Proyecto bases de datos y programación web:
Gestión de reservas en parcelas

Integrantes: Yerko Yuivar, Alex Rodríguez
Profesor: Benjamín Serrano
Fecha: 31/08/2024

Introducción:

Hoy en día, muchas personas dueñas de parcelas con salón de eventos se ven afectadas por la falta de visibilidad y una mala organización en el proceso de reservas lo cual es un problema común que impacta negativamente la experiencia del cliente y, por ende, los ingresos del negocio.

Problemática:

1. Bajas Ganancias:

- Muchos salones y parcelas no logran llenar sus espacios, lo que significa que se están perdiendo ingresos valiosos. Esto puede poner en riesgo la viabilidad del negocio.

2. Falta de Impulso y Visibilidad:

- Muchos de estos espacios no se conocen lo suficiente, lo que limita su capacidad para atraer a nuevos clientes. Sin una buena estrategia de marketing, pueden quedar en la sombra de algo que pudieron ser. Facebook Marketplace solo les ayuda al inicio pero luego, no logran concretar con seguridad las reservas.

3. Mala Organización:

- La gestión de reservas muchas veces se lleva a cabo de manera manual o con sistemas poco eficientes (via chat en Facebook). Esto puede llevar a confusiones, como doble reservas, y frustrar tanto a los clientes como al personal.

4. Limitaciones Tecnológicas:

- Muchos negocios aún dependen de métodos anticuados para manejar sus reservas, lo que dificulta la comunicación con los usuarios y la administración del espacio.

Objetivo:

Desarrollar una página web con una base de datos integrada que combine MongoDB y MySQL, con el fin de optimizar la gestión de reservas y mejorar la visibilidad online del negocio.
Cómo: Implementaremos una plataforma digital que agilizara los procesos internos de gestión de reservas, permitiendo a los clientes realizar reservas de manera sencilla y rápida.
Para qué: Esto facilitará una experiencia de usuario más fluida, atrayendo a un público más amplio y aumentando la tasa de reservas, lo que se traducirá en un incremento de los ingresos.
Por qué: Adoptar este enfoque moderno no solo mejorará la eficiencia operativa, sino que también generará confianza entre los clientes, al ofrecer un sistema seguro y accesible para gestionar sus reservas.

Requisitos funcionales:

1. Login del usuario:
- Los usuarios deben poder crear una cuenta proporcionando información básica (nombre, apellido, correo electrónico y contraseña)

- Debe tener la opción de iniciar sesión, cerrar sesión y recuperar contraseña

2. Reservar fechas:

- Permitir a los usuarios seleccionar una fecha y hora para el evento o arriendo.

- Visualización de disponibilidad de cada parcela.

3. Sistema de pago:

- Integrar métodos de pagos distintos (Paypal, tarjeta de crédito, binance, etc)

- Se debe confirmar el pago y enviar un recibo al correo electrónico.

- En caso de no tener cuenta registrada, se le debe dar al usuario la opción de agregar su correo.

4. Gestión de reservas:

- Cada usuario debe poder ver y gestionar sus reservas (reprogramar, cancelar, etc).

- Cada usuario recibirá una notificación por correo electrónico sobre el estado de su reserva

5. Visualización del centro de eventos

- Mostrar una lista sobre el espacio del salón y/o parcela junto a su capacidad.
- Mostrar otras parcelas disponibles.
- Filtrar búsqueda sobre salones, parcelas, o conjunto de ambos.

6. Comentarios y valoraciones:
   - Permitir a los usuarios dejar comentarios y valoraciones sobre el recinto, salón y cocina.

7. Soporte al cliente:

   - Sección de preguntas frecuentes (FAQ)

   - Opción de contacto para consultas generales.

8. Administración de parcelas/salones:

- Una interfaz para que los administradores puedan agregar, editar o eliminar información sobre las parcelas/salas.

- Gestión de usuario y reservas desde un panel de control

Requisitos no funcionales:

1. Usabilidad:

- La interfaz de la página web debe ser intuitiva y fácil de usar para todos los usuarios, independiente de su nivel técnico.

2. Rendimiento:

- La página debe cargar al menos en 4 segundos para conexiones estándares. (Según un estudio de Google, la probabilidad de rebote aumenta un 32% cuando el tiempo de carga pasa de 1 a 3 segundos.

- Capacidad para manejar 300 usuarios simultáneos sin perder rendimiento (Debido que eventos especiales o promociones, el tráfico puede aumentar significativamente)

3. Seguridad:
- Implementar https para asegurar la transmisión de datos.
- Protección contra ataques comunes (SQL injection).

4. Compatibilidad:
- La página debe ser compatible con móviles, ajustando el diseño necesario.
- La página podrá utilizarse desde cualquier navegador existente a la fecha.

5. Escalabilidad:
- La arquitectura del sistema debe permitir agregar nuevas funciones a la página web, de manera que se puedan agregar nuevos datos de parcela, nuevos vendedores, sin necesidad de reestructurar el código

6. Mantenimiento:
- El sistema debe estar diseñado para que el mantenimiento sea sencillo.
- El sistema debe mantener actualizaciones regulares sin interrupciones significativas.

7. Documentacion:
- Proporcionar documentación de la parcela y los salones para que los usuarios puedan verificar la veracidad del dueño del recinto.
- Se debe proporcionar una guía de cómo utilizar la página, ya sea para administradores como usuarios.

Diagrama casos de uso solución general

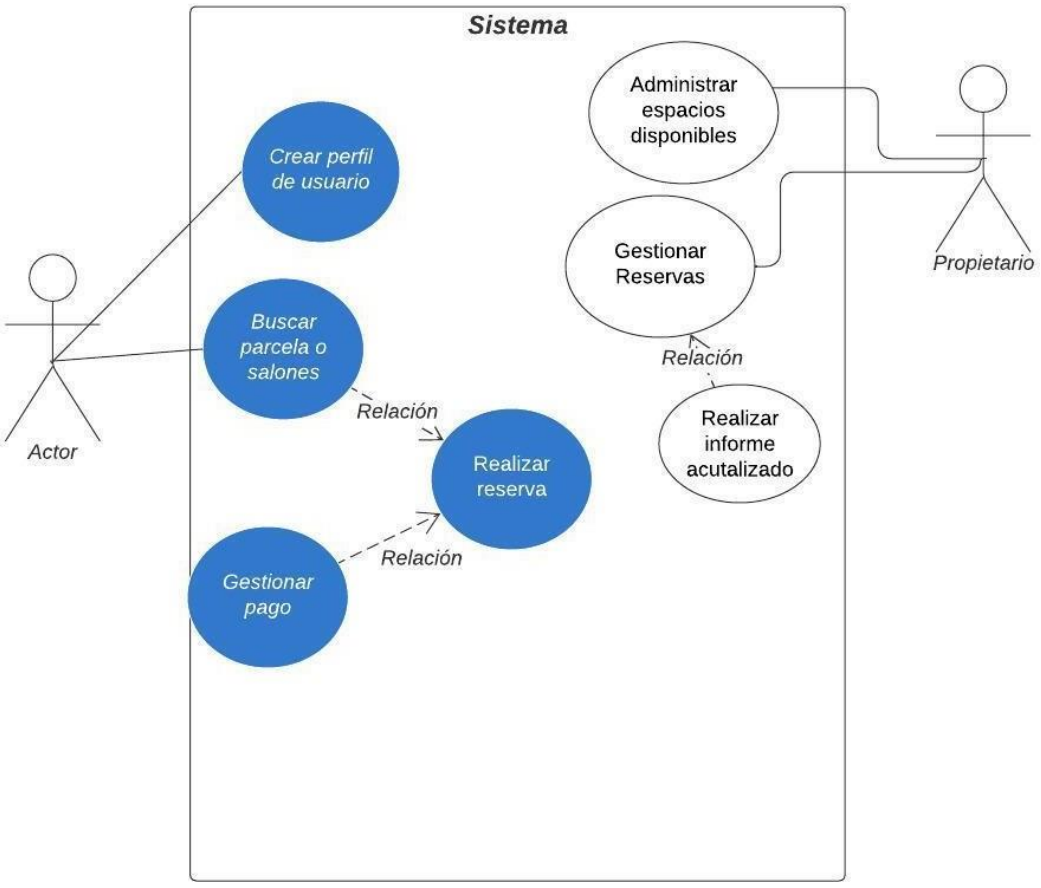
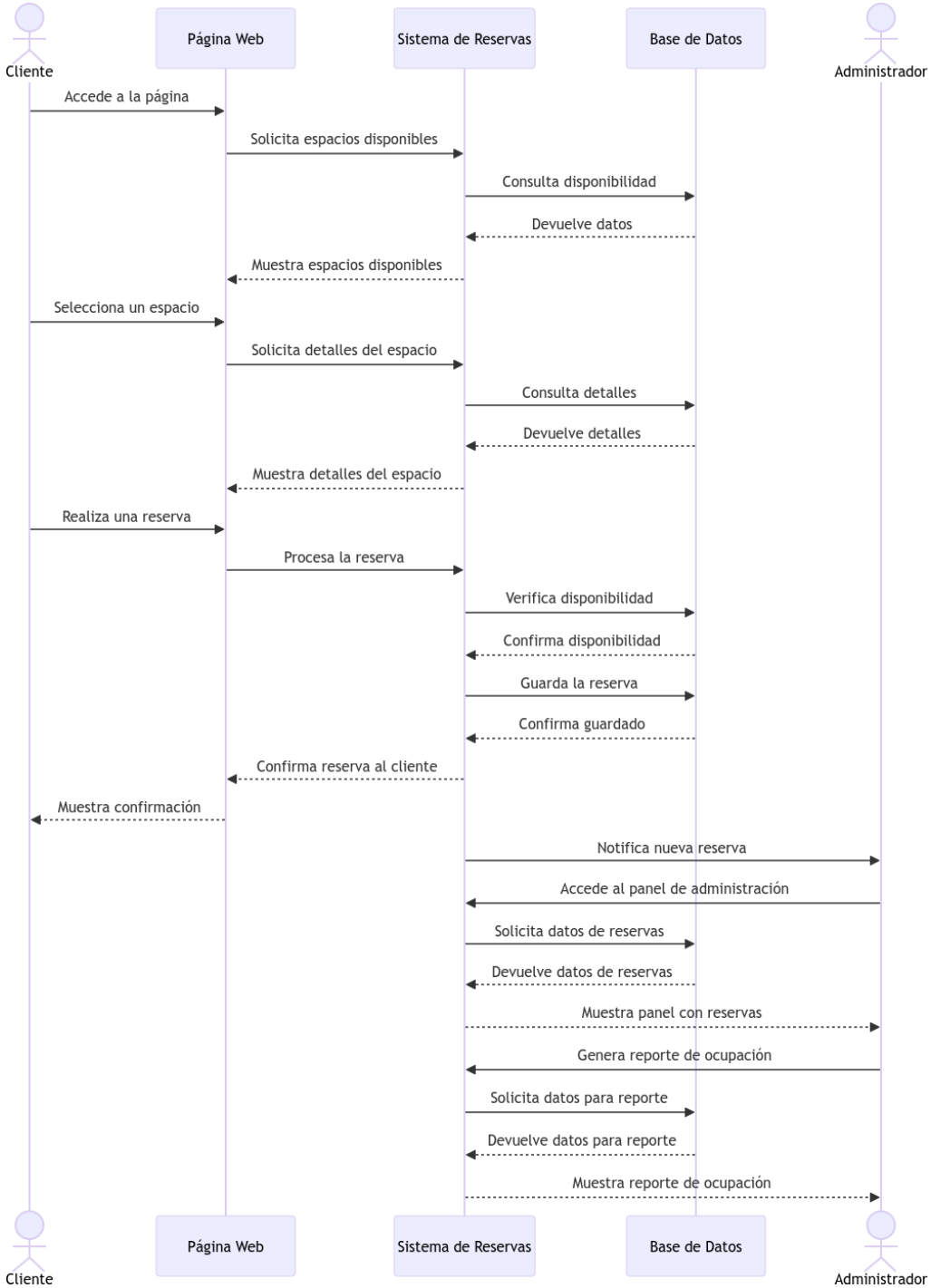
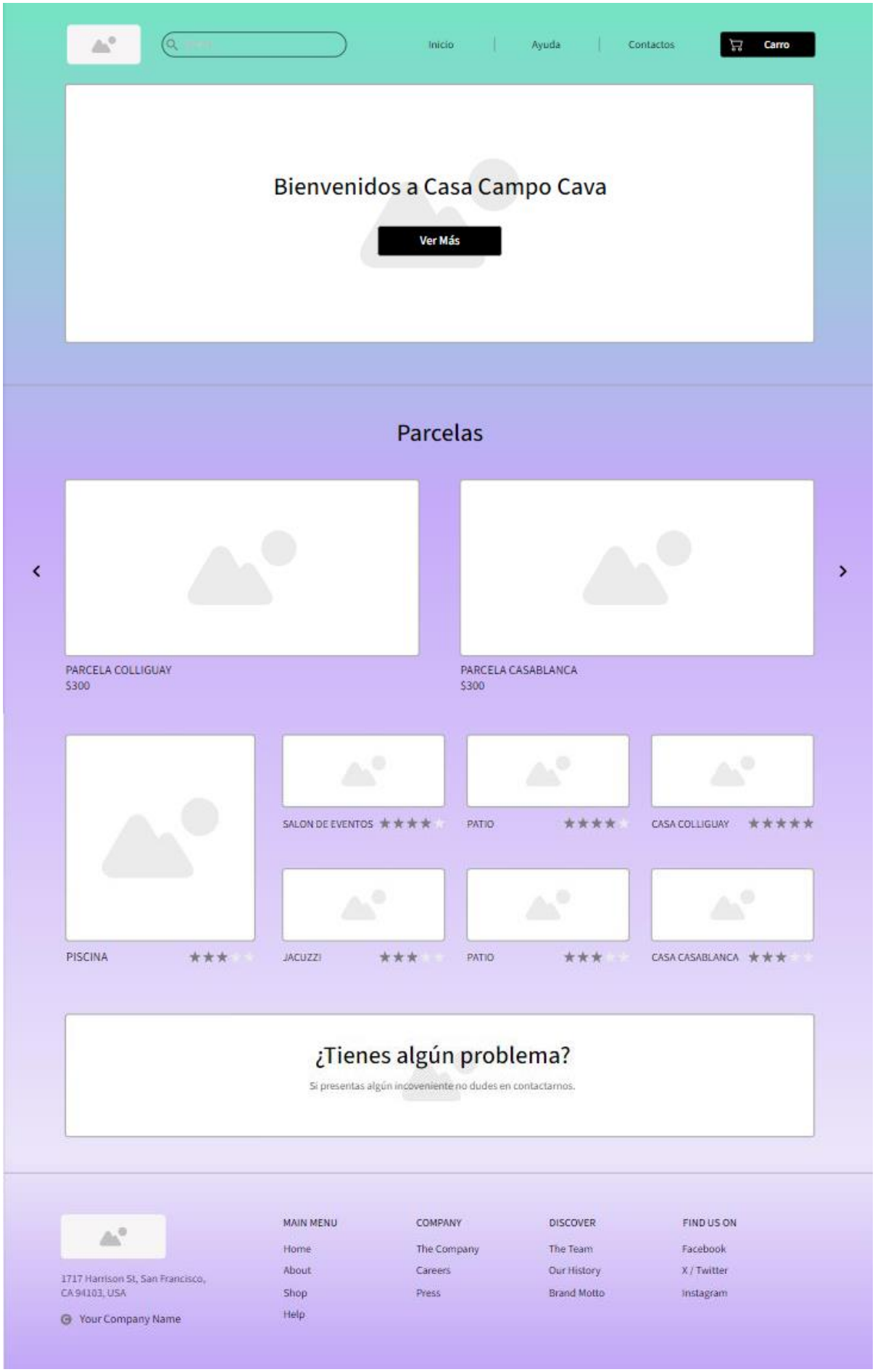


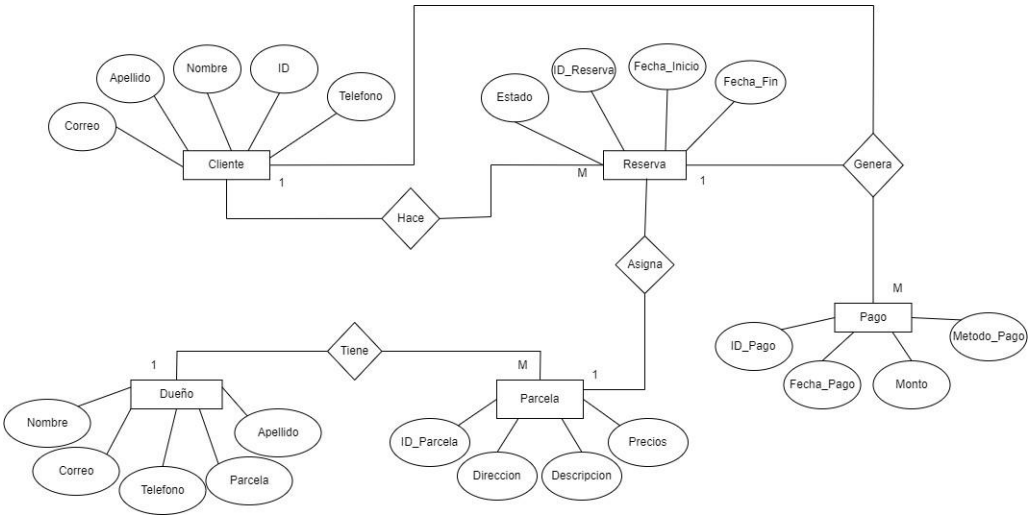
Diagrama de secuencia de la solución



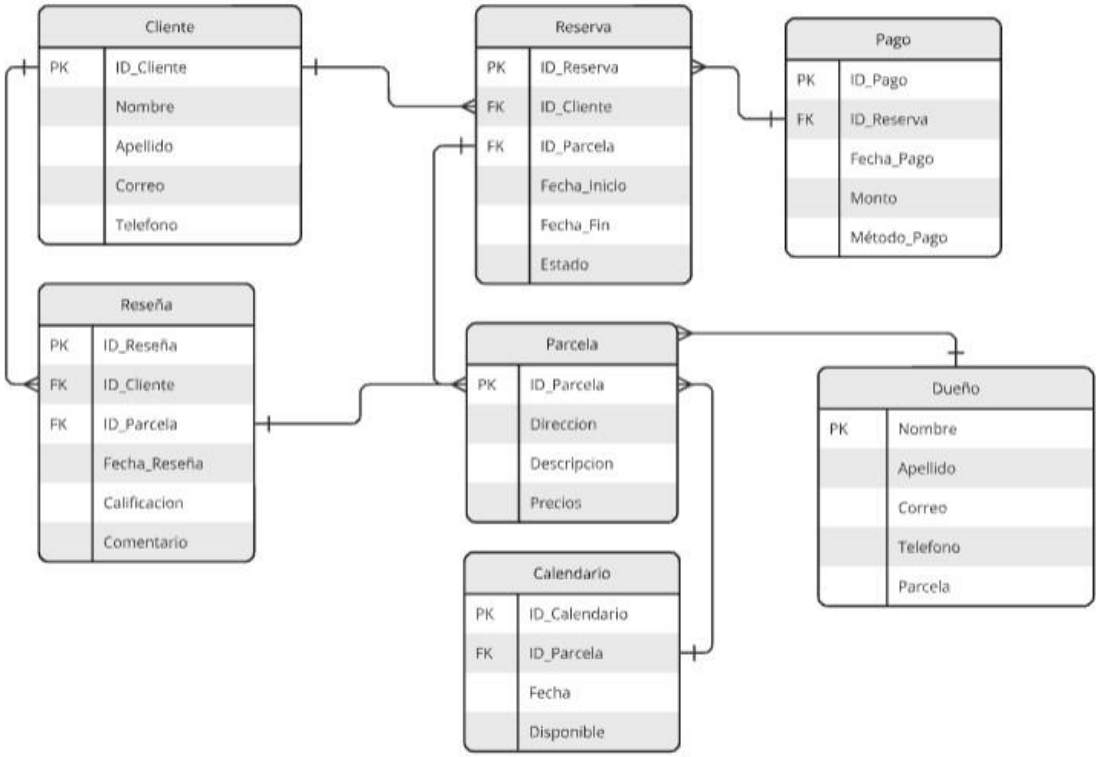
Mockups (front-end)



Modelo Entidad Relación



Modelo Relacional



1. Agregar una nueva columna a la tabla Cliente

```
mysql> ALTER TABLE Cliente ADD Fecha_Nacimiento DATE;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe cliente;
+------------------+-------------+------+-----+---------+----------------+
| Field            | Type        | Null | Key | Default | Extra          |
+------------------+-------------+------+-----+---------+----------------+
| ID_Cliente       | int         | NO   | PRI | NULL    | auto_increment |
| Nombre           | varchar(50) | YES  |     | NULL    |                |
| Apellido         | varchar(50) | YES  |     | NULL    |                |
| Correo           | varchar(100)| YES  |     | NULL    |                |
| Telefono         | varchar(20) | YES  |     | NULL    |                |
| Fecha_Nacimiento | date        | YES  |     | NULL    |                |
+------------------+-------------+------+-----+---------+----------------+
```

2. Modificar el tipo de dato de la columna Telefono en la tabla Dueño

```
mysql> ALTER TABLE Dueno MODIFY Telefono VARCHAR(15);
Query OK, 3 rows affected (0.03 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> describe dueno;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| Nombre   | varchar(50)  | YES  |     | NULL    |       |
| Apellido | varchar(50)  | YES  |     | NULL    |       |
| Correo   | varchar(100) | YES  |     | NULL    |       |
| Telefono | varchar(15)  | YES  |     | NULL    |       |
| Parcela  | int          | YES  | MUL | NULL    |       |
+----------+--------------+------+-----+---------+-------+
```

3. Mostrar la información de los clientes junto con las parcelas que han reservado

```
mysql> SELECT Cliente.Nombre, Cliente.Apellido, Parcela.Direccion, Reserva.Fecha_Inicio, Reserva.Fecha_Fin
    -> FROM Cliente
    -> JOIN Reserva ON Cliente.ID_Cliente = Reserva.ID_Cliente
    -> JOIN Parcela ON Reserva.ID_Parcela = Parcela.ID_Parcela;
+--------+----------+---------------------+--------------+------------+
| Nombre | Apellido | Direccion           | Fecha_Inicio | Fecha_Fin  |
+--------+----------+---------------------+--------------+------------+
| Juan   | Pérez    | Calle Falsa 123     | 2024-10-05   | 2024-10-10 |
| Maria  | González | Av. Siempre Viva 742 | 2024-10-07   | 2024-10-12 |
| Luis   | Ramírez  | Camino Verde 1      | 2024-10-09   | 2024-10-14 |
+--------+----------+---------------------+--------------+------------+
```

4. Mostrar la información de los pagos junto con el estado de las reservas correspondientes

```
mysql> SELECT Pago.ID_Pago, Pago.Monto, Pago.Metodo_Pago, Reserva.Estado
    -> FROM Pago
    -> JOIN Reserva ON Pago.ID_Reserva = Reserva.ID_Reserva;
+---------+--------+-----------------------+------------+
| ID_Pago | Monto  | Metodo_Pago           | Estado     |
+---------+--------+-----------------------+------------+
|       1 | 100.50 | Tarjeta de Crédito    | Confirmada |
|       2 | 150.00 | Paypal                | Pendiente  |
|       3 | 200.75 | Transferencia Bancaria | Cancelada  |
+---------+--------+-----------------------+------------+
```

5. Actualizar el estado de una reserva.

```
mysql> UPDATE Reserva
    -> SET Estado = 'Confirmada'
    -> WHERE ID_Reserva = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from reserva;
+------------+------------+------------+--------------+------------+------------+
| ID_Reserva | ID_Cliente | ID_Parcela | Fecha_Inicio | Fecha_Fin  | Estado     |
+------------+------------+------------+--------------+------------+------------+
|          1 |          1 |          1 | 2024-10-05   | 2024-10-10 | Confirmada |
|          2 |          2 |          2 | 2024-10-07   | 2024-10-12 | Confirmada |
|          3 |          3 |          3 | 2024-10-09   | 2024-10-14 | Cancelada  |
+------------+------------+------------+--------------+------------+------------+
```

6. Actualizar la calificación de una reseña.

```
mysql> UPDATE Resena
    -> SET Calificacion = 4, Comentario = 'Buen lugar, pero mejorable.'
    -> WHERE ID_Resena = 3;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from resena;
+-----------+------------+------------+-------------+--------------+-----------------------------------------+
| ID_Resena | ID_Cliente | ID_Parcela | Fecha_Resena | Calificacion | Comentario                              |
+-----------+------------+------------+-------------+--------------+-----------------------------------------+
|         1 |          1 |          1 | 2024-10-15   |            5 | Excelente experiencia. Muy recomendable. |
|         2 |          2 |          2 | 2024-10-20   |            4 | Muy buen lugar, aunque un poco ruidoso.  |
|         3 |          3 |          3 | 2024-10-25   |            4 | Buen lugar, pero mejorable.              |
+-----------+------------+------------+-------------+--------------+-----------------------------------------+
```

7. Insertar un nuevo cliente.

```
mysql> INSERT INTO Cliente (Nombre, Apellido, Correo, Telefono)
    -> VALUES ('Carlos', 'Fernandez', 'carlos.fernandez@mail.com', '555-9090');
Query OK, 1 row affected (0.00 sec)

mysql> select * from cliente;
+------------+--------+-----------+---------------------------+----------+------------------+
| ID_Cliente | Nombre | Apellido  | Correo                    | Telefono | Fecha_Nacimiento |
+------------+--------+-----------+---------------------------+----------+------------------+
|          1 | Juan   | Pérez     | juan.perez@mail.com       | 555-1234 | NULL             |
|          2 | Maria  | González  | maria.gonzalez@mail.com   | 555-5678 | NULL             |
|          3 | Luis   | Ramírez   | luis.ramirez@mail.com     | 555-8765 | NULL             |
|          4 | Carlos | Fernandez | carlos.fernandez@mail.com | 555-9090 | NULL             |
+------------+--------+-----------+---------------------------+----------+------------------+
```

8. Insertar una nueva parcela.

```
mysql> INSERT INTO Parcela (Direccion, Descripcion, Precios)
    -> VALUES ('Calle Los Robles 55', 'Parcela con piscina y jardín', 250.00);
Query OK, 1 row affected (0.00 sec)

mysql> select * from parcela;
+------------+----------------------+------------------------------+---------+
| ID_Parcela | Direccion            | Descripcion                  | Precios |
+------------+----------------------+------------------------------+---------+
|          1 | Calle Falsa 123      | Parcela con vistas al lago   |  100.50 |
|          2 | Av. Siempre Viva 742 | Parcela en la montaña        |  150.00 |
|          3 | Camino Verde 1       | Parcela con acceso a la playa |  200.75 |
|          4 | Calle Los Robles 55  | Parcela con piscina y jardín |  250.00 |
+------------+----------------------+------------------------------+---------+
```

9. Eliminar una reseña específica.

```
mysql> DELETE FROM Resena
    -> WHERE ID_Resena = 1;
Query OK, 1 row affected (0.00 sec)

mysql> select * from resena;
+-----------+------------+------------+--------------+-------------+-------------------------------------+
| ID_Resena | ID_Cliente | ID_Parcela | Fecha_Resena | Calificacion | Comentario                          |
+-----------+------------+------------+--------------+-------------+-------------------------------------+
|         2 |          2 |          2 | 2024-10-20   |           4 | Muy buen lugar, aunque un poco ruidoso. |
|         3 |          3 |          3 | 2024-10-25   |           4 | Buen lugar, pero mejorable.         |
+-----------+------------+------------+--------------+-------------+-------------------------------------+
```

10. Eliminar una tabla (por ejemplo, la tabla Calendario)

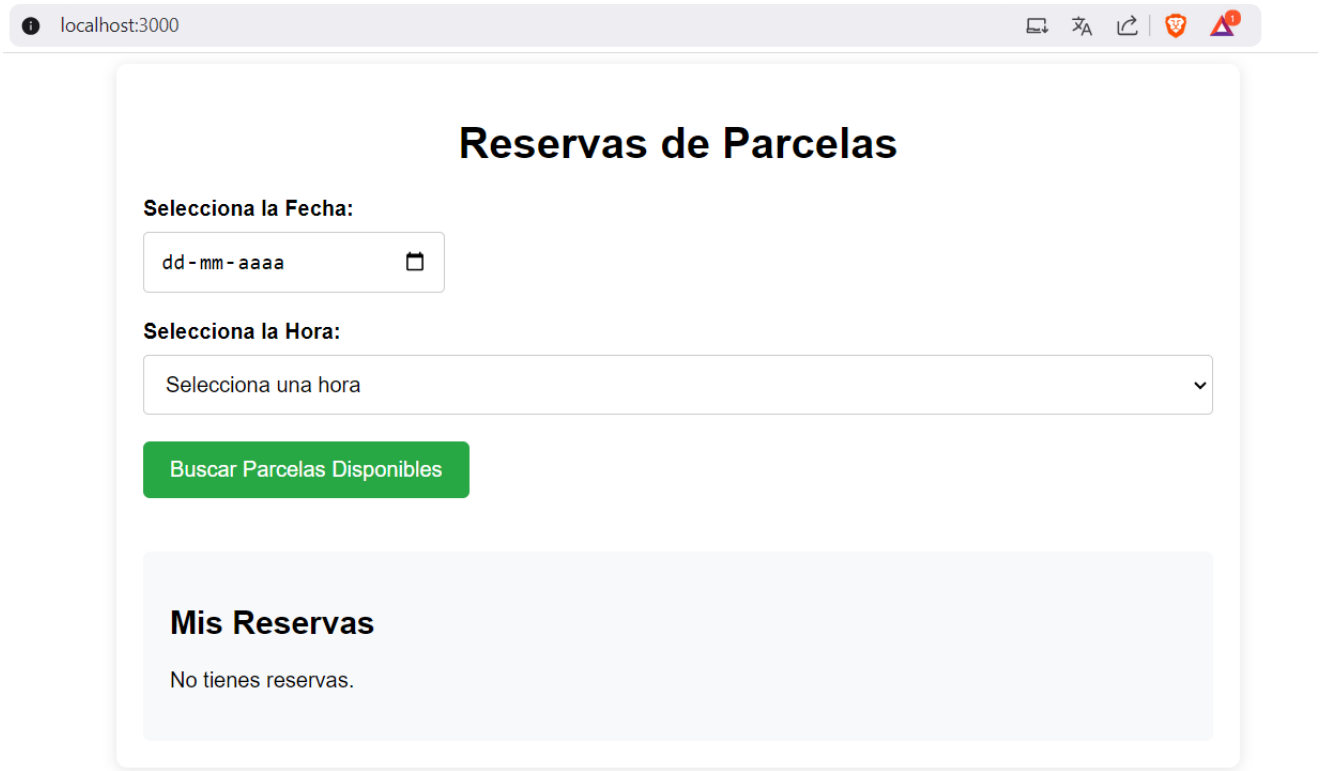
```
mysql> DROP TABLE Calendario;
Query OK, 0 rows affected (0.01 sec)

mysql> show tables;
+---------------------------+
| Tables_in_sistema_reserva |
+---------------------------+
| cliente                   |
| dueno                     |
| pago                      |
| parcela                   |
| resena                    |
| reserva                   |
+---------------------------+
```

Diccionario de Datos

Nombre atributo	Entidad/Relación a la que pertenece	Tipo de atributo	Nulo/No Nulo	En caso de ser compuesto indicar atributos	Descripción
ID_Cliente	Cliente	Entero	No Nulo	N/A	Identificador único del cliente.
Nombre	Cliente	Cadena de texto	No Nulo	N/A	Nombre del cliente.
Apellido	Cliente	Cadena de texto	No Nulo	N/A	Apellido del cliente.
Correo	Cliente	Cadena de texto	No Nulo	N/A	Correo electrónico del cliente.
Teléfono	Cliente	Cadena de texto	No Nulo	N/A	Número de teléfono del cliente.
ID_Reserva	Reserva	Entero	No Nulo	N/A	Identificador único de la reserva.
ID_Cliente	Reserva	Entero	No Nulo	N/A	Identificador del cliente que realiza la reserva.
ID_Parcela	Reserva	Entero	No Nulo	N/A	Identificador de la parcela reservada.
Fecha_Inicio	Reserva	Fecha	No Nulo	N/A	Fecha de inicio de la reserva.
Fecha_Fin	Reserva	Fecha	No Nulo	N/A	Fecha de fin de la reserva.
Estado	Reserva	Cadena de texto	No Nulo	N/A	Estado de la reserva (ej. Confirmada, Pendiente).
ID_Parcela	Parcela	Entero	No Nulo	N/A	Identificador único de la parcela.
Dirección	Parcela	Cadena de texto	No Nulo	N/A	Dirección de la parcela.
Descripción	Parcela	Cadena de texto	Nulo	N/A	Descripción de la parcela.
Precios	Parcela	Decimal	No Nulo	N/A	Precios de la parcela.
ID_Reseña	Reseña	Entero	No Nulo	N/A	Identificador único de la reseña.
ID_Cliente	Reseña	Entero	No Nulo	N/A	Identificador del cliente que realiza la reseña.
ID_Parcela	Reseña	Entero	No Nulo	N/A	Identificador de la parcela reseñada.
Fecha_Reseña	Reseña	Fecha	No Nulo	N/A	Fecha en que se realizó la reseña.
Calificación	Reseña	Entero	No Nulo	N/A	Calificación dada a la parcela.
Comentario	Reseña	Cadena de texto	Nulo	N/A	Comentario de la reseña.
ID_Calendario	Calendario	Entero	No Nulo	N/A	Identificador único del calendario.
ID_Parcela	Calendario	Entero	No Nulo	N/A	Identificador de la parcela correspondiente.
Fecha	Calendario	Fecha	No Nulo	N/A	Fecha específica registrada en el calendario.
Disponible	Calendario	Booleano	No Nulo	N/A	Indicador de disponibilidad para esa fecha (Sí/No).
ID_Pago	Pago	Entero	No Nulo	N/A	Identificador único del pago.
ID_Reserva	Pago	Entero	No Nulo	N/A	Identificador de la reserva relacionada con el pago.
Fecha_Pago	Pago	Fecha	No Nulo	N/A	Fecha en que se realizó el pago.
Monto	Pago	Decimal	No Nulo	N/A	Monto del pago.
Método_Pago	Pago	Cadena de texto	No Nulo	N/A	Método de pago utilizado (ej. Tarjeta, Transferencia).
Nombre	Dueño	Cadena de texto	No Nulo	N/A	Nombre del dueño.
Apellido	Dueño	Cadena de texto	No Nulo	N/A	Apellido del dueño.
Correo	Dueño	Cadena de texto	No Nulo	N/A	Correo electrónico del dueño.
Teléfono	Dueño	Cadena de texto	No Nulo	N/A	Número de teléfono del dueño.
Parcela	Dueño	Entero	No Nulo	N/A	Parcela que pertenece al dueño.

Imágenes front:



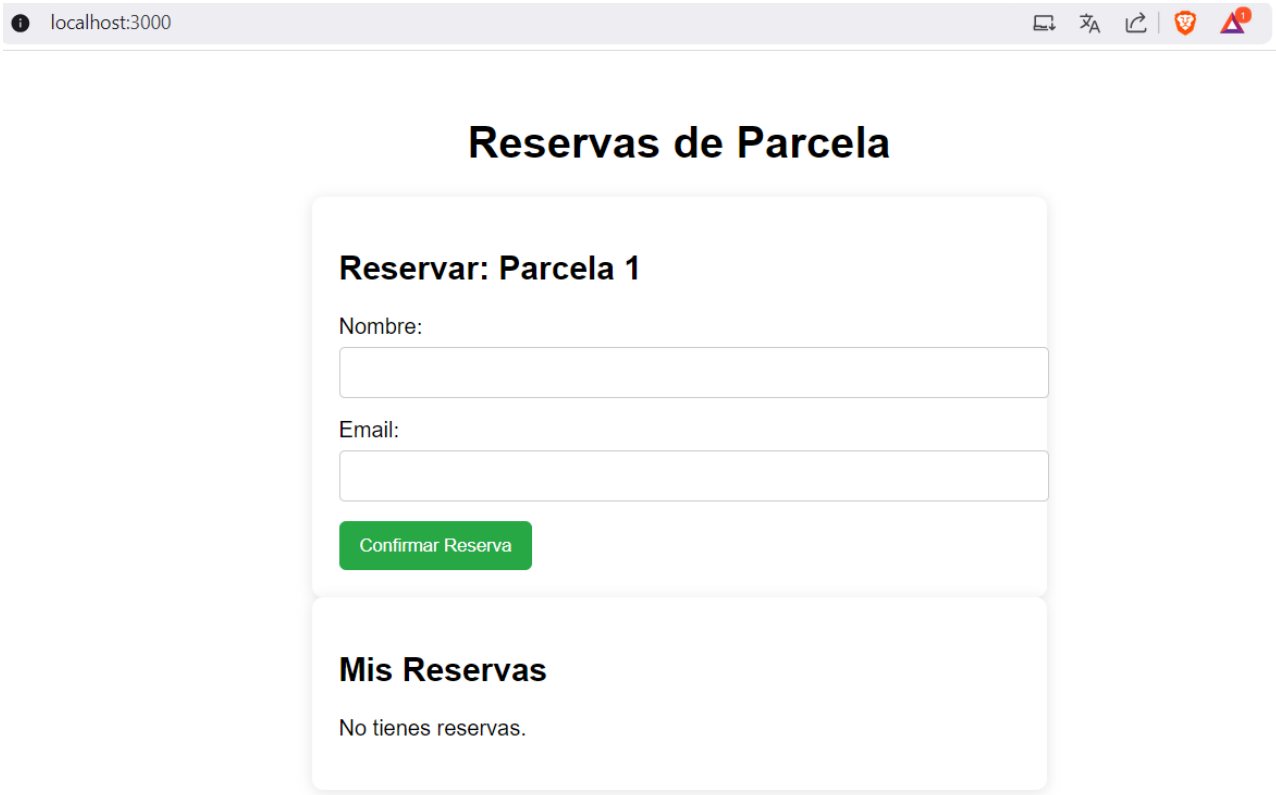
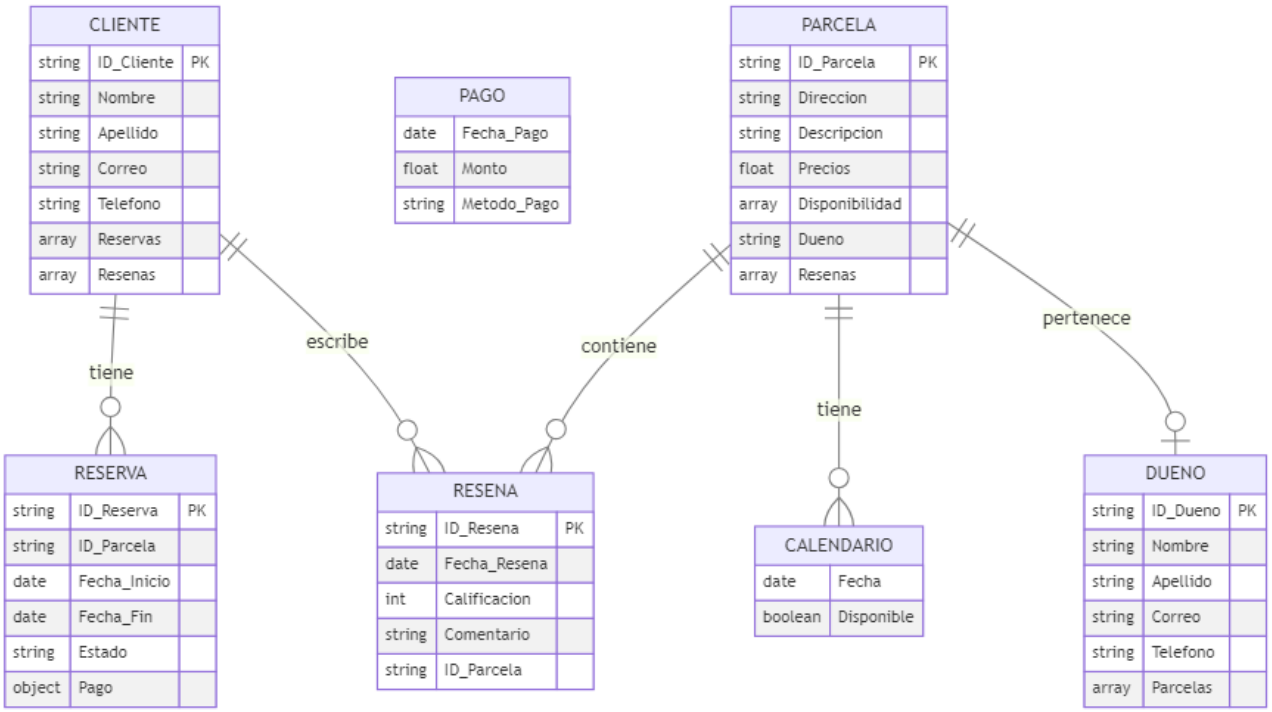


Diagrama NoSql



El diagrama que muestras describe una estructura de datos organizada en entidades como CLIENTE, PARCELA, RESERVA, etc., y sus relaciones. Este diseño define cómo se almacenarán y conectarán los datos. Según las características del diagrama, se parece más a un modelo orientado a **NoSQL** (bases de datos de documentos como MongoDB) debido a su uso de estructuras complejas

Campos complejos como arrays y objetos:

- En CLIENTE, se definen arrays como Reservas y Resenas.
- En RESERVA, hay un campo Pago, que parece un objeto.
- Las bases de datos NoSQL (como MongoDB) permiten almacenar arrays y objetos directamente dentro de un documento. En una base de datos SQL, estas estructuras no se manejan directamente, y es necesario dividirlas en tablas relacionadas.

Relaciones flexibles:

- Aunque hay relaciones entre las entidades (como CLIENTE tiene RESERVAS), estas no parecen tener claves foráneas tradicionales (como en SQL). En cambio, se usan referencias implícitas, lo que sugiere que los datos podrían estar embebidos dentro de documentos.

Modelado jerárquico:

- Por ejemplo, una parcela (PARCELA) tiene una lista de disponibilidad (Disponibilidad) y reseñas (Resenas). Esto refleja un enfoque donde los datos relacionados se guardan dentro del documento principal, algo común en bases de datos NoSQL.

Consultas NoSQL y CRUD

Consulta NoSQL: Contar reservas.

```
> db.reservas.countDocuments();
< 6
```

Consulta NoSQL: Eliminar

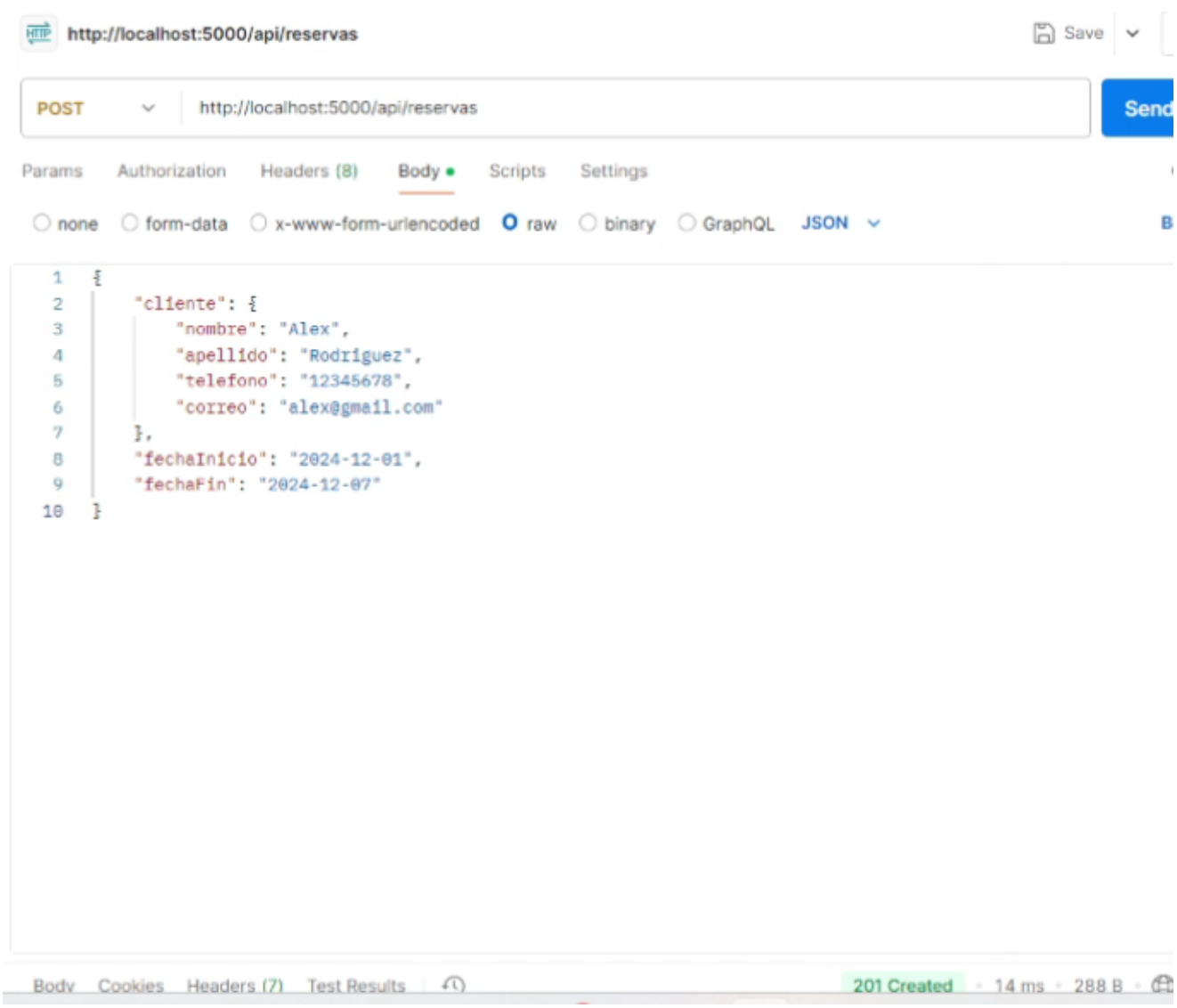
```
> db.reservas.deleteOne({ _id: ObjectId("675be06946916e9f289b8d3b") });
< {
    acknowledged: true,
    deletedCount: 1
  }
```

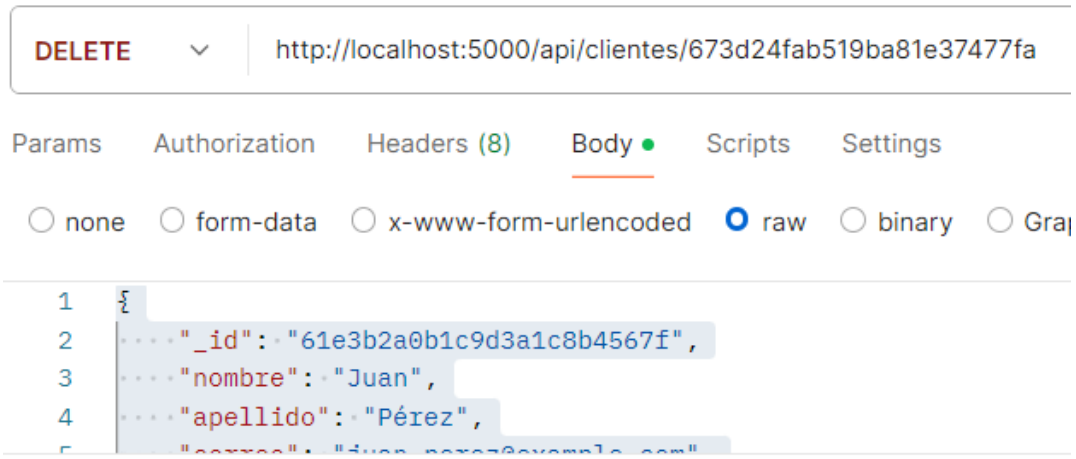
Consulta NoSql: Contar por apellido.

```
> db.reservas.countDocuments({ apellido: "Rodriguez" });
< 0
```

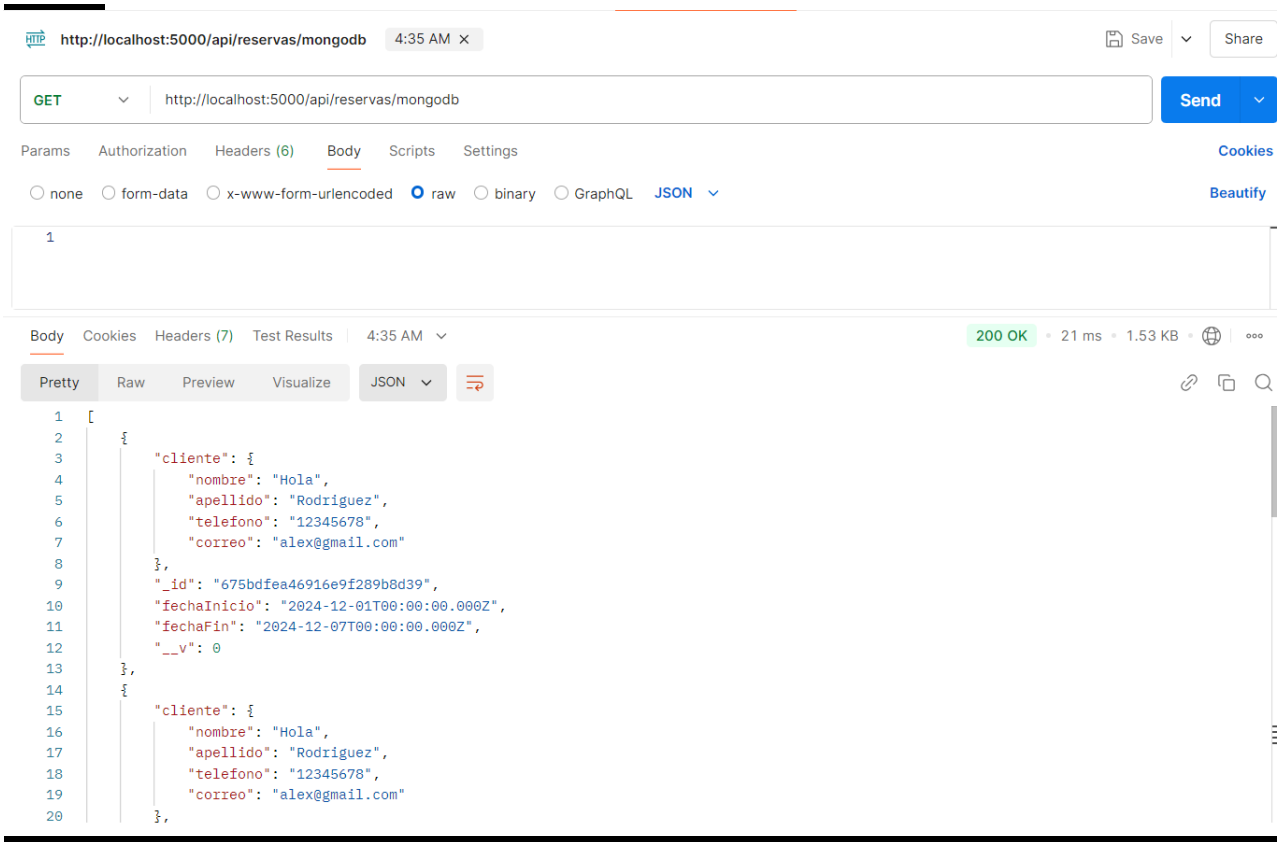
Consulta NoSQL: Eliminar nombres con Hola

```
> db.reservas.deleteMany({ "cliente.nombre": "Hola" });
< {
    acknowledged: true,
    deletedCount: 4
  }
```


http://localhost:5000/api/reservas
Save
POST
http://localhost:5000/api/reservas
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
{
"cliente": {
"nombre": "Alex",
"apellido": "Rodriguez",
"telefono": "12345678",
"correo": "alex@gmail.com"
},
"fechaInicio": "2024-12-01",
"fechaFin": "2024-12-07"
}
Body
Cookies
Headers (7)
Test Results
201 Created
14 ms
288 B


DELETE
http://localhost:5000/api/clientes/673d24fab519ba81e37477fa
Params
Authorization
Headers (8)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
{
"_id": "61e3b2a0b1c9d3a1c8b4567f",
"nombre": "Juan",
"apellido": "Pérez",
Body
Cookies
Headers (8)
Test Results
Pretty
Raw
Preview
Visualize
HTML
Cliente eliminado con éxito

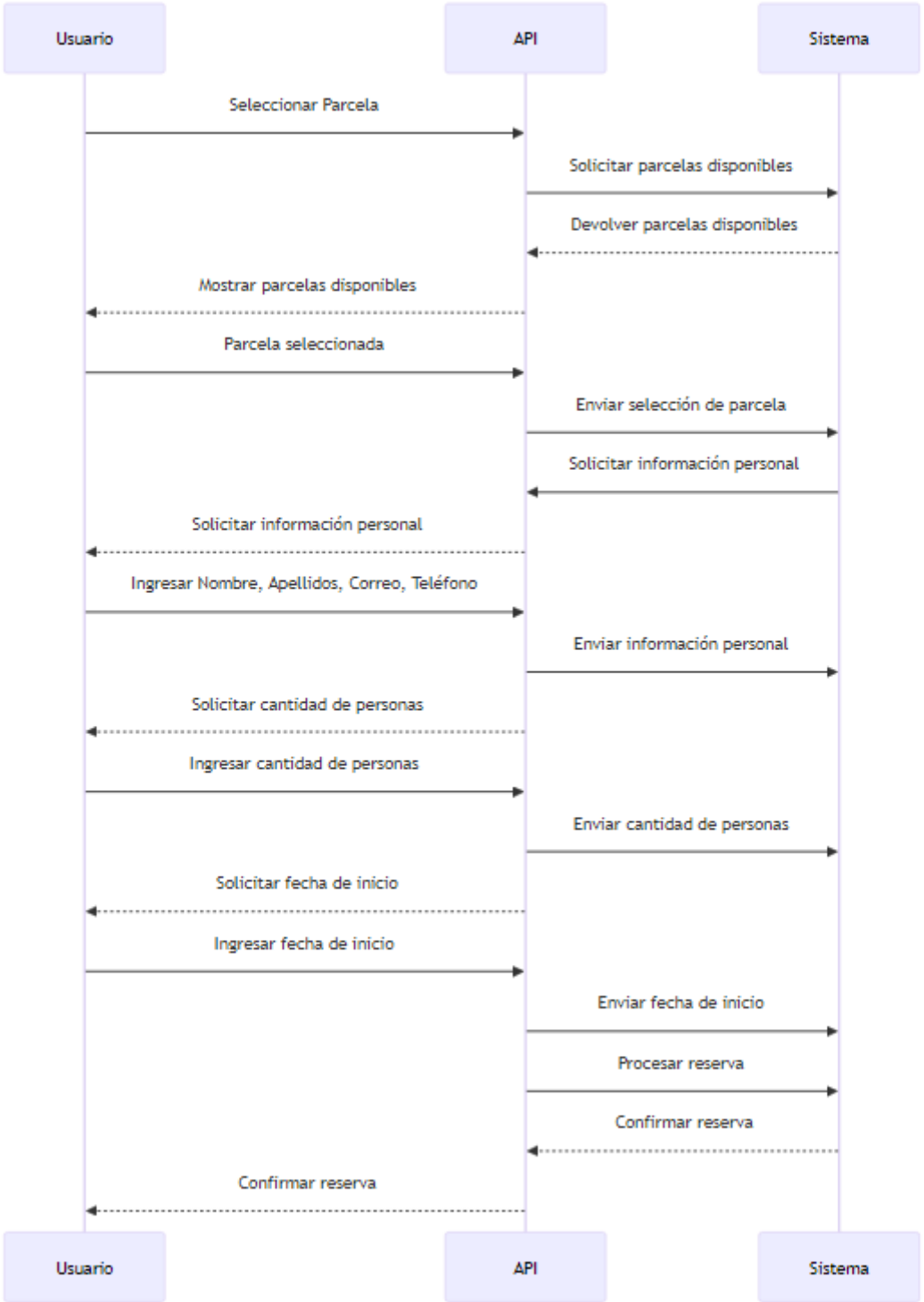
Get:


http://localhost:5000/api/reservas/mongodb
4:35 AM
Save
Share
GET
http://localhost:5000/api/reservas/mongodb
Send
Params
Authorization
Headers (6)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
1
Body
Cookies
Headers (7)
Test Results
4:35 AM
200 OK
21 ms
1.53 KB
Pretty
Raw
Preview
Visualize
JSON
[
{
"cliente": {
"nombre": "Hola",
"apellido": "Rodriguez",
"telefono": "12345678",
"correo": "alex@gmail.com"
},
"_id": "675bdfea46916e9f289b8d39",
"fechaInicio": "2024-12-01T00:00:00.000Z",
"fechaFin": "2024-12-07T00:00:00.000Z",
"__v": 0
},
{
"cliente": {
"nombre": "Hola",
"apellido": "Rodriguez",
"telefono": "12345678",
"correo": "alex@gmail.com"
},

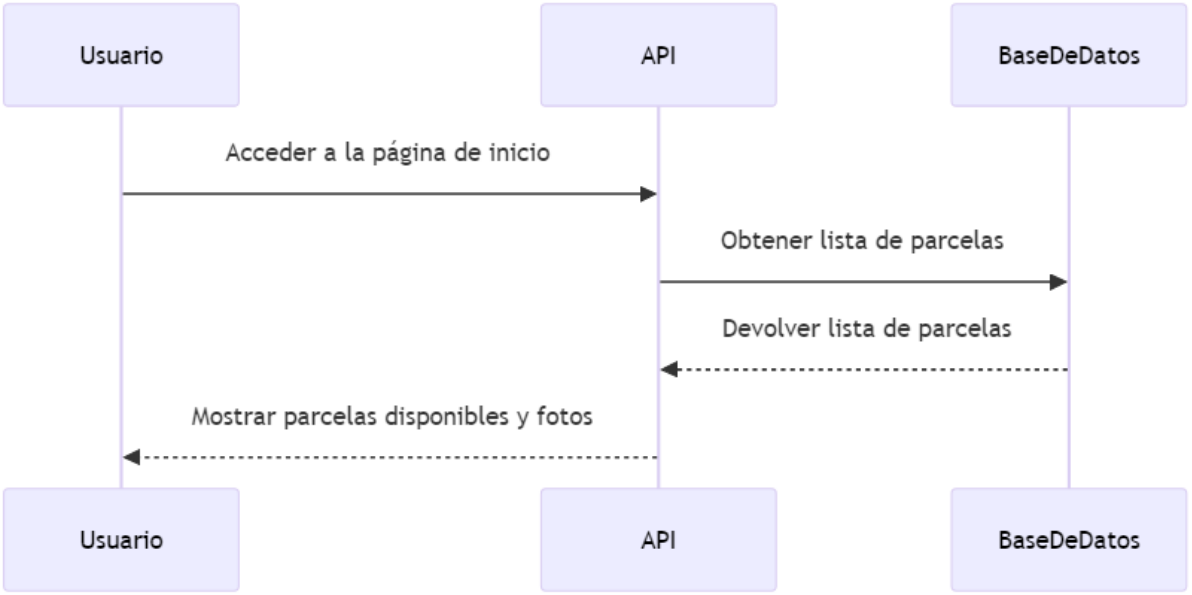
Diagramas de secuencia

Esto depende de la sección donde ingrese el usuario

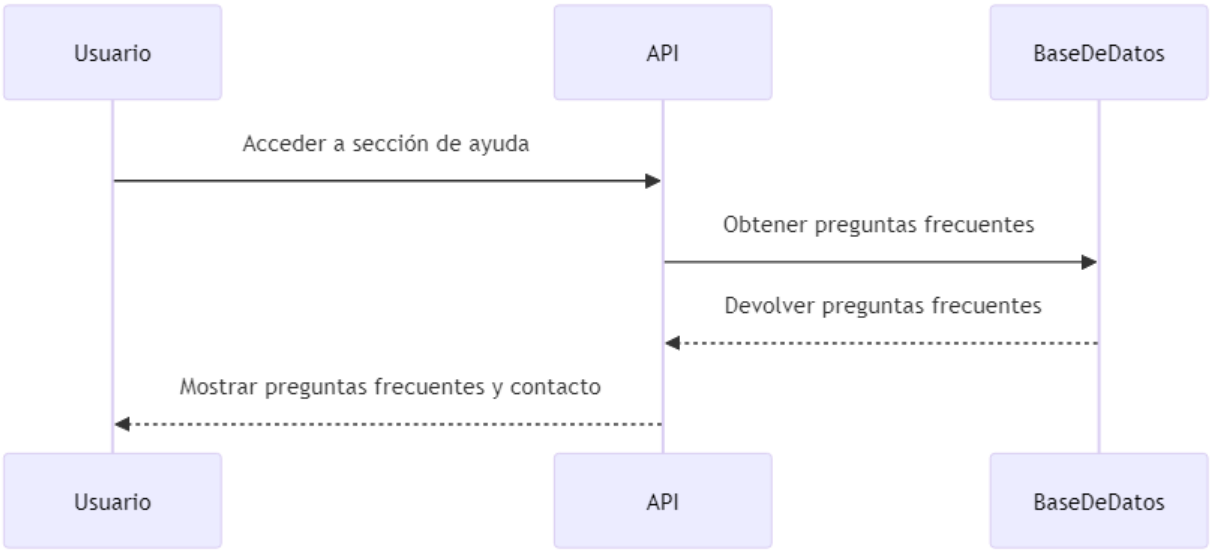
Sección de reservas



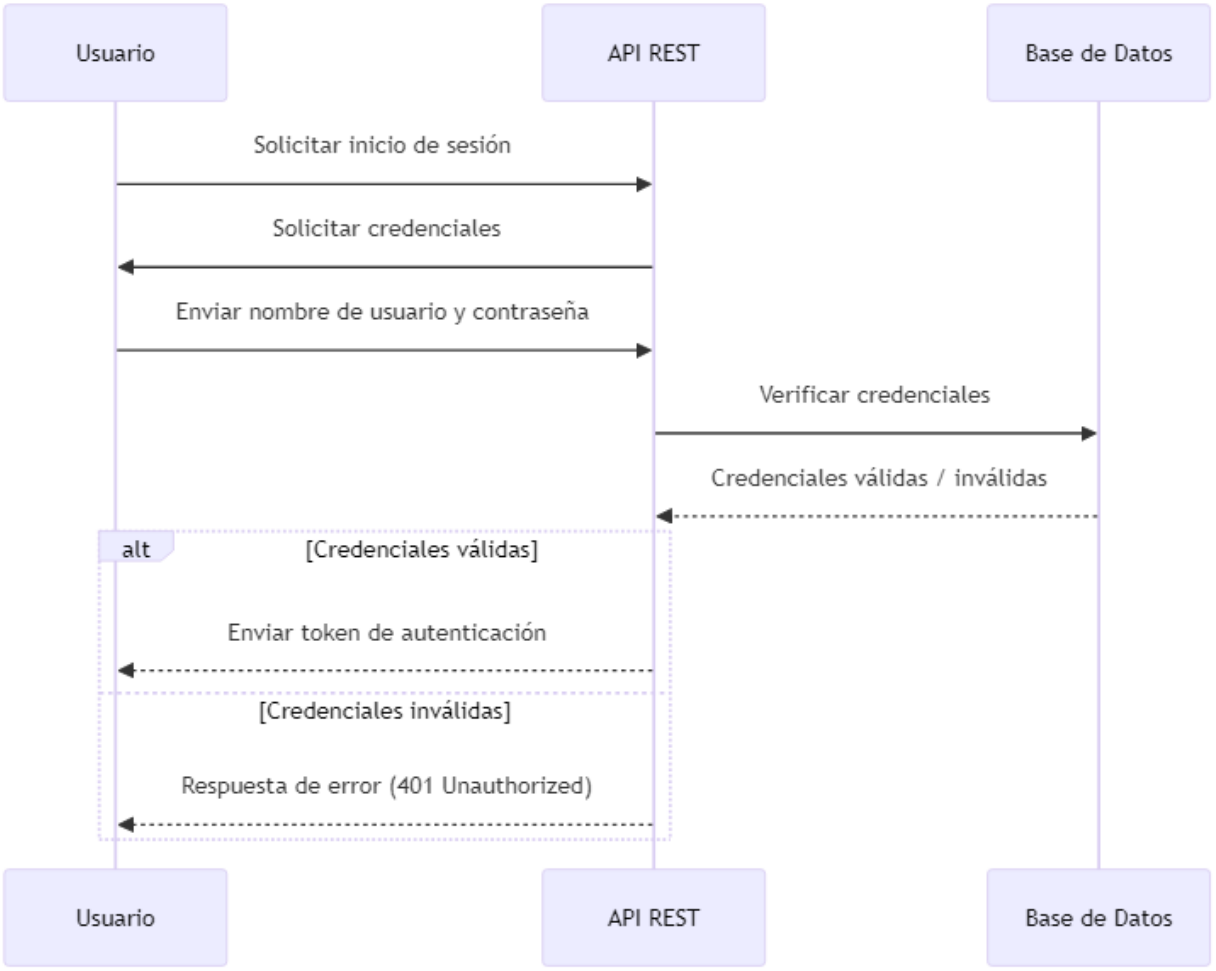
Pagina de inicio



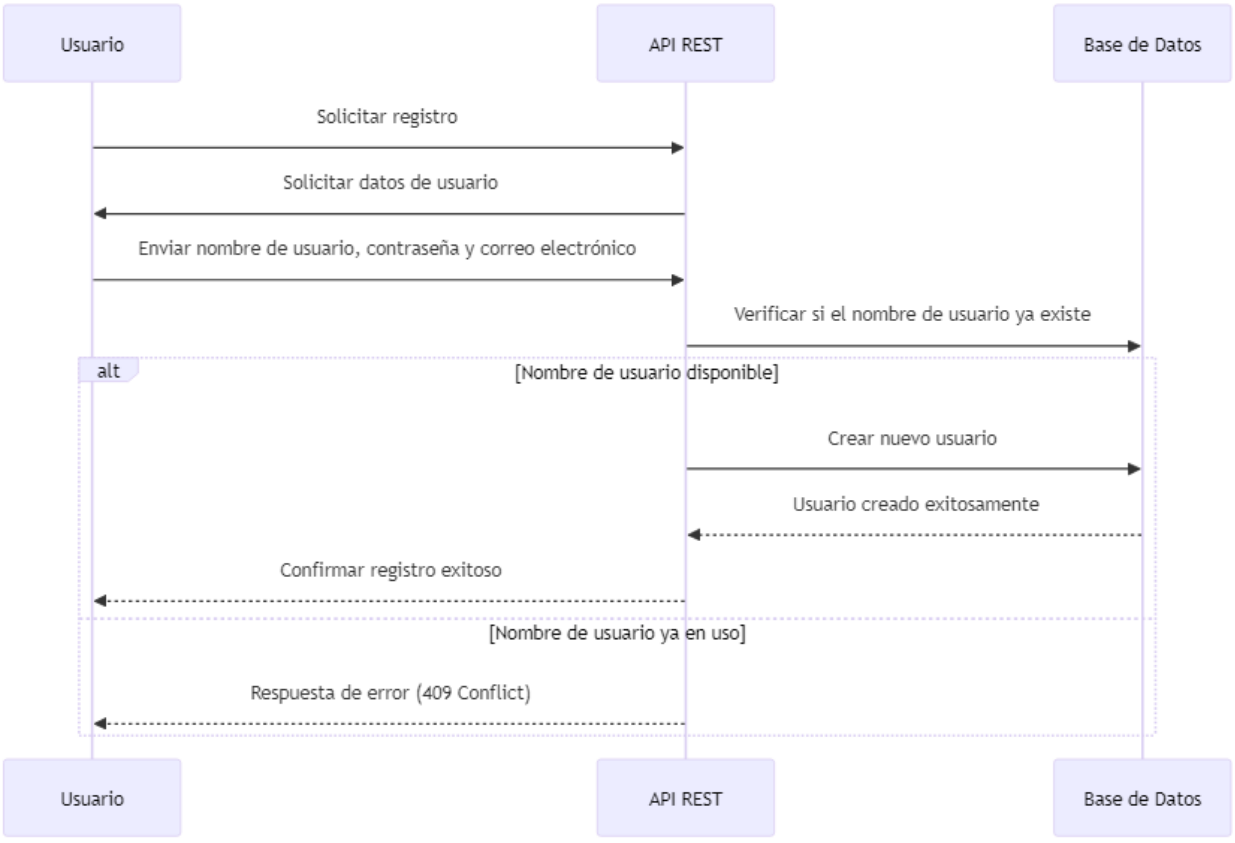
sección de ayuda al usuario



Inicio de sesión:



Registrar usuario:



Documentación de la API REST de Gestión de Reservas

Introducción
La API de Gestión de Reservas permite a los usuarios gestionar reservas de parcelas de forma eficiente. Proporciona un conjunto de endpoints para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) relacionadas con las reservas y los usuarios.

Autenticación
La autenticación se realiza mediante un sistema de registro de usuarios. Los usuarios deben registrarse proporcionando un nombre de usuario y una contraseña. Una vez registrados, pueden iniciar sesión para obtener un token que permite acceder a los recursos protegidos de la API.

Endpoints
1. Obtener Reservas
- **Método**: GET
- **Endpoint**: /reservas
- **Parámetros de entrada**:
  - usuarioId (opcional): Filtra reservas por usuario.
- **Respuesta esperada**:
  - 200 OK: Lista de reservas en formato JSON.
- **Códigos de estado**:
  - 200: Solicitud exitosa.
  - 404: No se encontraron reservas.

2. Crear Reserva
- **Método**: POST
- **Endpoint**: /reservas
- **Parámetros de entrada**:
  - usuarioId: ID del usuario que realiza la reserva.
  - parcelaId: ID de la parcela a reservar.
  - fecha: Fecha de la reserva.
- **Respuesta esperada**:
  - 201 Created: Reserva creada exitosamente.
- **Códigos de estado**:
  - 201: Reserva creada.
  - 400: Datos de entrada inválidos.

3. Actualizar Reserva
- **Método**: PUT
- **Endpoint**: /reservas/{id}
- **Parámetros de entrada**:
  - id: ID de la reserva a actualizar.
  - nuevoDatos: Nuevos datos de la reserva.
- **Respuesta esperada**:
  - 200 OK: Reserva actualizada exitosamente.
- **Códigos de estado**:
  - 200: Actualización exitosa.
  - 404: Reserva no encontrada.

4. Eliminar Reserva

- **Método**: DELETE
- **Endpoint**: /reservas/{id}
- **Parámetros de entrada**:
  - id: ID de la reserva a eliminar.
- **Respuesta esperada**:
  - 204 No Content: Reserva eliminada exitosamente.
- **Códigos de estado**:
  - 204: Eliminación exitosa.
  - 404: Reserva no encontrada.

Errores:

- **400 Bad Request: Datos de entrada inválidos.**
- **404 Not Found: Recurso no encontrado.**

Ejemplo de Solicitud para Crear Reserva:

POST /reservas
Content-Type: application/json

```
{
  "usuarioId": "123",
  "parcelaId": "456",
  "fecha": "2024-12-15"
}
```

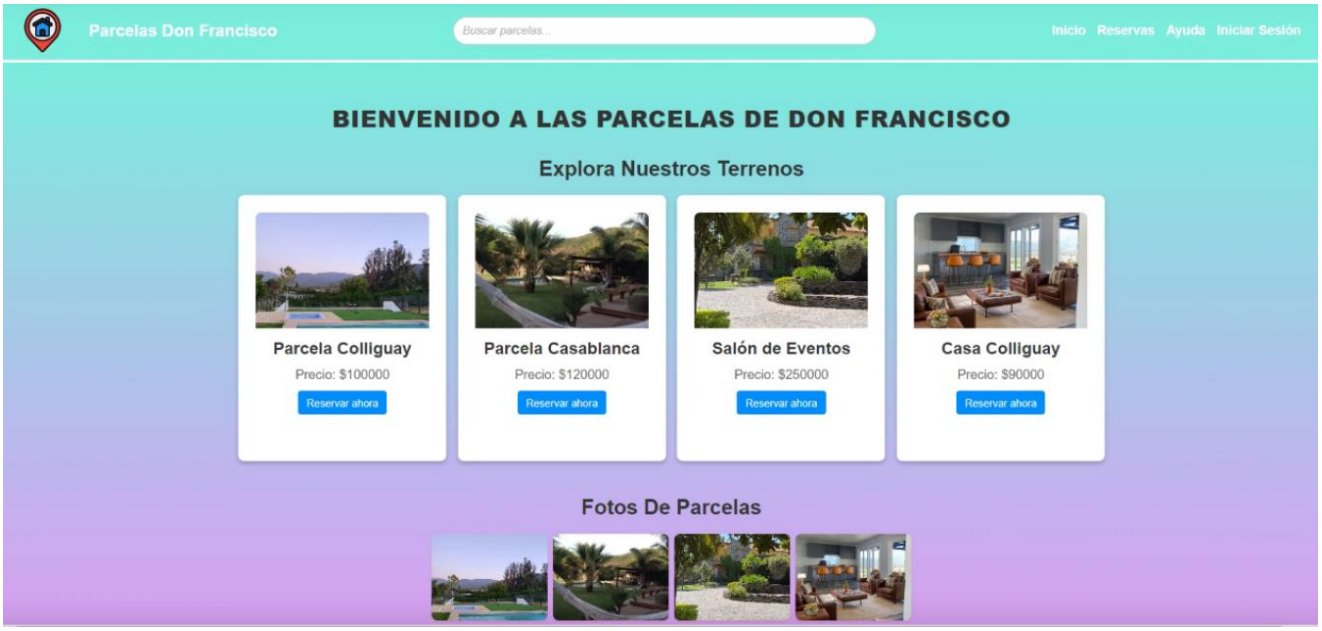
Ejemplo de Respuesta para Crear Reserva:

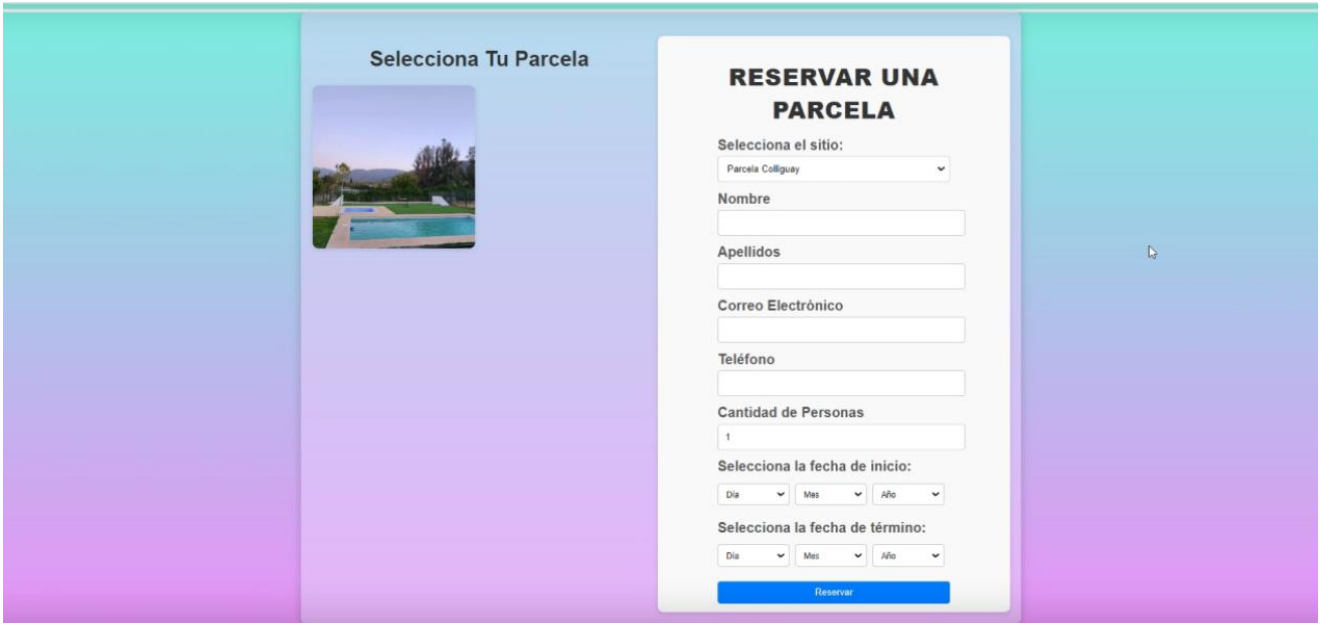
```
{
  "mensaje": "Reserva creada exitosamente.",
  "reservaId": "789"
}
```

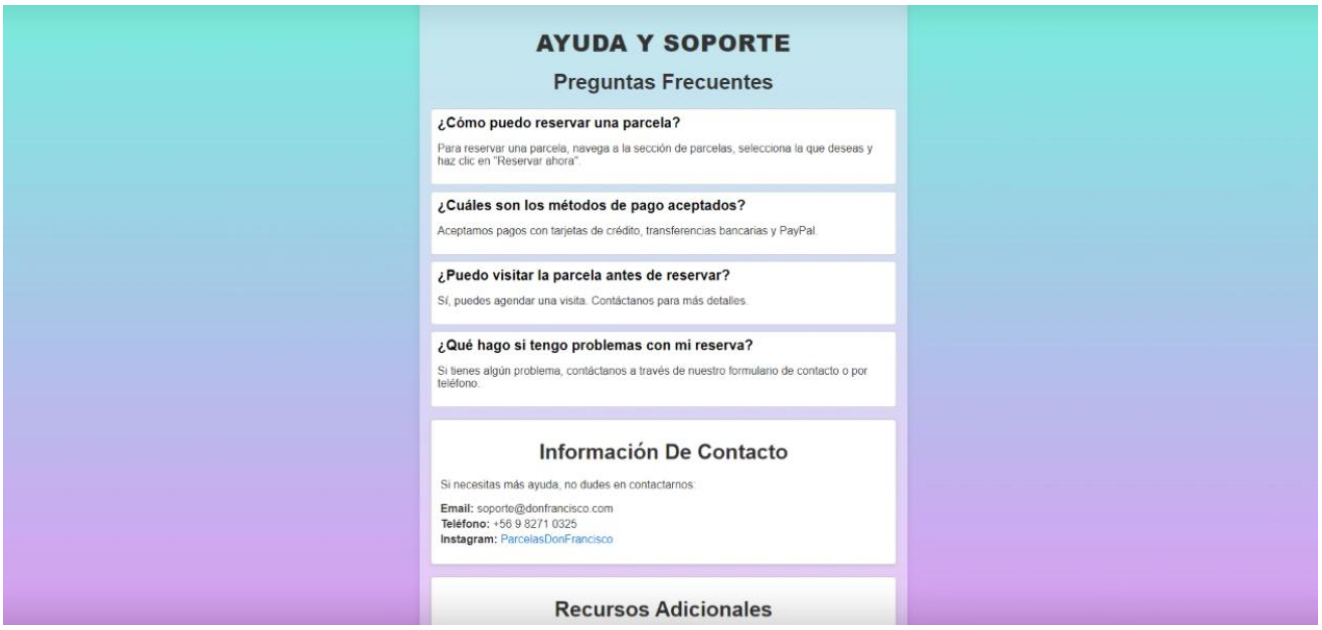
Consideraciones:

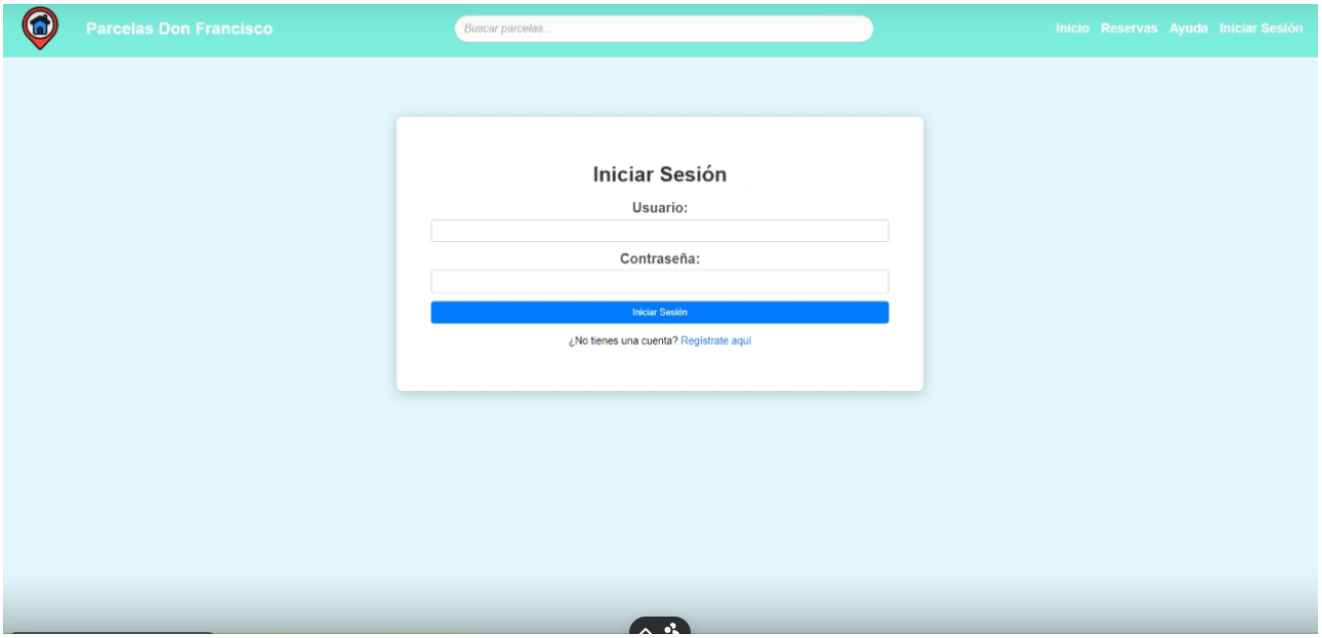
- **HTTPS**: Asegúrate de que tu API esté disponible a través de HTTPS para proteger la transmisión de datos.
- **Manejo de Errores**: Implementa un manejo de errores adecuado para las solicitudes que no cumplen con la autenticación y autorización.

Front Final

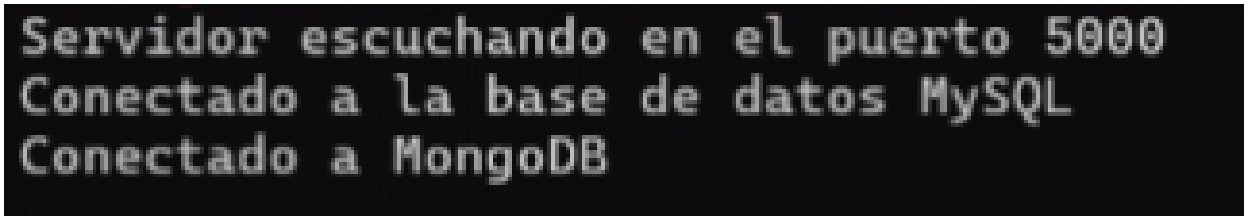

Parcelas Don Francisco
Buscar parcelas...
Inicio Reservas Ayuda Iniciar Sesión
BIENVENIDO A LAS PARCELAS DE DON FRANCISCO
Explora Nuestros Terrenos
Parcela Colliguay
Precio: $100000
Reservar ahora
Parcela Casablanca
Precio: $120000
Reservar ahora
Salón de Eventos
Precio: $250000
Reservar ahora
Casa Colliguay
Precio: $90000
Reservar ahora
Fotos De Parcelas


Selecciona Tu Parcela
RESERVAR UNA PARCELA
Selecciona el sitio:
Parcela Colliguay
Nombre
Apellidos
Correo Electrónico
Teléfono
Cantidad de Personas
1
Selecciona la fecha de inicio:
Día
Mes
Año
Selecciona la fecha de término:
Día
Mes
Año
Reservar


AYUDA Y SOPORTE
Preguntas Frecuentes
¿Cómo puedo reservar una parcela?
Para reservar una parcela, navega a la sección de parcelas, selecciona la que deseas y haz clic en "Reservar ahora".
¿Cuáles son los métodos de pago aceptados?
Aceptamos pagos con tarjetas de crédito, transferencias bancarias y PayPal.
¿Puedo visitar la parcela antes de reservar?
Sí, puedes agendar una visita. Contáctanos para más detalles.
¿Qué hago si tengo problemas con mi reserva?
Si tienes algún problema, contáctanos a través de nuestro formulario de contacto o por teléfono.
Información De Contacto
Si necesitas más ayuda, no dudes en contactarnos:
Email: soporte@donfrancisco.com
Teléfono: +56 9 8271 0325
Instagram: ParcelasDonFrancisco
Recursos Adicionales


Parcelas Don Francisco
Buscar parcelas...
Inicio
Reservas
Ayuda
Iniciar Sesión
Iniciar Sesión
Usuario:
Contraseña:
Iniciar Sesión
¿No tienes una cuenta? Regístrate aquí

Conexión de ambas Base de datos:



Actualizaciones

1. Se edito el objetivo, de tal manera que se pueda entender de mejor forma.

2. Se agregaron las imágenes front

3. Se agrego la documentación del API REST

4. Se agregaron los diagramas de secuencia

5. Se agrego el Modelo No SQL

6. Se explica el por qué es NoSQL

7. Se agregaron los CRUD y Consultas NoSQL

8. Se actualizan los front de la tercera entrega.

9. Se actualiza la api junto a la nueva autentificación (inicio de sesión).

10. Se agregaron las pruebas de las conexiones de las BD