

PROYECTO FINAL

NUEVAS TÉCNICAS DE PROGRAMACIÓN



Instituto
Tecnológico
Quito

ANÁLISIS EN SISTEMAS

NOMBRES: MAYRA CARRIÓN,
RICHARD ALTAMIRANO

NIVEL: 5 SEMESTRE

FECHA: 09
/02/2022

Contenido

- PROYECTO FINAL..... 3
 - 1. INTRODUCCIÓN 3
 - 2. NOMBRE DEL PROYECTO..... 3
 - 3. OBJETIVO GENERAL 3
 - 4. OBJETIVOS ESPECÍFICOS 4
 - 5. ANALISIS 4
 - 5.1.1 COLECCIONES EN MONGO DB 5
 - 5.1.2 SERVICIOS CON EXPRESS..... 6
 - 5.1.3 APLICACIÓN CON REACT 8
 - 6 CONCLUSIONES..... 10
 - 7 RECOMENDACIONES 10

PROYECTO FINAL

1. INTRODUCCIÓN

MERN significa MongoDB, Express, React, Node, después de las cuatro tecnologías clave que componen la pila, las cuales son:

- MongoDB - base de datos de documentos
- Express (.js) - Marco web Node.js
- React (.js): un marco de JavaScript del lado del cliente
- Nodo (.js): el principal servidor web de JavaScript

Cabe recalcar que, Express y Node conforman el nivel medio (aplicación). Express.js es un marco web del lado del servidor y Node.js es la popular y potente plataforma de servidor de JavaScript. Independientemente de la variante que elija, MERN es el enfoque ideal para trabajar con JavaScript y JSON, en todo momento.

2. NOMBRE DEL PROYECTO

- Sistema de reservaciones en un hotel

3. OBJETIVO GENERAL

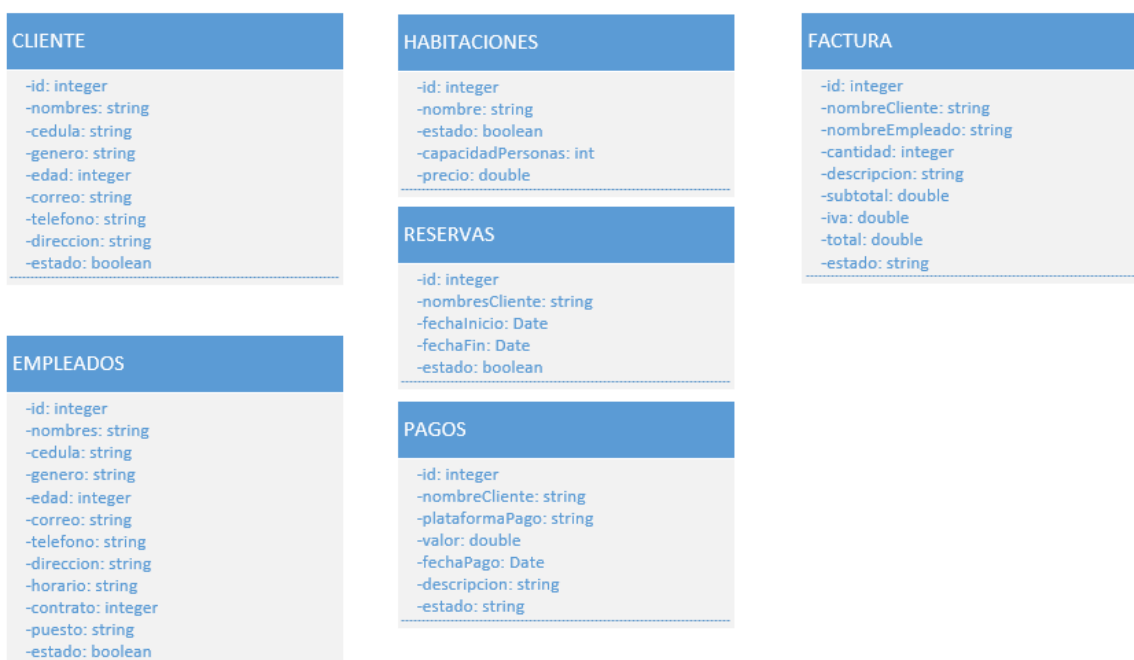
- Desarrollar un grupo de tecnologías Stack Mern en un sistema de reservaciones de habitaciones de un hotel, el cual permita poner en práctica todo lo aprendido en clases, a través de una base creada en MongoDB, el ingreso de datos en Express y una interfaz interactiva realizada con React.

4. OBJETIVOS ESPECÍFICOS

- Desarrollar las colecciones de la base de datos para el sistema de reservaciones, mediante la herramienta MongoDB, a fin de que los datos sean almacenados y utilizados de forma confiable.
- Desarrollar las rutas de datos (POST, GET, PUT, DELETE) para el sistema de reservaciones, mediante la herramienta Express, a fin de comprobar la utilización de las rutas.
- Realizar una interfaz interactiva para el sistema de reservaciones, mediante la herramienta React, con el fin de que utilizar funciones y componentes.

5. ANALISIS

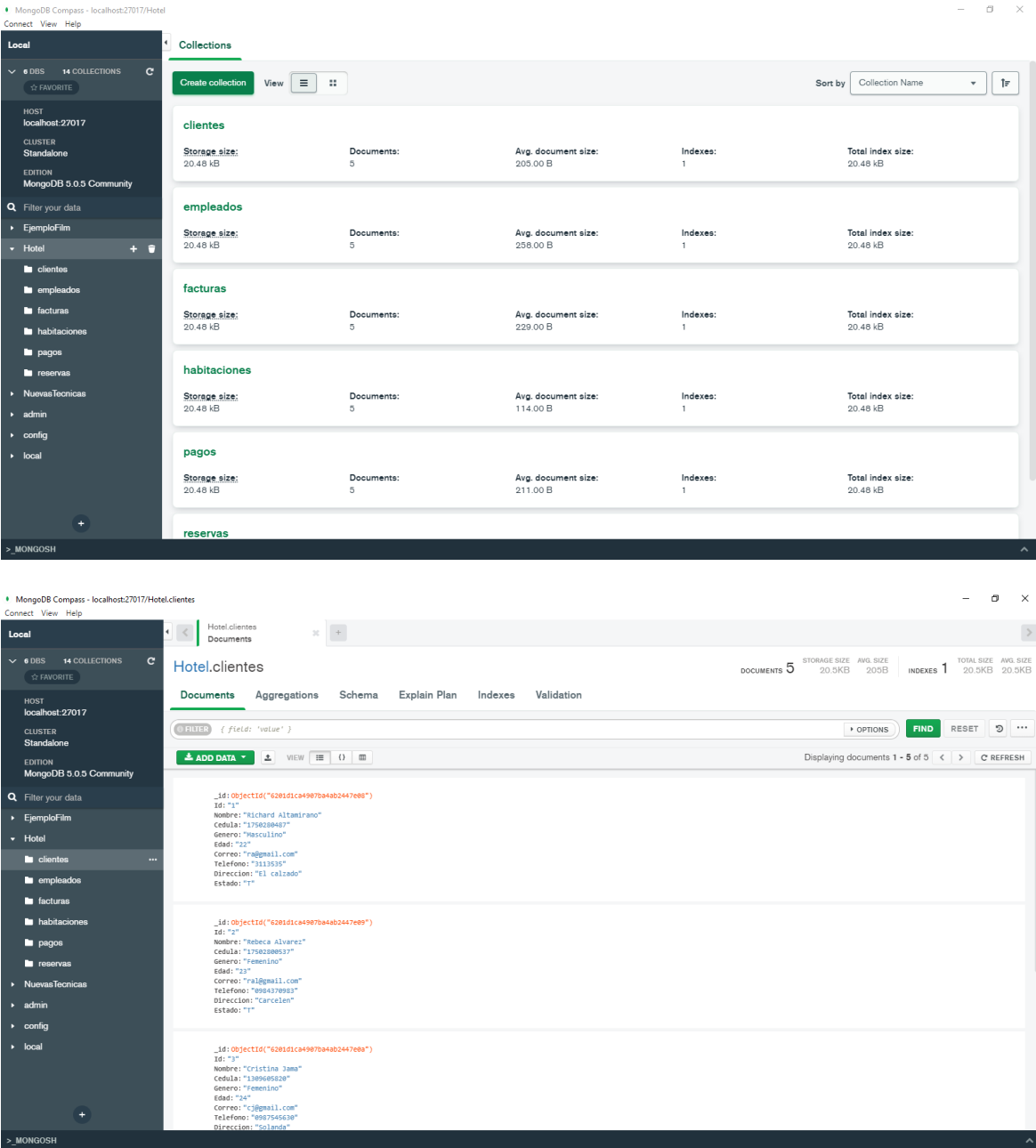
Para la realización del Stack Mern, realizamos un análisis de los datos que debería tener este sistema, para lo cual utilizamos el siguiente diagrama, en donde muestra que los empleados del hotel, serán registrados en una colección, los mismos que visualizaran si las habitaciones elegidas por el cliente están disponibles, para luego realizar la reserva, el pago correspondiente y finalmente la factura.



Luego de este procedimiento, se inició con la creación de Stack Mern, de la siguiente manera:

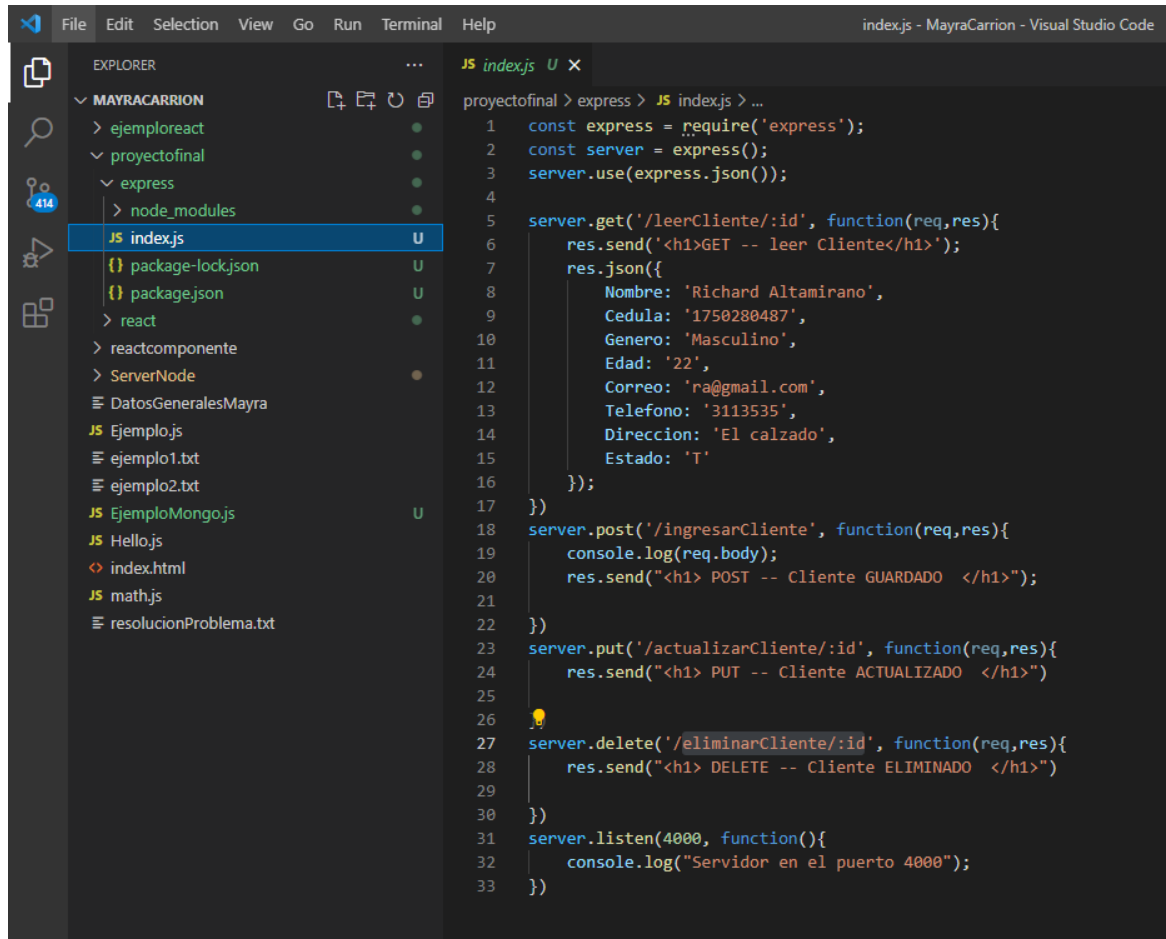
5.1.1 COLECCIONES EN MONGO DB

Se levanto el Mongoddb, se creó la base de datos, 6 colecciones con 5 datos en cada una de ellas, como se visualiza a continuación:



5.1.2 SERVICIOS CON EXPRESS

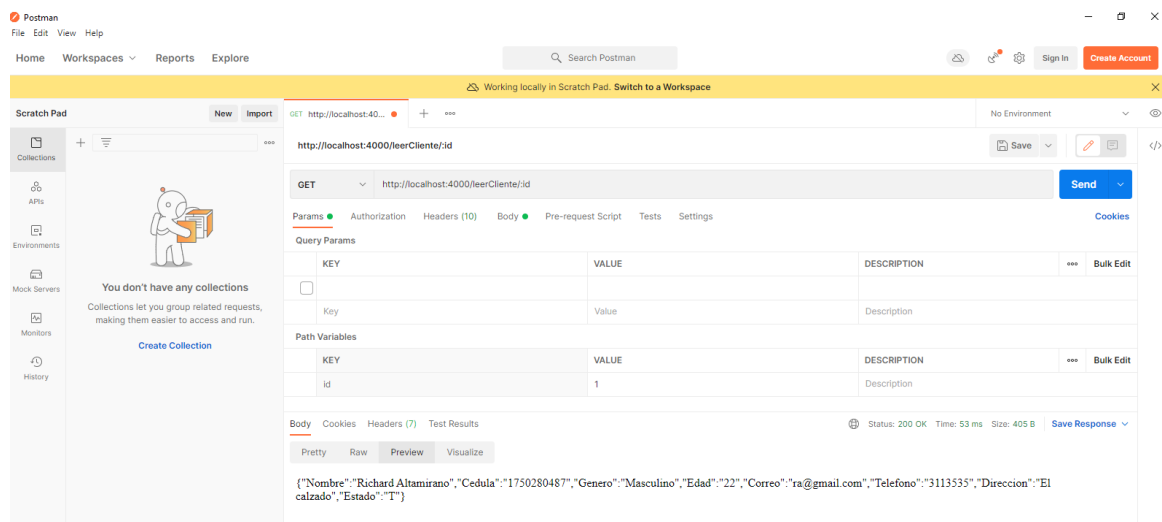
Se creo una carpeta para el la herramienta express, se inició el proyecto, se instaló express, y en la pagina de index.js, se realizó el código para leer, crear, actualizar y eliminar los datos, como se visualiza a continuación:



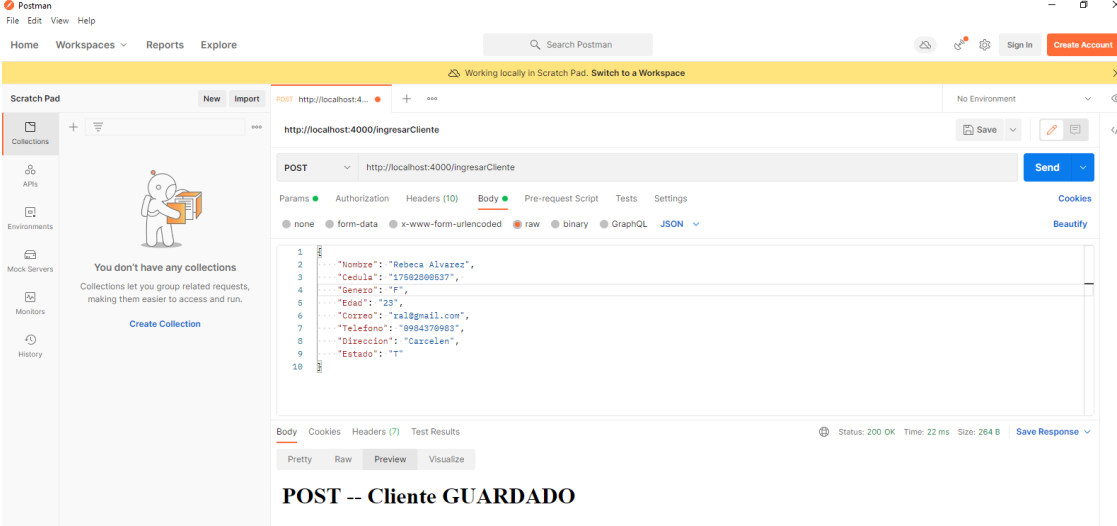
```
projectofinal > express > JS index.js > ...
1  const express = require('express');
2  const server = express();
3  server.use(express.json());
4
5  server.get('/leerCliente/:id', function(req,res){
6      res.send('<h1>GET -- leer Cliente</h1>');
7      res.json({
8          Nombre: 'Richard Altamirano',
9          Cedula: '1750280487',
10         Genero: 'Masculino',
11         Edad: '22',
12         Correo: 'ra@gmail.com',
13         Telefono: '3113535',
14         Direccion: 'El calzado',
15         Estado: 'T'
16     });
17 })
18 server.post('/ingresarCliente', function(req,res){
19     console.log(req.body);
20     res.send("<h1> POST -- Cliente GUARDADO </h1>");
21 })
22 server.put('/actualizarCliente/:id', function(req,res){
23     res.send("<h1> PUT -- Cliente ACTUALIZADO </h1>");
24 })
25
26
27 server.delete('/eliminarCliente/:id', function(req,res){
28     res.send("<h1> DELETE -- Cliente ELIMINADO </h1>");
29 })
30
31 server.listen(4000, function(){
32     console.log("Servidor en el puerto 4000");
33 })
```

Con el programa postman, se verifico que cada una de estas rutas estén funcionando correctamente, como se muestra a continuación:

Ruta GET: <http://localhost:4000/leerCliente/:id>



Ruta POST: http://localhost:4000/ingresarCliente



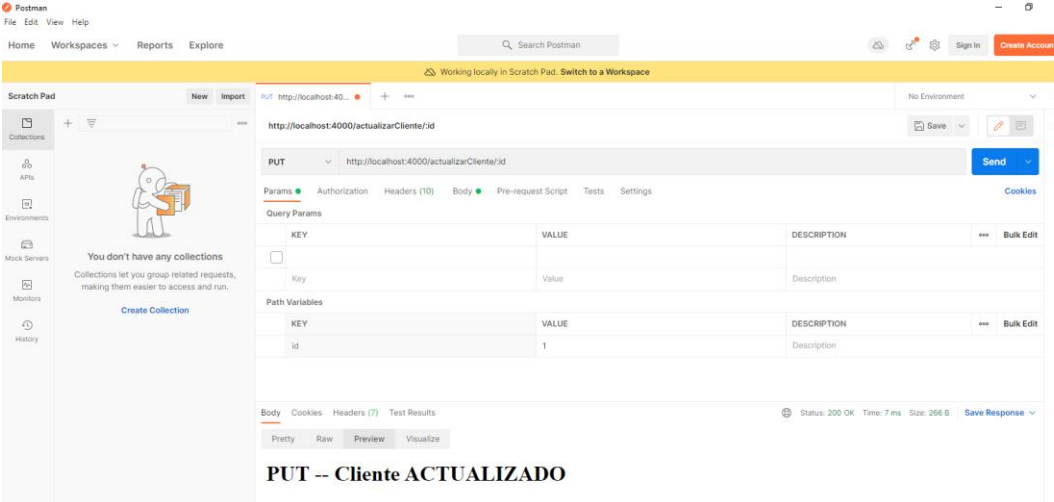
Postman interface showing a POST request to `http://localhost:4000/ingresarCliente`. The request body is a JSON object representing a client:

```
1 {
2   "Nombre": "Rebeca Alvarez",
3   "Cedula": "17582908537",
4   "Genero": "F",
5   "Edad": "23",
6   "Correo": "rai@gmail.com",
7   "Telefono": "9984378983",
8   "Direccion": "Cazcelen",
9   "Estado": "T"
10 }
```

The response status is 200 OK, Time: 22 ms, Size: 264 B. The response is saved.

POST -- Cliente GUARDADO

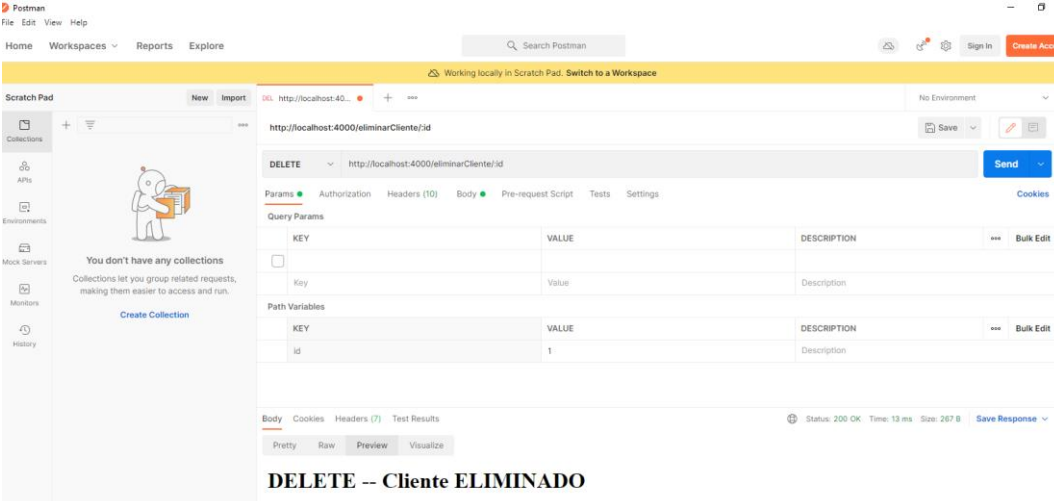
Ruta PUT: http://localhost:4000/actualizarCliente/:id



Postman interface showing a PUT request to `http://localhost:4000/actualizarCliente/:id`. The request has a path variable `id` set to 1. The response status is 200 OK, Time: 7 ms, Size: 266 B. The response is saved.

PUT -- Cliente ACTUALIZADO

Ruta DELETE: http://localhost:4000/eliminarCliente/:id

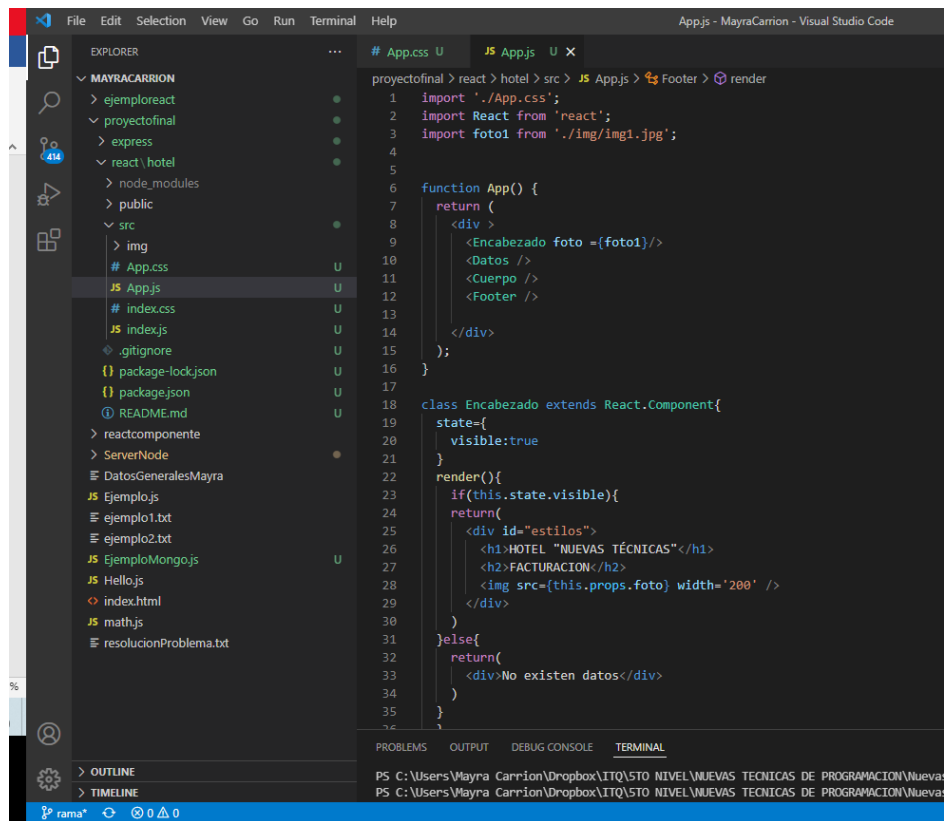


Postman interface showing a DELETE request to `http://localhost:4000/eliminarCliente/:id`. The request has a path variable `id` set to 1. The response status is 200 OK, Time: 13 ms, Size: 267 B. The response is saved.

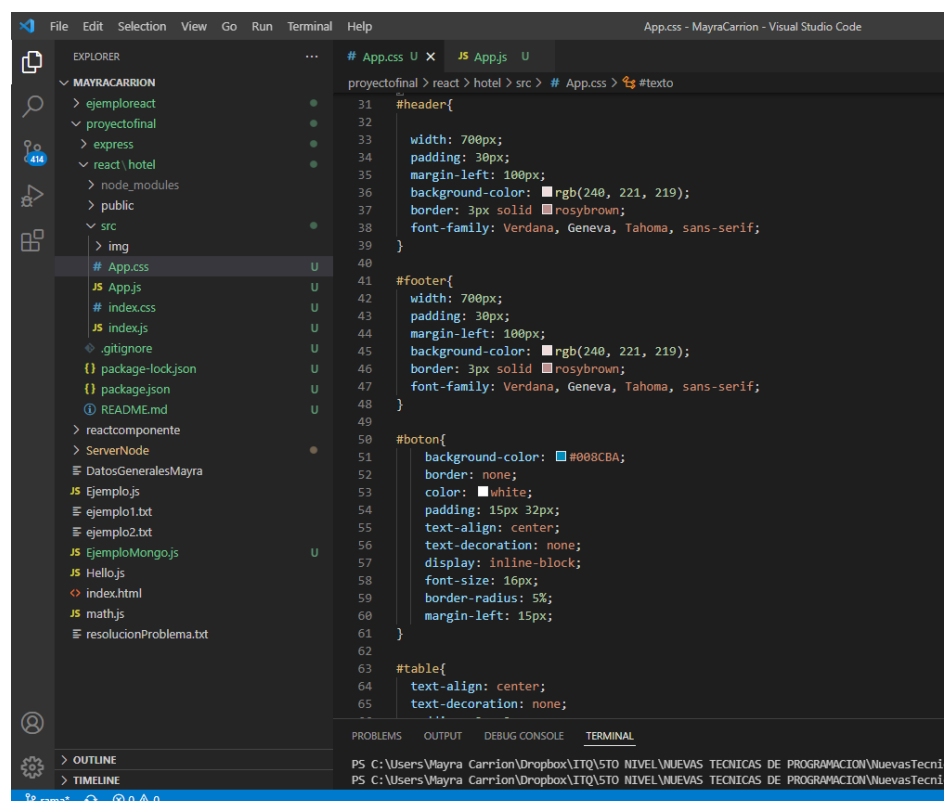
DELETE -- Cliente ELIMINADO

5.1.3 APLICACIÓN CON REACT

Se creo una carpeta para el la herramienta React, en donde se utilizó el comando npx créate-react-app nombre para iniciar el proyecto, luego se creó las funciones y componentes para cada sección de la pantalla de factura. Finalmente se realizó el css de la misma como se visualiza a continuación:



```
# App.css U JS App.js U X
projectofinal > react > hotel > src > JS App.js > Footer > render
1 import './App.css';
2 import React from 'react';
3 import foto1 from './img/img1.jpg';
4
5
6 function App() {
7   return (
8     <div>
9       <Encabezado foto ={foto1}/>
10      <Datos />
11      <Cuerpo />
12      <Footer />
13    </div>
14  );
15 }
16
17
18 class Encabezado extends React.Component{
19   state={
20     visible:true
21   }
22   render(){
23     if(this.state.visible){
24       return(
25         <div id="estilos">
26           <h1>HOTEL "NUEVAS TÉCNICAS"</h1>
27           <h2>FACTURACION</h2>
28           <img src={this.props.foto} width='200' />
29         </div>
30       )
31     }else{
32       return(
33         <div>No existen datos</div>
34       )
35     }
36   }
37 }
```



```
# App.css U X JS App.js U
projectofinal > react > hotel > src > # App.css > #texto
31 #header{
32
33   width: 700px;
34   padding: 30px;
35   margin-left: 100px;
36   background-color: #rgb(240, 221, 219);
37   border: 3px solid #rosybrown;
38   font-family: Verdana, Geneva, Tahoma, sans-serif;
39 }
40
41 #footer{
42   width: 700px;
43   padding: 30px;
44   margin-left: 100px;
45   background-color: #rgb(240, 221, 219);
46   border: 3px solid #rosybrown;
47   font-family: Verdana, Geneva, Tahoma, sans-serif;
48 }
49
50 #boton{
51   background-color: #008CBA;
52   border: none;
53   color: #white;
54   padding: 15px 32px;
55   text-align: center;
56   text-decoration: none;
57   display: inline-block;
58   font-size: 16px;
59   border-radius: 5%;
60   margin-left: 15px;
61 }
62
63 #table{
64   text-align: center;
65   text-decoration: none;
66 }
```


Finalmente, la pantalla quedo de esta manera:


React App x +

localhost:3000

Aplicaciones WhatsApp ITQ Smallpdf Pichincha YouTube Outlook GitHub peaje VSII-Mat

HOTEL "NUEVAS TÉCNICAS"

FACTURACION



NÚMERO FACTURA:

NOMBRE CLIENTE:

NOMBRE EMPLEADO:

DESCRIPCION:

CODIGO	CANTIDAD	DESCRIPCION	VALOR
1	1	Habitacion individual	17.85

SUBTOTAL:

IVA:

TOTAL:

ESTADO:

Windows Taskbar: Windows, Search, Task View, File Explorer, Chrome, Excel, Word, Teams, VS Code, Edge, PowerPoint

6 CONCLUSIONES

- Con este proyecto hemos llegado a la conclusión, de que las técnicas utilizadas como node, express y react son muy útiles para la realización de funciones y componentes en proyectos reales y funcionales.
- Se concluye finalmente que este proyecto nos permitió a desarrollar habilidades nuevas a la hora de programar y utilizar funciones y componentes, creación de colecciones en gestor de base de datos MongoDB y utilización de un control de versiones que nos ayudara mucho para futuros proyectos.

7 RECOMENDACIONES

- Se recomienda, indagar más sobre los temas de Stack Mern, la distintas formas de programar en estas, entre otros, ya que son de mucha ayuda para la implementación de sistemas.
- Se recomienda entender más a profundidad sobre las bases de datos no relaciones en el software de MongoDB ya que es muy interesante la aplicación de esta misma en varios proyectos y el control de versiones ya que es muy útil y necesario de aprender para los sistemas que podamos realizar.