

Informe 2º parcial Aplicación para Dispositivos Móviles

Alumnos

Chierichetti Nahuel Nicolás
Rodríguez Manzanero Tamara

Proyecto

El proyecto web desarrollado consiste en una app para el restaurante "Bell Resto". Nuestro cliente nos contacto ante la necesidad de facilitarle a sus usuarios un sistema de reserva de mesas para una mejor organización y distribución de los espacios en su restaurante.

Flujo de Uso

Se eligió realizar una aplicación distribuida en secciones para que el usuario pueda navegar de manera rápida utilizando el menú el cual se despliega desde un costado, pudiendo dirigirse de manera rápida y directa a una sección específica que sea de su interés.

Esto también ayuda a la hora de mostrar la información, ya que sirve para que no saturar al usuario, sino que el vaya eligiendo que ver y en que momento.

El funcionamiento de la app para el cliente nuevo. Desde el menú principal puede acceder a las secciones:

- Inicio para tener más información al restaurante.
- Menú: donde se encuentra una selección de platos destacados y podrá ver el menú completo.
- Reserva: Facilitando sus datos, día, hora y turno de la reserva, junto a los comensales y seleccionando o especificando alguna alergia posible. El usuario podrá realizar la reserva fácilmente desde la web.
- Mis Reservas: El usuario podrá ver los datos de su reserva una vez que la haya realizado o también tendrá la opción de cancelar la reserva si así lo desea.
- Contacto: si necesita un contacto personalizado y directo podrá llamar o enviar un email al restaurante.

Sección Inicio

Bell Resto

Una experiencia única para tu paladar

RESERVAR MESA

Bell Resto es un lugar íntimo en el que podes saborear a gusto y conversar de forma relajada.

Hacemos todo de forma casera y procuramos poner siempre un toque personal en nuestro trabajo diario, por eso somos un restaurante artesanal. Contamos con un gran equipo de profesionales dispuestos a cuidar cada detalle para que los platos sean una experiencia única en tu paladar.

Cocina rica, hecha con mucha ilusión, para gente que le gusta comer y disfrutar cada bocado de la vida... En Bell Resto vas a encontrar platos de carnes rojas, pastas de elaboración propia, menú veggie y postres artesanales.

Más de 10 años de experiencia nos avalan... ¡Te Esperamos!

Pescados de Mar

VER MENÚ

Seguinos en las redes sociales!



Sección Menú

Bell Resto

Menú

Pizza Muzzarella
★★★★★
\$ 650
Base de pan de trigo con salsa de tomate, muzzarella, olivas verdes y moradas.

Pizza Especial
★★★★★
\$ 680
Salsa de tomate, morrones rojos, queso muzzarella, jamón cocido y olivas verdes.

Gnocchi de Papa
★★★★★
\$ 850
Gnocchi de papa con salsa de tomate, receta original de la casa.

Spaghetti al Huevo
★★★★★
\$ 780
Spaghetti al huevo con salsa de tomate, receta auténtica de Bell Resto.

Sorrentinos Genoveses
★★★★★
\$ 890
Sorrentinos de espinaca con el verdadero pesto genovese.

Milanesa Napolitana
★★★★★
\$ 550
Milanesa napolitana con papas cortas finas.

Milanesa Con Puré
★★★★★
\$ 500
Milanesa napolitana con puré de papas o calabaza.

Filet de Merluza
★★★★★
\$ 690
Filet de merluza a la romana con finas hierbas.

Trucha a la Manteca de Hierbas
★★★★★
\$ 1200
Trucha a la manteca cubierta de finas hierbas.

Bife de Chorizo
★★★★★
\$ 980
Bife de Chorizo, a punto y con sabor argentino.

Seguinos en las redes sociales!



Sección Reserva

Bell Resto

Reserva

Nombre *

E-mail *

Fecha de reserva * dd/mm/aaaa

Turno

¿Desea notificar alergias?

Sí

En caso afirmativo defina la/s alérgia/s en el mensaje

(*) Campos obligatorios para hacer la reserva

ENVIAR

Seguinos en las redes sociales!



Aplicaciones para Dispositivos Móviles | Copyright © – Bell Resto

Sección Mis Reservas

Bell Resto

Mis reservas

Datos de la reserva

- Nombre: Nahuel
- Apellido: Chierichetti
- Email: nahuelchierichetti@gmail.com
- Número de contacto: 2216714060
- Fecha: 2021-06-23
- Horario: 21:30
- Turno: Cena

CANCELAR RESERVA

Seguinos en las redes sociales!



Aplicaciones para Dispositivos Móviles | Copyright © – Bell Resto

Sección Contacto

Bell Resto

Contacte con nosotros

Horario de atención:

Lunes cerrado

Martes a Domingo: 7:00 a.m. - 00:00 p.m.

+54 (11) 123-4567

contacto@bellresto.com

Seguinos en las redes sociales!

Aplicaciones para Dispositivos Móviles | Copyright © – Bell Resto

El funcionamiento de la web para el cliente frecuente.

Desde la cabecera y menú principal tiene un botón que lo lleva directo a la sección de reserva, debido a que el ya conoce los servicios que se ofrecen. Esto hace que el usuario no pierda tiempo y que pueda realizar de manera rápida y sencilla su objetivo principal en el sitio.



En ambos casos, el formulario específico indica que campos son obligatorios con un (*), en el caso de que el cliente no rellene el formulario correctamente se le dará notificación con el siguiente mensaje:

No ha sido posible realizar la reserva
Compruebe los siguientes datos obligatorios

- El nombre es obligatorio.
- El apellido es obligatorio.
- El email es obligatorio, para disfrutar de la reserva.
- El teléfono es obligatorio, para notificar el aviso de la reserva.
- La fecha es obligatoria, para realizar la reserva.

Sí el usuario ha llenado los campos obligatorios correctamente recibirá el siguiente mensaje:

Reserva realizada con éxito

Ya puede ver los datos de su reserva en el apartado "Mis reservas"

VER RESERVA

Una vez que el usuario haya realizado exitosamente la reserva, le aparecerá un botón con la opción de Ver Reserva, el cual lo redirigirá a la sección "Mis Reservas" donde le mostrará los datos que el usuario cargó, y una opción de cancelar la reserva en caso de que lo desee:

Mis reservas

Datos de la reserva

- Nombre: Nahuel
- Apellido: Chierichetti
- Email: nahuelchierichetti@gmail.com
- Número de contacto: 2216714060
- Fecha: 2021-06-23
- Horario: 21:00
- Turno: Cena

CANCELAR RESERVA

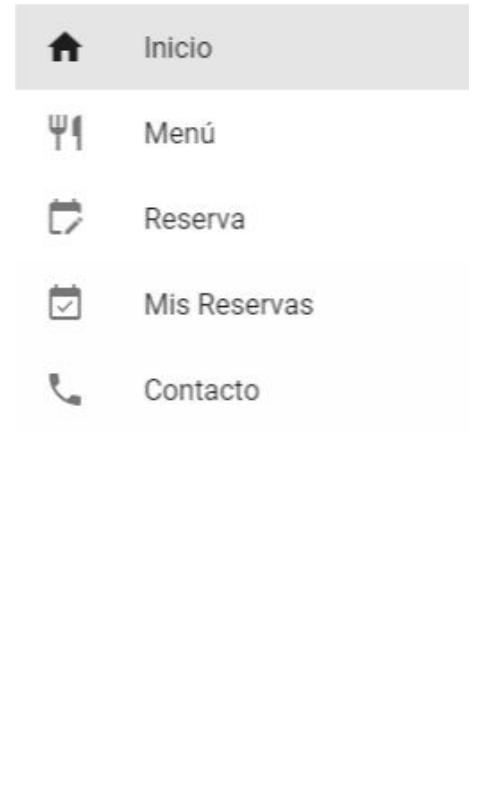
Funcionalidades Desarrolladas

Enrutamiento

El menú se realizó recorriendo por medio de un v-for, un array denominado "routes", este array contenía objetos que detallaban el nombre, ícono y el enlace de la ruta y nombre de componente. (Inicio, Menú, Reservá, Mis Reservas, Contacto).

```
let router = new VueRouter({
  routes: [
    { path: '/home', name: 'Home', component: componentinicio, },
    { path: '/menu', name: 'Menú', component: componentmenu, },
    { path: '/reserva', name: 'Reserva', component: componentreserva, },
    { path: '/mis/reservas', name: 'Mis Reservas', component: componentmostrar, },
    { path: '/contacto', name: 'Contacto', component: componentcontacto, },
    { path: '*', redirect: '/home' }
  ]
})
```

```
<v-navigation-drawer
  v-model="drawer"
  app>
<v-list>
  <v-list-item links
    v-for="item in items"
    :key="item.titulo"
    :to="item.path">
    <v-list-item-action>
      <v-icon>mdi-{{ item.icono }}</v-icon>
    </v-list-item-action>
    <v-list-item-content>
      <v-list-item-title>{{ item.titulo }}</v-list-item-title>
    </v-list-item-content>
  </v-list-item>
</v-list>
</v-navigation-drawer>
```

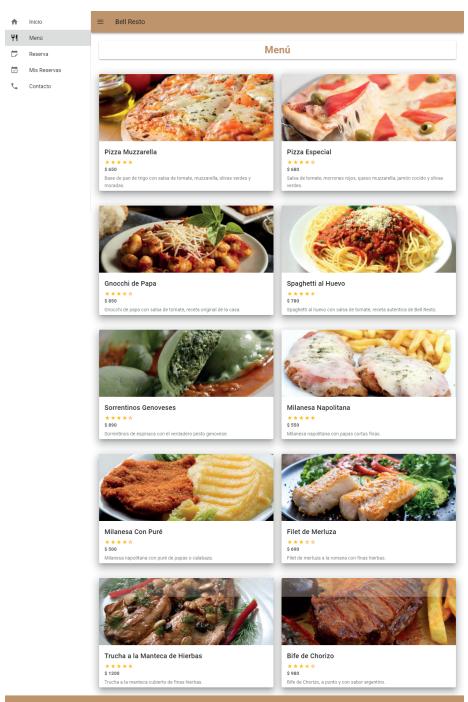


Menú Completo

El menú completo de los platos se realizó recorriendo por medio de un v-for, un array multilinea denominado "Platosfav", este array contenía objetos y elementos que detallaban los platos de la carta.

```
<div class="col-12 col-sm-6" v-for="plato in platosfav" v-bind:key="plato.id">
  <v-card class="mt-5" align="center" max-width="600" :elevation="14">
    <v-img height="200" :src="plato.src"></v-img>
    <v-card-title>{{plato.titulo}}</v-card-title>
    <v-card-text class="pl-0">
      <v-row align="center" class="mx-0">
        <ul class="pl-4">
          <li style="list-style:none">
            <v-rating :value="plato.value" colors="amber" dense half-increments readonly></v-rating>
          </li>
          <li style="list-style:none; font-weight: 800;">$ {{plato.precio}}</li>
          <li style="list-style:none" class="mt-1">{{plato.descripcion}}</li>
        </ul>
      </v-row>
    </v-card-text>
  </v-card>
</div>
```

Activar Windows



Aplicaciones para Dispositivos Móviles | Copyright © - Bell Resto



Filtros

Se utiliza el filtro “mayuscula” para aplicarle a los titulos y párrafos de la cabecera principal, como para otros textos de la web.

```
<v-row class="fill-height align="center">
  <div class="display-3 white--text">
    {{ item.titulo | mayuscula }}
  </div>
  <div class="text-center">
    <v-btn x-large href="#/menu" rounded>
```



Vue-Router / Componentes

Se utilizan los siguientes componentes para generar la navegación general de la aplicación y luego en el array routes se rutea cada componente.

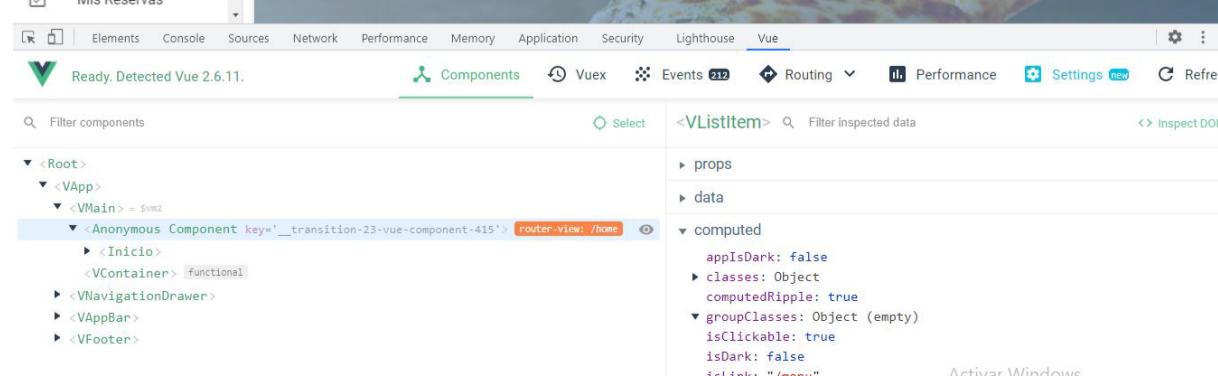
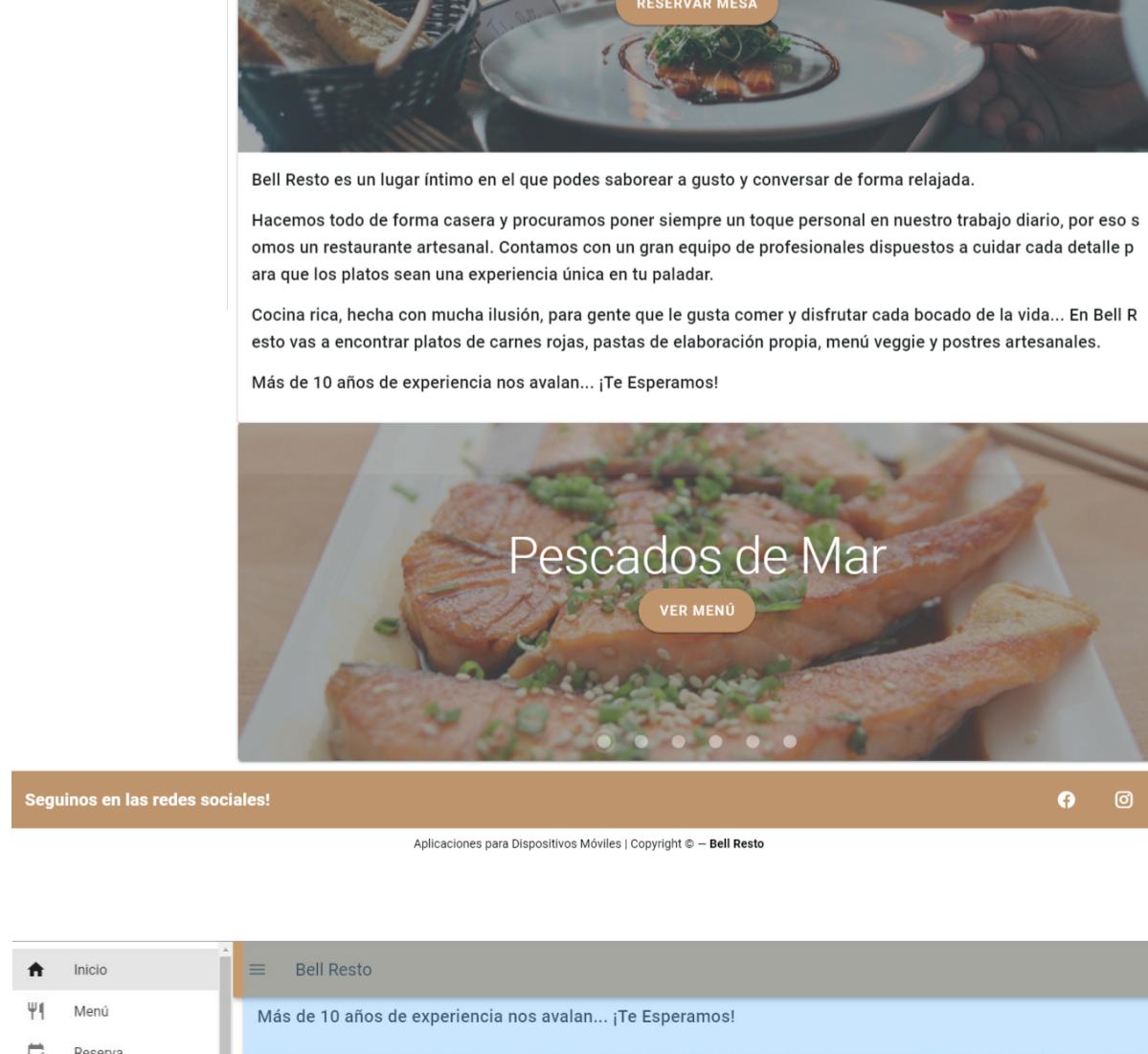
```
let componentinicio = { template: `<div class="title"><inicio></inicio></div>` }
let componentmenu = { template: `<div class="title"><menu></menu></div>` }
let componentreserva = { template: `<div class="title"><reserva></reserva></div>` }
let componentmostrar = { template: `<div class="title"><mostrar></mostrar></div>` }
let componentcontacto = { template: `<div class="title"><contacto></contacto></div>` }

let router = new VueRouter({
  routes: [
    { path: '/home', name: 'Home', component: componentinicio, },
    { path: '/menu', name: 'Menú', component: componentmenu, },
    { path: '/reserva', name: 'Reserva', component: componentreserva, },
    { path: '/mis/reservas', name: 'Mis Reservas', component: componentmostrar, },
    { path: '/contacto', name: 'Contacto', component: componentcontacto, },
    { path: '*', redirect: '/home' }
  ]
})
```

Componente inicio

Se utiliza el componente inicio para generar la sección de inicio donde por medio de un template se creará la estructura que contendrá esta sección.

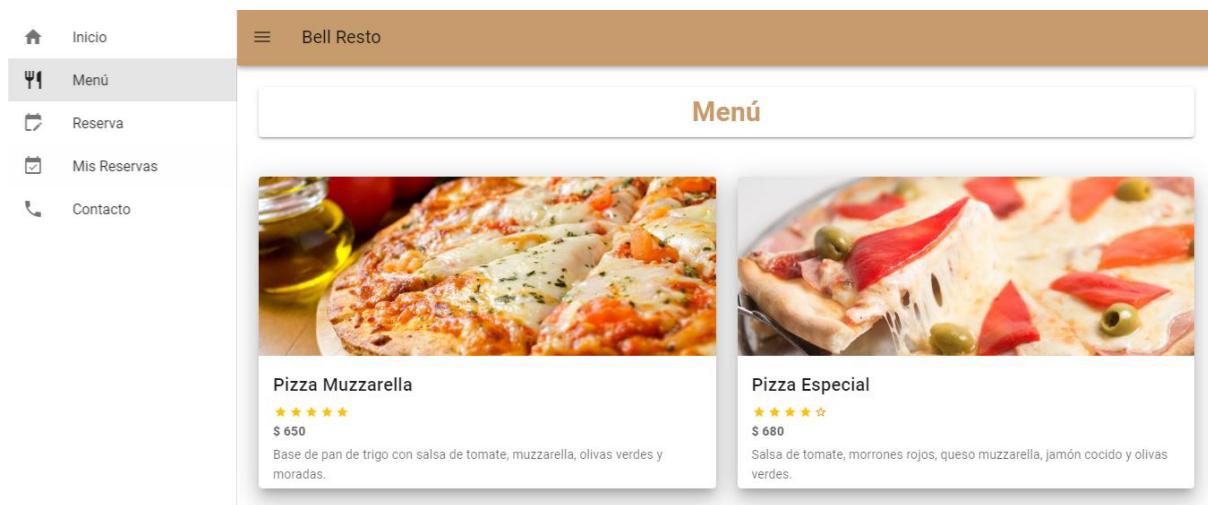
```
Vue.component('inicio', {
  template:
    `<div>
      <v-layout>
        <v-flex>
          <v-card>
            <v-img src="assets/img/banner-pc.jpg" aspect-ratio="2.75">
              <div class="container text-center p-5 mb-5 mt-9">
                <h1 class="container p-5 p-3 mt-9 white--text font-weight-bold">{{h1 | mayuscula}}</h1>
              </div>
            </v-img>
          </v-card>
        </v-flex>
      </v-layout>
    `)
```



Componente galería (menú)

Se utiliza el componente galería para generar la sección de menú donde se mostrarán los platos que se ofrecen al público. Por medio de un template se creará la estructura que contendrá esta sección.

```
Vue.component('galeria', {
  template:
    `<div class="container">
      <div class="row">
        <div style="display: contents">
          <div class="col-12">
            <v-card class="pt-3 pb-3">
              <div class="text-center">
                <h2 class="primary--text">Menú</h2>
              </div>
            </v-card>
          </div>
        </div>
      </div>
      <transition appear enter-active-class="animate__animated animate__backInRight">
        <div class="row">
          <div style="display: contents">
            <div class="col-12 col-sm-6" v-for="plato in platosfav" v-bind:key="plato.id">
              <v-card class="mt-5 align--center max-width="600" :elevation="14">
```



The screenshot shows the Vue DevTools interface with the 'Elements' tab selected. At the top, there's a toolbar with icons for device toolbar, console, sources, network, performance, memory, application, security, lighthouse, and vue. Below the toolbar, the component tree shows the structure of the application. A specific component, '<VListItem>', is selected and expanded, showing its properties: 'props', 'data', and 'computed'. The 'computed' block includes properties like 'appIsDark: false', 'classes: Object', 'computedDripple: true', 'groupClasses: Object (empty)', 'isClickable: true', 'isDark: false', 'isLink: "/menu"', and 'rootIsDark: false'. At the bottom right, there are links to 'Activar Windows' and 'Ve a Configuración para activar Windows.'

The component tree shows the following structure:

- <Root>
 - <VApp>
 - <VMain> = \$vm2
 - <Anonymous Component key="__transition-23-vue-component-417"> [router-view: /menu]
 - <Galeria>
 - <VContainer> [functional]
 - <VNavigationDrawer>
 - <VAppBar>
 - <VFooter>

Componente reserva

Se utiliza el componente reserva para generar la sección de reserva donde se mostrará un formulario donde el cliente introducirá los datos para su reserva. En el array form_data recibiremos los datos que el usuario cargue en el formulario de reserva.

Luego con la función hayErrores verificaremos hacemos un conteo de la cantidad de errores que puedan llegar a existir.

A través de un template se creará la estructura de esta sección.

```
Vue.component('reserva', {
  data:function(){
    return {
      form_data:{
        nombre: '',
        apellido: '',
        email: '',
        telefono: '',
        fecha: '',
        horario: '',
        select: null,
        checkbox: null,
        mensaje: ''
      },
      items: [
        'Almuerzo',
        'Cena',
      ],
      arr:[],
      errores:[],
      submitted: false,
      red: 'error',
      green: 'success',
      datos: true,
    }
  },
  computed:{
    hayErrores: function(){
      // Devuelve true en el caso de contener errores algún error
      return this.errores.length;
    }
  }
});
```

Este componente contiene un method con la función guardar y enviar. Guardar comprobará que los campos requeridos tengan información y en caso de que esto sea correcto lo almacenará en localStorage los datos de la reserva.

Enviar nos llevará a la función Mis Reservas.

```
methods:{
  guardar:function(form_data){
    this.datos = false;
    this.submitted = true;
    this.errores = [];

    if (!this.form_data.nombre) {
      this.errores.push('El nombre es obligatorio.');
    }
    if(this.form_data.nombre && this.form_data.nombre.length <3) {
      this.errores.push('Nombre invalido, debe tener mas de 3 caracteres.');
    }
    if (!this.form_data.apellido) {
      this.errores.push('El apellido es obligatorio.');
    }
    if(this.form_data.apellido && this.form_data.apellido.length <3) {
      this.errores.push('Apellido invalido, debe tener mas de 3 caracteres.');
    }
    if (!this.form_data.email) {
      this.errores.push('El email es obligatorio, para disfrutar de la reserva.');
    }
    if(this.form_data.email && this.form_data.email.length <3) {
      this.errores.push('Email invalido, debe tener mas de 3 caracteres.');
    }
    if (!this.form_data.telefono) {
```

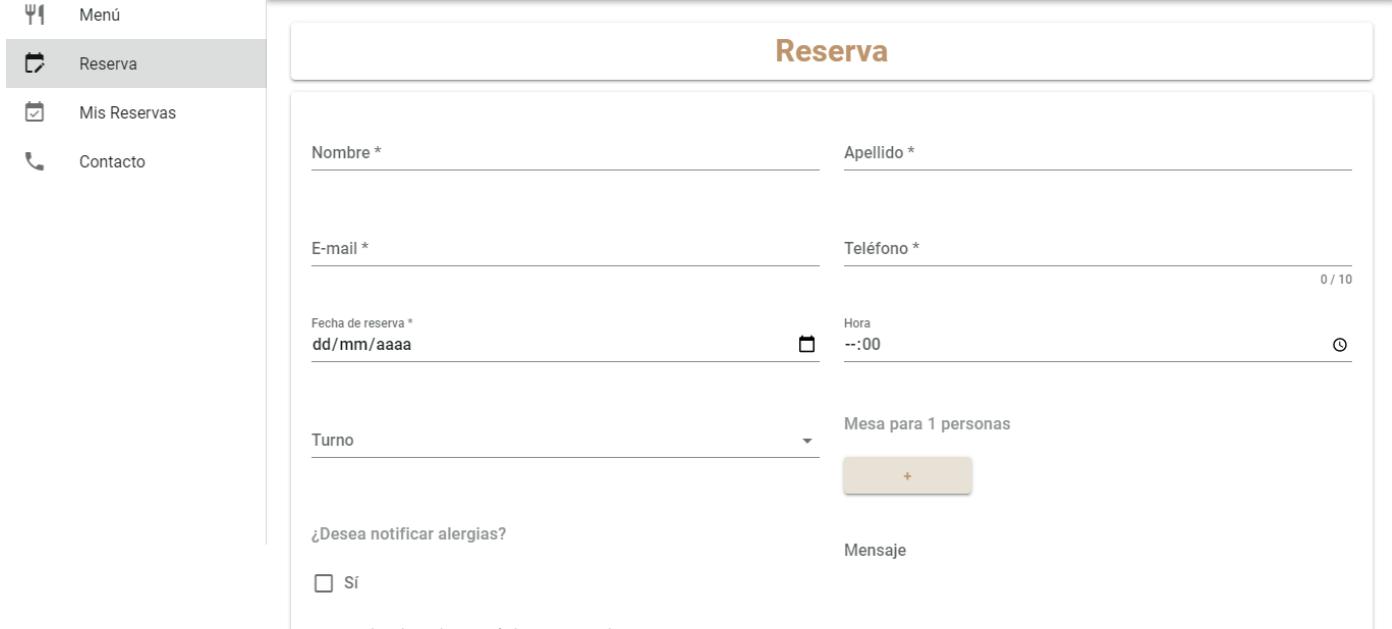
Activar Windows

Ve a Configuración

```
      if(this.errores.length == 0){
        this.datos = true;

        form_data = Object.assign({}, form_data,
        console.log(form_data)

        if(!localStorage.dato){
          arr=[];
        }else{
          arr=JSON.parse(localStorage.getItem("dato"))
          console.log(arr)
        }
        arr.push(form_data)
        localStorage.setItem("dato",JSON.stringify(arr))
      },
      enviar: function(){
        this.$router.push('/mis/reservas');
      },
    },
  )
};
```



Seguinos en las redes sociales!

Aplicaciones para Dispositivos Móviles | Copyright © – Bell Resto

Ready. Detected Vue 2.6.11.

Elements Console Sources Network Performance Memory Application Security Lighthouse Vue

Components Vuex Events Routing Performance Settings

Inspect DC

<Root>

<VApp>

<VMain> - \$vm2

<Anonymous Component key="__transition-23-vue-component-420"> router-view: /mis/reservas

<VContainer> functional

<VNavigationDrawer>

<VAppBar>

<VFooter>

props

data

computed

appliesDark: false

classes: Object

computedRipple: true

groupClasses: Object (empty)

clickable: true

isDark: false

isLink: "/menu"

rootIsDark: false

Activar Windows

Ve a Configuración para activar Windows.

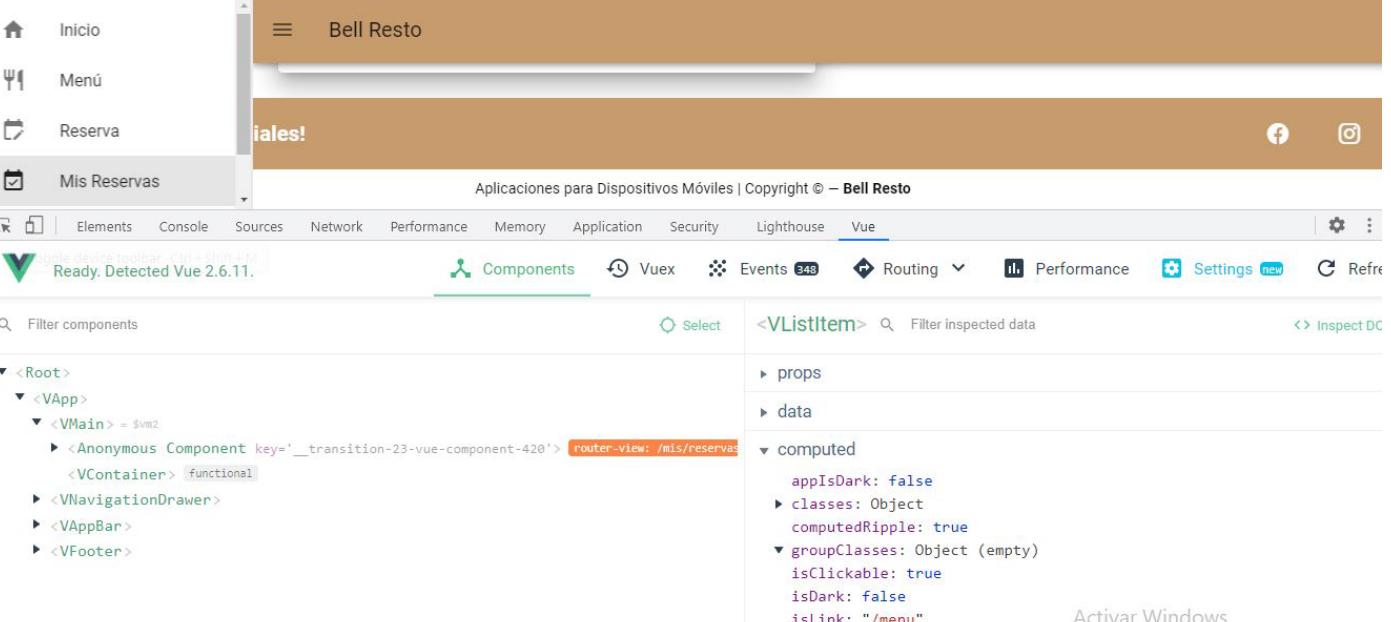
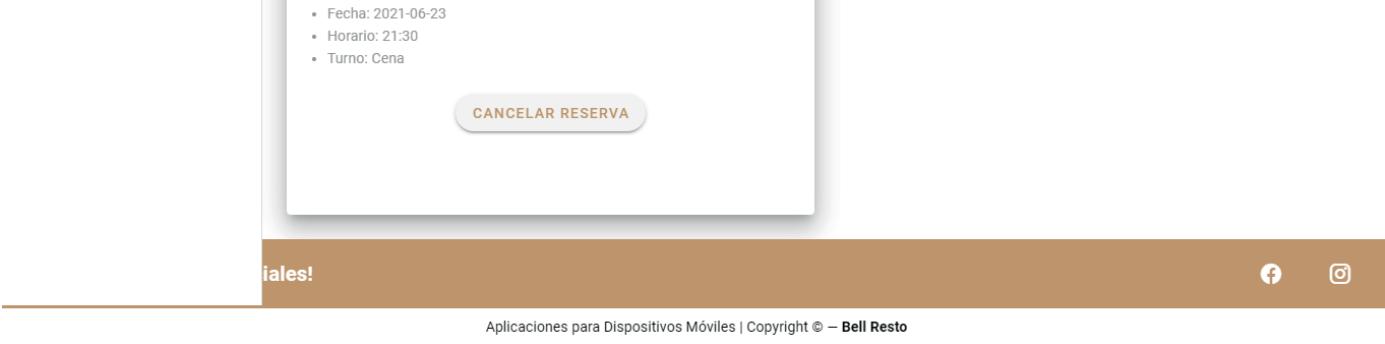
Componente mostrar

Se utiliza el componente mostrar para generar la sección de mis reservas donde se mostrarán los datos que el usuario registro en el formulario. El template generará la estructura de esta sección.

```
Vue.component("mostrar", {
  data:function(){
    return {
      arr:[],
      sin_datos: ""
    }
  },
  template:
  <div class="container">
    <div class="row">
      <div style="display: contents">
        <div class="col-12">
          <v-card class="pt-3 pb-3">
            <div class="text-center">
              <h2 class="primary--text">Mis reservas</h2>
            </div>
          </v-card>
        </div>
      </div>
    </div>
    <transition appear enter-active-class="animate__animated animate__backInRight">
      <div class="row">
        <div style="display: contents">
          <div v-if="this.arr.length == 0" class="col-12 text-center">
            <v-card class="mt-5 pt-6 mx-auto primary--text" align="center" max-width="600" :elevation="2">
              <p class="animate__animated animate__bounceIn animate__delay-2s animate__repeat-2">
```

Este componente contiene un method con la función ver_local y borrar. Ver_local comprobará que exista información guardada en localStorage para luego mostrarla en dicha sección o en caso de que no encuentre, dar un mensaje de aviso que no existen reservas hasta el momento. Borrar permitirá cancelar la reserva y borrar los datos del localStorage.

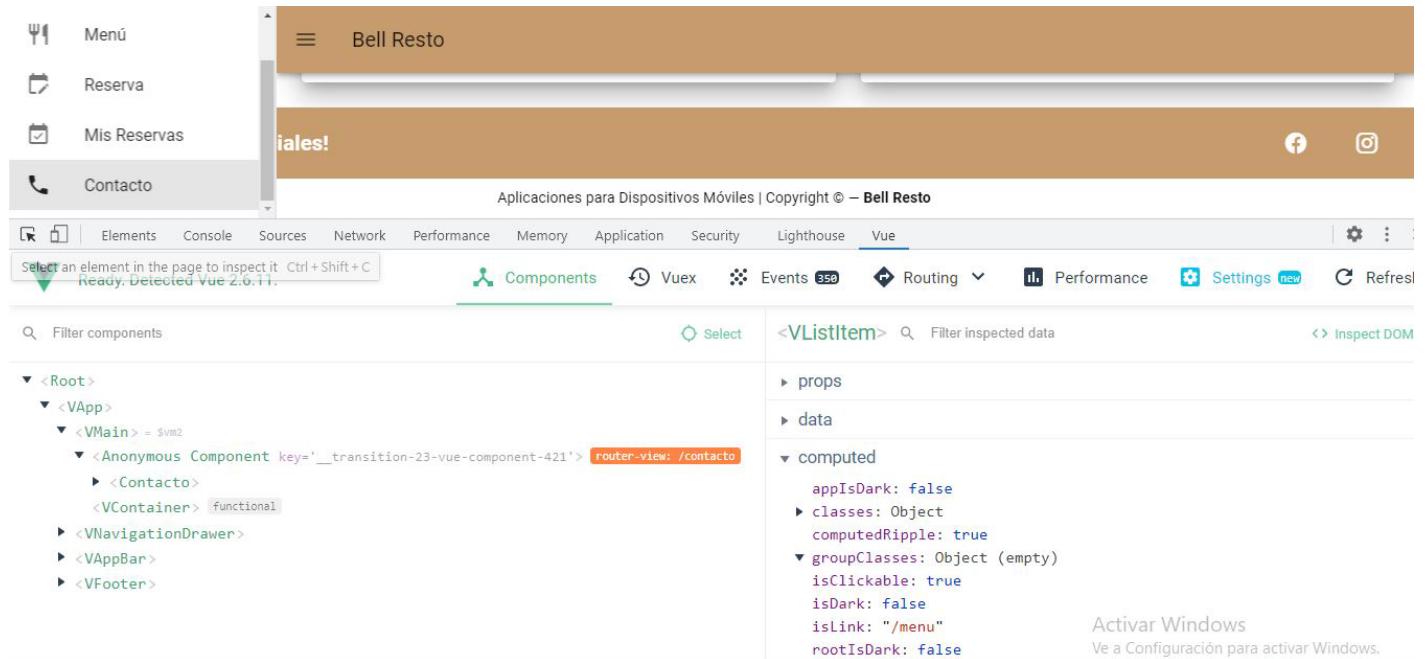
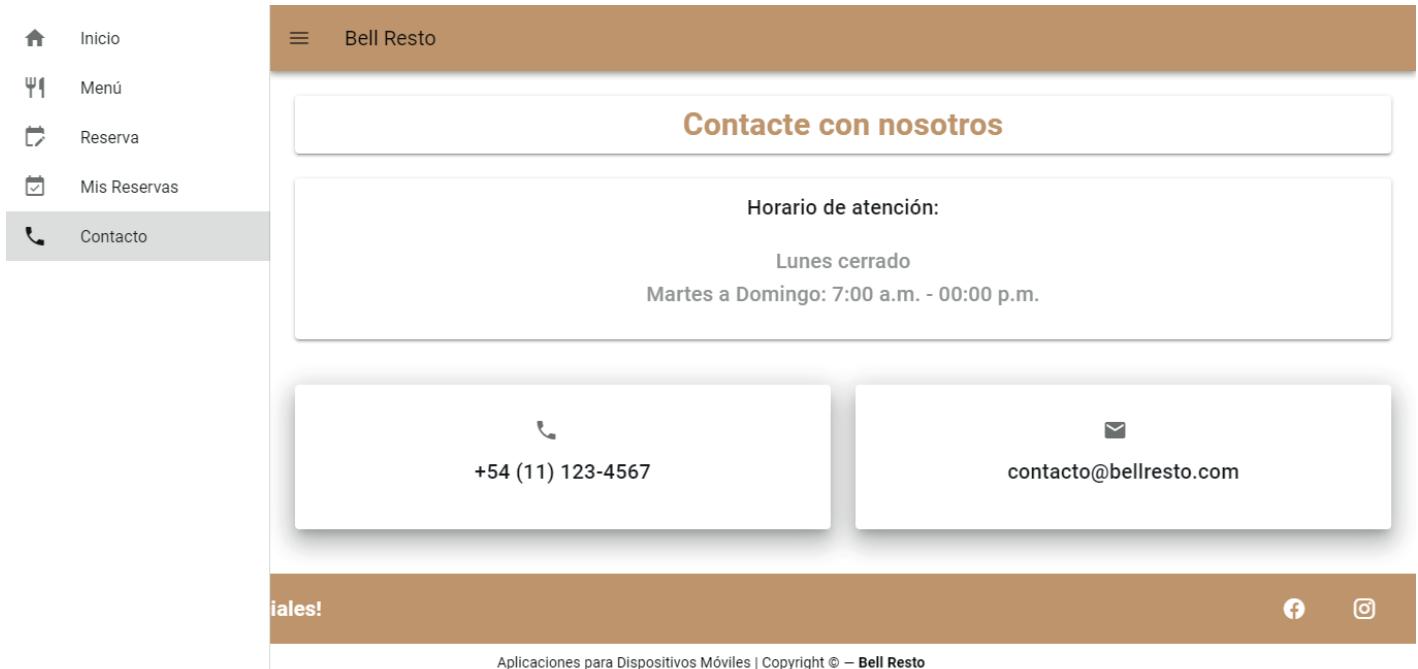
```
methods:{
  ver_local:function(){
    if(localStorage.getItem("dato")){
      var dame=JSON.parse(localStorage.getItem("dato"))
      this.arr = dame
    }
    if(this.arr.length == 0){
      this.sin_datos ="Aún no ha realizado ninguna reserva"
    }
  },
  borrar:function(item){
    arr= JSON.parse(localStorage.getItem ("dato"))
    for (var i=0; i < arr.length; i++){
      if (arr[i].fecha == item.fecha ) {
        var rta=confirm("¿Quieres cancelar tu reserva " + arr[i].nombre + "?")
        if (rta==true){
          arr.splice(i, 1);
        }
      }
    }
    localStorage.setItem("dato", JSON.stringify(arr))
    this.ver_local();
  }
})
```



Componente contacto

Se utiliza el componente contacto para generar la sección de contacto donde por medio de un template se creará la estructura que contendrá esta sección.

```
Vue.component('contacto', {
  template:
    `<div class="container">
      <div class="row">
        <div style="display: contents">
          <div class="col-12">
            <v-card class="pt-3 pb-3">
              <div class="text-center">
                <h2 class="primary--text">Contacte con nosotros</h2>
              </div>
            </v-card>
          </div>
        </div>
      </div>
    </div>
```



Componente btn-contador

Se utiliza el componente btn-contador para generar en el formulario de reserva de mesa un botón, el cual permite por medio de un v-on:click el incremento o decremento de personas que concurran a la reserva.

```
Vue.component('btn-contador', {
  data: function () {
    return {
      sumar: 1,
      texto1: "Mesa para",
      texto2: "personas"
    }
  },
  template:
  <div>
    <p style="font-size:0.8em" class="grey--text">{{texto1}} {{sumar}} {{texto2}}</p>
    <v-btn class="mr-4 col-3 primary--text" v-model="sumar" v-on:click="sumar++"> + </v-btn>
    <v-btn class="mr-4 col-3 primary--text" v-model="sumar" v-on:click="sumar--" v-if="sumar > 1"> - </v-btn>
  </div>
})
```

Mesa para 2 personas



V-if v-else

Se utiliza v-if para comprobar si hay errores en los campos obligatorios y en el caso de ser así, nos arroja el siguiente mensaje:

No ha sido posible realizar la reserva
Compruebe los siguientes datos obligatorios

- El nombre es obligatorio.
- El apellido es obligatorio.
- El email es obligatorio, para disfrutar de la reserva.
- El teléfono es obligatorio, para notificar el aviso de la reserva.
- La fecha es obligatoria, para realizar la reserva.

En el caso de que no haya errores se utiliza v-else y mostraría la confirmación de la reserva:

Reserva realizada con éxito

Ya puede ver los datos de su reserva en el apartado "Mis reservas"

VER RESERVA

V-bind / operador ternario

```
<div v-if="submitted === true" class="col-12 mt-5 white--text" v-bind:class="[datos ? green : red]" >
  <div v-if="hayErrores" class="reserva pt-2 pb-6 pl-7">
    <h3 class="mt-3 pt-3"><span>No ha sido posible realizar la reserva</span></h3>
    <p>Compruebe los siguientes datos obligatorios</p>
    <ul>
      <li v-for="x in errores">{{x}}</li>
    </ul>
  </div>
  <div v-else>
    <h3 class="mt-3 pt-7 text-center"><span>Reserva realizada con éxito</span></h3>
    <div class="container text-center pt-5 mb-5">
      <p style="font-size:0.7em" class="white--text">Ya puede ver los datos de su reserva en el apartado <a href="#">Ver Reserva</a>
    </div>
  </div>
</div>
```

Se ha utilizado v-bind:class con operador ternario para mostrar el error en caso de que estos existan, o la confirmación de la reserva si el usuario ingreso de manera correcta todos los campos obligatorios requeridos.

Vuetify

Hemos hemos utilizado la libreria de Vuetify de Vue para configurar el front-end de la aplicación.

```
<v-app id="inspire">
  <v-navigation-drawer
    v-model="drawer"
    app>
    <v-list>
      <v-list-item links
        v-for="item in items"
        :key="item.titulo"
        :to="item.path">
        <v-list-item-action>
          <v-icon>mdi-{{ item.icono }}</v-icon>
        </v-list-item-action>
        <v-list-item-content>
          <v-list-item-title>{{ item.titulo }}</v-list-item-title>
        </v-list-item-content>
      </v-list-item>
    </v-list>
  </v-navigation-drawer>

  <v-app-bar class="primary" app>
    <v-app-bar-nav-icon @click.stop="drawer = !drawer"></v-app-bar-nav-icon>
    <v-toolbar-title>Bell Resto</v-toolbar-title>
  </v-app-bar>
  <v-main>
    <v-container fluid>
      <v-fade-transition mode="out-in">
        <router-view></router-view>
      </v-fade-transition>
    </v-container>
  </v-main>
</v-app>
```

Transiciones / Animate.css

Hemos utilizado transiciones para generar un efecto más visual y no tan brusco en la entrada y muestra de la información.

```
<transition appear enter-active-class="animate__animated animate__backInRight">
  <div class="row">
    <div style="display: contents">
      <div class="col-12 col-sm-6" v-for="plato in platosfav" v-bind:key="plato.id">
        <v-card class="mt-5" align="center" max-width="600" :elevation="14">
          <v-img height="200" :src="plato.src"></v-img>
          <v-card-title>{{plato.titulo}}</v-card-title>
          <v-card-text class="pl-0">
            <v-row align="center" class="mx-0">
              <ul class="pl-4">
                <li style="list-style:none">
                  <v-rating :value="plato.value" color="amber" dense half-increments>
                    </li>
                  <li style="list-style:none; font-weight: 800;">$ {{plato.precio}}</li>
                  <li style="list-style:none" class="mt-1">{{plato.descripcion}}</li>
                </ul>
            </v-row>
          </v-card-text>
        </v-card>
      </div>
    </div>
  </transition>
```

Material Design

Utilizamos este framework de vue.js, el cual nos facilita el uso de iconos y genera una mejor navegación para el usuario.

```
<v-list-item-action>
  <v-icon>mdi-{{ item.icono }}</v-icon>
</v-list-item-action>
```

```
data: () => ({
  icons: [
    'mdi-facebook',
    'mdi-instagram'
  ],
  items:[
    { titulo: 'Inicio', icono: 'home', path:'home' },
    { titulo: 'Menú', icono: 'silverware-fork-knife', path:'menu' },
    { titulo: 'Reserva', icono: 'calendar-edit', path:'reserva' },
    { titulo: 'Mis Reservas', icono: 'calendar-check', path:'mis/reservas' },
    { titulo: 'Contacto', icono: 'phone', path:'contacto' },
  ],
  drawer: null,
}),
```