

#### ❑ **ACTIVITY 1 – Mostrar un usuario aleatorio**

- La página <https://randomuser.me/> permite obtener datos aleatorios de personas pensando en que los desarrolladores y otros profesionales puedan utilizarlos en sus pruebas y test.
- Las instrucciones de la API de ese servicio gratuito están en la URL: <https://randomuser.me/documentation>
- En todo caso la idea es hacer peticiones via GET a la URL: <https://randomuser.me/api/>
- Se pueden pasar parámetros para indicar cuántos usuarios aleatorios deseamos, el sexo, política de contraseñas, páginas, formato de respuesta, etc..
- En la página de documentación viene un ejemplo de la estructura JSON de las respuestas. Como resumen indicamos que es un objeto formado por 2 propiedades: **results** e **info**.

La primera es un array donde cada elemento lo forma un objeto con los datos del usuario aleatorio. La propiedad **info** contiene otros detalles entre los que destaca una semilla que permite repetir una petición con los mismo datos y datos de paginación (cuando deseamos dividir en páginas los resultados se debe usar la misma semilla)

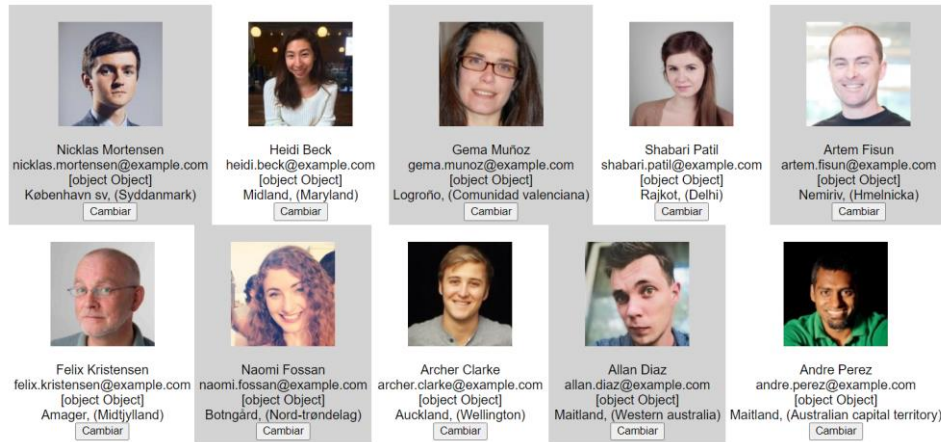
- La aplicación mostrará la foto, nombre, apellido, email, dirección y estado al que pertenece el usuario. Cada vez que actualicemos la página, se pedirá otro usuario. Ejemplo de resultado:



Nancy Melgar  
Email: nancy.melgar@example.com  
[object Object]  
Yurécuaro (TLAXCALA)

#### ❑ **ACTIVITY 2 – Mostrar 10 usuarios que cambian**

- Usaremos el mismo servicio de usuarios aleatorios para esta práctica
- Si pasamos un parámetro llamado **results** con valor **10**, conseguiremos obtener datos de 10 usuarios aleatorios
- Mostraremos esos datos en 2 filas. Distinguiremos el fondo de los pares e impares y, además, en cada usuario colocaremos un botón con el texto “**Cambiar**”
- Cuando hagamos clic en el botón “**Cambiar**”, el usuario en el que está situado el botón, se cambiará
- Mostraremos el mensaje **Cargando...** al principio, mientras llegan los datos de los 10 usuarios (cuando lleguen, se quita el mensaje). También cuando pulsemos **Cambiar** en el usuario a modificar, escribiremos **Esperando usuario nuevo...** en la celda de ese usuario (habremos, antes, quitado los datos del usuario) hasta que lleguen los nuevos datos.



### ❑ ACTIVITY 3 – Mapa meteorológico de la Aemet

- La agencia meteorológica española **AEMET** dispone de una API para datos abiertos en la que proporciona numerosas posibilidades de obtener información meteorológica.
- Para poder obtener los datos de esta agencia, necesitamos darnos de alta como desarrolladores en la dirección: <https://opendata.aemet.es/centrodedescargas/altaUsuario>
- Las instrucciones de funcionamiento de la API de AEMET están en: <https://opendata.aemet.es/dist/index.html?>
- En el alta necesitamos indicar un email que tendremos que confirmar para que se nos conceda una API Key, una clave única que se otorga a cada desarrollador. Esa API la deberemos enviar cada vez que hagamos peticiones a la AEMET
- Nuestra aplicación, inicialmente, muestra un cuadro de texto en el que introducir nuestra API y un botón de cargar mapa

## Imagen del día de la AEMET

Pega tu API Key

Cargar mapa

- Tras pegar nuestra API y pulsar el botón **Cargar mapa**, solicitaremos el mapa meteorológico del día en la dirección: <https://opendata.aemet.es/opendata/api/mapasygraficos/analisis>
- Se quitarán los controles de formulario y se mostrará (en horizontal) el mapa
- Hay que recordar que debemos enviar vía GET un parámetro llamado apikey, con nuestra clave (la cual se obtiene del formulario)

- Los datos que llegan, nos ofrecen un objeto JSON donde la propiedad llamada **datos**, es una URL a la imagen del mapa. La petición http nos proporciona ese enlace, pero no el mapa en sí.
- Por ello, hay una segunda petición (a esa URL devuelta por la propiedad datos) que es la que nos proporciona el mapa.

## ACTIVIDADES A ENTREGAR

### ❑ ACTIVITY 4 – Cambio de moneda

- La URL <https://exchangeratesapi.io/> proporciona una API para obtener cambios de moneda. Está documentado en la propia página. En la página aparece un ejemplo de devolución de datos:

```
{
  "success": true,
  "timestamp": 1519296206,
  "base": "USD",
  "date": "2021-03-17",
  "rates": {
    "GBP": 0.72007,
    "JPY": 107.346001,
    "EUR": 0.813399,
  }
}
```

- La propiedad **base** hace referencia a la moneda base del cambio, en el ejemplo anterior se toma 1 dólar como base. Así, por ejemplo podremos obtener el cambio de 1 dólar = 0.813399 euros

## Conversión de monedas

Moneda base: EUR ▼

Convertir a..

Convertir

Resultado: 3,6226

AUD  
BGN  
BRL  
CAD  
CHF  
CNY  
CZK  
DKK  
EUR  
GBP  
HKD  
HRK  
HUF  
IDR  
INR  
ILS  
ISK  
KRW  
MXN  
JPY

- Para obtener la última fecha de cambio, se usa <https://api.exchangeratesapi.io/latest>
- Podemos enviar el parámetro **symbols** para que nos devuelva solo los cambios a monedas concretas. Por ejemplo: <https://api.exchangeratesapi.io/latest?symbols=USD.GBP> nos consigue los cambios de Euro a Dólares EEUU y a Libras Esterlinas.

- También podemos cambiar la base de la moneda con el parámetro **base** al que podemos indicar una moneda diferente al Euro.
- Gracias a esta API, creamos el formulario de la Figura. Éste, deberá permitir realizar cualquier cambio de moneda. Para rellenar los cuadros de lista, necesitaremos hacer una primera petición y mostrar todas las monedas.

#### ❑ **ACTIVITY 5 – Imágenes desde la nube**

- El servicio *picsum* permite utilizar imágenes libremente. Posee una API al respecto. En este ejercicio podemos usar esta sintaxis: <https://picsum.photos/300/300?imagen=1>
- Este enlace obtiene la imagen número 1 a una resolución de 300 x 300 píxeles.
- Si añadimos el parámetro blur (no hace falta indicar valor en él), tendremos:  
<https://picsum.photos/300/300?image=!?blur>

La imagen sale desenfocada

- Sabiendo esto, genera una página que muestre 200 imágenes a 300 por 300 de resolución.
- Solo deben caber imágenes 4 en horizontal. Si la pantalla es pequeña, entonces, las imágenes se muestran más verticales. No debe hacer scroll horizontal. Las imágenes deben quedar centradas en la ventana.
- Si se hace clic en una imagen, esta se desenfoca
- Además aparece desenfocada durante una semana
- Si se hace clic en otra, entonces esa será la desenfocada la semana siguiente
- Si se hace clic en la que ya está desenfocada, entonces quedará sin desenfocar
- Usa las peticiones AJAX para enfocar y desenfocar.

Lista de fotos

Haz clic en la que te gusta más



## ACTIVIDAD OPCIONAL

#### ❑ **ACTIVITY 6 – Animación vistosa**

- Basándote en la API de **randomuser.me**, consigue crear una página web que muestre 50 caras aleatorias consiguiendo que cambien 100 veces, de forma aleatoria, cada 2 décimas de segundo

- Es una animación muy llamativa cuyo efecto se basa en que no haya un exceso de retraso al conseguir las fotos. Lo ideal es cargar las fotos por adelantado.
- Lo lógico es hacer una petición a esa API que devuelva muchas personas (por ejemplo 1000) y almacenarlas en un array. Luego el array se puede desordenar de forma aleatoria cada 2 o 3 décimas.
- El control de las décimas lo llevan los intervalos

