# Proyecto lunes 24 de Mayo

Proyecto Funcional - Weather App

## Objetivo

El objetivo de este laboratorio es crear una aplicación de consola que obtenga la temperatura de distintas ciudades a través de una API.

## Problema

Weather App: ***Aquí pueden ver un video de la aplicación***.

### Librerías útiles

- Inquirer: Esta librería los ayudará a realizar las preguntas y recibir los inputs del usuario. Para hacerla síncrona deben usar llamar a la función que la utilice con *await*.
- Prompt-sync: Otra librería que pueden usar para recibir el input del usuario de manera síncrona.
- Console-table-printer: Librería que les permitirá facilmente realizar las tablas en consola.
- Chalk: Librería que permite agregar color a la consola y otras personalizaciones.
- Figlet: Libería que permite hacer el título presentado en el video
- Axios: Libería que permite realizar requests a API's.

Para instalar librerías se recomienda fuertemente que usen proyectos aislados y que no las instalen globalmente. Para esto:

1. Crear una carpeta para el proyecto
2. Abrir la consola en la carpeta y correr el comando npm init ->Este iniciará un proyecto de node en la carpeta.
3. Instalar los paquetes con los comandos de las páginas, por ejemplo, npm install figlet

## Recomendaciones

Se recomienda fuertemente separar su aplicación en:

- app.js: Contiene la parte no funcional, tal como en el contador.
- view.js: Maneja toda la actualización de la vista de manera funcional. Con vista se entiende a lo que se muestra en consola.
- update.js: Actualiza el estado en base a lo que se ingresa en consola de manera funcional.

## Comportamiento de la app

### Parte 1(70 %): Crear la base de la aplicación

En esta primera parte deben implementar toda la funcionalidad de la aplicación, salvo conectarla a la API. Es decir, la aplicación debe poder:

- Agregar ciudades: El usuario debe poder agregar nuevas ciudades a la tabla y que estás se guarden en memoria. En esta primera parte las temperaturas a mostrar pueden ser un valor aleatorio. El usuario solo debe ingresar el nombre de la ciudad a agregar.
- Actualizar una ciudad: El usuario debe poder actualizar los valores de temperatura de una ciudad previamente creada. El usuario debe ser capaz de elegir cualquier ciuidad de las ya ingresadas en la tabla y las temperaturas deben actualizarse a un nuevo valor aleatorio.
- Eliminar una ciudad: El usuario debe ser capaz de elegir cualquier ciudad previamente agregada y eliminarla de la lista.
- Toda acción del usuario debe actualizar la tabla que se muestra en consola

### Parte 2(30 %): Conectar la aplicación a la API

La API a utilizar será Open Wheater API. Deben suscibirse a *Current Weather Data*, la opción gratuita, esto es más que suficiente. Una vez suscritos, debe obtener una API Key. En esta segunda parte deben implementar toda la funcionalidad necesaria para conectar su aplicación a la API:

- Agregar ciudades: Al agregar una ciudad se deben consultar sus temperaturas a la API.
- Actualizar una ciudad: Al actualizar una ciudad debe enviarse esta a la API y actualizar las temperaturas que se muestran (dado que es la versión gratuita lo más seguro es que no cambien, pero la llamada debe hacerse igualmente).
- Eliminar una ciudad: LA api no tiene efecto sobre esta funcionalidad.
- Toda acción del usuario debe actualizar la tabla que se muestra en consola.

**Parte de la dificultad de esta segunda parte es que logren conectarse a la API**

## Entrega

Suba su proyecto a Github antes del viernes 18 de Junio a las 19:00 PM, para esto:

Debes crear tu aplicación dentro del repositorio previamente clonado de Github Classroom en https://classroom.github.com/g/UAmLEMJO.

El proceso para entregar es el mismo que el del laboratorio 0: debes realizar un commit, que es una foto al estado actual de tu proyecto y luego subirlo a github, con los siguientes comandos desde una consola:

```
$ git add .
$ git commit -m "tu mensaje"
$ git push origin master
```