**Módulo Gestor de Ciudades**

Los sistemas informáticos de CAMPUS requieren la implementación de una funcionalidad para la gestión de datos de ciudades y departamentos. **Elabore una aplicación web** que permita la gestión de estos datos con las operaciones básicas de: **crear, modificar, listar y eliminar.**

La aplicación debe permitir realizar esta gestión tanto de departamentos como de ciudades teniendo en cuenta que la eliminación de un **Departamento** también eliminará las **ciudades correspondientes a ese departamento**.

Los datos de esta aplicación se manejarán en una base de datos que será administrada a través de **JSON Server** ymétodos **Fetch API** para acceder a éstos.

El siguiente es un modelo de datos que puede usar para la solución informática solicitada:

**{**

**“Departamentos”: [**

**{**

**“id”: 1,**

**“nomDepartamento”: “Santander”**

**},**

**{**

**“id”: 2,**

**“nomDepartamento”: “Cundinamarca”**

**}**

**],**

**“Ciudades”: [**

**{**

**“id”: 1,**

**“nomCiudad”: “Bucaramanga”,**

**“departamentoId”: 1,**

**“imagen”:** [**https://colombia.com/bucaramanga.jpg**](https://colombia.com/bucaramanga.jpg)**,**

**“coordenadas”:{“lat”:4, “lon”:72}**

**},**

**{**

**“id”: 2,**

**“nomCiudad”: “Barrancabermeja”,**

**“departamentoId”: 1,**

**“imagen”: “https://colombia.com/barranca.jpg” ,**

**“coordenadas”:{“lat”:4, “lon”:72}**

**},**

**{**

**“id”: 3,**

**“nomCiudad”: “Bogotá”,**

**“departamentoId”: 2,**

**“imagen”: “https://colombia.com/bogota.jpg” ,**

**“coordenadas”:{“lat”:4, “lon”:72}**

**},**

**{**

**“id”: 4,**

**“nomCiudad”: “Cajicá”,**

**“departamentoId”: 2,**

**“imagen”: “https://colombia.com/cajica.jpg” ,**

**“coordenadas”:{“lat”:4, “lon”:72}**

**}**

**]**

**}**

El sistema debe permitir elegir la apariencia de fondo o de tamaño de letra y recordar esta preferencia para la próxima vez que ingrese.

El sistema debe permitir conocer el estado atmosférico en tiempo real, de la ciudad que se elija, accediendo a la API externa <https://home.openweathermap.org> , para lo cual se manejan las coordenadas geográficas de la ciudad.

RECOMENDACIONES:

1. Para acceder a la API **Openweathermap** debe registrarse en la página de la plataforma con anticipación de dos horas para que se habilite su API Key y poder usarla en la solución actual. Este ejercicio ya se hizo anteriormente en clase, por lo tanto, debe tener una API Key activa. Si no lo hizo en clase, debe hacerlo antes de iniciar esta prueba.
2. Haga uso de **Local Storage** para manejar alguna preferencia de usuario respecto a la apariencia de la página como el color de fondo o el tamaño de letra.
3. Use el concepto de módulos para separar el programa principal Java Script de los scripts de acceso a la base de datos a través de la API. Esto facilitará la comprensión de la estructura del proyecto para usted mismo y para otros programadores.

**Rúbrica de evaluación**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Porcentaje | Impresionante | Bien hecho | Mejorable | Deficiente | Calificación |
|  |  |  |  |
| Diseño y maquetación  10% | El diseño de la aplicación es coherente con el tema y utiliza correctamente reglas de estilo.  5.0 | El diseño de la aplicación es visualmente adecuado, presenta errores en la definición de algunas reglas de estilo.  4.0 | El diseño presentado tiene baja calidad y no se ajusta a los cambios del tamaño de la resolución de la pantalla  3.5 | No presenta un diseño de acuerdo al proceso de entrenamiento.  2.0 |  |
| Estructura y orden del proyecto  20% | El proyecto posee una estructura que facilita la organización de los recursos como imágenes, archivos de JavaScript, importación de librerías etc.  5.0 | El proyecto posee algunas carpetas que permiten organiza los recursos internos.  4.0 | La estructura de carpetas no es adecuada y no favorece a las buenas prácticas del desarrollo de software.  3.0 | No tiene estructura definida y los archivos se encuentran en desorden.  2.0 |  |
| Gestión de Persistencia  20% | Utiliza JSON Server para la gestión de datos del aplicativo web.  5.0 | Gestiona los datos para el aplicativo con otro tipo de persistencia manteniendo el uso del formato JSON  4.0 | Gestiona los datos con otro mecanismo de persistencia sin usar el formato JSON  3.0 | No gestiona persistencia  1.0 |  |
| Uso de las API Fetch para: crear, modificar, listar y eliminar Ciudades  20% | La aplicación realiza las **cuatro** operaciones básicas para los datos de Ciudades  5.0 | La aplicación realiza **tres** de las operaciones básicas solicitadas para los datos de Ciudades  4.5 | La aplicación realiza **sólo dos** de las operaciones básicas solicitadas para los datos de Ciudades  2.0 | La aplicación realiza **una** de las cuatro operaciones básicas para los datos de Ciudades  1.0 |  |
| Uso de las API Fetch para: crear, modificar, listar y eliminar Departamentos  20% | La aplicación realiza las **cuatro** operaciones básicas para los datos de Departamentos  5.0 | La aplicación realiza **tres** de las operaciones básicas solicitadas para los datos Departamentos  4.5 | La aplicación realiza **sólo dos** de las operaciones básicas solicitadas para los datos de Departamentos  2.0 | La aplicación realiza **una** de las cuatro operaciones básicas para los datos de Departamentos  1.0 |  |
| Otros Requerimientos  10% | \*La aplicación muestra el estado del tiempo de una ciudad elegida  \*La aplicación “recuerda” la preferencia de usuario respecto a algún aspecto de apariencia como el color de fondo o el tamaño de letra.  5.0 | \*La aplicación muestra el estado del tiempo de una ciudad elegida  \*La aplicación NO maneja el tema de preferencias de usuario.  4.0 | \*La aplicación NO muestra el estado del tiempo de una ciudad elegida  \*La aplicación “recuerda” la preferencia de usuario respecto a algún aspecto de apariencia como el color de fondo o el tamaño de letra.  3.0 | \*La aplicación NO muestra el estado del tiempo de una ciudad elegida  \*La aplicación NO maneja el tema de preferencias de usuario.  1.0 |  |