.2 Objectiu de l'aplicació

L'objectiu principal de WeatherApp és facilitar un accés ràpid, visual i personalitzat a la informació meteorològica de diverses localitzacions, ajudant l'usuari a prendre decisions informades en el seu dia a dia.



### Objectius específics:

- ❖ Mostrar informació del temps actual segons la ubicació de l'usuari o ciutats configurades.
- ❖ Permetre la consulta de la previsió meteorològica horària i diària.

- ❖ Oferir una experiència visual moderna i elegant, amb una interfície fosca que millora la llegibilitat.
- ❖ Mostrar dades detallades: temperatura, sensació tèrmica, pressió, humitat, visibilitat, vent i índex UV.
- ❖ Incloure funcionalitats complementàries com la previsió astronòmica, amb sortida i posta del sol, fases de la lluna i signe zodiacal.
- ❖ Accedir a un radar meteorològic interactiu, amb capes com vent, pluja, temperatura i núvols.
- ❖ Permetre l'ús de diferents unitats de temperatura (°C / °F) i conservar configuracions gràcies a SharedPreferences.
- ❖ Adaptar-se a diferents idiomes (Català, Castellà, Anglès, segons la configuració del radar).

### 1.3 Problema que resol

Actualment, hi ha moltes aplicacions meteorològiques disponibles, però moltes pateixen problemes comuns:

- Interfícies complexes, amb massa informació o poca claredat visual.
- Publicitat invasiva o funcionalitats premium de pagament.
- Dades poc personalitzades o ubicacions no detectades amb precisió.
- Aplicacions massa carregades o lentes en dispositius més antics.

WeatherApp resol aquests problemes mitjançant:

- Una interfície neta, ràpida i intuïtiva sense anuncis.
- Informació visual i fàcil d'interpretar per a tots els nivells d'usuari.
- Ús de dades en temps real provinents d'una font fiable (OpenWeatherMap).
- Possibilitat de personalitzar ciutats i de mostrar la ubicació actual automàticament.
- Funcionalitats afegides que la diferencien de la competència (animacions, astronomia, radar interactiu).

## 1.4 Públic objectiu

L'aplicació està dissenyada per a un ampli ventall d'usuaris, però principalment enfocada a:

- ❖ Usuaris generals que necessiten consultar el temps cada dia abans de sortir.
- ❖ Estudiants, professionals i famílies que organitzen activitats a l'exterior.
- ❖ Docents i alumnes que volen conèixer un exemple funcional i complet de Flutter amb integració d'API.
- ❖ Persones que busquen una alternativa visualment atractiva i més senzilla que les aplicacions tradicionals.
- ❖ Professionals del transport, el turisme o l'agricultura, que depenen de condicions meteorològiques concretes.

En resum, WeatherApp està pensada per a tothom que valori una aplicació moderna, funcional, i visualment cuidada, amb especial èmfasi en la usabilitat, la precisió de la informació i l'estètica minimalista.

## 2. TECNOLOGIES UTILITZADES

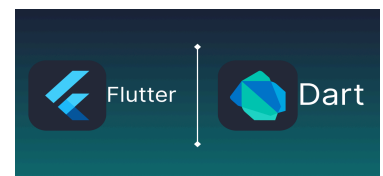
### 2.1 Llenguatges i frameworks

#### Flutter

El projecte ha estat desenvolupat íntegrament amb **Flutter**, el framework de codi obert creat per Google que permet construir aplicacions **multiplataforma** amb una sola base de codi. Flutter utilitza el llenguatge **Dart** per definir tant la lògica com la interfície gràfica de l'aplicació.

Beneficis clau:

- Una sola base de codi per a **Android i iOS**.
- Interfícies **reactives** i **altament personalitzables**.
- Integració senzilla amb **APIs REST**.
- Ampli suport per a **animacions i efectes visuals**.



## Dart

Dart és el llenguatge de programació emprat per Flutter. És un llenguatge orientat a objectes, segur i fàcil de llegir. Totes les classes, models, interfícies i serveis de l'aplicació estan desenvolupades en Dart, estructurades per funcionalitats.

## 2.2 Llenguatges i frameworks

L'aplicació utilitza diversos paquets de la comunitat Flutter i del sistema per afegir funcionalitats addicionals. A continuació es descriuen els més rellevants:

PAQUET	FINALITAT
<b>http</b>	Realitzar peticions a APIs externes (OpenWeatherMap, API d'Astronomia).
<b>intl</b>	Formatació de dates i hores segons el locale.
<b>geolocator</b>	Obtenir la ubicació actual del dispositiu (latitud i longitud).
<b>webview_flutter</b>	Mostrar el radar meteorològic embegut mitjançant una WebView.
<b>visibility_detector</b>	Detectar quan un element està a la vista per aplicar animacions.
<b>shared_preferences</b>	Guardar configuració local, com ciutats afegides, nom de la ubicació, etc.

## 2.3 Llenguatges i frameworks

EINA	US AL PROJECTE
<b>Android Studio</b>	Entorn de desenvolupament principal (IDE) per programar i emular l'app.
<b>Flutter SDK</b>	Plataforma i compilador per construir l'aplicació.

<b>DartPad i emuladors</b>	Per fer proves ràpides del codi i test de la interfície.
<b>Git &amp; GitHub</b>	Control de versions i dipòsit col·laboratiu del projecte.
<b>OpenWeatherMap API</b>	Proporciona dades meteorològiques en temps real i previsió per hores i dies.
<b>IPGeolocation API</b>	API astronòmica per obtenir informació com fases lunars i altitud solar/lunar.
<b>Asset bundles</b>	Per incloure icones i imatges personalitzades (meteorològiques i de fons).

## PETITA CONCLUSIÓ

Tot el codi Dart es troba estructurat en carpetes per components: screens, widgets, services, models, utils, etc., la qual cosa facilita la modularitat i el manteniment. I les animacions estan implementades amb widgets personalitzats (AnimatedGaugeTile, FadeInOnScroll, etc.), i no depenen de llibreries externes, optimitzant així el rendiment i la fluïdesa de l'app.

## 3.REQUISITS DEL SISTEMA

### 3.1 Sistemes operatius compatibles

L'aplicació WeatherApp ha estat desenvolupada i provada per funcionar correctament en dispositius mòbils amb els següents sistemes operatius:

- ❖ Android (recomanat: versió 8.0 o superior – API 26+)
- ❖ iOS (recomanat: iOS 13 o superior)

Tot i que el desenvolupament s'ha realitzat principalment en entorn Android, el codi és 100% compatible amb iOS gràcies a Flutter.



## 3.2 Dependències i llibreries necessàries

Perquè l'aplicació funcioni correctament, és imprescindible tenir accés a les següents llibreries i serveis externs:

### LLIBRERIES (AL PUBSPEC.YAML):

#### APIS EXTERNES REQUERIDES:

- OpenWeatherMap  
(<https://openweathermap.org/api>)
- Dades meteorològiques actuals i previsió horària/diària.
- IPGeolocation Astronomy API (<https://ipgeolocation.io>)
- Dades astronòmiques (fases lunars, alçada solar/lunar, zodíac).

```
dependencies:  
  flutter:  
    sdk: flutter  
  http: ^0.13.5  
  geolocator: ^9.0.2  
  shared_preferences: ^2.0.15  
  intl: ^0.18.0  
  webview_flutter: ^4.0.6  
  visibility_detector: ^0.4.0
```

#### PERMISOS NECESSARIS:



- Accés a la ubicació del dispositiu (ACCESS\_FINE\_LOCATION)
- Accés a Internet per consumir APIs
- Accés al sistema d'arxius intern per guardar preferències (SharedPreferences)

## 3.3 Espai requerit i requisits mínims del dispositiu

### Espai d'emmagatzematge:

- Aproximadament 40 MB (incloent fitxers d'icones, imatges i assets locals).

### Requisits mínims del dispositiu:

PAQUET	FINALITAT
CPU	ARMv8 (64-bit) o superior
RAM	Mínim 2 GB
RESOLUCIÓ	720x1280 px o superior
CONNEXIÓ	Wi-Fi o dades mòbils actives
SENSORS	Sensor GPS activat (per a funcionalitat de "La meua ubicació")

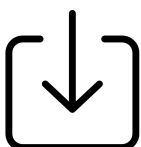
## CONSIDERACIONS ESPECIALS

- ❖ El rendiment pot veure's afectat si l'usuari bloqueja l'accés a la ubicació o té una connexió lenta.
- ❖ En cas que no es concedeixi el permís de localització, l'aplicació ofereix un mecanisme alternatiu per afegir ciutats manualment.
- ❖ L'aplicació s'adapta automàticament a diferents mides de pantalla i funciona tant en orientació vertical com horitzontal (encara que l'ús recomanat és en vertical).

## 4. INSTAL·LACIÓ I CONFIGURACIÓ INICIAL

### 4.1 Descàrrega del projecte

Per començar a utilitzar l'aplicació WeatherApp com a desenvolupador o tester tècnic, cal primer obtenir el codi font. Aquest està allotjat a un repositori de GitHub.



#### **Passos per descarregar el projecte:**

1. Accedeix al repositori del projecte des de l'enllaç proporcionat (ex. <https://github.com/usuari/weatherapp>).
2. Clica al botó Code i selecciona:
  - Download ZIP per baixar-lo manualment.
  - O copia la URL del repositori i clona amb Git:  
`git clone https://github.com/usuari/weatherapp.git`
3. Descomprimeix el projecte (si cal) i obre'l a Android Studio o Visual Studio Code amb suport per a Flutter.

### 4.2 Configuració de l'entorn (Flutter i Android Studio)

Per executar l'aplicació, has de tenir configurat correctament l'entorn de Flutter. A continuació t'expliquem com fer-ho:

#### **Requisits previs:**

- Flutter SDK (versió recomanada: 3.10.0 o superior)

- Android Studio o editor alternatiu com Visual Studio Code
- Dispositiu físic o emulador per a proves

### **Instal·lació de Flutter SDK (si no el tens):**

<https://docs.flutter.dev/get-started/install>

### **Després d'instal·lar, comprova que tot està correcte amb:**

**flutter doctor**

### **Assegura't que no hi hagi errors en cap dels següents:**

- ❖ Flutter SDK
- ❖ Android toolchain
- ❖ Android Studio
- ❖ Connected device


## 4.3 Execució inicial de l'aplicació

### **PASSOS DETALLATS:**

1. Obre la carpeta del projecte amb Android Studio.
2. Assegura't de tenir seleccionat un dispositiu virtual (AVD) o el teu mòbil connectat via USB amb depuració activada.
3. Executa la comanda següent al terminal integrat: **flutter pub get**



Aquesta comanda descarrega totes les dependències especificades al fitxer pubspec.yaml.

4. Un cop descarregades, fes clic al botó Run  de la barra superior o escriu: **flutter run**
5. Després de compilar, es carregarà la pantalla d'inici animada (**SplashScreen**) i es redirigirà a la pantalla de configuració inicial.

## 4.4 Configuració del fitxer pubspec.yaml

Aquest fitxer conté totes les dependències, assets i configuracions bàsiques del projecte. Assegura't que conté:

Exemple de secció crítica:

**Important:** Si afegeixes nous arxius a la carpeta **assets/**, executa de nou:

**flutter pub get**

```
dependencies:
  flutter:
    sdk: flutter
  http: ^0.13.5
  geolocator: ^9.0.2
  shared_preferences: ^2.0.15
  intl: ^0.18.0
  webview_flutter: ^4.0.6
  visibility_detector: ^0.4.0

flutter:
  assets:
    - assets/icons/
    - assets/images/
```



## 4.5 Permisos necessaris

Perquè l'aplicació funcioni correctament, cal garantir que tingui accés als recursos següents:

PERMÍS	PROPÒSIT
ACCESS_FINE_LOCATION	Obtenir la ubicació precisa de l'usuari.
INTERNET	Consultar l'API de dades meteorològiques.
ACCESS_COARSE_LOCATION	Ubicació aproximada com a alternativa.

Aquests permisos s'inclouen automàticament a AndroidManifest.xml. En cas que no es mostrin correctament, comprova que el fitxer conté:

**Android/app/src/main/AndroidManifest.xml:**

`<uses-permission android:name="android.permission.INTERNET"/>`

`<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>`

### Altres consells de configuració:

- ❖ Depuració per USB activada al mòbil (si es prova en dispositiu real).
- ❖ Si no funciona la localització, comprova que els serveis de localització del dispositiu estiguin activats.
- ❖ Es recomana l'ús d'emuladors Android x86 o ARM64 per garantir compatibilitat.

## 5. INSTRUCCIONS D'ÚS

A continuació descriurem totes les funcionalitats principals de l'aplicació **WeatherApp**, explicant pantalla per pantalla com utilitzar-la. Aquest apartat està pensat per a usuaris finals i conté indicacions visuals per acompanyar cada explicació.



## 5.1 Pantalla de benvinguda i SplashScreen

Quan l'usuari obre l'aplicació per primer cop, apareix una pantalla inicial amb un logotip animat i un fons degradat de tons foscos. Aquesta pantalla serveix com a intro visual elegant mentre es carreguen les dades.


- ❖ La durada de l'animació és de 4 segons i, posteriorment:
  - Si l'usuari ja havia configurat ciutats prèviament, es redirigeix automàticament a la Home.
  - Si és la primera vegada, es mostra la pantalla de configuració inicial.

 Splash Screen.mp4

## 5.2 Configuració inicial de ciutats (MainCitiesScreen)

- Aquesta pantalla només es mostra la primera vegada que s'instal·la l'aplicació, o si es fa un reset manual de preferències. L'usuari pot:
- Confirmar la ubicació actual, mostrada automàticament mitjançant el GPS.
- Afegir fins a 4 ciutats addicionals manualment, escrivint el nom i prement Enter.
- Eliminar ciutats tocant la "X" de cada etiqueta.
- Quan ha seleccionat les ciutats desitjades, pot prémer "Continuar" per accedir a la pantalla principal.
- La configuració es guarda automàticament a **SharedPreferences** per a futurs llançaments.

La configuració es guarda automàticament a **SharedPreferences** per a futurs llançaments.

 Main Cities Screen.mp4

## 5.3 Pantalla principal – Home (HomeScreen)

La pantalla principal mostra la previsió actual de totes les ciutats configurades, inclosa la ubicació actual. Cada ciutat es mostra com una targeta interactiva (WeatherCard) amb:

- ❖ Nom de la ciutat (amb icona de localització si és la ubicació actual) 📌.
- ❖ Icona del clima actual i fons dinàmic en funció del temps ☁️.
- ❖ Temperatura actual, màxima i mínima 🌡️.
- ❖ Índex UV amb color adaptat 🌞.
- ❖ Hora actual 🕒.

### **Altres funcionalitats:**

- ❖ Buscador de ciutats a la part superior 🔍.
- ❖ Botó per canviar unitats °C / °F ➕.
- ❖ Botó per accedir al radar meteorològic 🌩️.

Pots eliminar una ciutat fent swipe (arrossegar lateralment) la targeta.

📺 Home Screen.mp4

## 5.4 Pantalla de ciutat – Detalls (CityScreen)

Quan l'usuari toca sobre una ciutat, entra a la pantalla detallada on es mostra:

- ❖ Header de la ciutat:
  - Nom de la ciutat
  - Icona del clima
  - Temperatura i descripció
  - Màxima i mínima del dia
  -
- ❖ Previsió horària (ForecastHourly):
  - Les 12 hores properes amb temperatura i icona
  - Vista horitzontal i desplaçable
- ❖ Previsió diària (ForecastDaily):
  - Els pròxims 7 dies amb màxima i mínima
  - Icona per cada jornada
- ❖ Detalls meteorològics ampliat (WeatherDetailGrid):
  - Sensació tèrmica
  - Pressió atmosfèrica
  - Humitat
  - Visibilitat
  - Índex UV (amb color)
  - Velocitat i angle del vent

A la part superior hi ha un botó per accedir a la pantalla astronòmica 🗺️.

📺 City Screen.mp4



## 5.5 Pantalla astronòmica (AstronomyScreen)

Aquesta pantalla mostra informació relacionada amb l'estat actual del cel:

- Fase lunar actual (amb símbol i nom traduït)
- Hora de sortida i posta del sol
- Alçada solar i lunar
- Signe zodiacal actual segons la data
- Dades obtingudes de l'API d'astronomia (ipgeolocation.io)

Tot es mostra en un disseny minimalista amb targetes i icones visuals.

📺 Astronomy Screen.mp4

## 5.6 Pantalla del radar meteorològic (RadarScreen)

L'usuari pot visualitzar un radar interactiu amb mapes reals en viu. El radar ofereix diverses capes:

- Vent
- Pluja
- Temperatura
- Núvols
- Ones
- Pressió



També permet seleccionar l'idioma de visualització i reiniciar la ubicació amb un botó flotant.

📺 Radar Screen.mp4

## 5.7 Canvi d'unitats (°C / °F)

Des de la pantalla principal, l'usuari pot canviar les unitats de temperatura amb un botó flotant. L'aplicació fa el càlcul automàtic:

→ °C → °F:  $(\text{temp} * 9/5) + 32$

→ °F → °C:  $((\text{temp} - 32) * 5) / 9$

Aquest canvi afecta totes les pantalles de la interfície de forma immediata.

## 5.8 Afegir i eliminar ciutats

- ❖ Per afegir una ciutat, només cal escriure-la al buscador i prémer Enter.
- ❖ Per eliminar-la, es pot fer:
  - Des de la pantalla principal: arrossegant la targeta
  - Des de la configuració inicial: tocant la "X" de la ciutat

Els canvis es guarden automàticament a SharedPreferences.

## 5.9 Navegació entre pantalles

La navegació és simple i està basada en accions clares de l'usuari:

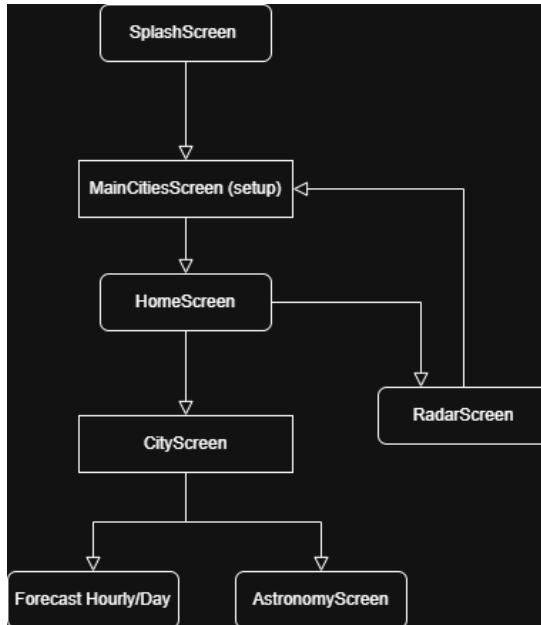
- **Tocar una ciutat** → entra a CityScreen
- **Des de CityScreen** → accés a AstronomyScreen amb icona estelada
- **Des de la Home** → accés a Radar amb el botó flotant
- El botó enrere torna sempre a la pantalla anterior

Aquestes són totes les funcionalitats clau que componen l'aplicació **WeatherApp**, pensades perquè qualsevol usuari pugui obtenir informació meteorològica fiable i elegant amb facilitat.



## 6. Flux de navegació

### 6.1 Diagrama general de navegació



### 6.2 Transicions entre pantalles

A continuació explicarem totes les transicions disponibles dins l'app:

- ❖ SplashScreen → MainCitiesScreen
  - Quan l'usuari no té ciutats configurades.
  - Es mostra només una vegada.
- ❖ SplashScreen → HomeScreen
  - Si ja hi ha ciutats guardades a SharedPreferences.
- ❖ MainCitiesScreen → HomeScreen
  - Un cop l'usuari ha afegit ciutats i prem el botó "Continuar".
- ❖ HomeScreen → CityScreen
  - Tocant sobre qualsevol targeta del temps d'una ciutat.
  - Es carrega tota la informació detallada.
- ❖ CityScreen → AstronomyScreen
  - Prement la icona estelada (☀️) a la part superior dreta.
- ❖ HomeScreen → RadarScreen

- Prement el botó flotant amb icona de radar (🌩️).
- Retorn entre pantalles
  - Totes les pantalles tenen botó de retrocés (←) que retorna a la pantalla anterior.
  - No es perden dades ni configuracions en cap moment del procés.

### 6.3 Comportament especial del flux

L'app recorda les ciutats seleccionades i la unitat de temperatura. L'usuari pot tornar a CityScreen tantes vegades com vulgui des de Home. Les dades es carreguen de forma dinàmica i les animacions suavitzen les transicions visuals.

## 7. ESTRUCTURA DEL CODI I EXPLICACIÓ DELS FITXERS .DART

L'aplicació WeatherApp segueix una estructura modular i clara, amb fitxers separats per responsabilitats: pantalles (screens), components visuals (widgets), models de dades (models) i serveis (services).

A continuació s'explica cada fitxer rellevant del projecte:

### 7.1 Punt d'entrada de l'aplicació (main.dart)

És el fitxer principal. Aquí es configura:

- El tema visual (ThemeData.dark)
- El títol de l'app: App del Temps
- La pantalla inicial (SplashScreen)

També s'inicialitza WidgetsFlutterBinding per assegurar que Flutter estigui llest abans d'accedir a recursos del sistema.

### 7.2 Pantalles (screens/)

#### → Splash\_screen.dart:

- ❖ Mostra el logotip animat amb una animació d'escala i un missatge de benvinguda.

- ❖ Després de 4 segons, consulta SharedPreferences per saber si hi ha ciutats guardades.
  - Si n'hi ha: navega a HomeScreen
  - Si no: navega a MainCitiesScreen

Important: Aquí es defineix la primera lògica de navegació basada en preferències.

→ **Main\_cities\_screen.dart:**

- ❖ Permet afegir fins a 4 ciutats i mostrar la ubicació actual.
- ❖ Guarda la llista a SharedPreferences amb clau 'ciutats'.
- ❖ També mostra errors si la ubicació no és accessible.
- ❖ En prémer "Continuar", es navega cap a HomeScreen.

→ **Home\_screen.dart:**

- ❖ Mostra les targetes del temps (WeatherCard) per cada ciutat configurada.
- ❖ Inclou:
  - Botó per canviar unitats (°C/°F)
  - Botó per accedir a RadarScreen
  - Buscador per afegir ciutats
- ❖ Gestiona la lectura de dades mitjançant WeatherService.

→ **City\_screen.dart:**

- ❖ Pantalla detallada amb informació meteorològica completa d'una ciutat.
- ❖ Compon amb diversos widgets:
  - CityHeader
  - ForecastHourly
  - ForecastDaily
  - WeatherDetailGrid
- ❖ També ofereix accés a AstronomyScreen.

→ **Radar\_screen.dart:**

- ❖ Mostra un WebView amb el radar meteorològic de windy.com.
- ❖ Permet escollir la capa (vent, pluja, núvols...) i l'idioma.

→ **Astronomy\_screen.dart:**

- ❖ Consulta l'API de ipgeolocation.io.
- ❖ Mostra:
  - Fase lunar (amb emoji i nom)
  - Hora de sortida i posta del sol
  - Altitud solar i lunar
  - Signe zodiacal

## 7.3 Components visuals (widgets/)

**Weather\_card.dart:**

- Targeta resum amb dades actuals del clima per ciutat.
- Toca per accedir a CityScreen.
- Mostra temperatura, descripció, UV i icona meteorològica amb fons personalitzat.

**Forecast\_hourly.dart:**

- Llista horitzontal amb previsió de 12 hores futures.
- Temperatura, hora i icona.

**Forecast\_daily.dart:**

- Mostra la previsió per als pròxims 7 dies.
- Inclou temperatura màxima i mínima i el dia de la setmana.

**City\_header.dart:**

- Capçalera de la ciutat: icona, temperatura actual, màxima/minima i descripció.

**Weather\_detail\_grid.dart:**

Mostra un grid animat amb:

- Sensació tèrmica
- Pressió
- Humitat
- Índex UV
- Visibilitat

Utilitza AnimatedGaugeTile i DetailTile.

## 7.4 Model de dades (models/weather\_model.dart)

```
String ciutat;  
double temperatura;  
String descripcio;  
String icona;  
int humitat;  
int pressio;  
double ventVelocitat;  
int windDeg;  
int visibilitat;  
double sensacioTermica;  
DateTime sortidaSol;  
DateTime postaSol;  
double latitud;  
double longitud;
```

**Classe Weather;** Model per desar tota la informació meteorològica d'una ciutat: Té constructor `fromJson` per parsejar dades de l'API.

## 7.5 Servei d'API (services/weather\_service.dart)

**Classe WeatherService:** Encapsula totes les crides a OpenWeatherMap:

- `fetchCurrentWeather(city)`
- `fetchForecast(city)`
- `fetchWeatherByCoords(lat, lon)`
- `fetchForecastByCoords(lat, lon)`
- `fetchUvIndex(lat, lon)`

També conté l'`apiKey`, la `baseUrl` i gestiona errors de xarxa.

## 7.6 Experiència visual de l'usuari

L'ús d'aquestes animacions no només aporta dinamisme, sinó que contribueix a:

- Millorar la claredat i jerarquia visual
- Fer que el contingut sigui més comprensible
- Aportar una experiència moderna i agradable

L'objectiu ha estat que l'app es mogui com si estigués viva, sense resultar excessiva o molesta.

## 8.MANEIG DE DADES I PERSISTÈNCIA

### 8.1 Accés i ús de SharedPreferences

L'aplicació WeatherApp fa ús de SharedPreferences, una llibreria de Flutter per a l'emmagatzematge local i persistent de dades senzilles com cadenes, llistes o valors bàsics. Això permet mantenir configuracions de l'usuari encara que l'app es tanqui.

#### Dades que es guarden localment:

1. Ciutats: Llista de ciutats afegides per l'usuari.
2. nomUbicacio: Nom de la ubicació actual detectada.

### 8.2 Gestió de preferències i ciutats

L'usuari pot afegir ciutats manualment o confiar en la ubicació automàtica. Aquestes dades es persistiran entre sessions:

- Les ciutats afegides es visualitzen a HomeScreen.
- Les ciutats es poden eliminar fent swipe lateral o tocant "X" a MainCitiesScreen.
- La configuració es guarda cada cop que hi ha canvis.

A més, l'ordre de les ciutats es conserva, i sempre es manté "La meua ubicació" com a primer element.

### 8.3 Tractament d'errors i dades sensibles

Tot i que l'app no gestiona informació confidencial (com dades bancàries o comptes), es tenen en compte les bones pràctiques següents:

- No es guarden dades personals sensibles.
- Les peticions a l'API es fan via HTTPS.
- Si falla una crida a l'API, es mostra un missatge d'error elegant i no es bloqueja l'app:

```
ScaffoldMessenger.of(context).showSnackBar(  
  SnackBar(  
    content: Text('Error carregant dades: $e'),  
    backgroundColor: Colors.red,  
  ),  
);
```

## 9. ÚS DE LES APIS EXTERNES

L'aplicació WeatherApp es basa en dues fonts de dades externes per oferir informació meteorològica i astronòmica en temps real: OpenWeatherMap i IPGeolocation. Les dades s'obtenen mitjançant peticions HTTP i es processen en format JSON.

### 9.1 OpenWeatherMap: endpoints i exemples

#### Base URL:

- <https://api.openweathermap.org/data/2.5/>

#### API key personal integrada al fitxer weather\_service.dart.

#### Endpoints utilitzats:

- **/weather:** Obtenir el temps actual per ciutat o coordenades.
- **/forecast:** Obtenir la previsió meteorològica horària i diària.
- **/onecall:** Obtenir l'índex UV per coordenades geogràfiques.

### 9.2 IPGeolocation: dades astronòmiques

#### DADES OBTINGUDES:

- Hora de sortida i posta del sol.
- Altura solar i lunar.
- Fase lunar actual.
- Signe zodiacal (calculat a partir de la data).

### 9.3 Gestió de respostes i dades

Les dades rebudes es processen mitjançant:

- `http.get()` per fer la petició.
- `json.decode(response.body)` per convertir a format usable.
- Models com `Weather.fromJson()` per transformar-ho en objectes de Flutter.

En cas d'error (per exemple, ciutat no trobada), es llença una excepció controlada amb missatge informatiu per a l'usuari.

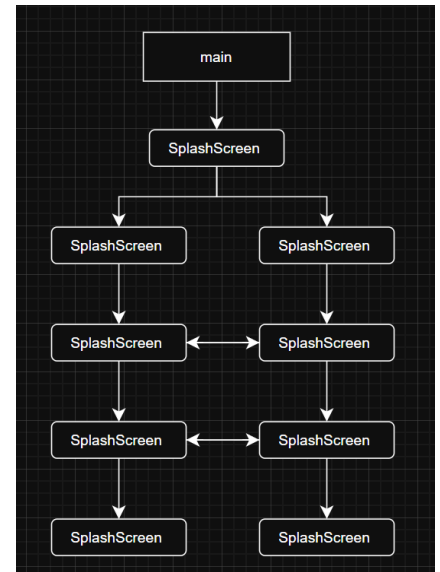
# 10. ARQUITECTURA DEL SISTEMA I FLUX GENERAL

Aquest gràfic resumeix les principals connexions entre pantalles, serveis i models de dades.

## 10.1 Interacció entre pantalles i serveis

L'aplicació es basa en una arquitectura centrada en pantalles i serveis, amb una separació clara de responsabilitats:

- **Pantalles (Screens):** gestionen la interfície d'usuari i la navegació.
- **Widgets:** són components reutilitzables com targetes, llistes i detalls visuals.
- **Serveis (WeatherService):** encapsulen tota la comunicació amb les APIs.
- **Models (Weather):** estructuren les dades obtingudes i permeten fàcil accés i manipulació.
- **SharedPreferences:** gestiona la persistència local i decisions de flux.



## 10.2 Flux de dades entre components

### Flux típic:

1. L'usuari obre l'app → SplashScreen comprova configuració guardada.
2. Segons les dades, es mostra MainCitiesScreen o HomeScreen.
3. HomeScreen demana dades a WeatherService per a cada ciutat.
4. Les dades es processen al model Weather i es mostren en WeatherCard.
5. En tocar una ciutat → CityScreen mostra detalls i crida més dades.
6. L'usuari pot accedir a RadarScreen o AstronomyScreen des d'allà.

### Tots els serveis són reactius i asíncrons:

- async/await garanteixen que l'usuari només vegi dades carregades.
- Els errors es controlen per evitar talls o fallades visuals.



# 11. CONCLUSIONS I MILLORES FUTURES

## 11.2 Reflexions del desenvolupament

Durant el desenvolupament de **WeatherApp**, s'ha pogut aconseguir una aplicació completa, estable i visualment atractiva, que compleix amb els requisits definits al principi del projecte:

- Integració amb una API meteorològica real (OpenWeatherMap).
- Consulta del temps actual i previsió horària i diària.
- Accés a funcionalitats avançades com informació astronòmica i radar en viu.
- Interfície intuïtiva i moderna amb animacions suaus.
- Ús de mecanismes locals de persistència (**SharedPreferences**).
- Desenvolupament modular, amb codi reutilitzable i fàcil de mantenir.

A més, s'ha tingut especial cura en l'experiència d'usuari, tant en l'aspecte visual com en la fluïdesa de la navegació, fet que diferencia l'aplicació de moltes alternatives més carregades o menys personalitzables.

## 11.2 Funcionalitats pendents o suggerides

Tot i que l'app és funcional i completa, s'han identificat algunes millores que podrien implementar-se en el futur per ampliar el valor i la personalització per part de l'usuari:

- **Mode clar/fosc configurable manualment.**
- **Suport multilingüe complet** (català, castellà, anglès a tot l'UI, no només radar/API).
- **Mapa interactiu amb ubicació actual** i punts destacats.
- **Sistema d'alertes meteorològiques push** per pluja o canvis bruscos de temps.
- **Exportació o backup de configuracions** (amb Firebase o localment).
- **Historial de dades o estadístiques** (temperatura mitjana setmanal, etc.).

## 11.3 Possibilitats d'escalat o comercialització

L'aplicació, per la seva estructura modular, pot evolucionar fàcilment cap a una **versió professional o comercial**, afegint:

- **Subscripció per eliminar límit de ciutats** o afegir funcionalitats premium.
- **Integració amb més APIs** (com WeatherAPI, AccuWeather) per oferir comparatives.
- **Publicació a Google Play o App Store**, afegint analytics i sistema de versions.
- **Funcionalitats d'intel·ligència artificial** per prediccions personalitzades (machine learning).
- **Panell d'administració web** per controlar usuaris o configuracions (amb Firebase o Supabase).

## CONCLUSIÓ FINAL:

**WeatherApp** no només ha estat un repte tècnic aconseguit, sinó també una aplicació real, funcional i atractiva, preparada per evolucionar i créixer. Representa un projecte sòlid, tant en experiència d'usuari com en arquitectura de codi, i està alineat amb els estàndards actuals del desenvolupament mòbil modern.