

Universidad Tecnológica Nacional
Facultad Regional Avellaneda



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio III

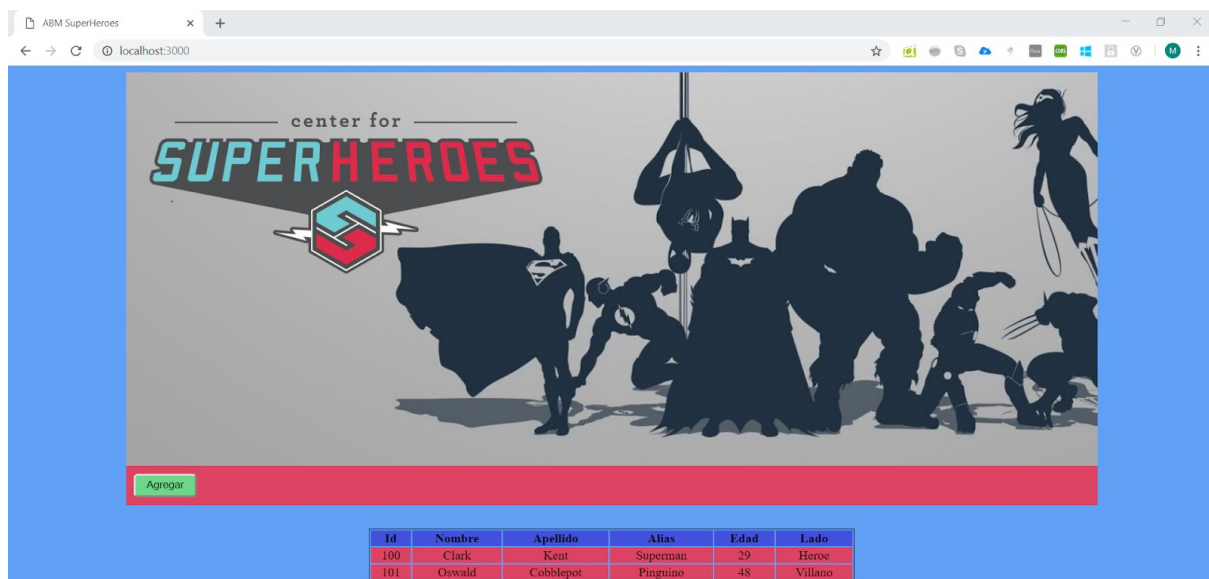
Apellido:		Fecha:	27/11/2018
Nombre:		Docente ⁽²⁾ :	Baus/Mutti
División:		Nota ⁽²⁾ :	
Legajo:		Firma ⁽²⁾ :	
Instancia ⁽¹⁾ :	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;">PP</div> <div style="border: 1px solid black; width: 20px; height: 20px;"></div> <div style="text-align: center;">RPP</div> <div style="text-align: center;">X</div> <div style="text-align: center;">SP</div> <div style="border: 1px solid black; width: 20px; height: 20px;"></div> <div style="text-align: center;">RSP</div> <div style="border: 1px solid black; width: 20px; height: 20px;"></div> <div style="text-align: center;">FIN</div> <div style="border: 1px solid black; width: 20px; height: 20px;"></div> </div>		

(1) Las instancias validas son: 1^{er} Parcial (**PP**), Recuperatorio 1^{er} Parcial (**RPP**), 2^{do} Parcial (**SP**), Recuperatorio 2^{do} Parcial (**RSP**), Final (**FIN**) . Marque con una cruz.

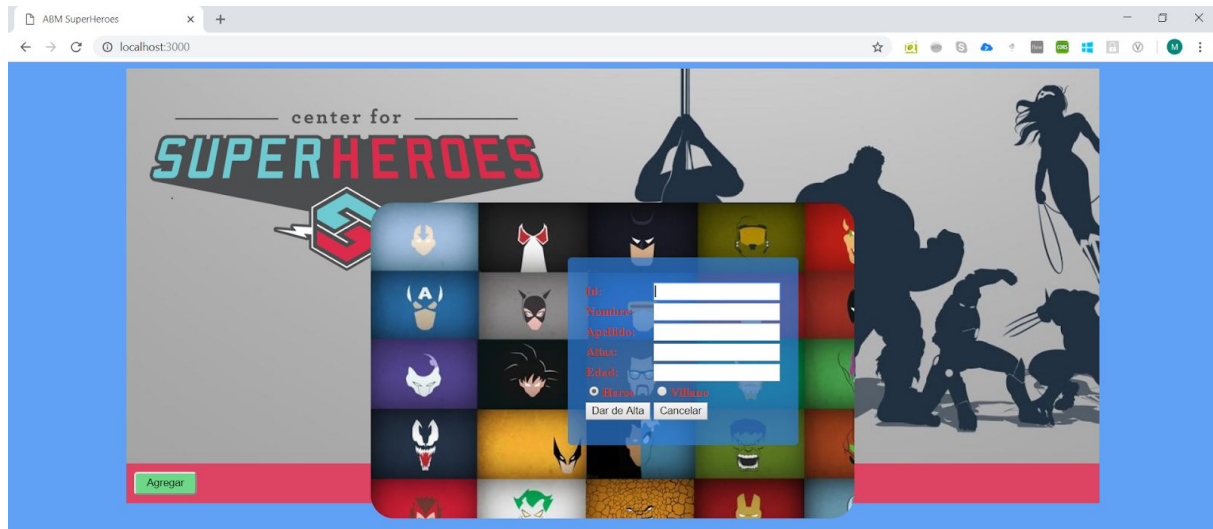
(2) Campos a ser completados por el docente.

Realizar una ABM de Super Heroes y Villanos.

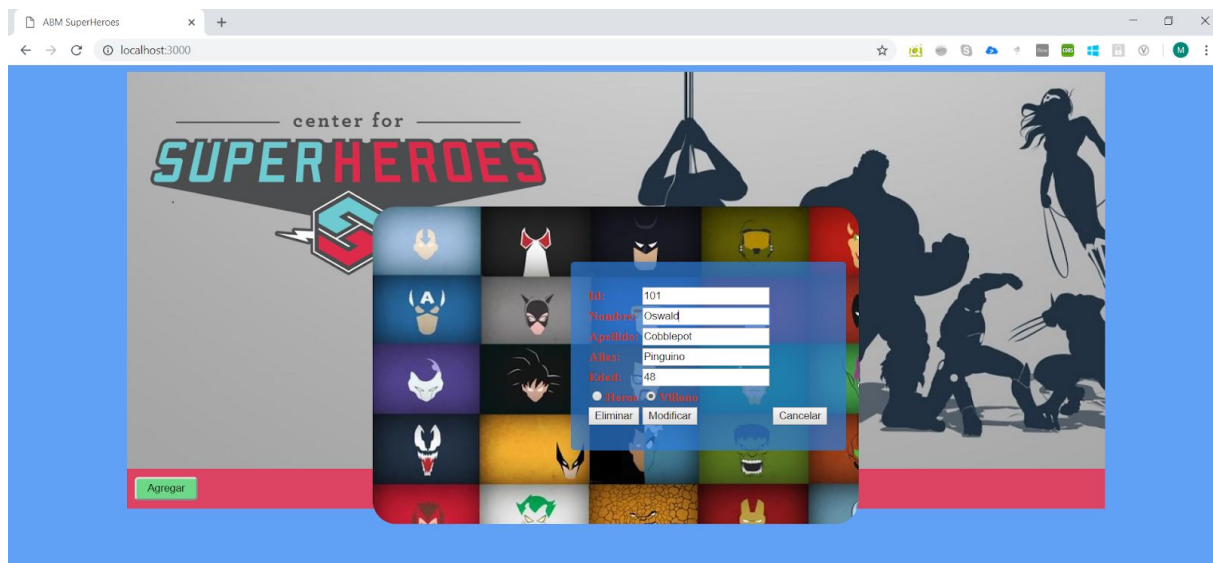
- Desarrollar una aplicación **en capas**, que cuente con una única página donde exista el listado de los héroes y villanos dados de alta, así como un botón para dar de alta a nuevos personajes. Para esto deberá usar HTML y CSS para darle agradable “look and feel”.



- El botón de alta abrirá un formulario(utilizando alguna animación) donde se podrán ingresar los datos del nuevo personaje. Asimismo, el formulario contendrá botones para aceptar y cancelar. Mientras se espera la respuesta del servidor, mostrar un spinner. Todos los campos del formulario deberán tener al menos una validación.



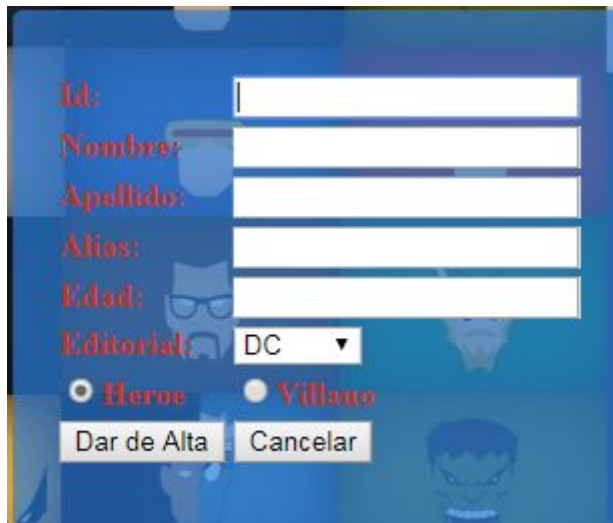
- La lista de personajes deberá contar con algún manejador de eventos, tal que al hacer click en algún elemento de la lista, se muestre un nuevo formulario que nos permita eliminar o modificar el personaje.



- Utilizar JQuery en al menos un método de AJAX.
- Utilizar JS nativo(XMLHttpRequest) en al menos un método de AJAX.
- El servidor será provisto en un archivo llamado server.js y no deberá ser modificado. Para correrlo, bastará con correr por línea de comando: "node server". El servidor quedará escuchando en el puerto 3000.
- El navegador por default va a devolvernos **index.html** al apuntarle al puerto 3000 de localhost.

RECUPERATORIO:

- Agregar un campo al formulario que permita ingresar la editorial a la que pertenece el super heroe.
Los valores pueden ser: "DC", "Marvel", "Otro". Este último se usará por defecto para los personajes dados de alta con anterioridad.



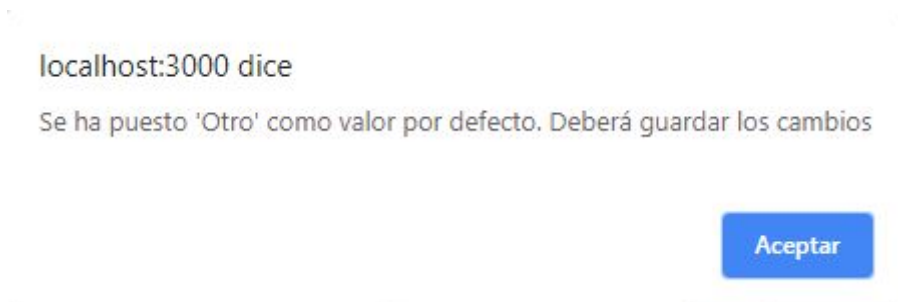
Formulario de creación de personaje con los siguientes campos:

- Id:** Campo de texto vacío.
- Nombre:** Campo de texto vacío.
- Apellido:** Campo de texto vacío.
- Alias:** Campo de texto vacío.
- Edad:** Campo de texto vacío.
- Editorial:** Selector de lista desplegable con "DC" seleccionado.
- Lado:** Radio buttons para "Heroe" (seleccionado) y "Villano".
- Botones:** "Dar de Alta" y "Cancelar".

- Para los personajes agregados anteriormente, mostrar como "No definido" en la tabla

Id	Nombre	Apellido	Alias	Edad	Lado	Editorial
100	Clark	Kent	Superman	29	Heroe	No definido
101	Oswald	Cobblepot	Pinguino	48	Villano	No definido

- Al hacer click en uno de los héroes que no tienen Editorial definido, mostrar un alerta:

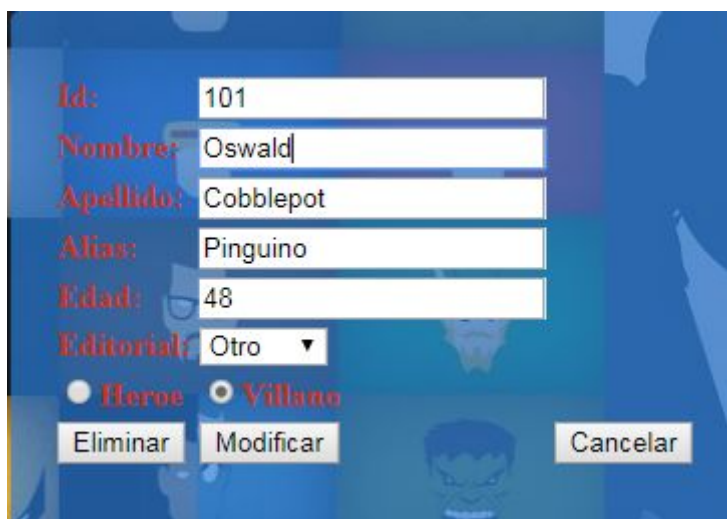


Alerta de confirmación con el siguiente mensaje:

localhost:3000 dice

Se ha puesto 'Otro' como valor por defecto. Deberá guardar los cambios

Aceptar



Formulario de edición de personaje con los siguientes campos:

- Id:** Campo de texto con "101".
- Nombre:** Campo de texto con "Oswald".
- Apellido:** Campo de texto con "Cobblepot".
- Alias:** Campo de texto con "Pinguino".
- Edad:** Campo de texto con "48".
- Editorial:** Selector de lista desplegable con "Otro" seleccionado.
- Lado:** Radio buttons para "Heroe" y "Villano" (seleccionado).
- Botones:** "Eliminar", "Modificar" y "Cancelar".

SERVER:

Contrato con el servidor:

1. ALTA:

request(POST):

- url: /agregar

```
var data = {  
  "collection": "heroes",  
  "hero": hero  
}
```

- **héroe** es un objeto creado en base a los datos del formulario.
- Se debe usar la función de JSON que convierte a string los objetos al momento de realizar la llamada AJAX.
- Agregar el header "Content-Type":
 - **JS**: xhr.setRequestHeader("Content-Type", "application/json");
 - **JQuery**: contentType: 'application/json'
-

response:

```
var response = {  
  message: "Alta exitosa",  
}
```

2. BAJA:

request(POST):

- url: /eliminar

```
var data = {  
  "collection": "heroes",  
  "id": hero.id  
}
```

- **héroe** es un objeto creado en base a los datos del formulario. Para el borrado me alcanza con mandar el id.
- Se debe usar la función de JSON que convierte a string los objetos al momento de realizar la llamada AJAX.
- Agregar el header "Content-Type":
 - **JS**: xhr.setRequestHeader("Content-Type", "application/json");
 - **JQuery**: contentType: 'application/json'

-

response:

```
var response = {  
  message: "Baja exitosa"  
}
```

3. MODIFICACIÓN

request(POST):

- url: /modificar

```
var data = {  
  "collection": "heroes",  
  "hero": hero  
}
```

- **héroe** es un objeto creado en base a los datos del formulario.
- Se debe usar la función de JSON que convierte a string los objetos al momento de realizar la llamada AJAX.
- Agregar el header "Content-Type":
 - **JS**: xhr.setRequestHeader("Content-Type", "application/json");
 - **jQuery**: contentType: 'application/json'

-

response:

```
var response = {  
  message: "Modificacion exitosa",  
}
```

4. Traer

request(GET):

- url: /traer
- Pasar por url el atributo "collection" con el valor "heroes".

response:

```
var response = {  
  message: "Carga exitosa",  
  "data": array  
}
```

