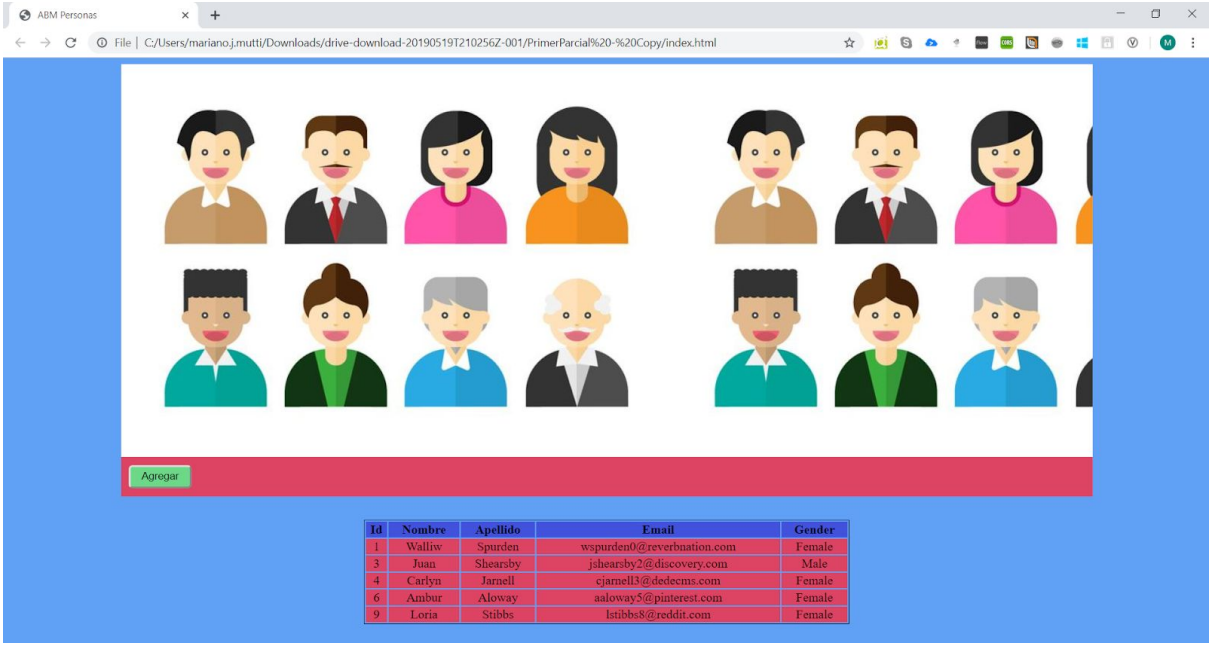
Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio III										
Apellido:				Fecha:	21/05/2019					
Nombre:				Docente(2):	Baus/Mutti					
División:				Nota(2):						
Legajo:				Firma(2):						
Instancia(1):	PP	X	RPP		SP		RSP		FIN	

(1) Las instancias validas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do Parcial (**RSP**), Final (**FIN**) . Marque con una cruz.
(2) Campos a ser completados por el docente.

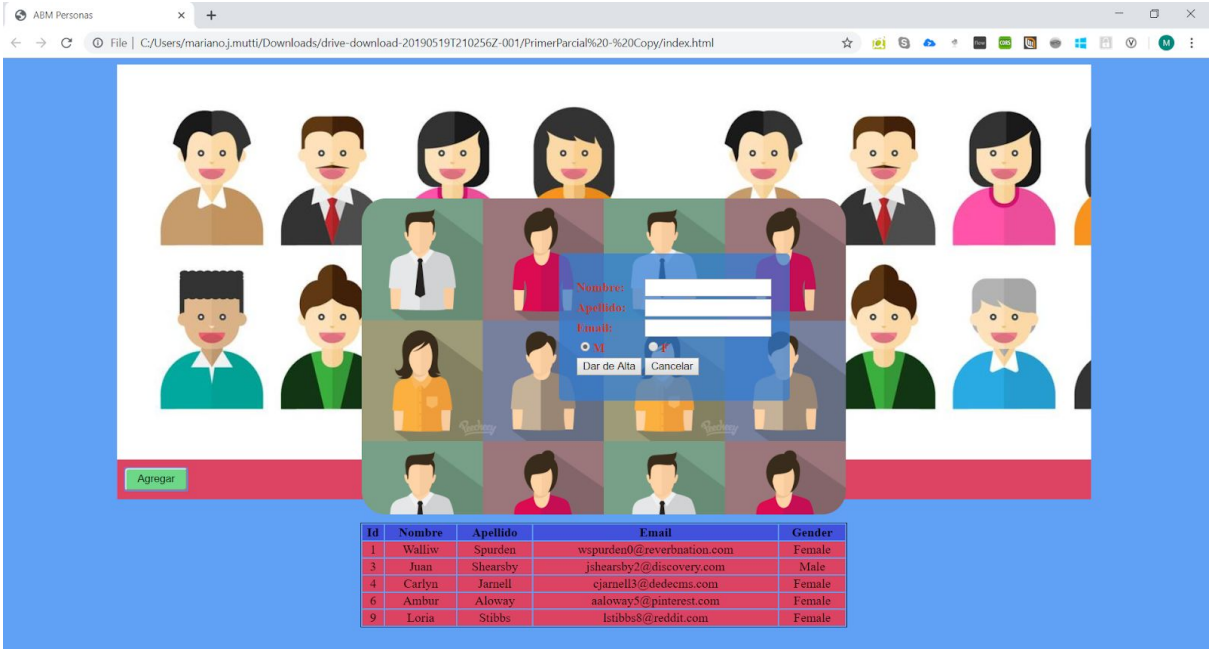
Realizar una ABM de Personas.

- Desarrollar una aplicación que cuente con una única página donde exista el listado de las personas dadas de alta(mostrar un spinner mientras cargan), así como un botón para dar de alta a nuevas personas. Para esto deberá usar HTML y CSS para darle agradable “look and feel”.

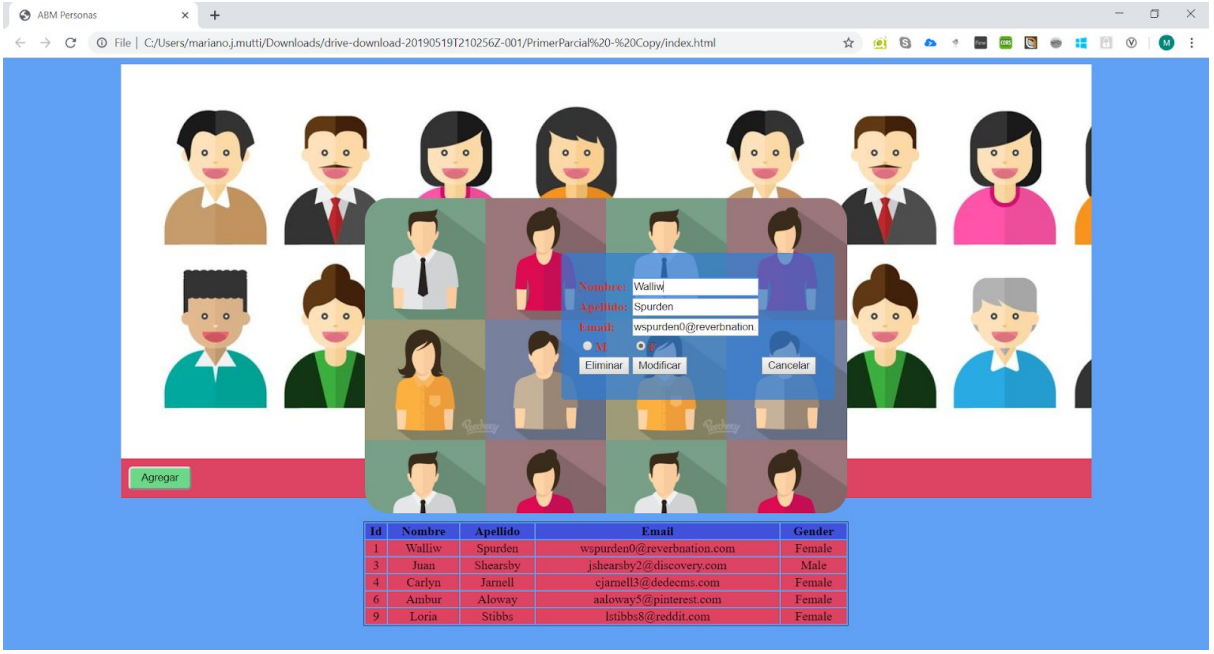


- El botón de alta abrirá un formulario donde se podrán ingresar los datos de la nueva persona. Asimismo, el formulario contendrá botones para aceptar y cancelar.

- El formulario deberá contar con un control apropiado para ingresar emails. También deberá contar con una validación para hacer los datos requeridos.



- 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 a la persona seleccionada.



- 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 body = {
    'collection':'personas',
    'objeto' : persona
}
```

   - ***obejto*** 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 body = {
    'collection':'personas',
    'id' : id
}
```

   - Para el borrado me alcanza con mandar el id de la persona.

- 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 body = {
    'collection':'personas',
    'objeto' : persona
}
```

- ***persona*** 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 “personas”.

response:

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