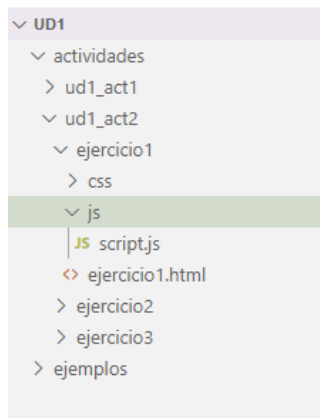


## UD1 ACT3. Interacción con el usuario I: Central de Medidas

Crea (si no la tienes ya) una carpeta que llamarás **ud1**, dentro habrá una llamada **actividades** dentro habrá otra llamada **ud1\_act3**. Dentro de la misma, cada ejercicio irá ubicado en una carpeta que llamarás **ejercicio1**, **ejercicio2**, etc... respectivamente. En los nombres de las carpetas de los ejercicios no uses espacios ni mayúsculas. Dentro de cada ejercicio:

- Los archivos Javascript irán ubicados en la carpeta **js**
- Los estilos, de haberlos, irán ubicados en la carpeta **css**.
- Las imágenes, de haberlas, irán ubicadas en la carpeta **img**.



El código debería entregarse convenientemente formateado y comentado si fuera necesario. Todas las páginas deberían tener el nombre del ejercicio y la actividad en el título.

La entrega de la actividad se realizará vía GitHub, aunque deberá constar como entregada en el Moodle de la clase previamente a la fecha límite de entrega.

## Ejercicio 1: Gestionando la lógica

- Queremos implementar un script en JavaScript que nos permita gestionar temperaturas mensuales [suponemos que el mes tiene 30 días] relativas a varias estaciones meteorológicas situadas en distintas ciudades. Para ello vamos a crear la clase **CentralMedidas** con las siguientes características:

Clase <b>CentralMedidas</b>			
Propiedades			
Nombre	Tipo	Visibilidad	Descripción
<b>Medidas</b>	entero[][]	Privada	Contiene una tabla de enteros correspondientes a las medidas tomadas durante un mes en distintas ciudades. La primera columna almacenará el nombre de la ciudad. Inicialmente el array está vacío.
Métodos			
Nombre	Parámetros	Devuelve	Descripción
<b>insertaMedidas(ciudad, valores)</b>	ciudad:cadena valores:entero[]	Booleano	Crea una nueva fila en la tabla de medidas correspondiente a la ciudad <b>ciudad</b> con los valores del array <b>valores</b> . Devuelve verdadero si se ha podido insertar las medidas y falso si no (si la ciudad ya existe, o el segundo parámetro no contiene 30 valores).
<b>insertaAleatorio(ciudad)</b>	ciudad:cadena	Booleano	Crea una nueva fila en la tabla de medidas correspondiente a la ciudad <b>ciudad</b> con valores aleatorios. Devuelve verdadero si se ha podido insertar las medidas y falso si no (si la ciudad ya existe).
<b>mediaMedidas(ciudad)</b>	ciudad:cadena	Entero	Devuelve la temperatura media de la ciudad indicada durante el mes
<b>mediaMedidasTotal()</b>	-	Entero	Devuelve la temperatura media de todas las ciudades durante el mes
<b>eliminaCiudad(ciudad)</b>	ciudad:cadena	Booleano	Elimina las medidas de la ciudad. Devuelve verdadero si se ha podido eliminar la fila y falso si no (pues la ciudad no existe).
<b>toConsole()</b>	-	-	Se muestran por consola las medidas correspondientes a cada ciudad.

Además, podrán crearse aquellos métodos que se considere necesario para la consecución del ejercicio.

La tabla medidas tendrá 31 columnas (una primera columna para el nombre de la ciudad y las 30 siguientes que contienen las medidas del mes) y una fila por cada ciudad. Por ejemplo, si tenemos 5 ciudades (Santander, Oviedo, Madrid, Valencia y Cádiz) la tabla sería la siguiente:

	0	1	2	3	...	30
0	Santander	5	6	10	...	2
1	Oviedo	4	2	8	...	0
2	Madrid	2	1	-1	...	-5
3	Valencia	8	10	15	...	5
4	Cádiz	10	12	18	...	7

Las medidas pueden tener un valor en el rango comprendido entre -10 y 40.

Deberán implementarse los siguientes archivos:

- **centralmedidas.js**. Definición de la clase **CentralMedidas**.
- **central\_medidas\_prueba.js**. Prueba de la clase con distintos casos de prueba

## Ejercicio 2: Gestionando la interacción con el usuario

Vamos a añadir un método a la clase **CentralMedidas** con las siguientes características:

Clase <b>CentralMedidas</b>																																																																																																		
Métodos																																																																																																		
Nombre	Parámetros	Devuelve	Descripción																																																																																															
<b>toHTML()</b>	-	<b>cadena</b>	Devuelve una cadena que representa una tabla HTML con la misma estructura que la propiedad <b>medidas</b> , añadiendo una columna adicional que representa la media de las medidas del mes para esa fila. Algo así: <div><table><tr><th></th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>Media</th></tr><tr><td>SANTANDER</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td><td>10</td></tr><tr><td>OVIEDO</td><td>40</td><td>11</td><td>28</td><td>27</td><td>17</td><td>20</td><td>40</td><td>8</td><td>4</td><td>31</td><td>14</td><td>34</td><td>5</td><td>25</td><td>31</td><td>25</td><td>-5</td><td>20</td><td>27</td><td>-1</td><td>35</td><td>22</td><td>38</td><td>-8</td><td>18</td><td>-1</td><td>-2</td><td>21</td><td>7</td><td>16</td><td>18.2</td></tr></table></div>		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Media	SANTANDER	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	OVIEDO	40	11	28	27	17	20	40	8	4	31	14	34	5	25	31	25	-5	20	27	-1	35	22	38	-8	18	-1	-2	21	7	16	18.2
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Media																																																																			
SANTANDER	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10																																																																				
OVIEDO	40	11	28	27	17	20	40	8	4	31	14	34	5	25	31	25	-5	20	27	-1	35	22	38	-8	18	-1	-2	21	7	16	18.2																																																																			

Ahora ya estamos listos para incorporar la parte visual. Para ello contamos con el siguiente formulario, definido en **temperaturas.html**, con la hoja de estilos correspondiente.

### Control de temperaturas

Ciudad

Medidas

☒ Manual ☐ Aleatorio

El formulario nos va a permitir añadir nuevas ciudades (campo Ciudad) y medidas (campo Medidas).

- El texto que escribamos en el campo de Ciudad se convertirá a mayúsculas de manera automática.
- En función de los botones de radio se podrán introducir las medidas de dos maneras:
  - Manual. Se introducirán 30 valores, separados entre comas:

## Control de temperaturas

Ciudad

Medidas 

10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,  
10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,  
10, 10, 10, 10

☒ Manual ☐ Aleatorio

- o Aleatorio. Los 30 Valores se generarán de manera automática (con valores en el rango permitido), pasando el campo a ser de sólo lectura:

## Control de temperaturas

Ciudad

Medidas

☐ Manual ☒ Aleatorio

Al pulsar uno de los botones de radio, el valor del campo medidas deberá borrarse.

- Al pulsar en el botón **Guardar**, se deberán realizar las siguientes validaciones:
  - El campo ciudad tiene contenido y no está repetido (no existe en la tabla).
  - El campo de medidas contiene 30 valores separados por comas, y todos los valores son enteros en el rango de temperaturas permitido.

En caso de haber algún error, se mostrará en la capa con identificador **errores** (que debe ocultarse una vez que se han subsanado los errores):

## La ciudad ya existe

Si las validaciones son correctas, se mostrará en la capa con id **tabla-medidas** una tabla con las ciudades que se han insertado hasta el momento, incluyendo la ciudad actual. Por ejemplo, en la siguiente captura se han insertado dos ciudades:

## Control de temperaturas

Ciudad

Medidas

☒ Manual ☐ Aleatorio

Además, la capa con id **temperatura-media** mostrará la temperatura media de todas las ciudades:

Temperatura media: 14.1

- Al pulsar el botón **Borrar** se consultará el campo Ciudad. En caso de no estar vacío y contener un nombre de ciudad existente, se borrará la fila correspondiente, actualizando la tabla y la temperatura media. Si las validaciones no son correctas, se mostrará el error en el campo de error.
- Al cerrar la página, la información introducida en la página debería guardarse en una entrada de `localStorage` automáticamente. Además, al cargar la página, debería comprobarse si existe la entrada en `localStorage` y en caso afirmativo cargar los datos, tanto en nuestro modelo (instancia de `CentralMedidas`) como en la propia vista Web.
- Puedes implementar los siguientes métodos en `CentralMedidas` a modo de ayuda:

Clase <b>CentralMedidas</b>			
Métodos			
Nombre	Parámetros	Devuelve	Descripción
<b>toJSON()</b>	-	<b>cadena</b>	Devuelve una cadena con la representación en formato JSON de la tabla de medidas
<b>fromJSON()</b>	<b>medidas:cadena</b>	-	Recibe una cadena con la representación en formato JSON de la tabla de medidas. Integra dicha tabla en la propiedad privada <b>Medidas</b>

## RESULTADOS DE APRENDIZAJE Y CRITERIOS DE EVALUACIÓN

Esta actividad contribuye a la consecución del RA1. *Genera interfaces gráficas de usuario mediante editores visuales utilizando las funcionalidades del editor y adaptando el código generado.*

Los criterios de evaluación que permiten valorar la consecución del resultado de aprendizaje son los siguientes:

*1a) Se han analizado las herramientas y librerías disponibles para la generación de interfaces gráficos.*

*1b) Se ha creado un interfaz gráfico utilizando las herramientas de un editor visual.*

*1f) Se ha modificado el código generado por el editor visual.*

En el Moodle de la asignatura se detalla la rúbrica de corrección para la actividad.