# Módulo 4 – App de Clima (Lógica y Estadísticas en JavaScript)

**1) Propósito**
Reforzar los fundamentos de programación en **JavaScript** implementando la lógica interna de la App de Clima: modelar los datos de clima de los lugares en estructuras de datos, recorrerlos con ciclos, aplicar condicionales y funciones para calcular **estadísticas semanales** y actualizar el DOM.
El foco de esta iteración es la **lógica y el manejo de datos**, no el diseño visual. Mantén la temática que elegiste (playas, montañas, planetas, rutas de trekking, regiones fantásticas, etc.), pero ahora concéntrate en que el comportamiento de la app esté bien resuelto en JavaScript, preparando el proyecto para futuras funcionalidades como **alertas** y conexión a una **API de clima**.

**2) Objetivos de aprendizaje**

- Representar la información de clima de distintos **lugares** utilizando **variables, arreglos y objetos** en JavaScript.
- Utilizar **ciclos** y **sentencias condicionales** para procesar pronósticos y calcular estadísticas.
- Definir y reutilizar **funciones** para separar responsabilidades y evitar duplicar código.
- Manipular el **DOM** para mostrar dinámicamente datos, estadísticas y mensajes en la interfaz.
- Gestionar el proyecto con **Git/GitHub** (commits descriptivos, trabajo incremental, README actualizado).

**3) Alcance**

Se mantiene la estructura base del proyecto:

- **Inicio (Home)**: listado de lugares con clima actual.
- **Detalle de lugar**: información ampliada del lugar seleccionado.

En esta entrega se agrega un foco claro en **datos y cálculos**:

- Los datos de clima **dejan de estar "quemados" en el HTML** y pasan a estar definidos en un archivo JS como un **arreglo de objetos**, por ejemplo:

```
const lugares = [
  {
    id: 1,
    nombre: "Playa del Viento",
    tempActual: 22,
    estadoActual: "Soleado",
    pronosticoSemanal: [
      { dia: "Lunes", min: 18, max: 24, estado: "Soleado" },
      { dia: "Martes", min: 17, max: 23, estado: "Nublado" },
      // ...
    ]
  },
  // más lugares...
];
```

En la **vista de detalle** de cada lugar se añade una sección **“Estadísticas de la semana”** que muestre, al menos:

- Temperatura **mínima**, **máxima** y **promedio** de la semana.
- Cantidad de días por tipo de clima (ej.: cuántos días soleados, nublados, lluviosos), puede cambiar según tu implementación.
- Un pequeño **resumen textual**, por ejemplo:
  - “Semana mayormente soleada.”
  - “Semana fría con varias lluvias.”
  -

**4) Requisitos funcionales mínimos**

- Home debe mostrar **≥ 5 lugares** (pueden ser los mismos del módulo anterior o nuevos), con su clima actual.
- Al seleccionar un lugar (click en card/botón/enlace), se debe mostrar la **vista de detalle** correspondiente.
- En la vista de detalle se debe visualizar:
  - El **pronóstico diario** (lista o cards con día, min, max, estado).
  - La sección **“Estadísticas de la semana”** con:
    - Mínimo, máximo y promedio de temperatura de la semana.
    - Cantidad de días por tipo de clima (al menos 2 tipos: por ejemplo, “soleado” y “lluvioso”).
    - Un mensaje de resumen generado a partir de esos datos.
- El cálculo de estadísticas debe hacerse **en JavaScript** a partir del arreglo pronosticoSemanal de cada lugar (no se permite escribir los resultados “a mano” en el HTML).

**5) Requisitos técnicos**

**Modelado de datos**

- Definir en JavaScript un **arreglo de lugares** (lugares), donde cada lugar sea un **objeto** con al menos:
  - id, nombre, tempActual, estadoActual.
  - pronosticoSemanal: arreglo de objetos con dia, min, max y estado.
- No se permite obtener datos desde una API aún; todo el pronóstico debe estar definido en el código JS.

**Variables, condicionales, ciclos y funciones**

- Utilizar **variables** y constantes para guardar datos intermedios (suma de temperaturas, contadores, etc.).
- Utilizar **ciclos** (for, while o similar) para recorrer el pronóstico semanal y:
  - Calcular mínimo, máximo y promedio.
  - Contar cuántos días hay de cada tipo de clima.
- Utilizar **condicionales** (if, else if, else) para:
  - Evaluar estados del tiempo (ej.: si hay más días soleados que nublados → “Semana mayormente soleada”).
  - Generar el resumen textual de la semana.
- Definir al menos **dos funciones**:
  - Una función para buscar y obtener el **objeto lugar** a partir de un id o nombre.
  - Una función para **calcular estadísticas** a partir del pronosticoSemanal de un lugar y devolver un objeto con los resultados.

## 6) Entregables

- Proyecto comprimido en un único archivo **.zip** que contenga:
  - Archivos HTML.
  - Archivos de estilos (CSS/SASS compilado) ya existentes del módulo anterior.
  - Archivos JavaScript con el modelo de datos y la lógica implementada.
  - Carpeta de recursos (imágenes, íconos, etc.), si corresponde.
- Archivo **README.md** con:
  - Descripción breve de la App de Clima y su temática.
  - Explicación sencilla de cómo están modelados los datos (arreglo de lugares, pronóstico semanal).
  - Resumen de las estadísticas que calcula la app en esta versión.
  - **Enlace al repositorio público de GitHub** del proyecto.

## Rúbrica de evaluación

| Criterio | Excelente (3 pts) | Adecuado (2 pts) | Básico (1 pt) | Insuficiente (0 pts) |
|---|---|---|---|---|
| **Modelado de datos (arrays y objetos)** | Todos los lugares y pronósticos están en un arreglo de objetos claro y completo; nada “quemado” en HTML. | Arreglo de objetos correcto, pero quedan algunos datos sueltos o repetidos fuera del modelo. | Mezcla importante de datos en JS y en HTML; objetos incompletos o poco consistentes. | No hay modelo de datos usable en JS (sin arreglo de lugares o con errores graves). |
| **Lógica JS (ciclos, condicionales, funciones)** | Usa ciclos y condicionales correctamente; funciones separan bien responsabilidades y casi no hay código duplicado. | Usa ciclos y condicionales, pero la organización en funciones es parcial o algo repetitiva. | Uso mínimo o incorrecto de ciclos/condicionales; pocas funciones o mal definidas. | No implementa la lógica requerida (sin ciclos/condicionales ni funciones relevantes). |
| **Estadísticas del clima** | Calcula bien min, max, promedio y conteos; el resumen textual coincide con los datos mostrados. | Calcula la mayoría de estadísticas con detalles menores (p.ej. promedio sin redondeo claro). | Faltan estadísticas o algunas son incorrectas; el resumen no siempre refleja los datos. | No muestra estadísticas o son evidentemente erróneas/incoherentes. |
| **DOM e interfaz de detalle** | Pronóstico y estadísticas se generan dinámicamente desde JS; la vista de detalle es clara y legible. | La mayor parte del contenido se genera desde JS; algunos elementos siguen fijos o poco claros. | Solo una parte se actualiza con JS; mucho contenido fijo en HTML, interfaz confusa. | No se usa el DOM para mostrar pronóstico/estadísticas; la vista no refleja la lógica. |
| **Git/GitHub y README** | ≥3 commits descriptivos; repo público; README explica estructura de datos, estadísticas e incluye enlace al repo. | ≥3 commits aceptables; repo público; README con descripción básica y enlace. | <3 commits o mensajes genéricos; README muy breve o incompleto. | Sin repo público o sin enlace; sin README o vacío. |