# MANUEAL DE USUARIO

## CLIENTE:

cliente.component.html

```
<div class="contenedor-seccion">
  <h2 class="titulo-seccion">Datos del Cliente</h2>

  <ion-item>
    <ion-label position="stacked">Nombre completo</ion-label>
    <ion-input [(ngModel)]="nombre" placeholder="Juan Pérez"></ion-input>
  </ion-item>

  <ion-item>
    <ion-label position="stacked">Teléfono</ion-label>
    <ion-input [(ngModel)]="telefono" type="tel" placeholder="1234567890"></ion-input>
  </ion-item>

  <ion-item>
    <ion-label position="stacked">Correo electrónico</ion-label>
    <ion-input [(ngModel)]="correo" type="email" placeholder="correo@ejemplo.com"></ion-input>
  </ion-item>

  <!-- Botón para guardar los datos -->
  <ion-button expand="full" (click)="emitirDatos()">Guardar Datos</ion-button>

  <!-- Mostrar los clientes guardados -->
  <div *ngIf="clientes.length > 0">
    <h3>Clientes Guardados</h3>
    <ion-list>
      <ion-item *ngFor="let cliente of clientes">
        <ion-label>
          <p><strong>Nombre:</strong> {{ cliente.nombre }}</p>
          <p><strong>Teléfono:</strong> {{ cliente.telefono }}</p>
          <p><strong>Correo:</strong> {{ cliente.correo }}</p>
        </ion-label>
      </ion-item>
    </ion-list>
  </div>
</div>
```

cliente.component.ts

```
import { CommonModule } from '@angular/common';
import { Component, EventEmitter, Output } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { IonicModule } from '@ionic/angular';

@Component({
  selector: 'app-cliente',
  standalone: true,
  imports: [IonicModule, CommonModule, FormsModule],
  templateUrl: './cliente.component.html',
  styleUrls: ['./cliente.component.scss'],
})
export class ClienteComponent {

  nombre: string = '';
  telefono: string = '';
  correo: string = '';
  clientes: Array<any> = []; // Lista de clientes

  emitirDatos() {
    // Crear un objeto con los datos del cliente
    const cliente = {
      nombre: this.nombre,
      telefono: this.telefono,
      correo: this.correo
    };

    // Agregar el cliente a la lista
    this.clientes.push(cliente);

    // Limpiar los campos del formulario
    this.nombre = '';
    this.telefono = '';
    this.correo = '';
  }
}
```

Este componente de Angular llamado ClienteComponent es un formulario interactivo que permite al usuario ingresar y guardar datos de clientes. A continuación, se explica cada parte del componente:

---

**1. Archivo TypeScript (**cliente.component.ts**)**

Este archivo contiene la lógica del componente:

- **Propiedades:**
  - nombre, telefono, correo: Variables enlazadas al formulario para capturar los datos del cliente.
  - clientes: Un arreglo que almacena los datos de todos los clientes ingresados.
- **Método** emitirDatos()**:**
  - Crea un objeto con los datos ingresados (nombre, telefono, correo).
  - Agrega este objeto al arreglo clientes.
  - Limpia los campos del formulario para permitir nuevos datos.

---

**2. Archivo HTML (**cliente.component.html**)**

Este archivo define la estructura visual del componente:

- **Formulario:**
  - Contiene tres campos de entrada (ion-input) para capturar el nombre, teléfono y correo del cliente.
  - Cada campo está enlazado a las propiedades del componente mediante [(ngModel)].
- **Botón Guardar:**
  - Un botón (ion-button) que ejecuta el método emitirDatos() al hacer clic, guardando los datos ingresados.
- **Lista de Clientes:**
  - Si hay clientes guardados (clientes.length > 0), se muestra una lista (ion-list) con los datos de cada cliente.

- La lista utiliza *ngFor para iterar sobre el arreglo clientes y mostrar cada cliente.

---

**3. Archivo SCSS (cliente.component.scss)**

Actualmente, este archivo está vacío, pero puede ser utilizado para agregar estilos personalizados al componente.

---

**Flujo del Componente:**

1. El usuario ingresa los datos en los campos del formulario.
2. Al hacer clic en "Guardar Datos", los datos se guardan en el arreglo clientes.
3. Los datos guardados se muestran dinámicamente en una lista debajo del formulario.

---

**Tecnologías Utilizadas:**

- **Angular:** Para la estructura del componente.
- **Ionic Framework:** Para los elementos visuales (ion-item, ion-input, ion-button, etc.).
- **Two-Way Data Binding ([(ngModel)]):** Para enlazar los datos del formulario con las propiedades del componente.
- **Directivas Angular (*ngIf, *ngFor):** Para mostrar u ocultar elementos dinámicamente.

Este componente es útil para aplicaciones que requieren capturar y mostrar datos de manera sencilla, como un sistema de gestión de clientes.

MESA:

mesa.component.html

src > app > mesa > mesa.component.html > ion-card

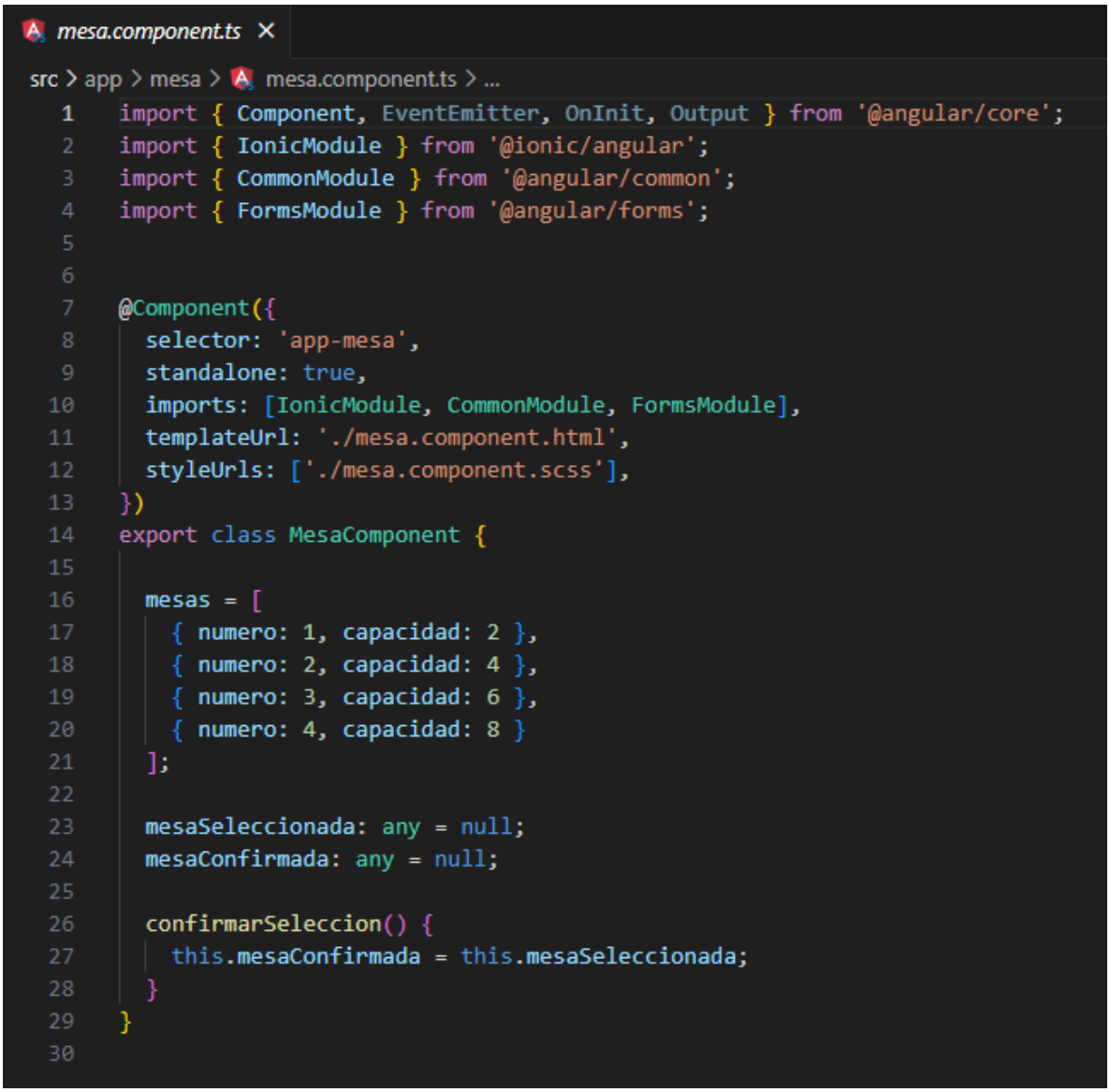
Go to component

```html
<ion-card>
  <ion-card-header>
    <ion-card-title>Seleccionar Mesa</ion-card-title>
  </ion-card-header>

  <ion-card-content>
    <ion-list>
      <ion-radio-group [(ngModel)]="mesaSeleccionada">
        <ion-item *ngFor="let mesa of mesas">
          <ion-label>Mesa {{ mesa.numero }} ({{ mesa.capacidad }} personas)</ion-label>
          <ion-radio slot="start" [value]="mesa"></ion-radio>
        </ion-item>
      </ion-radio-group>
    </ion-list>

    <ion-button expand="block" (click)="confirmarSeleccion()" [disabled]="!mesaSeleccionada">
      Confirmar Mesa
    </ion-button>

    <div *ngIf="mesaConfirmada" class="mesa-confirmada">
      <ion-text color="success">
        <h3>Mesa Confirmada:</h3>
        <p>Número: {{ mesaConfirmada.numero }}</p>
        <p>Capacidad: {{ mesaConfirmada.capacidad }} personas</p>
      </ion-text>
    </div>
  </ion-card-content>
</ion-card>

```

mesa.component.scss

src > app > mesa > mesa.component.scss > ion-card

```scss
ion-card {
  margin: 10px;
}

ion-item {
  --min-height: 48px;
}
```




Este componente de Angular llamado MesaComponent permite al usuario seleccionar una mesa de una lista y confirmar su elección. A continuación, se explica cada parte del componente:

---

**1. Archivo TypeScript (mesa.component.ts)**

Este archivo contiene la lógica del componente:

- **Propiedades:**
    - mesas: Un arreglo que contiene información sobre las mesas disponibles, incluyendo su número y capacidad.

    - mesaSeleccionada: Variable que almacena la mesa seleccionada por el usuario.
    - mesaConfirmada: Variable que almacena la mesa confirmada después de que el usuario haga clic en el botón de confirmación.

- **Método** confirmarSeleccion()**:**
    - Asigna la mesa seleccionada (mesaSeleccionada) a la variable mesaConfirmada, indicando que la selección ha sido confirmada.

---

**2. Archivo HTML (mesa.component.html)**

Este archivo define la estructura visual del componente:

- **Lista de Mesas:**
    - Utiliza un ion-radio-group para permitir al usuario seleccionar una mesa de la lista.
    - Cada mesa se representa como un ion-item con un ion-radio que permite la selección.
    - La directiva *ngFor itera sobre el arreglo mesas para generar dinámicamente los elementos de la lista.
- **Botón Confirmar:**
    - Un botón (ion-button) que ejecuta el método confirmarSeleccion() al hacer clic.
    - El botón está deshabilitado ([disabled]="!mesaSeleccionada") hasta que el usuario seleccione una mesa.
- **Confirmación de Mesa:**
    - Si hay una mesa confirmada (mesaConfirmada), se muestra un mensaje con los detalles de la mesa seleccionada (número y capacidad).

---

**3. Archivo SCSS (mesa.component.scss)**

Este archivo contiene estilos personalizados para el componente:

- **ion-card:** Define un margen de 10px para las tarjetas.
- **ion-item:** Ajusta la altura mínima de los elementos de la lista.

---

**Flujo del Componente:**

1. El usuario ve una lista de mesas disponibles con su número y capacidad.
2. Selecciona una mesa utilizando los botones de radio (ion-radio).
3. Hace clic en el botón "Confirmar Mesa" para confirmar su selección.
4. Los detalles de la mesa confirmada se muestran en la parte inferior del componente.

---

**Tecnologías Utilizadas:**

- **Angular:** Para la estructura del componente.
- **Ionic Framework:** Para los elementos visuales (ion-card, ion-radio-group, ion-button, etc.).
- **Two-Way Data Binding ([(ngModel)]):** Para enlazar la selección de la mesa con la propiedad mesaSeleccionada.
- **Directivas Angular (*ngFor, *ngIf):** Para generar dinámicamente la lista de mesas y mostrar u ocultar elementos según las condiciones.

Este componente es útil para aplicaciones que requieren la selección de opciones, como un sistema de reservas de mesas en un restaurante.

RESERVA:

reserva.component.html

src > app > reserva > reserva.component.html > div.contenedor-seccion

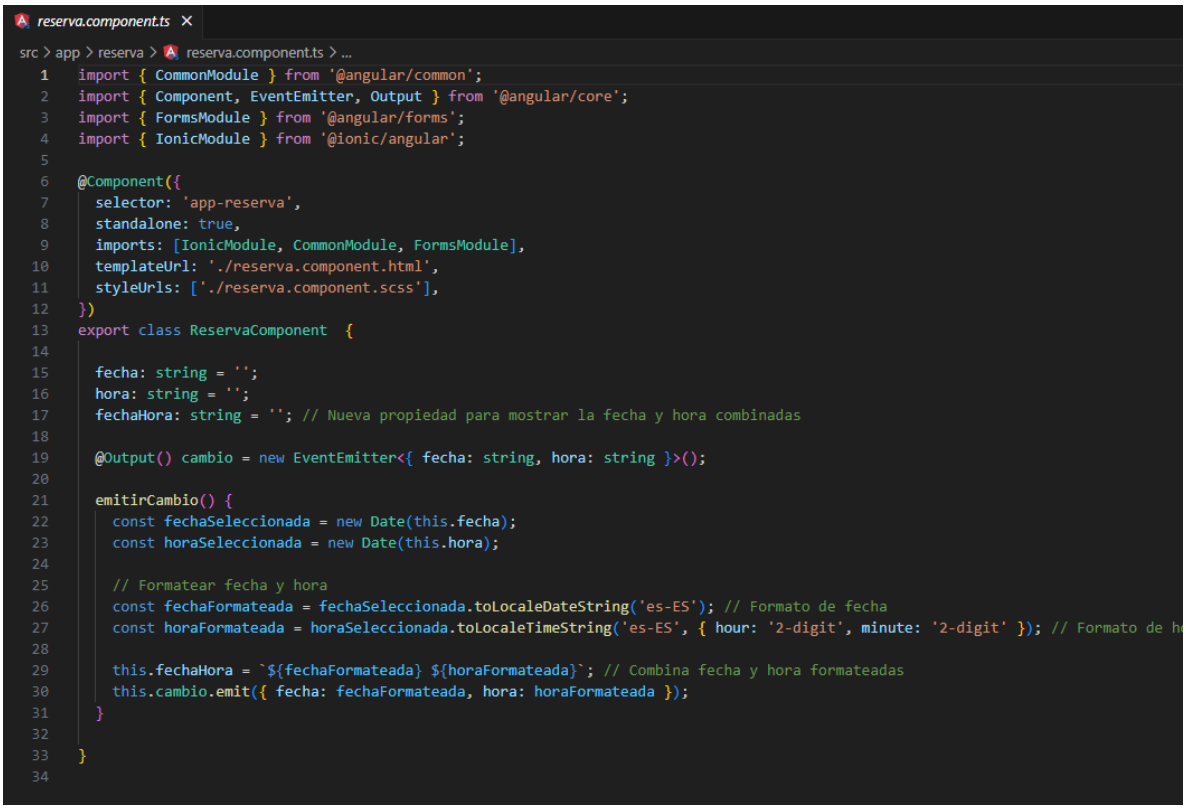
Go to component

```html
<div class="contenedor-seccion">
  <h2 class="titulo-seccion">Selecciona fecha y hora</h2>

  <ion-item>
    <ion-label position="stacked">Fecha</ion-label>
    <ion-datetime presentation="date" [(ngModel)]="fecha"></ion-datetime>
  </ion-item>

  <ion-item>
    <ion-label position="stacked">Hora</ion-label>
    <ion-datetime presentation="time" [(ngModel)]="hora"></ion-datetime>
  </ion-item>

  <!-- Botón para confirmar la selección -->
  <ion-button expand="block" (click)="emitirCambio()">Confirmar selección</ion-button>

  <div *ngIf="fechaHora">
    <h3>Fecha y Hora seleccionadas:</h3>
    <p>{{ fechaHora }}</p>
  </div>
</div>

```

reserva.component.scss

src > app > reserva > reserva.component.scss > .contenedor-seccion

```scss
.contenedor-seccion {
  padding: 5px;
  border-radius: 10px;
  background-color: #1e1e1e;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);
  width: auto; /* Reduce el ancho máximo del contenedor */
  height: auto; /* Reduce la altura máxima del contenedor */
  margin: 0 auto;   /* Centrado horizontal */
}

.titulo-seccion {
  font-size: 1.2rem;
  font-weight: 600;
  color: #ffffff;
  text-align: center;
  margin-bottom: 20px;
}

ion-item {
  --background: transparent;
  margin-bottom: 1px;
}

ion-label {
  color: #cccccc;
  font-size: 1rem;
}

ion-datetime {
  width: 100%;
  height: 100%;
  transform: scale(0.85); // Escala más pequeña
  color: #ffffff;
}

```




El componente ReservaComponent es un formulario interactivo que permite al usuario seleccionar una fecha y una hora, combinarlas y emitir un evento con los datos seleccionados. A continuación, se explica cada parte del componente:

---

### 1. Archivo TypeScript (reserva.component.ts)

Este archivo contiene la lógica del componente:

- **Propiedades:**
  - fecha: Variable enlazada al selector de fecha (ion-datetime) para almacenar la fecha seleccionada.
  - hora: Variable enlazada al selector de hora (ion-datetime) para almacenar la hora seleccionada.
  - fechaHora: Propiedad que combina la fecha y la hora seleccionadas en un formato legible.
- **Decorador @Output:**

    - cambio: Un evento que emite un objeto con la fecha y hora seleccionadas al componente padre.

- **Método** emitirCambio()**:**
    - Convierte las variables fecha y hora en objetos Date.
    - Formatea la fecha y la hora en un formato legible (es-ES).
    - Combina la fecha y la hora formateadas en la propiedad fechaHora.
    - Emite un evento con los datos formateados (fecha y hora) al componente padre.

---

**2. Archivo HTML (reserva.component.html)**

Este archivo define la estructura visual del componente:

- **Selector de Fecha:**
    - Un campo de entrada (ion-datetime) con presentación de tipo date para seleccionar una fecha.
    - Está enlazado a la propiedad fecha mediante [(ngModel)].
- **Selector de Hora:**
    - Un campo de entrada (ion-datetime) con presentación de tipo time para seleccionar una hora.
    - Está enlazado a la propiedad hora mediante [(ngModel)].
- **Botón Confirmar:**
    - Un botón (ion-button) que ejecuta el método emitirCambio() al hacer clic.
    - Este botón confirma la selección de fecha y hora.
- **Visualización de Fecha y Hora Seleccionadas:**
    - Si la propiedad fechaHora tiene un valor, se muestra un mensaje con la fecha y hora seleccionadas.

---

**3. Archivo SCSS (reserva.component.scss)**

Este archivo contiene estilos personalizados para el componente:

- **contenedor-seccion:**
    - Define un contenedor con bordes redondeados, fondo oscuro, y sombra para un diseño moderno.
    - Centra el contenido horizontalmente y ajusta el tamaño automáticamente.
- **titulo-seccion:**
    - Estiliza el título con un tamaño de fuente mediano, color blanco, y alineación centrada.
- **Estilos de ion-item, ion-label, y ion-datetime:**
    - Personalizan los elementos de entrada para que se integren con el diseño oscuro.
    - Reducen el tamaño de los selectores de fecha y hora para un diseño más compacto.

---

**Flujo del Componente:**

1. El usuario selecciona una fecha y una hora utilizando los selectores (ion-datetime).
2. Al hacer clic en "Confirmar selección", el método emitirCambio() combina y formatea los datos.
3. La fecha y hora seleccionadas se muestran en pantalla y se emite un evento al componente padre con los datos.

---

**Tecnologías Utilizadas:**

- **Angular:** Para la estructura del componente.
- **Ionic Framework:** Para los elementos visuales (ion-datetime, ion-item, ion-button, etc.).
- **Two-Way Data Binding ([(ngModel)]):** Para enlazar los selectores de fecha y hora con las propiedades del componente.

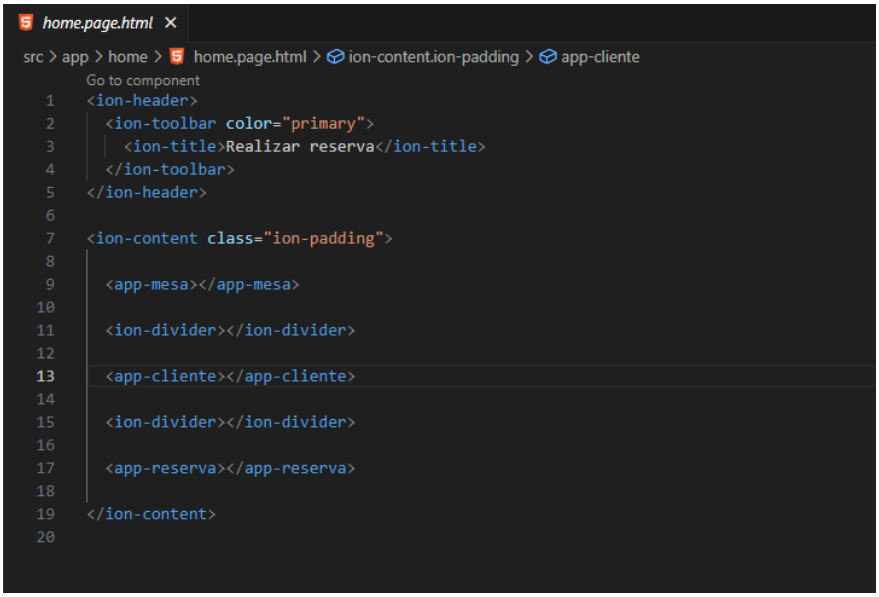
- **EventEmitter (@Output):** Para emitir eventos al componente padre.

---

**Uso del Componente:**

Este componente es ideal para aplicaciones que requieren la selección de una fecha y hora, como sistemas de reservas, citas o eventos. Su diseño moderno y funcionalidad lo hacen fácil de usar e integrar en aplicaciones más grandes.

HOME:



home.page.scss

src > app > home > home.page.scss > #container strong

```scss
#container {
  text-align: center;

  position: absolute;
  left: 0;
  right: 0;
  top: 50%;
  transform: translateY(-50%);
}

#container strong {
  font-size: 20px;
  line-height: 26px;
}

#container p {
  font-size: 16px;
  line-height: 22px;

  color: #8c8c8c;

  margin: 0;
}

#container a {
  text-decoration: none;
}
```

home.page.ts

src > app > home > home.page.ts > HomePage

```ts
import { Component, CUSTOM_ELEMENTS_SCHEMA } from '@angular/core';
import { ReservaComponent } from "../reserva/reserva.component";
import { IonicModule } from '@ionic/angular';
import { ClienteComponent } from '../cliente/cliente.component';
import { MesaComponent } from '../mesa/mesa.component';
import { CommonModule } from '@angular/common';


@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
  standalone: true,
  imports: [IonicModule, CommonModule, ClienteComponent, ReservaComponent, MesaComponent],
  schemas: [CUSTOM_ELEMENTS_SCHEMA],  // Agrega esto para soportar elementos web
})
export class HomePage {
  constructor() {}
}
```

El componente HomePage es la página principal de la aplicación, que actúa como un contenedor para los componentes MesaComponent, ClienteComponent y ReservaComponent. Su propósito es integrar estos componentes para que el usuario pueda realizar una reserva completa (selección de mesa, ingreso de datos del cliente y selección de fecha y hora). A continuación, se explica cada parte del componente:

---

**1. Archivo TypeScript (home.page.ts)**

Este archivo define la lógica básica de la página:

- **Importaciones:**
  - Importa los componentes MesaComponent, ClienteComponent y ReservaComponent para utilizarlos dentro de esta página.
  - También importa módulos esenciales como IonicModule y CommonModule.
- **Decorador @Component:**
  - Define el selector app-home, que se utiliza en el archivo HTML.
  - Especifica los estilos (home.page.scss) y la plantilla (home.page.html) de la página.
  - Incluye los componentes importados en la propiedad imports para que puedan ser utilizados en la plantilla.
  - Utiliza CUSTOM_ELEMENTS_SCHEMA para permitir el uso de componentes personalizados como app-mesa, app-cliente y app-reserva.
- **Clase** HomePage**:**
  - Actualmente, no contiene lógica adicional, ya que su propósito principal es servir como un contenedor para los componentes.

---

**2. Archivo HTML (home.page.html)**

Este archivo define la estructura visual de la página:

- **Encabezado (ion-header):**
  - Contiene una barra de herramientas (ion-toolbar) con el título "Realizar reserva".
- **Contenido Principal (ion-content):**
  - Incluye los tres componentes principales:

1. **<app-mesa>:** Renderiza el componente para seleccionar una mesa.
2. **<app-cliente>:** Renderiza el componente para ingresar los datos del cliente.
3. **<app-reserva>:** Renderiza el componente para seleccionar la fecha y hora de la reserva.

- Los componentes están separados por divisores (ion-divider) para mejorar la organización visual.

---

**3. Archivo SCSS (home.page.scss)**

Este archivo contiene estilos personalizados para la página:

- **#container:**
  - Define un contenedor centrado verticalmente y horizontalmente.
  - Aunque no se utiliza directamente en el archivo HTML actual, puede ser útil para centrar contenido adicional en la página.
- **Estilos de texto (#container strong, #container p, #container a):**
  - Estilizan elementos de texto con tamaños de fuente específicos y colores.
  - Estos estilos no están aplicados a los componentes actuales, pero podrían ser utilizados en futuras modificaciones.

---

**Flujo del Componente:**

1. El usuario selecciona una mesa utilizando el componente MesaComponent.
2. Ingresa los datos del cliente en el componente ClienteComponent.
3. Selecciona la fecha y hora de la reserva en el componente ReservaComponent.
4. Cada componente funciona de manera independiente, pero juntos forman un flujo completo para realizar una reserva.

---

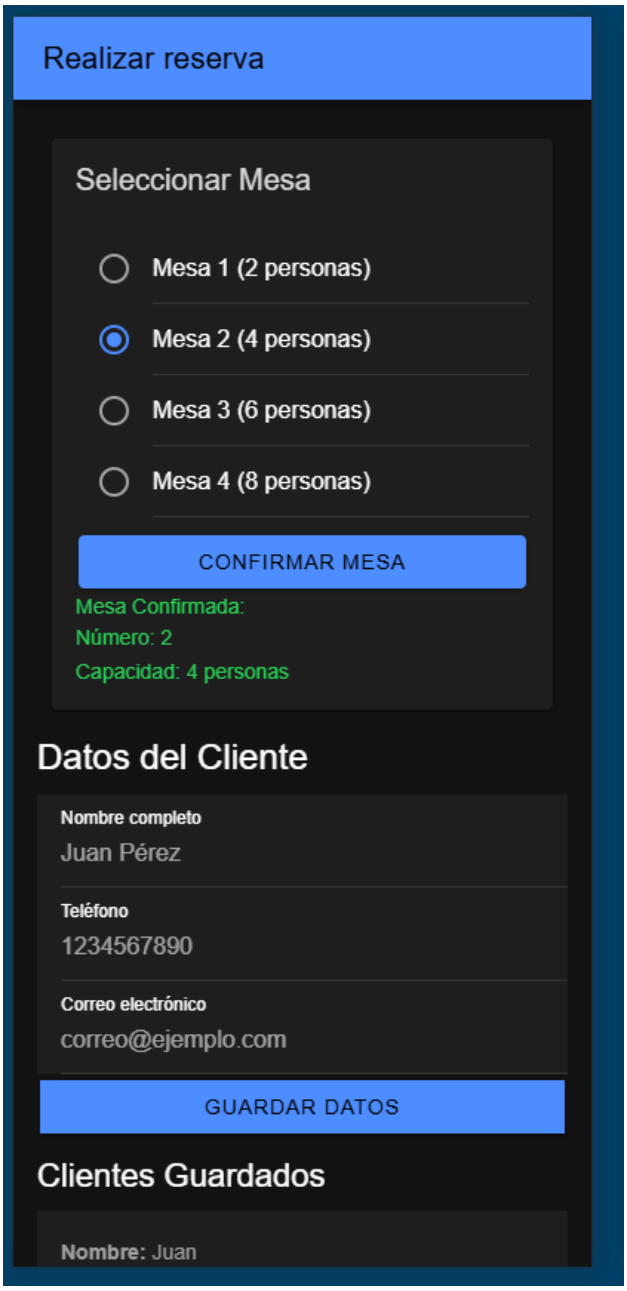
**Tecnologías Utilizadas:**

- **Angular:** Para la estructura de la página y la integración de componentes.
- **Ionic Framework:** Para los elementos visuales (ion-header, ion-toolbar, ion-content, etc.).
- **Componentes Personalizados:** Los componentes MesaComponent, ClienteComponent y ReservaComponent son reutilizados dentro de esta página.

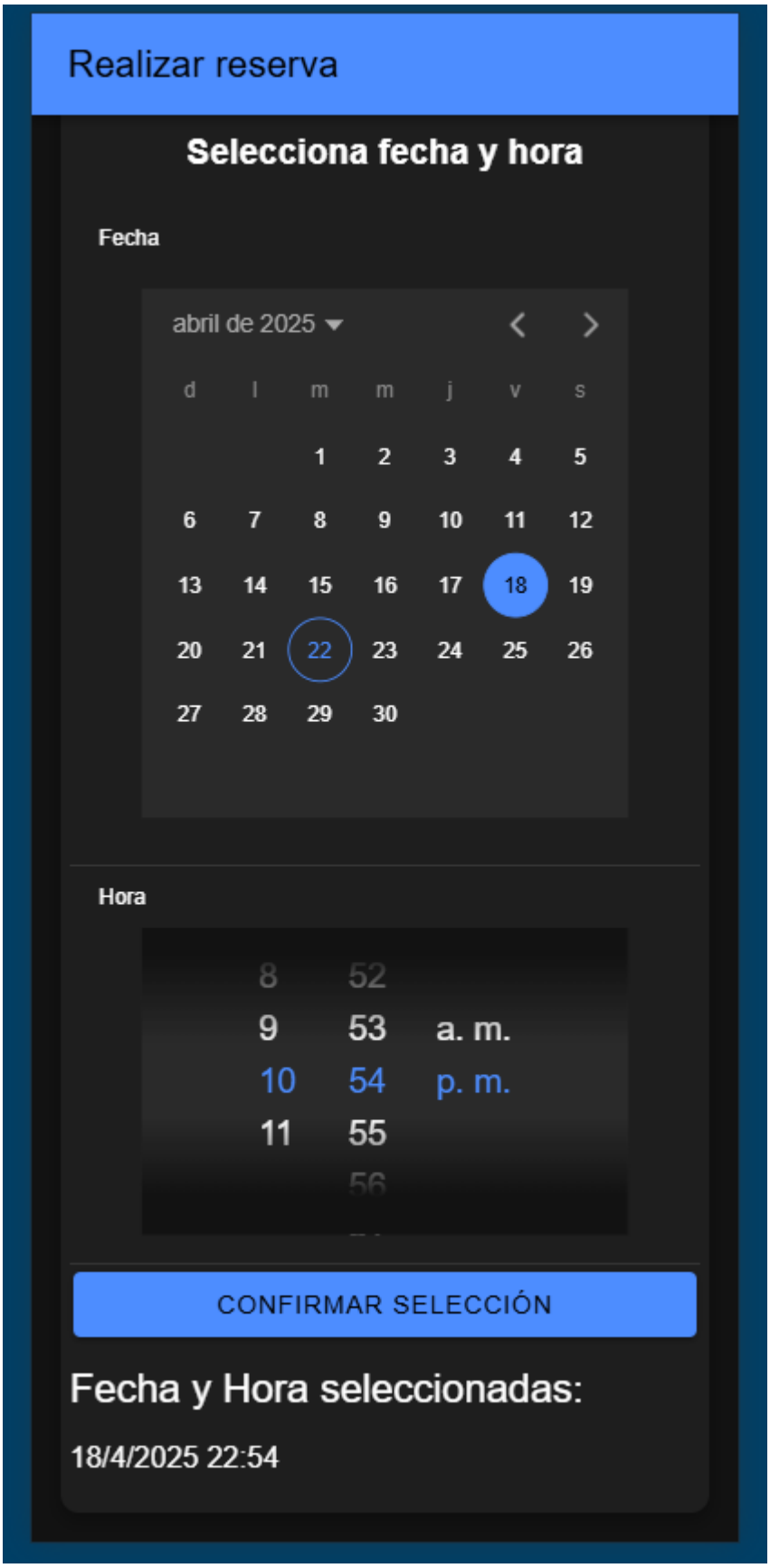
---

**Propósito del Componente:**

El componente HomePage sirve como una interfaz principal para que el usuario pueda realizar una reserva completa. Su diseño modular permite que cada parte del proceso (selección de mesa, datos del cliente, fecha y hora) sea manejada por un componente independiente, lo que facilita la escalabilidad y el mantenimiento del código.


Realizar reserva
Seleccionar Mesa
Mesa 1 (2 personas)
Mesa 2 (4 personas)
Mesa 3 (6 personas)
Mesa 4 (8 personas)
CONFIRMAR MESA
Mesa Confirmada:
Número: 2
Capacidad: 4 personas
Datos del Cliente
Nombre completo
Juan Pérez
Teléfono
1234567890
Correo electrónico
correo@ejemplo.com
GUARDAR DATOS
Clientes Guardados
Nombre: Juan


Realizar reserva
Clientes Guardados
Nombre: Juan
Teléfono: 3123123
Correo: Juan@gmail.com
Selecciona fecha y hora
Fecha
abril de 2025
d l m m j v s
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
Hora
9 53
10 54 a. m.
11 55 p. m.
56
57


Realizar reserva
Selecciona fecha y hora
Fecha
abril de 2025
d l m m j v s
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
Hora
8 52
9 53 a. m.
10 54 p. m.
11 55
56
CONFIRMAR SELECCIÓN
Fecha y Hora seleccionadas:
18/4/2025 22:54