

Universidad Tecnológica Nacional

Facultad Regional Avellaneda



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio III

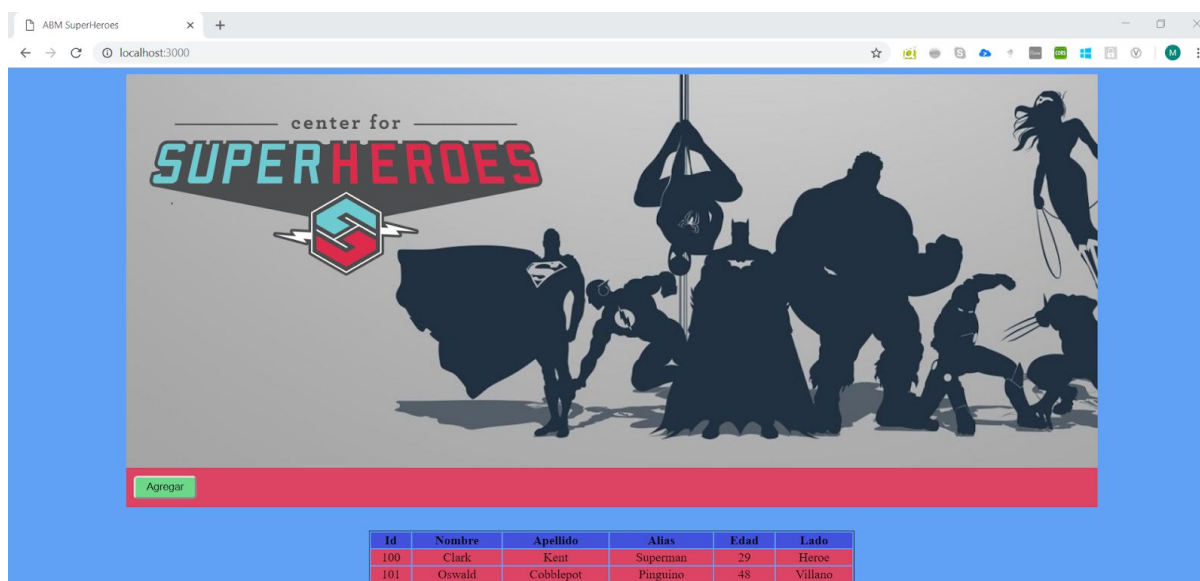
Apellido:		Fecha:	23/10/2018
Nombre:		Docente ⁽²⁾ :	Baus/Mutti
División:		Nota ⁽²⁾ :	
Legajo:		Firma ⁽²⁾ :	
Instancia ⁽¹⁾ :	<div style="display: flex; justify-content: space-around;"> <div>PP</div> <div>X</div> <div>RPP</div> <div></div> <div>SP</div> <div></div> <div>RSP</div> <div></div> <div>FIN</div> <div></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.

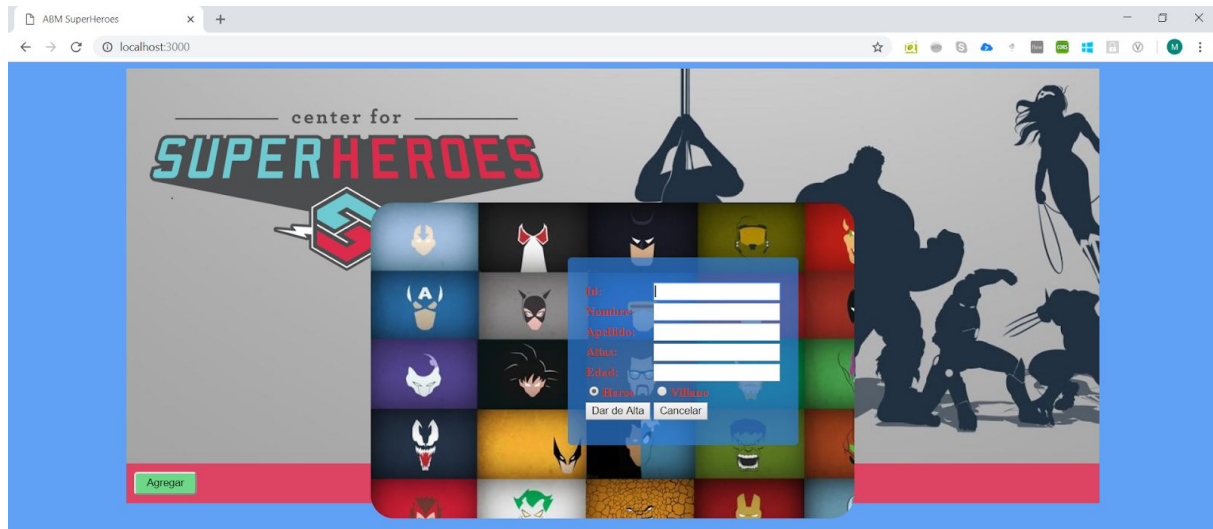
(2) Campos a ser completados por el docente.

Realizar una ABM de Super Heroes y Villanos.

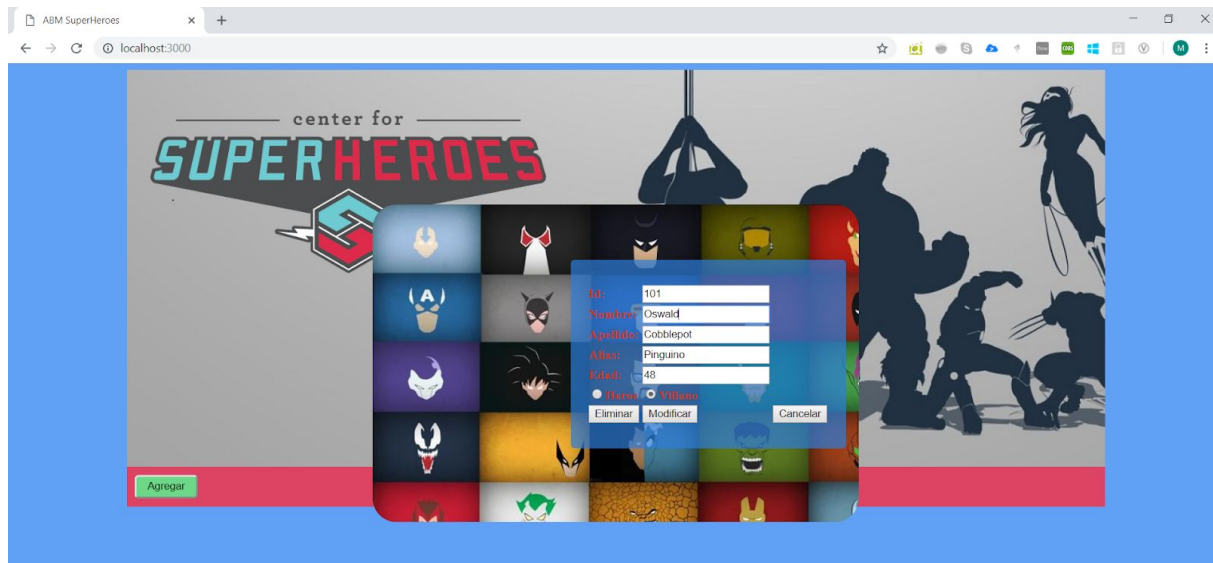
- Desarrollar una aplicación **en capas**, que cuente con una única página donde exista el listado de los héroes y villanos dados de alta, así como un botón para dar de alta a nuevos personajes. Para esto deberá usar HTML y CSS para darle agradable “look and feel”.



- 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.

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  
}
```

