



# Piscine Mobile - 2

## API and data management

*Summary: This document contain the subject for the Module02 of the Piscine Mobile.*

*Version: 2.00*

# Contents

<b>I</b>	<b>Instructions</b>	<b>2</b>
<b>II</b>	<b>Specific instructions</b>	<b>3</b>
<b>III</b>	<b>Introduction</b>	<b>4</b>
<b>IV</b>	<b>Exercise 00: Where are we?</b>	<b>5</b>
<b>V</b>	<b>Exercise 01: Searcher</b>	<b>7</b>
<b>VI</b>	<b>Exercise 02: Fill the views</b>	<b>9</b>
<b>VII</b>	<b>Exercise 03: What's wrong with you!</b>	<b>11</b>
<b>VIII</b>	<b>Submission and peer-evaluation</b>	<b>13</b>

# Chapter I

## Instructions

- Only this page will serve as reference. Do not trust rumors.
- Read attentively the whole document before beginning.
- Your exercises will be corrected by your piscine colleagues.
- The document can be relied upon, do not blindly trust the demos or pictures example which can contain not required additions.
- Got a question? Ask your peer on the right. Otherwise, try your peer on the left.
- By Odin, by Thor ! Use your brain !!!



Intra indicates the date and the hour of closing for your repositories. This date and hour also corresponds to the beginning of the peer-evaluation period for the corresponding piscine day. This peer-evaluation period lasts exactly 24h. After 24h passed, your missing peer grades will be completed with a 0.

# Chapter II

## Specific instructions


This module is the continuation of the previous one. For the sake of clarity, you will copy the project from the previous module into a new repository to work on.

# **Chapter III**

## **Introduction**

# Chapter IV

## Exercise 00: Where are we?

	Exercise :
Where are we?	
Turn-in directory : <b>Module102</b>	
Files to turn in : <i>weatherAppV2_proj</i> and all necessary files	
Forbidden functions : <b>None</b>	

So, in the previous module, you have created a weather app that displays a simple text for now.

It's time to add a real data to your app.  
To do this, you will need :

- A weather API.
- A geolocation from the GPS of the device.
- A Geocoding API to get the city name from the coordinates

Start with Geolocation.

At the start of the application or when the user clicks on a geolocation button, you need to know the location of the device to get the weather.  
You must use the GPS of the device to get the coordinates.

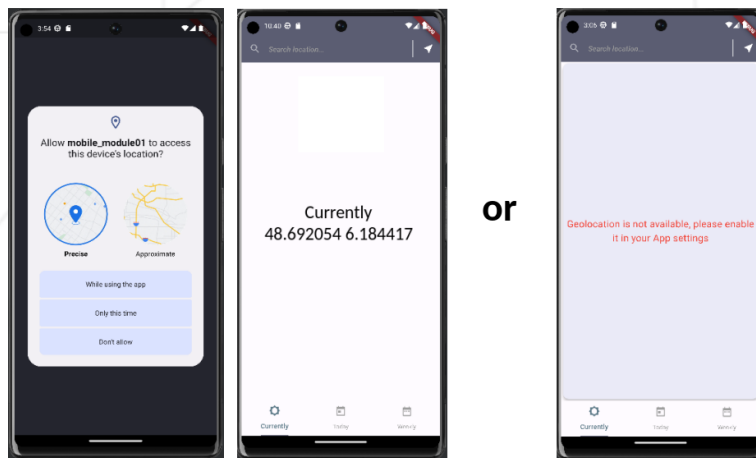
This implies that you have to get the user's permission to retrieve and use the location. You will have to deal with both cases: the user accepts and the user refuses.

If the user accepts, you will have to get the coordinates and use them to get the weather. For now, just display the coordinates in a text.

If the user refuses, your application should still work.  
The user will still be able to enter a city name in the search textfield to get the weather,

but you must inform him not have access to his location.


So now at the start of your application you have something like this:



You should not use an external api for geolocation, you will be forced to use the GPS of the device to get coordinates

# Chapter V

## Exercise 01: Searcher

	Exercise :
	Searcher
Turn-in directory : <b>Module102</b>	
Files to turn in : <i>weatherAppV2_proj</i> and all necessary files	
Forbidden functions : <b>None</b>	

You have also the search bar to get the weather from a city name, country, region, etc.

You will have to use the weather API to get the weather.  
You will have to use the Geocoding API to get the city name from the coordinates or coordinates from the city name.

When you request the Api for the weather by a name of a city, you will often receive a list of cities that match the name you entered.

You will have to display a suggestion list of cities, and each suggestion will have to contain:

- The name of the city.
- The region of the city.
- The country of the city.

To let the user choose the city he wants.

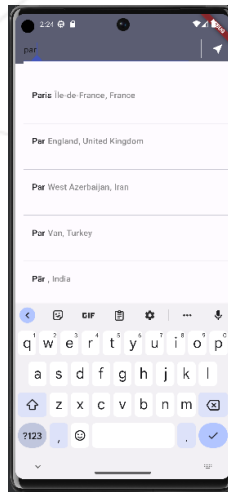
The list of suggestions need to be updated when the user type in the search bar.

When user select a city in the list, you will have to get the weather for this selectedcity.

The user should also search without selecting a city in the list.



You must have something like this:




For this project we have use :

- [Geocoding API](#) from Open-Meteo. to get the coordinates from the city name and fill the search list.

# Chapter VI

## Exercise 02: Fill the views

	Exercise :
Fill the views	
Turn-in directory : <b>Module102</b>	
Files to turn in : <i>weatherAppV2_proj</i> and all necessary files	
Forbidden functions : <b>None</b>	

Now that you have the coordinates, you can use them to get the weather. You have also the search bar to get the weather from a city name, country, region, etc.

You will have to use the weather API to get the weather. You will have to use the Geocoding API to get the city name from the coordinates or coordinates from the city name.

So, it's time to fill the views with the data.

In your first tab "Current" you will have to display:

- The location (city name, region and country).
- The current temperature (in Celsius).
- The current weather description (cloudy, sunny, rainy, etc.).
- the current wind speed (in km/h).

In your second tab "Today" you will have to display:

- The location (city name, region and country).
- The list of the weather for the day with the following data:
  - The hours.
  - The temperatures by hours.

- The weather description (cloudy, sunny, rainy, etc.) by hours.
- The wind speed (in km/h) by hours.

In your third tab “Weekly” you will have to display:

- The location (city name, region and country).
- The list of the weather for each day of the week with the following data:
  - The date.
  - The min and max temperatures of the day
  - The weather description (cloudy, sunny, rainy, etc.).

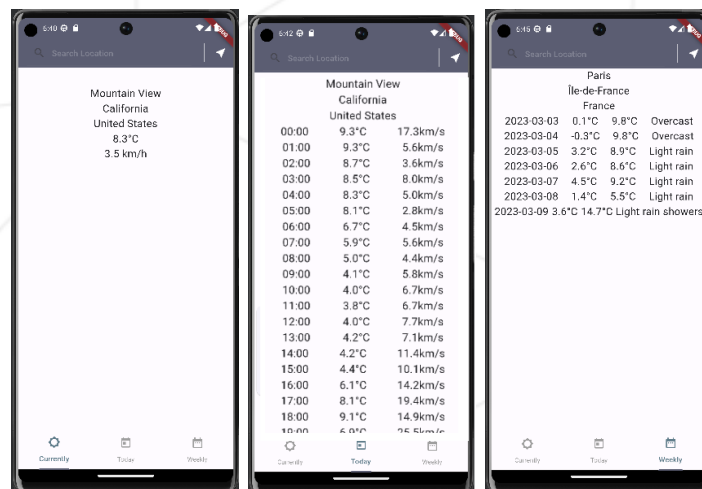
You must pay attention to the following points:

- When you run the application, you must be on the first tab “Current”.
- When you launch a search, you must stay on the tab where you were.
- When you switch tabs, you must always see the data of the last search.

For the moment do not pay attention to the design, just make sure that the information is displayed.

You will have to improve the design in the next module.

You must have something like this:




For this project we will use the following APIs:

- [Weather Forecast API](#) from Open-Meteo.

# Chapter VII

## Exercise 03: What's wrong with you!

	Exercise :
What's wrong with you!	
Turn-in directory : <code>Module102</code>	
Files to turn in : <i>weatherAppV2<sub>proj</sub></i> and all necessary files	
Forbidden functions : None	

You have done a great job!  
But there is still work to be done.

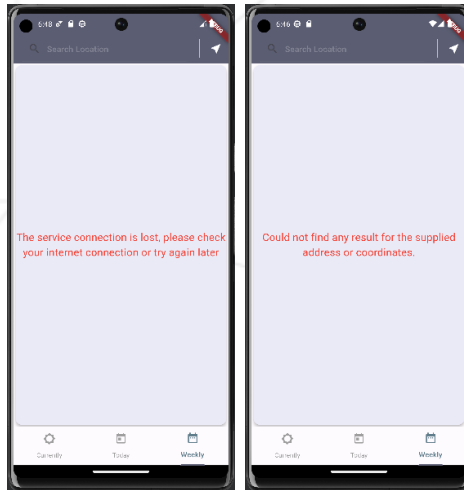
You have already dealt with the case where the user refuses to give you his location, but there are still other cases to deal with:

- The user enters a city name that does not exist.
- The connection to the API fails.

In this both cases, you must inform the user that the city name is not valid or that the connection to the API failed.  
The message should not disappear until the user enters a valid city or until the connection to the API is successful.

Never forget the power of the user to break your application.  
You must always be ready to deal with all cases.

You must have something like this:



# Chapter VIII

## Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.



The evaluation process will happen on the computer of the evaluated group.