

# Trabajo Práctico Especial I

Diseño e Implementación de una Aplicación Web para el  
Manejo de una Casa Inteligente.

Interacción Hombre-Computadora  
Grupo 7

Francisco Delgado 57101

Lucas Emery 57341

Felipe Gorostiaga 57200

Leandro Reina 54011

## Indice

<b>Indice</b>	<b>2</b>
<b>Introducción</b>	<b>3</b>
<b>Lineamientos Generales / Decisiones de Diseño</b>	<b>3</b>
<b>Actividades</b>	<b>4</b>
<b>Instalación</b>	<b>7</b>

## Introducción

La idea principal de nuestra aplicación web consiste en proveer una interfaz intuitiva para el usuario, con manejo fácil, instantáneo y automatizado de los distintos dispositivos en una casa inteligente.

Esto se logra mediante un sistema que permite realizar control directo sobre cada dispositivo específico o mediante el uso de funciones automatizadas. Estas consisten en un conjunto de uno o más dispositivos asociados donde cada uno de estos tiene acciones a realizarse. Ejecutando una de estas funciones, se llevan a cabo todas las acciones indicadas en los dispositivos.

En el presente informe comenzaremos detallando los lineamientos generales y decisiones de diseño utilizadas durante el proceso de desarrollo del programa. Luego, comentaremos sobre los elementos de navegación que se presentan durante la utilización de la aplicación, junto con capturas de pantalla para ayudar a ilustrar el flujo del programa.

Se mostrarán detalles específicos de acerca de la implementación del programa el cual fue llevada a cabo con React (una librería de JavaScript creada por Facebook) y se explicara las decisiones tomadas respecto al diseño con sus diferencias y similitudes con los prototipos presentados en el primer informe.

Finalmente, explicaremos como instalar la aplicación en un servidor web junto con todas sus dependencias.

## Lineamientos Generales y Decisiones de Diseño

Con el fin de crear una aplicación visualmente agradable, funcional y user-friendly adoptamos todo el conocimiento teórico adquirido en la materia.

La estrategia para lograr este cometido fue crear un diseño minimalista, con poca información pero concisa e intuitiva, usando iconos grandes y que expresen su uso correctamente en el modelo mental del usuario.

Se apuntó a simplificar donde sea posible, usando nombres cortos, concisos, e intuitivos, manteniendo similitudes con otros programas, como el uso de sliders y botones de manera similar, y logar diseño homogéneo en toda la aplicación; todo esto teniendo siempre en cuenta los usuarios representativos establecidos en la anterior entrega de este trabajo. Tratando de cumplir con las expectativas tanto de una persona joven con alta comodidad en el manejo de tecnología como las de una persona mayor con dificultades a la hora de manejar tecnología y espera que la aplicación sea fácil de usar, brindando una mejora a su vida diaria y no más complicaciones.

El uso de colores fue fundamental a la hora de pensar el diseño de los componentes de la página. La gama de colores principal de la aplicación es blanco-gris-azul-negro, esto se replicó en todas las pestañas para mantener el estilo y se eligieron estos colores particulares porque se acoplan al diseño minimalista que queríamos, siendo visualmente agradable y sin desviar la atención del usuario como sería el caso con colores brillantes.

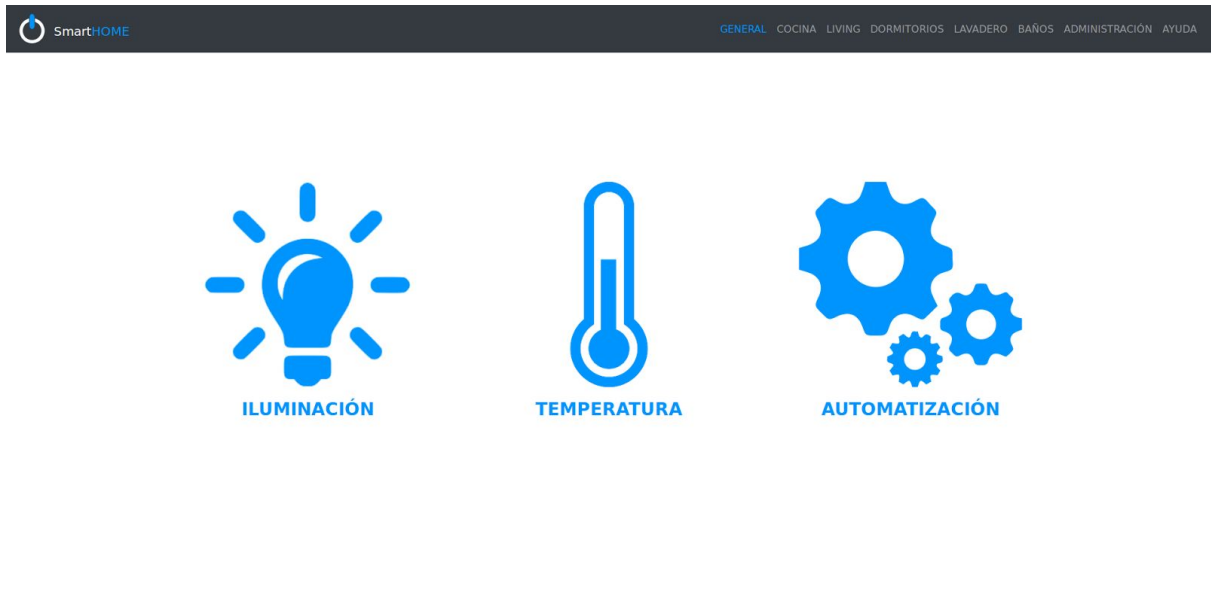
Como veremos a continuación en las capturas de pantalla, para las advertencias se usaron 3 colores: amarillo para indicar que faltan completar campos o no se indicó una opción y rojo para indicar que se presentó un error a la hora de realizar la acción y verde para indicar que se pudo realizar con éxito.

Cabe destacar que durante la implementación de la aplicación web nos encontramos con ciertos impasses, tanto de limitaciones del back-end como de complejidad de implementar en el front-end que nos llevaron a ajustar ciertos cambios respecto a los prototipos presentados.

## Diseño y Usabilidad

Como se fue mencionado anteriormente, la aplicación se diseñó con el fin de mejorar la fluidez de navegación dentro de la misma. En esta sección veremos cómo se desenvuelve este diseño al realizar distintas tareas que los usuarios utilizarían y comparando con los prototipos de diseño propuestos en el primer informe.

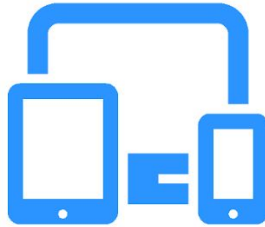
### Pagina Principal



Vista general - Prototipo



ADMINISTRAR AMBIENTES



ADMINISTRAR DISPOSITIVOS



AUTOMATIZACIÓN

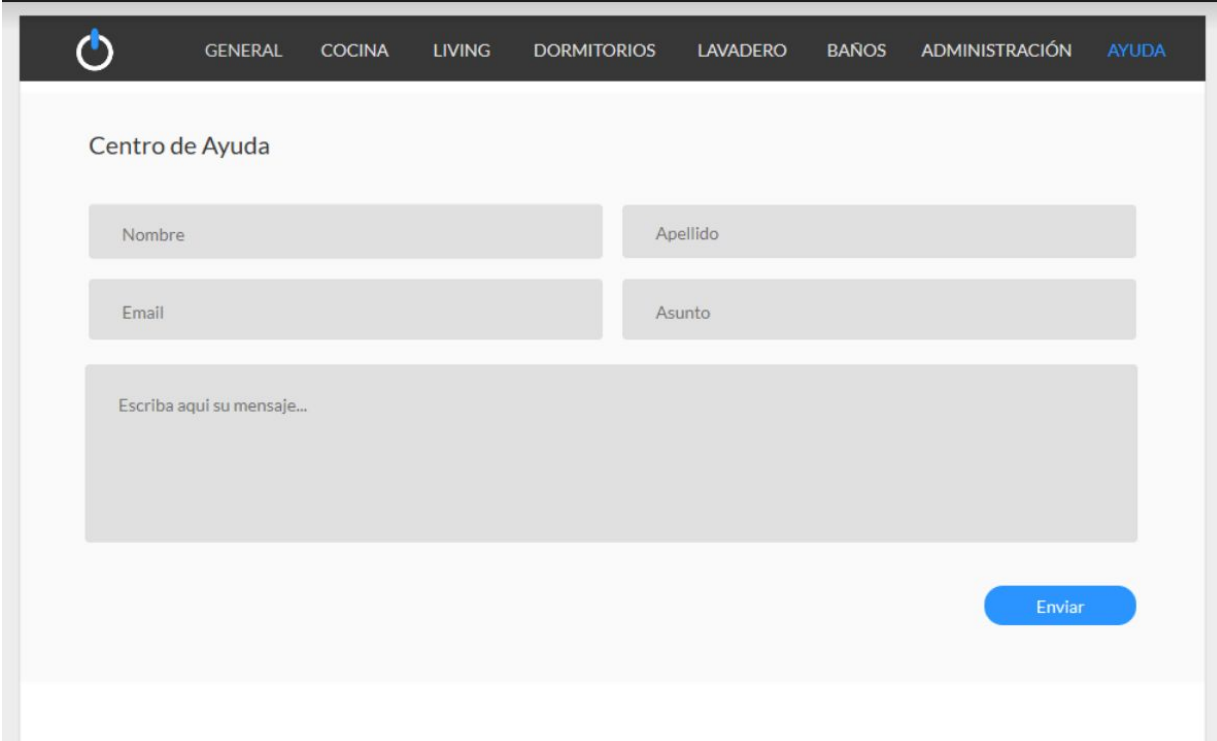
### Vista General - Aplicacion Web

Como se puede observar en las capturas de pantalla, la pagina principal fue cambiada notablemente. La API provista por la cátedra incluye la funcionalidad de crear ambientes, por esto nos vimos obligados a modificar la barra de navegación, haciendo que se vayan agregando los ambientes a medida que el usuario cree nuevos. Las pestañas “General” y “Ayuda” se mantuvieron presentes, con pequeños cambios respecto al estilo.

En la página principal también hubo grandes cambios, decidimos quitar Iluminación y Temperatura, ya que nos pareció más simple que los usuarios manejen sus dispositivos de iluminación y temperatura dependiendo en la habitación que se encuentren. Se agregaron los iconos “Administrar Ambientes” y “Administrar Dispositivos” a esta ventana con iconos que describen sus funcionalidades.

Aunque hubo un cambio notable respecto de la funcionalidad, visualmente se mantuvo un diseño casi idéntico, usando iconos grandes y representativos para brindar una cálida bienvenida al usuario y fácil control de la aplicación.

## Ventana de Ayuda



Centro de Ayuda

Nombre

Apellido

Email

Asunto

Escriba aquí su mensaje...

Enviar

Centro de Ayuda - Prototipo

## Centro de Ayuda

Debes ingresar todos los datos que se piden!

Nombre:

Felipe

Apellido

Gorostiaga

Email

name@example.com

Asunto

Falla en automatizacion

Comentario

Escriba aqui su mensaje...

Enviar

### Centro de Ayuda - Aplicacion

Aquí podemos visualizar el contenido de la pestaña “Ayuda” . Como se puede notar este formulario mantiene un diseño muy simple y minimalista. Es muy similar al prototipo de la misma con ligeros cambios de estilo que aplicamos para una mejor interfaz visual. Como se puede notar, es necesario ingresar datos en todos los campos y también un email válido, sino se presenta una alerta que lo indica.

Su consulta ha sido enviada con éxito!

Nos comunicaremos con usted a la brevedad para solucionar su problema. Se le enviara un mail a la casilla ingresada.

Gracias por confiar en nosotros!







Aceptar

En caso de que todos los datos hayan sido completados correctamente y el usuario hace click en el botón “Enviar” , se pasa a la pestaña mostrada arriba. Se utilizó el color verde para mostrar que la acción fue llevada a cabo con éxito ya que acompaña dicha idea con el modelo mental que presentan la gran mayoría de los usuarios.



## Ventana Automatización

### Funciones Personalizadas

Test			
Test1			
HCI Rules			
soloColor			
Lampara y Horno			

Crear Función

### Crear Función

×

Nombre

Ambiente

Todas

Acciones

Seleccione

Seleccione

+

Crear

Cancelar

Crear Funcion - Aplicación

Al hacer clic en el icono de Automatización, se presenta la pestaña en el primer screenshot de arriba. Como podemos observar se presentan las funciones personalizadas creadas por el usuario, en formato de lista, con un diseño simple y fácil de comprender. En el costado derecho de cada Componente de la lista incluimos tres iconos muy característicos acompañados con colores representativos, para acceder a las funcionalidades de correr la función, modificarla o eliminarla. La combinación de los iconos y el color presentan una funcionalidad implícita sin tener que agregar texto y “ensuciar” el diseño minimalista.

Haciendo click en el botón Crear Función se abre una pestaña para completar los campos de la nueva función. El diseño de la misma fue pensado para continuar con el mismo estilo simple y intuitivo. Cuenta con indicaciones y alertas con colores apropiados para cada tipo de notificación.



**La función se ejecutó correctamente**

Entendido

Aquí podemos ver un ejemplo de alerta que se abre cuando se corre una función personalizada correctamente. Esto es muy útil para el usuario ya que no deja dudas sobre si hubo errores o no, caso en el que se muestra la alerta correspondiente.

## Automatización

### Rutinas

- ▼ Cocina
  - Café 6am
  - Luces
  - Preparar horno 9pm
- ▼ Living
  - TV 10pm
  - Luces
- ▼ Dormitorio 1
  - Prender aire acondicionado
  - Prender PC 6pm



### Funciones Personalizadas

- ▼ Lavadero
  - Lavado chuavechito
  - Secado rapido
- ▼ Living
  - Modo cine
  - Modo gaming
  - Modo partido
  - Apagar todo
- ▼ Cocina
  - Desayuno juanita
  - Desayuno 1 pepe
  - Desayuno 2 pepe
  - Preparar para cocinar



Automatización - Prototipo

Observando el prototipo presentado en el informe podemos ver que hubieron grandes cambios en la pestaña “Automatización”, por cuestiones de que no se encontraban en la API propuesta por la cátedra quitamos la funcionalidad de rutinas. Las funciones personalizadas ya no se muestran por ambiente en esta pestaña sino que se muestran todas y dentro de cada ambiente también se pueden encontrar. Esto fue diseñado de esta manera por simpleza de implementación y por el hecho de que no presentaba grandes beneficios al usuario y hasta podría llegar a complicarlo al tenerlo en dos lugares distintos. Cabe destacar que los iconos del prototipo no tienen colores distintos como en la aplicación, lo cual fue cambiado para brindar una user-interface más intuitiva.

## Ventana de Ambiente

Creando un ambiente nuevo, se agregara a la barra de navegación dinámicamente y haciendo clic en esta llevará al usuario a la siguiente ventana.

### Dispositivos

lampara de lava	Modificar ⚙
jetsons oven	Modificar ⚙

### Funciones Personalizadas

soloColor	▶ ⚙ 🗑
-----------	-------

### Ambiente - Aplicación web

#### Dispositivos

<input checked="" type="radio"/> Horno	⚙ <input type="checkbox"/>
<input checked="" type="radio"/> Lavavajilla	⚙ <input type="checkbox"/>
<input checked="" type="radio"/> Heladera	⚙ <input checked="" type="checkbox"/>
<input checked="" type="radio"/> Tostadora	⚙ <input checked="" type="checkbox"/>

#### Funciones Personalizadas

Desayuno juancito	▶
Tostadas y cafe	▶
Pepe merienda	▶

#### Rutinas

Cafe 6am	<input checked="" type="checkbox"/>
Luces	<input type="checkbox"/>
Preparar horno 9pm	<input checked="" type="checkbox"/>

### Ambiente - Prototipo


Observando ambas capturas se puede notar los cambios en estilo que realizamos como grupo respecto del prototipo. Se agregó más color para representar las acciones y hacerlas más descriptivas para el usuario y no se implementó el uso de rutinas. El diseño es más simple y visualmente más agradable sin tantos iconos en la pantalla y mejor distribuidos.

Haciendo click en el botón Modificar de algún dispositivo se abre la ventana de modificación correspondiente al dispositivo (en este caso hacemos clic en la lámpara de lava).


## lámpara de lava

---

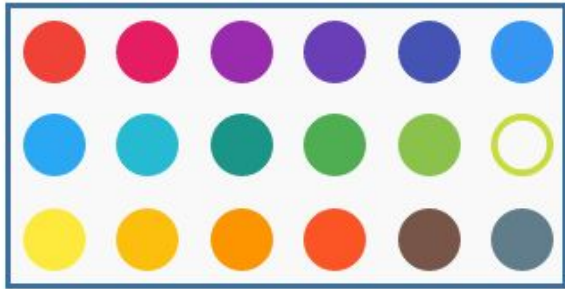
Intensidad:



Activar:



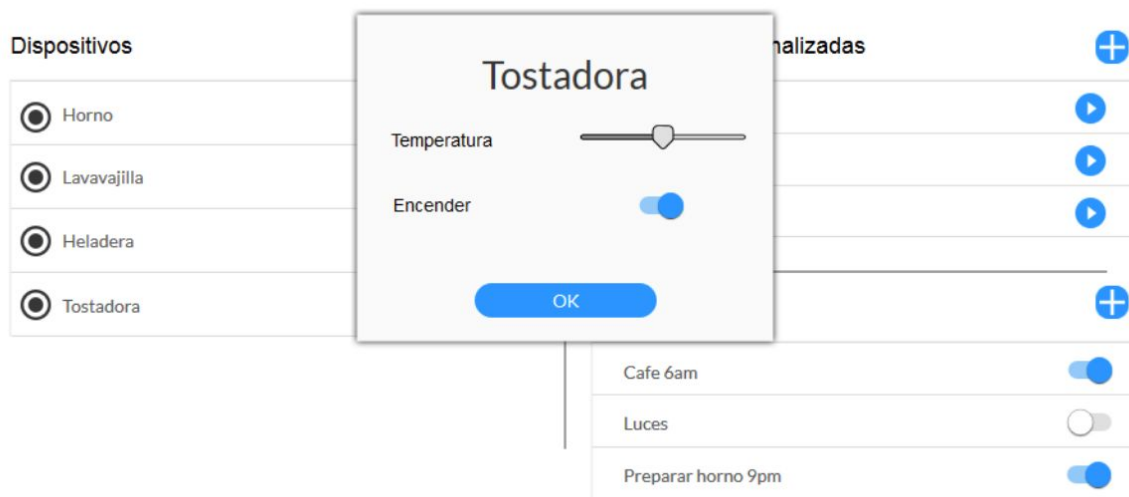
Color:



Cerrar

Guardar

Modificación de Dispositivo dentro de Ambiente - Aplicación web



Modificación de Dispositivo dentro de Ambiente - Prototipo

No se presentaron notables cambios en la modificación de un dispositivo dentro de un ambiente, como se puede ver en las capturas.

En la captura del proyecto final podemos ver cómo se mantiene el diseño minimalista y simple presente en toda la aplicación. Las funcionalidades son presentadas de manera tal que los usuarios puedan hacer uso de estas sin complicaciones, es decir que esta interfaz gráfica de usuario es de muy fácil uso para todos los usuarios representativos.

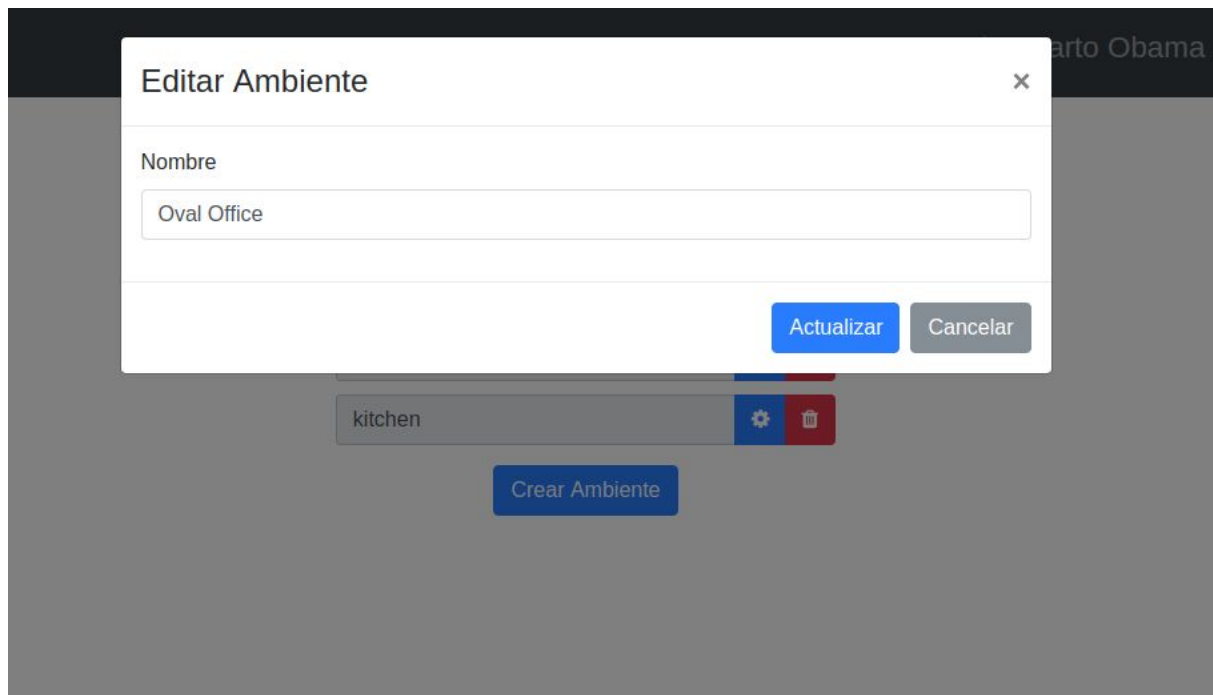
## Administrar Ambiente

**Ambientes**

Cuarto Obama		
Oval Office		
Living Room		
kitchen		

[Crear Ambiente](#)

Ventana Ambiente - Aplicación web



**Editar Ambiente** ×

Nombre

Oval Office

[Actualizar](#) [Cancelar](#)

kitchen

[Crear Ambiente](#)

Ventana Ambiente - Aplicación web

Como se puede observar en las capturas de pantalla presentadas arriba, la misma presenta el mismo estilo y uso de iconos descriptivos que las otras ventanas mostradas en el informe, manteniendo la consistencia de la aplicación. Esto brinda facilidad a la hora de realizar tareas ya que el usuario sabe cómo manejarse en todas las ventanas una vez que comprende el funcionamiento en general.

Haciendo click en el icono de modificar se abre la ventana mostrada en la captura de pantalla arriba, en donde se puede modificar el nombre de dicho ambiente.

## Alertas



**¿Esta seguro de que desea eliminar el ambiente?**

Eliminar

Cancelar



**Hubo un error al intentar crear la función**

Entendido



**La función se creó correctamente**

Entendido



**Debe especificar un nombre**

Entendido

Ejemplos de alertas en la Aplicación



## Implementación

Como fue mencionado previamente, decidimos crear esta aplicación web usando la librería de JavaScript, React. Esta librería fue creada por la empresa Facebook y brinda la facilidad de crear componentes reutilizables e interactivos para interfaces de usuario. Cada componente puede contener propiedades (props) y estados (state), y es mostrado en la ventana al crearse. Cada vez que se modifique el estado de un componente, el mismo volverá a cargarse en la ventana y consecuentemente sus hijos. React utiliza un estilo de js conocido como JSX que luego al ser compilado se transforma en Javascript puro.

Ejemplo de Componente en React:

```
class GenerateNav extends React.Component{
  constructor(props){
    super(props);
    this.state = {rooms: []};
  }

  componentWillMount(){
    api.room.list()
      .done((data)=>this.setState({rooms: data.rooms}))
  }

  render() {
    const listNavItems = this.state.rooms.map((room)=>{
      return(
        <NavItem>
          <NavLink active={this.props.active === room.id} onClick={() => this.props.callback(room.id)}
            href="#">{room.name.toUpperCase()}</NavLink>
        </NavItem>
      );
    });

    return (
      <Nav className="ml-auto" navbar>
        <NavItem>
          <NavLink active={this.props.active === "general"} onClick={() => this.props.callback("general")} href="#">GENERAL</NavLink>
        </NavItem>
        {listNavItems}
        <NavItem>
          <NavLink active={this.props.active === "administracion"} onClick={() => this.props.callback("administracion")}
            href="#">ADMINISTRACIÓN</NavLink>
        </NavItem>
        <NavItem>
          <NavLink active={this.props.active === "ayuda"} onClick={() => this.props.callback("ayuda")}
            href="#">AYUDA</NavLink>
        </NavItem>
      </Nav>
    );
  }
}
```

El código presentado en la imagen arriba es una muestra de cómo creamos una barra de navegación dinámica con React. Cada componente debe tener un constructor y un método `render()` el cual es ejecutado cada vez que se altera un estado interno. Este componente presenta un estado y es la lista de Ambientes que devuelve el API. El método `render` retorna componentes los cuales tienen un estilo similar al de XML y se le pasan propiedades como atributos. Este lenguaje no es javascript puro sino que como se mencionó anteriormente en JSX el cual una vez compilado con Babel u otro compilador se pasa a js puro.

## Conclusión

En conclusión creamos como grupo haber realizado una aplicación web que sobrepasa nuestras expectativas. La página presenta un estilo minimalista con buen uso de los colores, iconos y con buena usabilidad.

Creemos haber desarrollado un sistema de fácil comprensión y uso para todos nuestros usuarios representativos, tanto expertos en tecnología como personas con nada o poca experiencia en el ámbito tecnológico y web. Contiene buen manejo de errores, con advertencias indicando el error encontrado con colores correspondientes para hacer énfasis en dicha advertencia.

El manejo de dispositivos tiene un menú personalizado para cada tipo de dispositivo, donde cada menú es muy fácil de utilizar y personalizado, como pudimos observar en la captura de pantalla en el caso de un dispositivo de tipo lámpara.

El producto final tuvo muchas modificaciones respecto a los prototipos entregados en el primer informe como se pudo observar en las capturas de pantalla y esto fue mayoritariamente porque no conocíamos el funcionamiento exacto del back-end en ese entonces. También hubo cambios en el estilo y uso de colores los cuales decidimos realizar para hacer nuestra pagina mas intuitiva. Como se puede observar en el producto final, implementamos muchos conceptos vistos en la teoría de HCI a nuestra aplicación web. Algunos de estos son la Ley de Fitts, ya que diseñamos las ventanas para que el usuario tenga que hacer los mínimos movimientos posibles de mouse para apretar botones, iconos y/o llenar input tags. También se puede ver el buen uso de componentes GUI, el caso de Drop-down menus, iconos visualmente descriptivos a la acción que realizan y Botones claros y con color dependiendo de su funcionamiento. Es relevante al caso que la aplicación cumple con los requisitos de usabilidad que aprendimos en las teóricas de las primeras clases; es fácil de aprender a utilizar debido a su estilo simple y minimalista, contiene un buen manejo de errores, hay “information feedback” en todas las acciones, es decir se le avisa al usuario el resultado luego de realizar una acción (éxito,error, etc..). Se mantiene la consistencia de estilo y funcionalidad en toda la página web y el uso de acciones se

presenta tan fácil e intuitivamente que el usuario se siente en control, el cual es una de las pautas que plantean muchos expertos para brindar una buena user-experience.

El uso de React para la implementación de nuestra aplicación es muy discutible, es decir que trajo muchas consecuencias negativas pero otras positivas. El principal problema fue el aprendizaje de esta librería. Nos adentramos en un mundo de la programación web más extenso de lo que creíamos y consecuentemente más complejo. Nos llevó mucho tiempo aprender a hacer buen uso de esta herramienta correctamente y esta fue la principal razón por la que no pudimos entregar el trabajo en la primer fecha de entrega. Luego de extensas horas de prueba y error con React logramos “agarrarle la mano” y recién en esta instancia fue fructífero para nuestro proyecto.

## Navegadores Soportados

- Google Chrome
- Mozilla Firefox

## Instalación

Las siguientes instrucciones fueron testeadas en una pc con un Linux Mint KDE 18.2 live USB y Mozilla Firefox 56.0.

1. Instalar node.js 6 (<https://nodejs.org/en/download>) mediante cualquier método deseado, como por ejemplo utilizando aptitude:

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

2. Instalar la API facilitada por la cátedra, versión 1.1.1, según las instrucciones provistas con la misma, por ejemplo en la carpeta ~/hci/api.

3. Abrir una terminal en la carpeta donde se instaló la API en el paso previo y ejecutar “npm start” (sin comillas):

```
cd ~/hci/api  
npm start
```

4. Descargar el código de la carpeta “tpe1” dentro del repositorio “hci-2017b-07” de BitBucket provisto por la cátedra y colocarlo, por ejemplo, dentro de ~/hci/tpe1.

5. Abrir otra terminal en la carpeta donde se descargo el código de la aplicación (sin cerrar la terminal anterior) y ejecutar “npm install” para instalar las dependencias:

```
cd ~/hci/tpe1  
npm install
```

6. Finalmente, para correr la aplicación, ejecutar el comando “npm start” en la misma carpeta del paso anterior

```
npm start
```

7. (Opcional) Aunque el proceso detallado anteriormente exitosamente ejecuta el programa, el mismo se encuentra en modo de desarrollo, resultando en un desempeño reducido, en favor de un mejor feedback para simplificar el debugging del programa.

Para compilar la aplicación en lugar de ejecutarla en modo de desarrollo, se puede usar el comando “npm run build” (sin las comillas) para compilar el código fuente.