



TECNOLÓGICO
NACIONAL DE MÉXICO
CAMPUS MONCLOVA

PROYECTO FINAL

Aplicación Cliente-Servidor

7mo semestre

Programación en Ambiente Cliente-Servidor
Arely Aide Covarrubias Garcia
Arisbeth Leija Garza



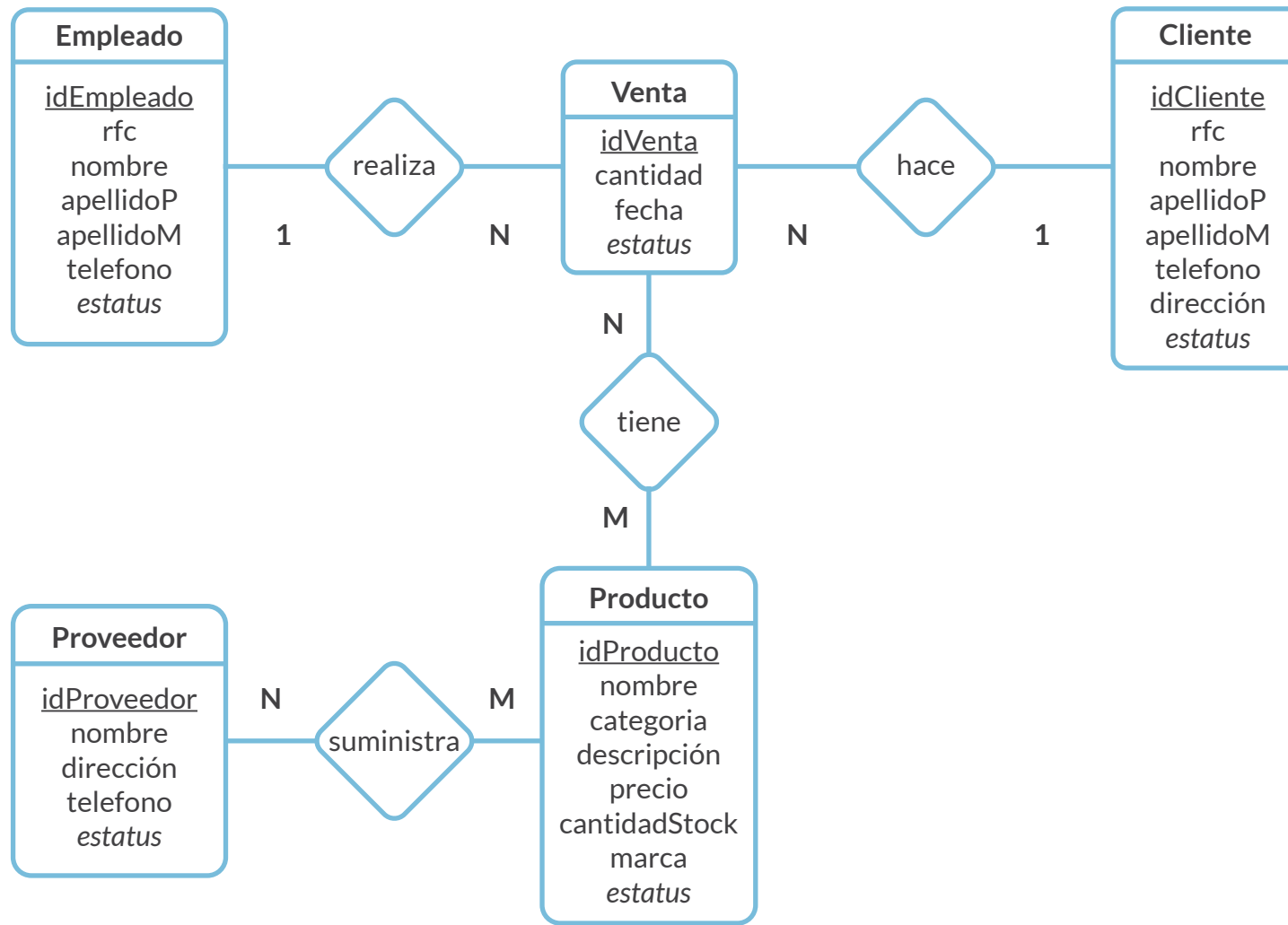
CONTENIDO

1.- Modelado de la base de datos	3
1.1.- Diagrama Entidad-Relación	4
1.2.- Modelo Racional	5
1.3.- Diccionario de Datos	5
2.- Construcción de la Web API	8
2.1.- Clase: BD	9
2.2.- Clase: Users PHP	11
2.3.- Comprobación del funcionamiento de la API Restfull con Postman	12
3.- Lanzamiento de API RestFull por medio del servidor IIS de Windows	15
4.- Implementación del cliente REACT	18
4.1.- Pruebas del cliente	24
5.- Implementación de cliente: Aplicación Web de ASP NET CORE	27

Modelado de la base de datos



Diagrama Entidad-Relación



1.2.- Modelo Racional

- **Cliente** (idCliente, nombre, apPaterno, apMaterno, direccion, telefono, estatus)
- **Empleado** (idEmpleado, rfc, nombre, apellidoP, apellidoM, direccion, telefono, estatus)
- **Producto** (idProducto, nombre, categoría, descripción, precio, cantidadStock, marca, estatus)
- **Proveedor** (idProveedor, nombre, direccion, telefono, estatus)
- **Venta** (idVenta, cantidad, fecha, detalles, estatus, idEmpleado, idCliente)
- **ProductoVenta** (idProducto, idVenta, nombreProducto, cantidadProducto, estatus)
- **ProductoProveedor** (idProducto, idProveedor, estatus)

1.3.- Diccionario de Datos

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	Proveedor	idProveedor	int	10	NO	SI	NO	SI	Identificador de proveedor	Numero entero positivo	Unico/ Requerido
TiendaApiDB	Proveedor	nombre	varchar	25	NO	NO	NO		Nombre de proveedor	Dato alfanumerico	Requerido
TiendaApiDB	Proveedor	direccion	varchar	100	NO	NO	NO		Direccion de proveedor	Dato alfanumerico	Requerido
TiendaApiDB	Proveedor	telefono	varchar	20	NO	NO	NO		telefono	Dato alfanumerico	Requerido
TiendaApiDB	Proveedor	estatus	bit	1	NO	NO	NO		Estado del proveedor	0/1	Requerido

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	Cliente	idCliente	int	10	NO	SI	NO	SI	Identificador de cliente	Numero entero positivo	Unico/ Requerido
TiendaApiDB	Cliente	nombre	varchar	25	NO	NO	NO		Nombre del cliente	Dato alfanumerico	Requerido
TiendaApiDB	Cliente	apellidoP	varchar	20	NO	NO	NO		Apellido paterno	Dato alfanumerico	Requerido
TiendaApiDB	Cliente	apellidoM	varchar	20	NO	NO	NO		Apellido materno	Dato alfanumerico	Requerido
TiendaApiDB	Cliente	direccion	varchar	100	NO	NO	NO		direccion del cliente	Dato alfanumerico	Requerido
TiendaApiDB	Cliente	telefono	varchar	20	NO	NO	NO		telefono	Dato alfanumerico	Requerido
TiendaApiDB	Cliente	estatus	bit	1	NO	NO	NO		Estado del cliente	0/1	Requerido

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	Empleado	idEmpleado	int	10	NO	SI	NO	SI	Identificador de empleado	Numero entero positivo	Unico/ Requerido
TiendaApiDB	Empleado	rfc	varchar	25	NO	NO	NO		Registros Federal Contribuyentes	Dato alfanumerico	Requerido
TiendaApiDB	Empleado	nombre	varchar	25	NO	NO	NO		Nombre del empleado	Dato alfanumerico	Requerido
TiendaApiDB	Empleado	apellidoP	varchar	20	NO	NO	NO		Apellido paterno	Dato alfanumerico	Requerido
TiendaApiDB	Empleado	apellidoM	varchar	20	NO	NO	NO		Apellido materno	Dato alfanumerico	Requerido
TiendaApiDB	Empleado	direccion	varchar	100	NO	NO	NO		Direccion del cliente	Dato alfanumerico	Requerido
TiendaApiDB	Empleado	telefono	varchar	20	NO	NO	NO		Telefono	Dato alfanumerico	Requerido
TiendaApiDB	Empleado	estatus	bit	1	NO	NO	NO		Estado del cliente	0/1	Requerido

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	Producto	idProducto	int	10	NO	SI	NO	SI	Identificador de producto	Numero entero positivo	Unico/ Requerido
TiendaApiDB	Producto	nombre	varchar	25	NO	NO	NO		Nombre del producto	Dato alfanumerico	Requerido
TiendaApiDB	Producto	categoria	varchar	20	NO	NO	NO		Categoria	Dato alfanumerico	Requerido
TiendaApiDB	Producto	descripción	varchar	20	NO	NO	NO		Descripcion del producto	Dato alfanumerico	Requerido
TiendaApiDB	Producto	precio	decimal	10,2	NO	NO	NO		Precio del producto	Número con punto decimal	Requerido
TiendaApiDB	Producto	cantidadStock	int	20	NO	NO	NO		Cantidad del stock	Numero entero positivo	Requerido
TiendaApiDB	Producto	marca	varchar	25	NO	NO	NO		Marca del producto	Dato alfanumerico	Requerido
TiendaApiDB	Producto	estatus	bit	1	NO	NO	NO		Estado del producto	0/1	Requerido

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	Venta	idVenta	int	10	NO	SI	NO	SI	Identificador de venta	Numero entero positivo	Unico/ Requerido
TiendaApiDB	Venta	cantidad	int	10	NO	NO	NO		Cantidad de venta	Número entero positivo	Requerido
TiendaApiDB	Venta	fecha	datetime	20	NO	NO	NO		Fecha de venta	Dato alfanumerico	Requerido
TiendaApiDB	Venta	estatus	bit	1	NO	NO	NO		Estatus de venta	0/1	Requerido
TiendaApiDB	Venta	idCliente	int	10	NO	NO	SI		Identificador de cliente	Número entero positivo	Requerido
TiendaApiDB	Venta	idEmpleado	int	11	NO	NO	SI		Identificador de empleado	Número entero positivo	Requerido

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	ProductoVenta	idProductoVenta	int	10	NO	SI	NO	SI	Identificador de productos de venta	Numero entero positivo	Unico/ Requerido
TiendaApiDB	ProductoVenta	idProducto	int	10	NO	NO	SI		Identificador de producto	Número entero positivo	Requerido
TiendaApiDB	ProductoVenta	idVenta	int	11	NO	NO	SI		Identificador de producto	Número entero positivo	Requerido
TiendaApiDB	ProductoVenta	nombre	varchar	1	NO	NO	NO		Nombre de venta	Dato alfanumerico	Requerido
TiendaApiDB	ProductoVenta	cantidadProducto	int	10	NO	NO	SI		Cantidad de producto	Número entero positivo	Requerido
TiendaApiDB	ProductoVenta	estatus	bit	11	NO	NO	SI		Estado de los pedidos o productos	0/1	Requerido

Base de Datos	Tabla	Campo	Tipo	Tamaño	Nulo	PK	FK	Auto Incremento	Descripción	Dominio	Restricciones
TiendaApiDB	Producto Proveedor	idProductoProveedor	int	10	NO	SI	NO	SI	Identificador de productos y proveedor	Numero entero positivo	Unico/ Requerido
TiendaApiDB	Producto Proveedor	idProducto	int	10	NO	NO	SI		Identificador de producto	Número entero positivo	Requerido
TiendaApiDB	Producto Proveedor	idProveedor	int	11	NO	NO	SI		Identificador de proveedor	Número entero positivo	Requerido
TiendaApiDB	Producto Proveedor	estatus	bit	1	NO	NO	NO		Nombre de venta	Estado de los pedidos o productos	Requerido

Construcción de la web API



PDO significa PHP Data Objects, Objetos de Datos de PHP, una extensión para acceder a bases de datos. PDO permite acceder a diferentes sistemas de bases de datos con un controlador específico (MySQL, SQLite, Oracle...) mediante el cual se conecta. Independientemente del sistema utilizado, se emplearán siempre los mismos métodos, lo que hace que cambiar de uno a otro resulte más sencillo.

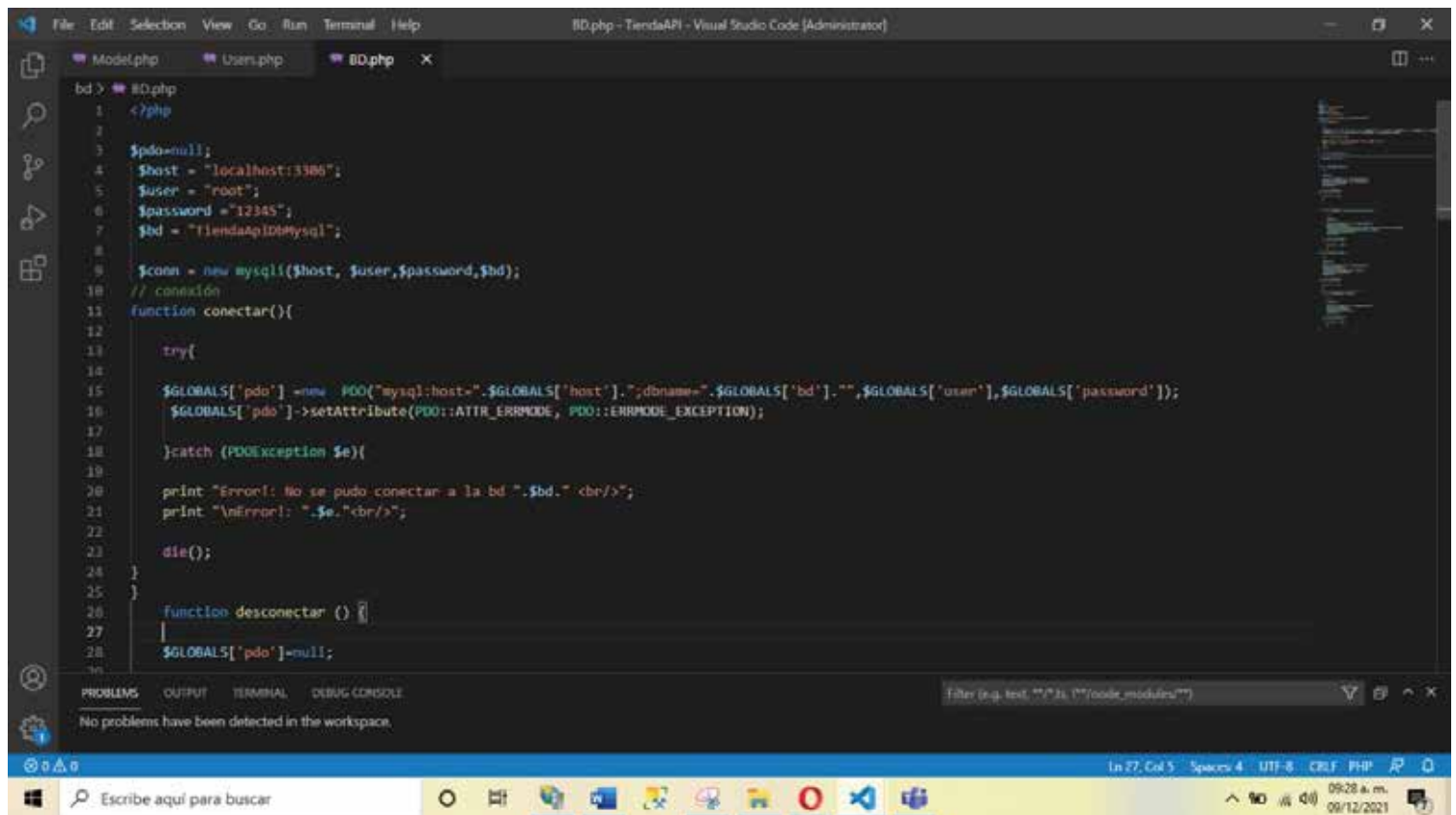
La clase PDO se encarga de mantener la conexión a la base de datos y otro tipo de conexiones específicas como transacciones, además de crear instancias de la clase PDOStatement. Es ésta clase, PDOStatement, la que maneja las sentencias SQL y devuelve los resultados. La clase PDOException se utiliza para manejar los errores.

SE realizo una API restfull utilizando lenguajes php en Visual Studio Code conectado a la base de datos de MySQL por medio de la implementacion de la capa de abstraccion de datos PDO.

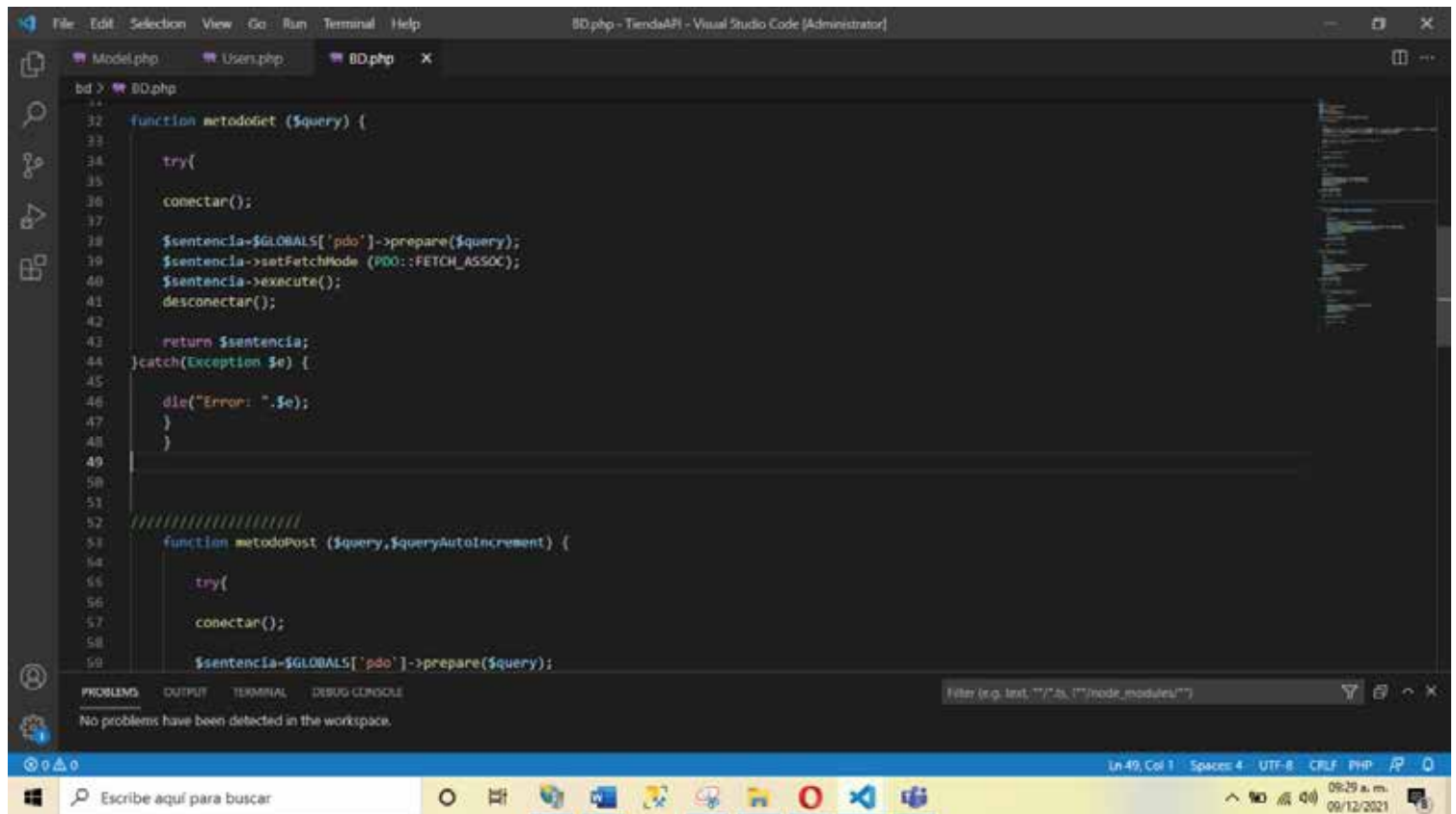
2.1.- Clase: BD

En esta clase se crean variables con los parametros de conexion a la base de datos de mysql y posteriormente se realiza la funcion de conexion en la cual se utiliza la capa de abstraccion de datos PDO para enviar los paarmetros de cadena de conexion.

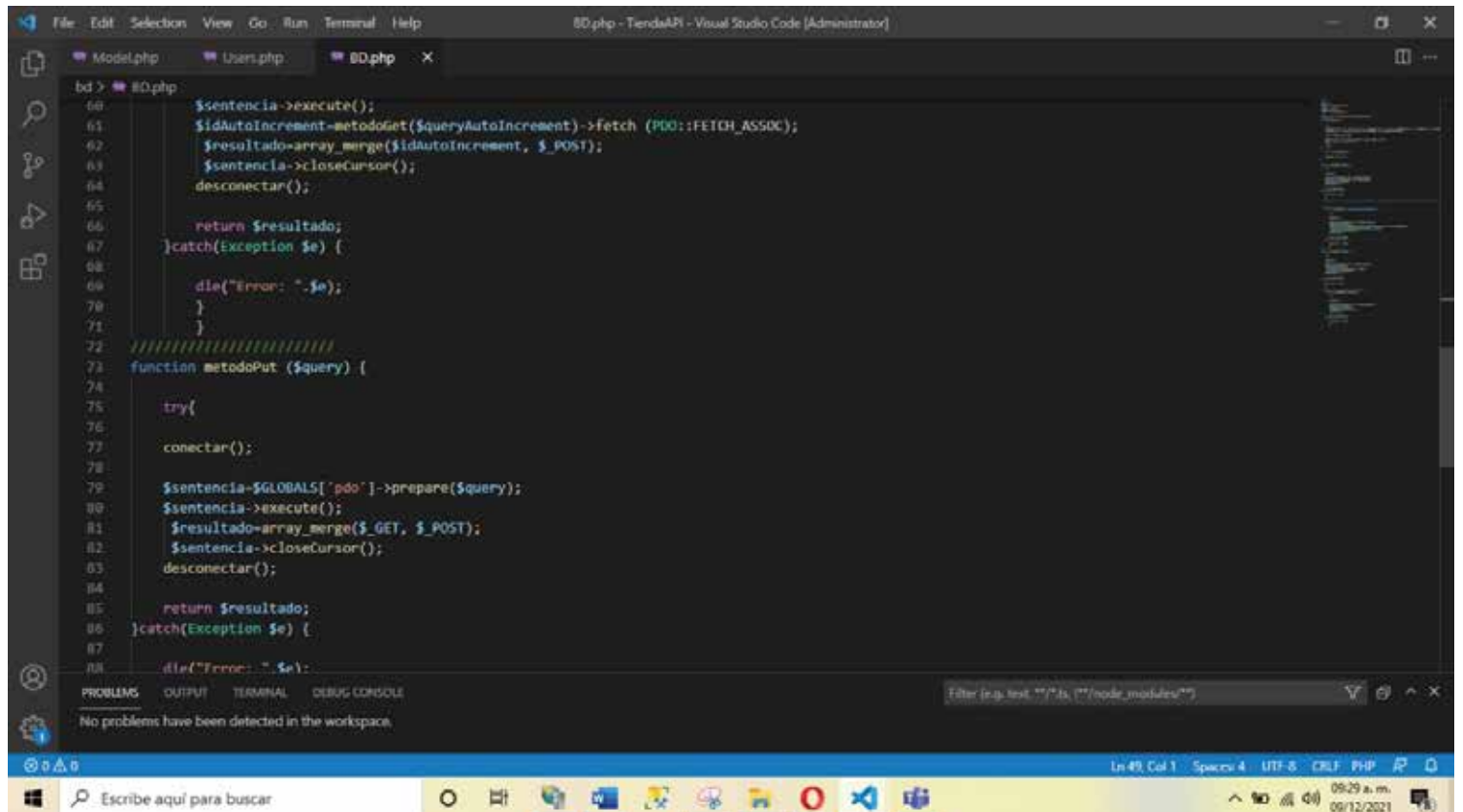
En el resto de la clase se encuentrasn las respectivas funciones de crud para la base de datos



```
bd > BD.php
1  <?php
2
3  $pdo=null;
4  $host = "localhost:3306";
5  $user = "root";
6  $password = "12345";
7  $bd = "tiendaApiDbmysql";
8
9  $conn = new mysqli($host, $user,$password,$bd);
10 // conexión
11 function conectar(){
12
13     try{
14
15         $GLOBALS['pdo'] =new PDO("mysql:host=".$GLOBALS['host'].";dbname=".$GLOBALS['bd']."",$GLOBALS['user'],$GLOBALS['password']);
16         $GLOBALS['pdo']->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
17
18     }catch (PDOException $e){
19
20         print "Error!: No se pudo conectar a la bd ".$bd."<br/>";
21         print "\nError!: ".$e."<br/>";
22
23     }
24     die();
25 }
26
27 function desconectar () {
28     $GLOBALS['pdo']=null;
29 }
```



```
bd > BD.php
32 function metodoGet ($query) {
33
34     try{
35
36         conectar();
37
38         $sentencia=$GLOBALS['pdo']->prepare($query);
39         $sentencia->setFetchMode (PDO::FETCH_ASSOC);
40         $sentencia->execute();
41         desconectar();
42
43         return $sentencia;
44     }catch(Exception $e) {
45
46         die("Error: ".$e);
47     }
48 }
49
50
51
52
53 //////////////////////////////////////////////////
54 function metodoPost ($query,$queryAutoIncrement) {
55
56     try{
57
58         conectar();
59
60         $sentencia=$GLOBALS['pdo']->prepare($query);
```



```
60 $sentencia->execute();
61 $idAutoIncrement=metodoGet($queryAutoIncrement)->fetch (PDO::FETCH_ASSOC);
62 $resultado=array_merge($idAutoIncrement, $_POST);
63 $sentencia->closeCursor();
64 desconectar();
65
66 return $resultado;
67 }catch(Exception $e) {
68
69     die("Error: ".$e);
70 }
71 }
72
73 //////////////////////////////////////////////////
74 function metodoPut ($query) {
75
76     try{
77
78         conectar();
79
80         $sentencia=$GLOBALS['pdo']->prepare($query);
81         $sentencia->execute();
82         $resultado=array_merge($_GET, $_POST);
83         $sentencia->closeCursor();
84         desconectar();
85
86         return $resultado;
87     }catch(Exception $e) {
88
89         die("Error: ".$e);
90     }
91 }
```

```

bd > BD.php
87
88     die("Error: ".$e);
89 }
90 }
91
92 //////////////////////////////////////////////////
93 function metodoDelete ($query) {
94
95     try{
96         conectar();
97
98         $sentencia=$GLOBALS['pdo']->prepare($query);
99         $sentencia->execute();
100         $sentencia->closeCursor();
101         desconectar();
102
103         return $_GET['id'];
104     }catch(Exception $e) {
105
106         die("Error: ".$e);
107     }
108 }
109

```

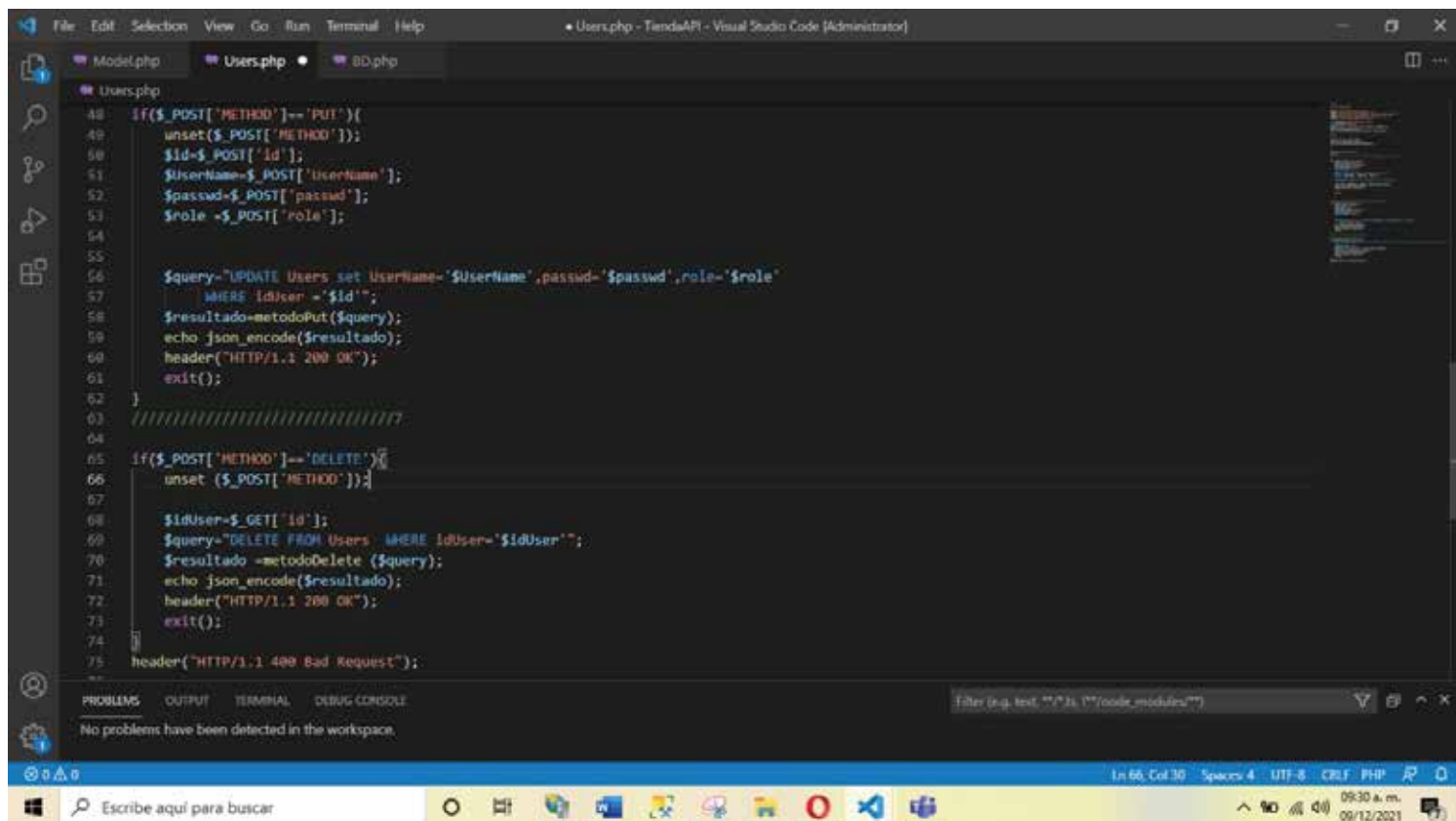
2.2.- Clase Users.PHP

Para poder realizar el crud en cada una de las tablas contenidas en a base de datos, se creo una clase para cada respectiva entidad de la base de datos. En ella se colocaron condicionales que envian un comando de ejecucion de query segun el metodo requerido por el programa (GET, POST, DELETE, PUT). Y especificando los paráremtros que debe recibir y utilizar en cada comando.

```

Users.php - TiendaAPI - Visual Studio Code [Administrator]
Users.php
1 <?php
2
3 include 'bd/BD.php';
4
5 header('Access-Control-Allow-origin: *');
6 header('Access-Control-Allow-Methods: GET, POST, PUT, DELETE');
7 header('Access-Control-Allow-Headers: X-Requested-With');
8 header('Content-type: application/json');
9
10 if ($_SERVER['REQUEST_METHOD']=='GET'){
11     if(isset($_GET['id'])){
12         $query="select * from Users where idUser=".$_GET['id'];
13         $resultado=metodoDelete($query);
14         echo json_encode($resultado->fetch(PDO::FETCH_ASSOC));
15     }else{
16
17         $query ="select *from Users";
18         $resultado =metodoGet ($query);
19         echo json_encode($resultado->fetchAll());
20     }
21 }
22
23
24 header("HTTP/1.1 200 OK");
25 exit();
26
27 //////////////////////////////////////////////////
28 if($_POST['METHOD'] == 'POST'){
29     //metodo de insertar usuarios
30

```



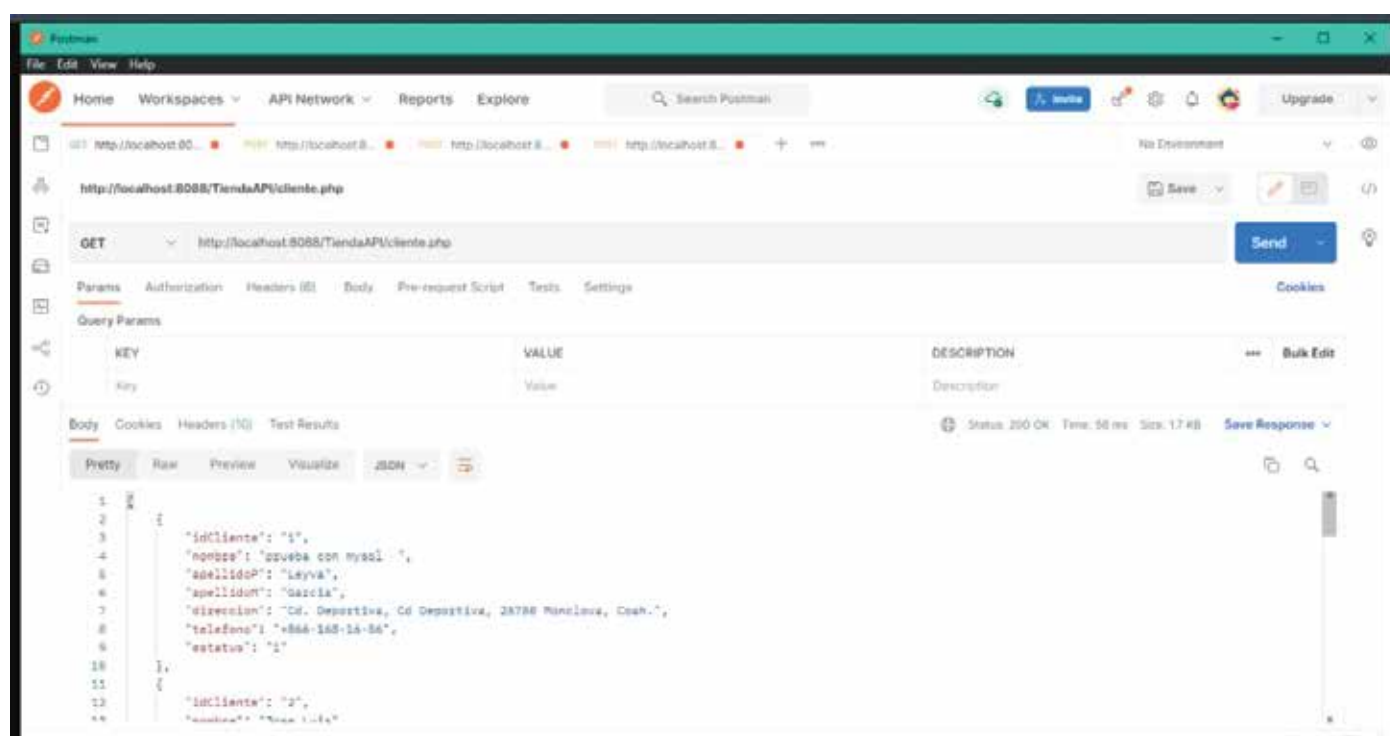
```
48 if($_POST['METHOD']=='PUT'){
49     unset($_POST['METHOD']);
50     $id=$_POST['id'];
51     $userName=$_POST['userName'];
52     $passwd=$_POST['passwd'];
53     $role=$_POST['role'];
54
55     $query="UPDATE Users set UserName='$userName',passwd='$passwd',role='$role'
56         WHERE idUser = '$id'";
57     $resultado=metodoPut($query);
58     echo json_encode($resultado);
59     header("HTTP/1.1 200 OK");
60     exit();
61 }
62
63 ///////////////////////////////////////////////////
64
65 if($_POST['METHOD']=='DELETE'){
66     unset($_POST['METHOD']);
67
68     $idUser=$_GET['id'];
69     $query="DELETE FROM Users WHERE idUser='$idUser'";
70     $resultado=metodoDelete($query);
71     echo json_encode($resultado);
72     header("HTTP/1.1 200 OK");
73     exit();
74
75     header("HTTP/1.1 400 Bad Request");
76 }
```

En el presente documento se muestra únicamente la clase de users.php ya que el resto de las clases comparten una sintaxis similar.

2.3.- Comprobación del funcionamiento de la API RESTFULL CON POSTMAN:

Una vez terminada la API en en php con PDO como capa de abstracción, se realizaron las siguientes pruebas de CRUD:

1.- Prueba del método GET :



2.- Prueba del método POST

Postman interface showing a POST request to `http://localhost:8088/TiendaAPI/cliente.php?id=11`. The request body is a JSON object with the following fields:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> id	11	
<input checked="" type="checkbox"/> nombre	Actualizar	
<input checked="" type="checkbox"/> apellidoP	Perez	
<input checked="" type="checkbox"/> apellidoM	Caballero	

The response is a 200 OK status with a time of 43 ms and a size of 513 B. The response body is a JSON object with the following fields:

```
1 {
2   "id": "11",
3   "nombre": "Luis",
4   "apellidoP": "Hernandez",
5   "apellidoM": "Aguilar",
6   "direccion": "Bosques",
7   "telefono": "8666632",
8   "estatus": "1"
9 }
```

3.- Prueba del método PUT

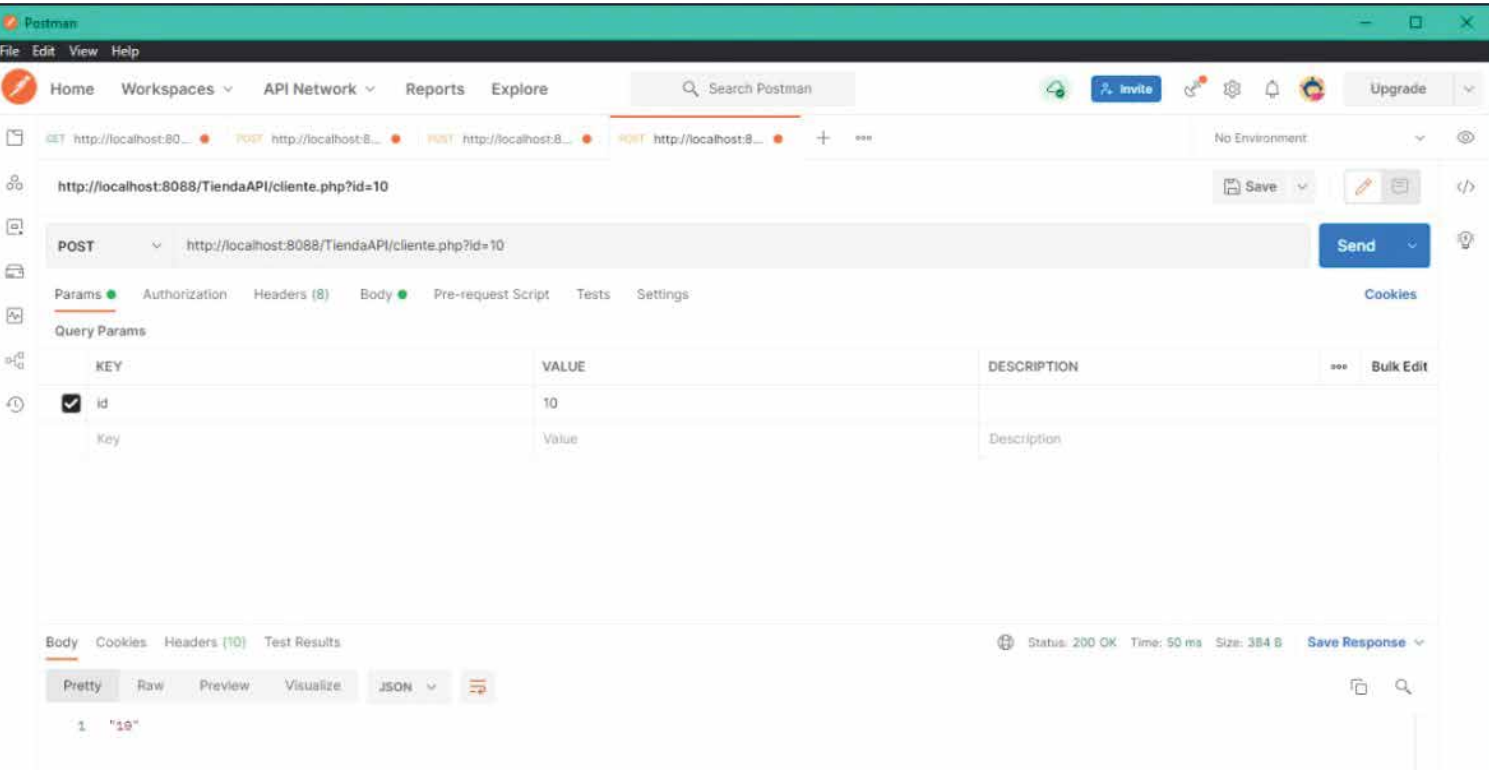
Postman interface showing a PUT request to `http://localhost:8088/TiendaAPI/cliente.php?id=11`. The request body is a JSON object with the following fields:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> id	11	
<input checked="" type="checkbox"/> nombre	Actualizar	
<input checked="" type="checkbox"/> apellidoP	Perez	
<input checked="" type="checkbox"/> apellidoM	Caballero	

The response is a 200 OK status with a time of 43 ms and a size of 513 B. The response body is a JSON object with the following fields:

```
1 {
2   "id": "11",
3   "nombre": "Luis",
4   "apellidoP": "Hernandez",
5   "apellidoM": "Aguilar",
6   "direccion": "Bosques",
7   "telefono": "8666632",
8   "estatus": "1"
9 }
```

4.- Prueba del método DELETE



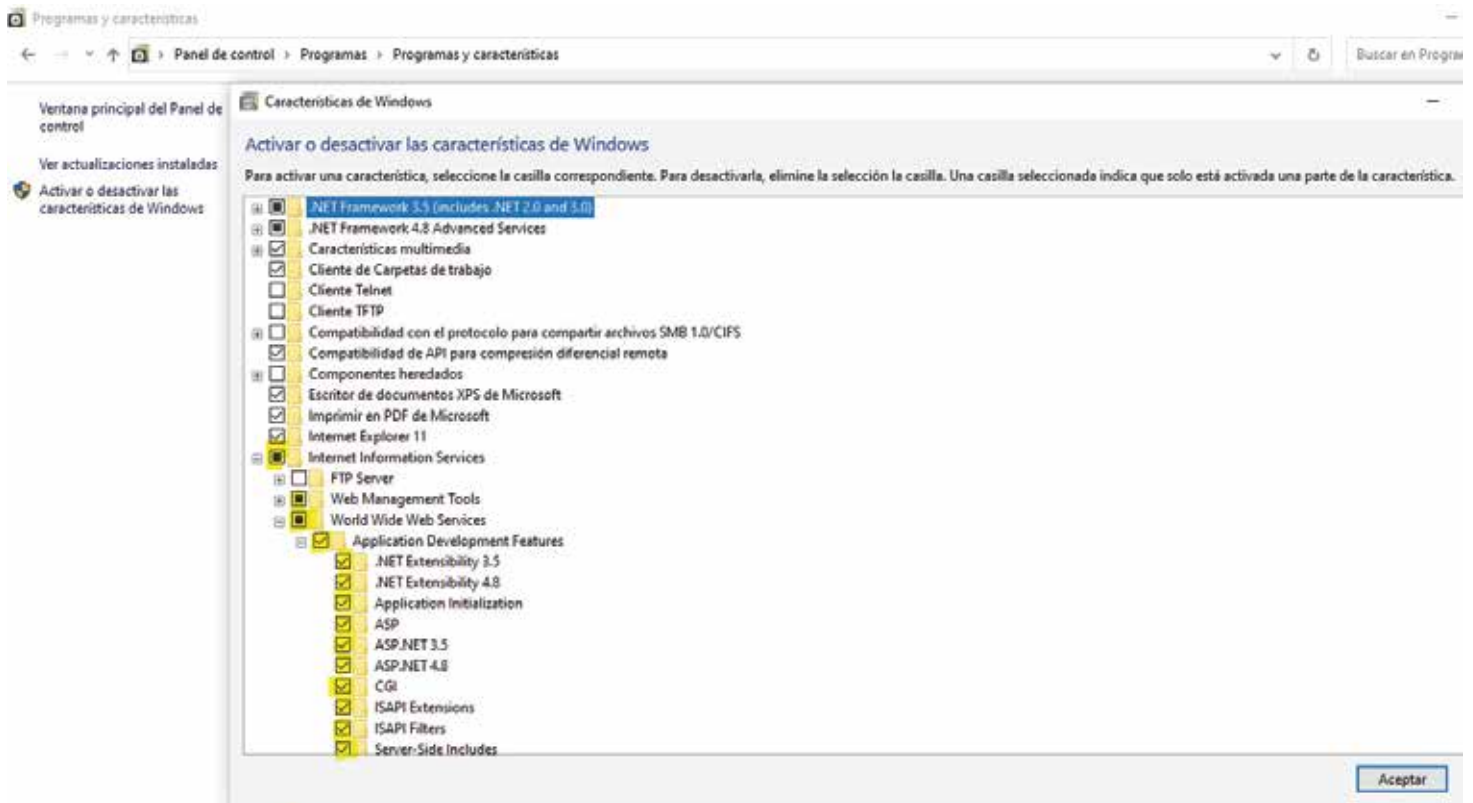
Lanzamiento de API RestFull por medio del servidor IIS de Windows



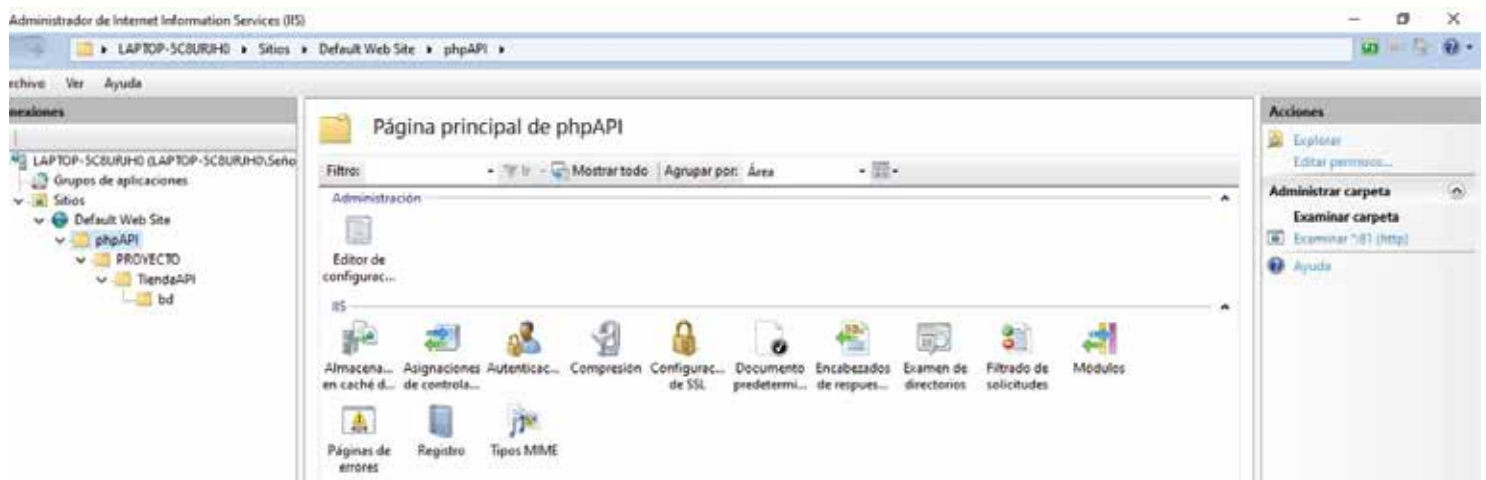
Para lanzar la API con ayuda del servidor de IIS de Windows, se navegó por las siguientes configuraciones de windows:

Panel de control > Programas > Programas y características

Y despues marcar las casillas sombreadas con amarillo:



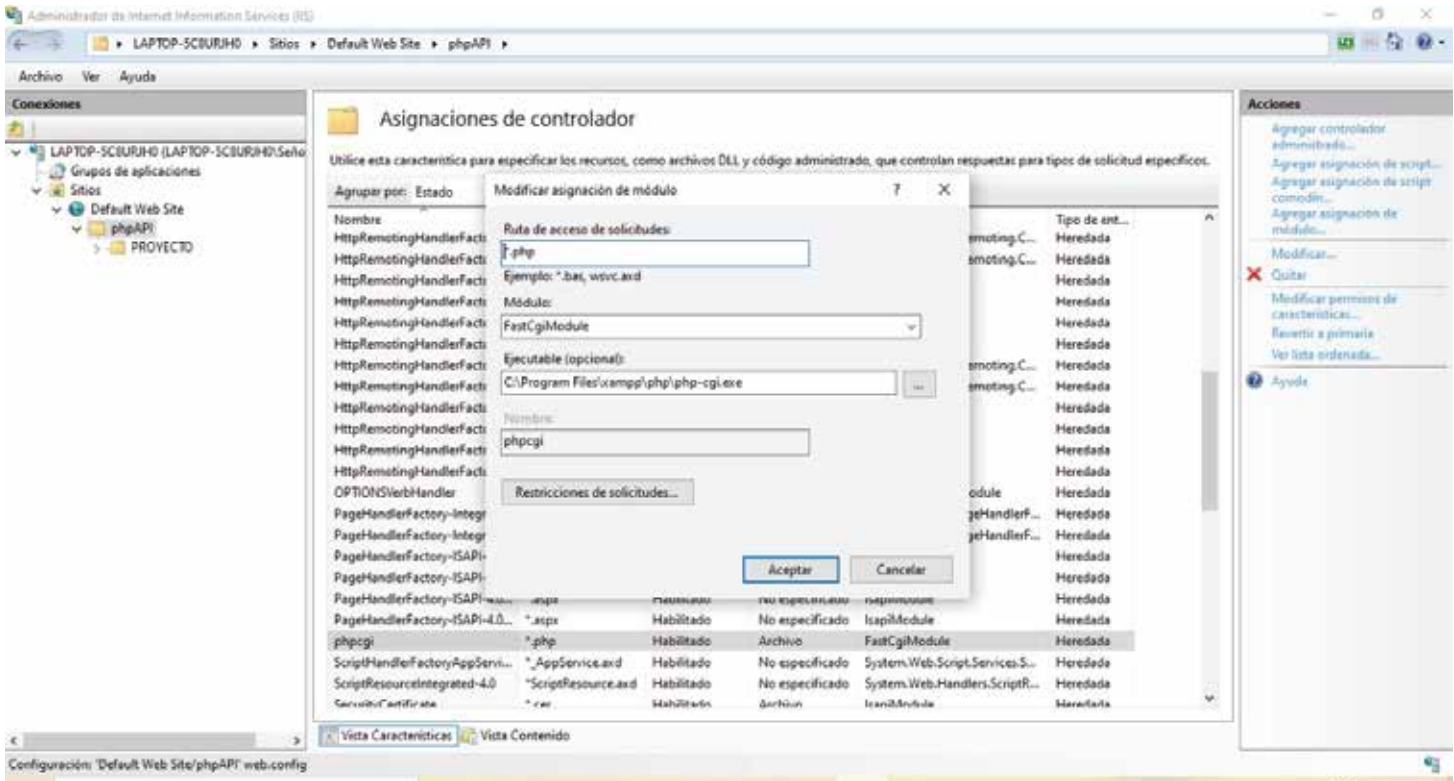
Lo siguiente es, una vez realizada la instalacion de los modulos necesarios para el servidor IIS, se ejecutó el administrador.



Para poder lanzar el proyecto al servicio IIS, es necesario colocar la carpeta con los archivos de la web api en el directorio del disco inicial de la maquina, el cual es el siguiente:

C:\inetpub\wwwroot

2.- Una vez cargados los archivos, en la pantalla principal del adimisitrador de servicios IIS, se navegó por asignaciones de controlador > Agregar asignación de controlador y realizar las configuraciones del controlador de php para la api restfuñ

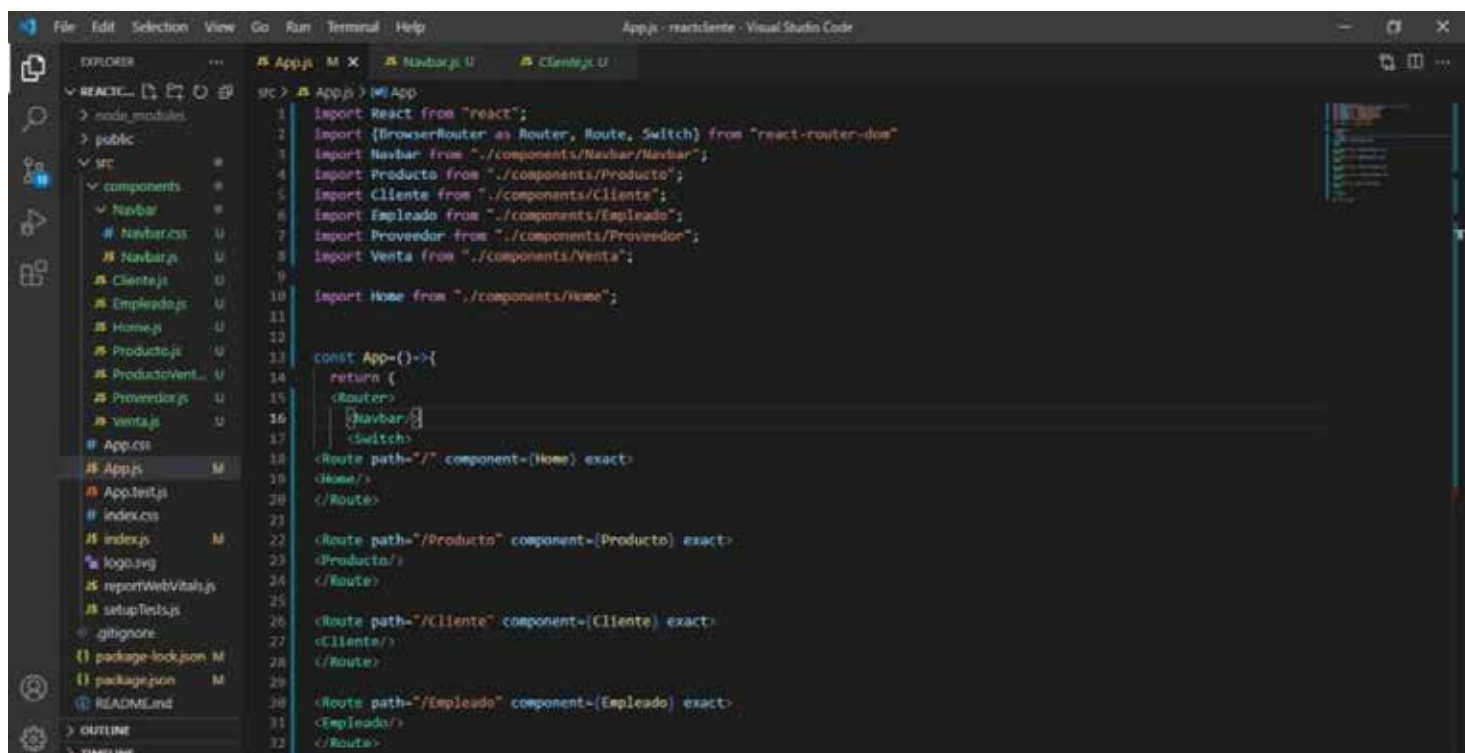


Implementación del cliente REACT

Para poder crear la aplicación REACT de Node JS, se realizó una instalación de los componentes necesarios por medio de la terminal de visual studio code, siendo los siguientes comandos utilizados:

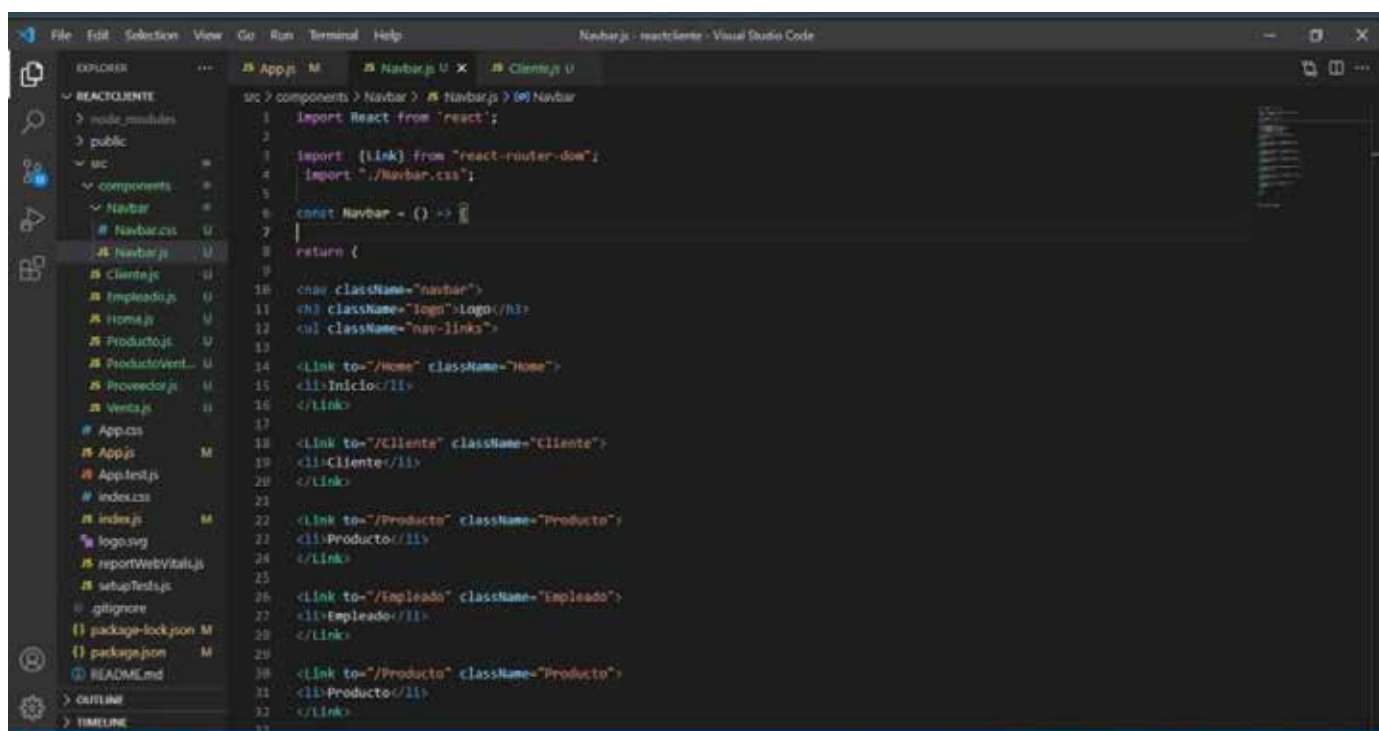
- `npx create-react-app my-app`
- `[npx react-router-dom`

Primero, en la clase App.js se realizaron las siguientes configuraciones de las rutas de acceso a cada una de las secciones de la aplicación, es decir, el muestreo de las tablas de la base de datos:



```
1 import React from "react";
2 import {BrowserRouter as Router, Route, Switch} from "react-router-dom";
3 import NavBar from "../components/NavBar/NavBar";
4 import Producto from "../components/Producto";
5 import Cliente from "../components/Cliente";
6 import Empleado from "../components/Empleado";
7 import Proveedor from "../components/Proveedor";
8 import Venta from "../components/Venta";
9
10 import Home from "../components/Home";
11
12
13 const App=()=>{
14   return (
15     <Router>
16       <NavBar/>
17       <Switch>
18         <Route path="/" component={Home} exact>
19           <Home/>
20         </Route>
21         <Route path="/Producto" component={Producto} exact>
22           <Producto/>
23         </Route>
24         <Route path="/Cliente" component={Cliente} exact>
25           <Cliente/>
26         </Route>
27         <Route path="/Empleado" component={Empleado} exact>
28           <Empleado/>
29         </Route>
30       </Switch>
31     </Router>
32   )
33 }
```

En la clase NavBar.js se crearon los controladores respectivos para redirigir a cada una de los directorios especificados anteriormente.



```
1 import React from "react";
2
3 import {Link} from "react-router-dom";
4 import "../NavBar.css";
5
6 const NavBar = () => {
7   return (
8     <div className="navbar">
9       
10       <ul className="nav-links">
11         <li><Link to="/" className="Home">Inicio</Link></li>
12         <li><Link to="/Cliente" className="Cliente">Cliente</Link></li>
13         <li><Link to="/Producto" className="Producto">Producto</Link></li>
14         <li><Link to="/Empleado" className="Empleado">Empleado</Link></li>
15       </ul>
16     </div>
17   )
18 }
```

Luego se continuó con la creación de cada una de las clases a utilizar para los métodos CRUD de las tablas de la base de datos. En este caso, se mostrará como ejemplo la clase de la tabla Cliente:

En el inicio de la clase se añadió la URL base con el enlace local de la api, además de la definición de las variables a utilizar en los llamados a los métodos:

```
src > components > Clientes > Cliente > handleChange > setFrameworkSeleccionado() callback
1 import React, {useState, useEffect} from 'react';
2 import 'bootstrap/dist/css/bootstrap.min.css';
3 import {Modal, ModalBody, ModalFooter, ModalHeader} from 'reactstrap';
4 import axios from 'axios';
5
6 const Cliente=()=>{
7   const baseUrl="http://localhost:81/TiendaAPI/cliente.php";
8   const [data, setData]=useState({});
9   const [modalInsertar, setModalInsertar]= useState(false);
10  const [modalEditar, setModalEditar]= useState(false);
11  const [modalEliminar, setModalEliminar]= useState(false);
12  const [frameworkSeleccionado, setFrameworkSeleccionado]=useState({
13    idcliente: '',
14    nombre: '',
15    apellidoP: '',
16    apellidoM: '',
17    direccion: '',
18    telefono: '',
19    estatus: ''
20  });
21
22  const handleChange=e=>{
23    const {name, value}=e.target;
24    setFrameworkSeleccionado((prevState)=>({
25      ...prevState,
26      [name]: value
27    }));
28    console.log(frameworkSeleccionado);
29  }
30
31  const abrirCerrarModalInsertar=()=>{
32    setModalInsertar(!modalInsertar);
```

```
src > components > Clientes > Cliente > handleChange > setFrameworkSeleccionado() callback
29  }
30
31  const abrirCerrarModalInsertar=()=>{
32    setModalInsertar(!modalInsertar);
33  }
34
35  const abrirCerrarModalEditar=()=>{
36    setModalEditar(!modalEditar);
37  }
38
39  const abrirCerrarModalEliminar=()=>{
40    setModalEliminar(!modalEliminar);
41  }
42
43  const petitionGet=async()=>{
44    await axios.get(baseUrl)
45      .then(response=>{
46        setData(response.data);
47      }).catch(error=>{
48        console.log(error);
49      })
50  }
51
52  const petitionPost=async()=>{
53    var f = new FormData();
54    f.append("nombre", frameworkSeleccionado.nombre);
55    f.append("apellidoP", frameworkSeleccionado.apellidoP);
56    f.append("apellidoM", frameworkSeleccionado.apellidoM);
57    f.append("direccion", frameworkSeleccionado.direccion);
58    f.append("telefono", frameworkSeleccionado.telefono);
59    f.append("estatus", frameworkSeleccionado.estatus);
60    f.append("METHOD", "POST");
```

En los siguientes formularios se realizó la configuración de los botones de la pagina principal y los llamados a los métodos del crud.

```
File Edit Selection View Go Run Terminal Help Clientes - reactcliente - Visual Studio Code
src > components > Clientes > Cliente > handleChange > setFrameworkSeleccionado() callback
68 }
69
70 const petitionPut-async()->{
71   var f = new FormData();
72   f.append("nombre", frameworkSeleccionado.nombre);
73   f.append("apellidoP", frameworkSeleccionado.apellidoP);
74   f.append("apellidoM", frameworkSeleccionado.apellidoM);
75   f.append("direccion", frameworkSeleccionado.direccion);
76   f.append("telefono", frameworkSeleccionado.telefono);
77   f.append("estatus", frameworkSeleccionado.estatus);
78   f.append("METHOD", "PUT");
79   await axios.post(baseUrl, f, {params: {idCliente: frameworkSeleccionado.idCliente}})
80   .then(response=>{
81     var dataNueva= data;
82     dataNueva.map(framework=>{
83       if(framework.idCliente===frameworkSeleccionado.idCliente){
84         framework.nombre=frameworkSeleccionado.nombre;
85         framework.apellidoP=frameworkSeleccionado.apellidoP;
86         framework.apellidoM=frameworkSeleccionado.apellidoM;
87         framework.direccion=frameworkSeleccionado.direccion;
88         framework.telefono=frameworkSeleccionado.telefono;
89         framework.estatus=frameworkSeleccionado.estatus;
90       }
91     });
92   });
93   setData(dataNueva);
94   abrirCerrarModalEditar();
95   }).catch(error=>{
96     console.log(error);
97   })
98 }
99
100 const petitionDelete-async()->{
```

```
File Edit Selection View Go Run Terminal Help Clientes - reactcliente - Visual Studio Code
src > components > Clientes > Cliente > handleChange > setFrameworkSeleccionado() callback
100
101 const petitionDelete-async()->{
102   var f = new FormData();
103   f.append("METHOD", "DELETE");
104   await axios.post(baseUrl, f, {params: {idCliente: frameworkSeleccionado.idCliente}})
105   .then(response=>{
106     setData(data.filter(framework=>framework.idCliente!==frameworkSeleccionado.idCliente));
107     abrirCerrarModalEliminar();
108   }).catch(error=>{
109     console.log(error);
110   })
111 }
112
113 const seleccionarFramework=(framework, caso)->{
114   setFrameworkSeleccionado(framework);
115   (caso==="Editar")?
116     abrirCerrarModalEditar():
117     abrirCerrarModalEliminar()
118 }
119
120 useEffect(()=>{
121   petitionGet();
122 },[])
123
124 return (
125   <div style={{textAlign: 'center'}}>
126     <button className="btn btn-success" onClick={()=>abrirCerrarModalInsertar()}>Insertar</button>
127     <br /><br />
128     <table className="table table-striped">
129       <thead>
130         <tr>
```



```
File Edit Selection View Go Run Terminal Help
Cliente.js - reactcliente - Visual Studio Code

JS App.js M JS Navbar.js U JS Cliente.js U X
src > components > JS Cliente.js > [w] Cliente > [w] handleChange > [w] setFrameworkSeleccionado() callback

Search (Ctrl+Shift+F) (
125 <div style={{textAlign: 'center'}}>
126 <br />
127 <button className="btn btn-success" onClick={()=>abrirCerrarModalInsertar()}>Insertar</button>
128 <br /><br />
129 <table className="table table-striped">
130 <thead>
131 <tr>
132 <th>ID</th>
133 <th>Nombre</th>
134 <th>Apellido</th>
135 <th>Apellido</th>
136 <th>Direccion</th>
137 <th>Telefono</th>
138 <th>Estatus</th>
139 <th>Acciones</th>
140 </tr>
141 </thead>
142 <tbody>
143 {data.map((framework=>{
144 <tr key={framework.idCliente}>
145 <td>{framework.idCliente}</td>
146 <td>{framework.nombre}</td>
147 <td>{framework.apellidoP}</td>
148 <td>{framework.apellidoM}</td>
149 <td>{framework.direccion}</td>
150 <td>{framework.telefono}</td>
151 <td>{framework.estatus}</td>
152 <td>
153 <button className="btn btn-primary" onClick={()=>seleccionarFramework(framework, "Editar")}>Editar</button> (" ")
154 <button className="btn btn-danger" onClick={()=>seleccionarFramework(framework, "Eliminar")}>Eliminar</button>
155 </td>
```

```
File Edit Selection View Go Run Terminal Help
Cliente.js - reactcliente - Visual Studio Code

JS App.js M JS Navbar.js U JS Cliente.js U X
src > components > JS Cliente.js > [w] Cliente > [w] handleChange > [w] setFrameworkSeleccionado() callback

164 <Modal isOpen={modalInsertar}>
165 <ModalHeader>Insertar Framework</ModalHeader>
166 <ModalBody>
167 <div className="form-group">
168 <label>Nombre: </label>
169 <br />
170 <input type="text" className="form-control" name="nombre" onChange={handleChange}/>
171 <br />
172 <label>Apellido P: </label>
173 <br />
174 <input type="text" className="form-control" name="apellidoP" onChange={handleChange}/>
175 <br />
176 <label>Apellido M: </label>
177 <br />
178 <input type="text" className="form-control" name="apellidoM" onChange={handleChange}/>
179 <br />
180 <label>Direccion: </label>
181 <br />
182 <input type="text" className="form-control" name="direccion" onChange={handleChange}/>
183 <br />
184 <label>Telefono: </label>
185 <br />
186 <input type="text" className="form-control" name="telefono" onChange={handleChange}/>
187 <br />
188 <label>Estatus: </label>
189 <br />
190 <input type="text" className="form-control" name="estatus" onChange={handleChange}/>
191 <br />
192 </div>
193 </ModalBody>
194 <ModalFooter>
```

```
File Edit Selection View Go Run Terminal Help Clientes - reactcliente - Visual Studio Code
src > components > Clientes > Cliente > handleChange > setFrameworkSeleccionado() callback
231
232
233 <Modal isOpen={modalEditar}>
234   <ModalHeader>editar </ModalHeader>
235   <ModalBody>
236     <div className="form-group">
237       <label>Nombre: </label>
238       <br />
239       <input type="text" className="form-control" name="nombre" onChange={handleChange} value={frameworkSeleccionado.id && frameworkSeleccionado.nombre} />
240     </div>
241     <label>Apellido p: </label>
242     <br />
243     <input type="text" className="form-control" name="apellidoP" onChange={handleChange} value={frameworkSeleccionado.id && frameworkSeleccionado.apellidoP} />
244     <br />
245     <label>Apellido m: </label>
246     <br />
247     <input type="text" className="form-control" name="apellidoM" onChange={handleChange} value={frameworkSeleccionado.id && frameworkSeleccionado.apellidoM} />
248     <br />
249     <label>Direccion: </label>
250     <br />
251     <input type="text" className="form-control" name="direccion" onChange={handleChange} value={frameworkSeleccionado.id && frameworkSeleccionado.direccion} />
252     <br />
253     <label>Telefono: </label>
254     <br />
255     <input type="text" className="form-control" name="telefono" onChange={handleChange} value={frameworkSeleccionado.id && frameworkSeleccionado.telefono} />
256     <br />
257     <label>Estatus: </label>
258     <br />
259     <input type="text" className="form-control" name="estatus" onChange={handleChange} value={frameworkSeleccionado.id && frameworkSeleccionado.estatus} />
260     <br />
261   </div>
262 </ModalBody>
263 </Modal>
```

```
File Edit Selection View Go Run Terminal Help Clientes - reactcliente - Visual Studio Code
src > components > Clientes > Cliente > handleChange > setFrameworkSeleccionado() callback
231 </div>
232 </ModalBody>
233 <ModalFooter>
234   <button className="btn btn-primary" onClick={()=>peticionPut()}>Editar</button>{" "}
235   <button className="btn btn-danger" onClick={()=>abrirCerrarModalEditar()}>Cancelar</button>
236 </ModalFooter>
237 </Modal>
238
239 <Modal isOpen={modalEliminar}>
240   <ModalBody>
241     ¿Estás seguro que deseas eliminar el Cliente {frameworkSeleccionado.id && frameworkSeleccionado.idCliente}?
242   </ModalBody>
243   <ModalFooter>
244     <button className="btn btn-danger" onClick={()=>peticionDelete()}>
245       Sí
246     </button>
247     <button
248       className="btn btn-secondary"
249       onClick={()=>abrirCerrarModalEliminar()}
250     >
251       No
252     </button>
253   </ModalFooter>
254 </Modal>
255
256 </div>
257 );
258 }
259
260
261 export default Cliente;
```

4.1.- Pruebas del cliente

Se realizaron pruebas con los métodos de CRUD con la tabla Cliente de la base de datos.

1.- Comprobación del método SELECT

Al lanzar la aplicación y redirigirse al controlador de la tabla clientes, se puede observar una tabla con todos los registros de dicha entidad y sus respectivos botones de eliminar y de editar.

Insertar								
IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	García	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq, Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodríguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villalón	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldua	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar

2.- Comprobación del método PUT

Se comprobó el método PUT para la inserción, el cual muestra una ventana en la cual se solicitan los datos a seleccionar en el nuevo registro.

IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	García	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq, Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodríguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villalón	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldua	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar

Nombre:

Luis Angel

Apellido P:

Leyva

Apellido M:

García

Dirección:

Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.

Teléfono:

+8661681656

Estatus:

1

Insertar

Cancelar

- Producto
- Proveedor
- Venta

Una vez realizado el insert, se comprobaron los cambios realizados:

Insertar

IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	García	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq. Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodriguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villaral	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldúa	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar
10	Luis Ángel	Leyva	García	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+8661681656	1	Editar	Eliminar

3.- Comprobación del método POST

Se actualizaron los datos de un registro de la tabla una vez guardados se comprobaron los cambios:

Editar

Proveedores

Venta

IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	García	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq. Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodriguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villaral	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldúa	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar
10	Luis Ángel	Leyva	García	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+8661681656	1	Editar	Eliminar

Insertar

IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	Garcia	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq. Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodriguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villarial	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldua	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar
10	Actualización	Actualización	Actualización	Actualización	Actualización	1	Editar	Eliminar

4.- Comprobación del método DELETE

Al dar clic en el botón de eliminar, se comprobó el funcionamiento del método delete exitosamente:

¿Estás seguro que deseas eliminar el Cliente 10?								
<div>SI No</div>								
IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	Garcia	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq. Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodriguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villarial	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldua	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar
10	Actualización	Actualización	Actualización	Actualización	Actualización	1	Editar	Eliminar

Insertar

IDENTIFICACIÓN	Nombre	Apellido	Apellido	Dirección	Teléfono	Estatus	Acciones	
1	prueba con mysql	Leyva	Garcia	Cd. Deportiva, Cd Deportiva, 25750 Monclova, Coah.	+866-168-16-56	1	Editar	Eliminar
2	José Luis	Hernandez	Garza	Av. Huemac, Brasil Esq. Anáhuac, 25750 Monclova, Coah.	+866-198-11-56	1	Editar	Eliminar
3	Flor Isabel	Flores	Perez	Brasil #201, Guadalupe, 25750 Monclova, Coah.	+844-188-56-34	1	Editar	Eliminar
4	Marcos	Mireles	Tovar	Valparaíso 212, Guadalupe, 25750 Monclova, Coah.	+866-111-13-59	1	Editar	Eliminar
5	Nataly	Meza	Rodriguez	Calle Guatemala S/N, Guadalupe, 25750 Monclova, Coah.	+866-677-23-12	1	Editar	Eliminar
6	Samuel Israel	Vázquez	Villarial	Blvd Harold R. Pape 6201, Guadalupe, 25750 Monclova, Coah.	+844-190-23-45	1	Editar	Eliminar
7	Juan de Dios	Del Bosque	Saldua	Carretera 30, Magisterio, 25716 Frontera, Coah.	+866-166-78-01	1	Editar	Eliminar

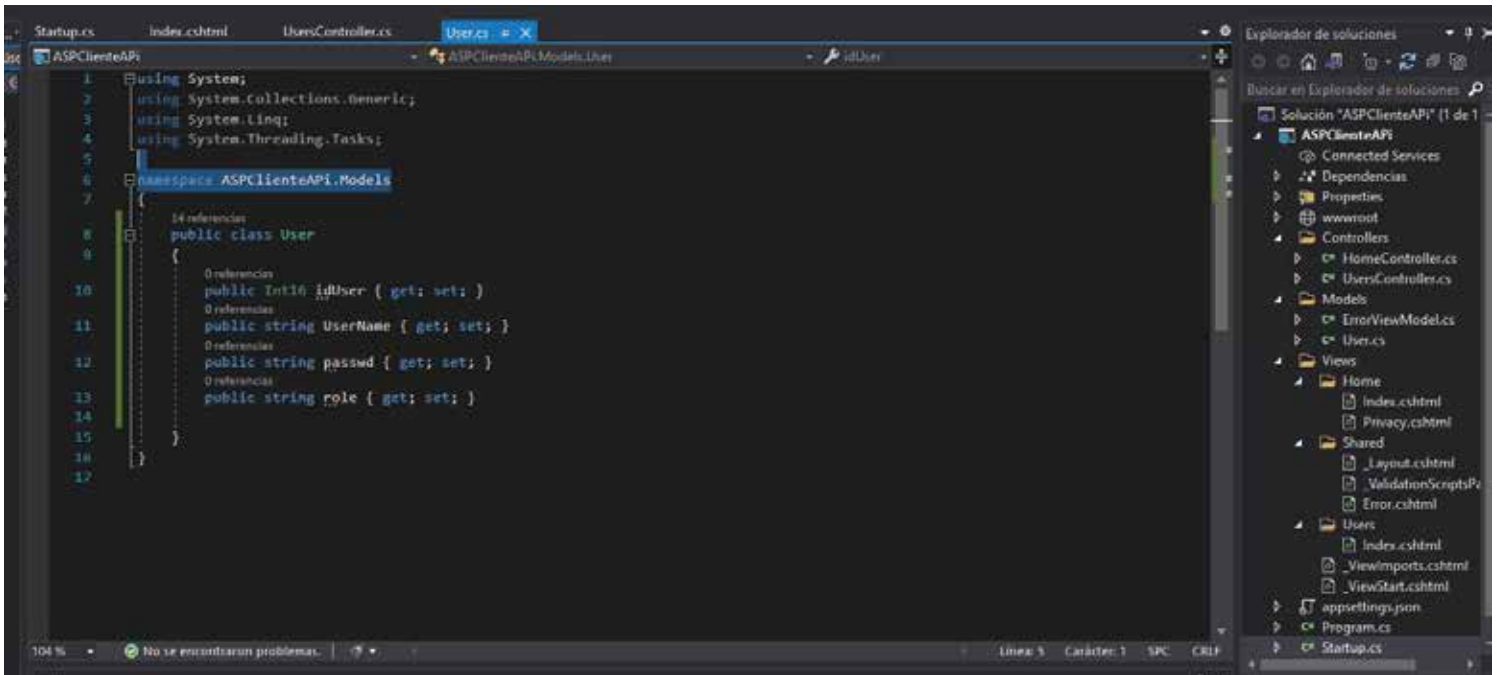
Implementación de cliente: Aplicación Web de ASP NET CORE



Implementación de cliente: Aplicación Web de ASP NET CORE

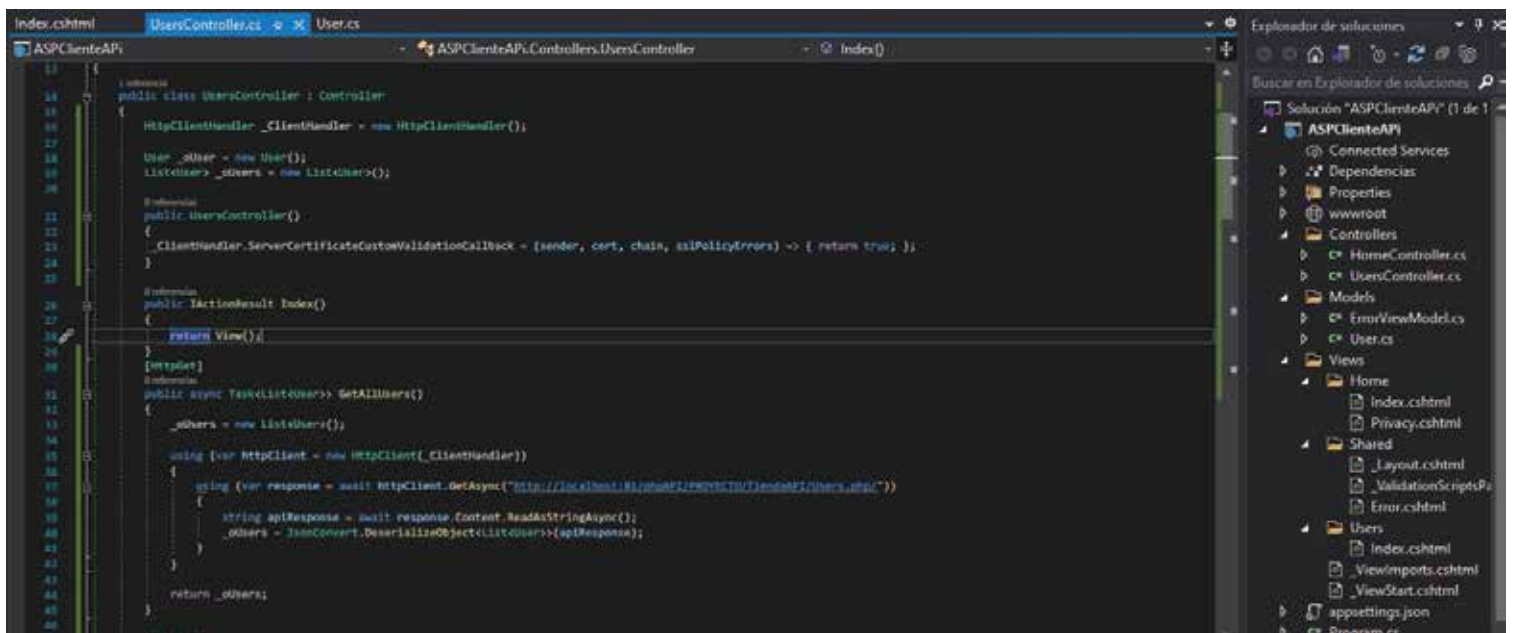
ASP.NET Core es un marco multiplataforma de código abierto y de alto rendimiento que tiene como finalidad compilar aplicaciones modernas conectadas a Internet y habilitadas para la nube

Una vez creada la aplicación Web de ASP Net Core, se crearon las clases especiales para cada una de las entidades de la base de datos, especificando un método para recibir la información de cada uno de los campos de las tablas. A continuación, se muestra como ejemplo la clase de la tabla “Usuarios”:



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Threading.Tasks;
5
6 namespace ASPClienteAPI.Models
7 {
8     public class User
9     {
10         public int idUser { get; set; }
11         public string UserName { get; set; }
12         public string passwd { get; set; }
13         public string role { get; set; }
14     }
15 }
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Una vez realizado, se creo un nuevo controlador para cada una de las clases de las tablas, llevando su respectivo nombre. En dichos controladores se especificaron los métodos de CRUD a utilizar. El método Select que muestra las tablas y su contenido, es el siguiente:



```
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
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

A continuación, los métodos GET para el muestreo de registros según una ID especificada, y el método POST para la actualización de registros:

```
[HttpGet]
0 referencias
public async Task<User> GetById(int userId)
{
    _oUser = new User();

    using (var httpClient = new HttpClient(_clientHandler))
    {
        using (var response = await httpClient.GetAsync("http://localhost:81/phpAPI/PROYECTO/TiendaAPI/Users.php/" + userId))
        {
            string apiResponse = await response.Content.ReadAsStringAsync();
            _oUser = JsonConvert.DeserializeObject<User>(apiResponse);
        }
    }

    return _oUser;
}

[HttpPost]
0 referencias
public async Task<User> AddUpdateUser(User user)
{
    _oUser = new User();

    using (var httpClient = new HttpClient(_clientHandler))
    {
        StringContent content = new StringContent(JsonConvert.SerializeObject(user), Encoding.UTF8, "application/json");
        using (var response = await httpClient.PostAsync("http://localhost:81/phpAPI/PROYECTO/TiendaAPI/Users.php/", content))
        {
            string apiResponse = await response.Content.ReadAsStringAsync();
            _oUser = JsonConvert.DeserializeObject<User>(apiResponse);
        }
    }

    return _oUser;
}
```

Por último, el método DELETE para la eliminación de los registros de las tablas

```
}

[HttpDelete]
0 referencias
public async Task<string> Delete(int userId)
{
    string message = "";

    using (var httpClient = new HttpClient(_clientHandler))
    {
        using (var response = await httpClient.DeleteAsync("http://localhost:81/phpAPI/PROYECTO/TiendaAPI/Users.php/" + userId))
        {
            message = await response.Content.ReadAsStringAsync();
        }
    }

    return message;
}
```


Por último, se creó un formulario para cada una de las tablas y sus procesos CRUD, añadiendo botones y una tabla con los registros, además de cuadros de texto con los cuales ingresar nuevos datos a la tabla. La siguiente parte del código se utiliza para crear los controladores en el formulario:

```
Index.cshtml  UsersController.cs  User.cs
1  ViewBag.Title = "User";
2  Layout = "~/Views/Shared/_Layout.cshtml";
3
4
5
6  <script src="~/lib/jquery/dist/jquery.min.js"></script>
7
8  <h2>Add a User</h2>
9
10 <div class="form-group" style="display:none;">
11   <input id="txtUserId" class="form-control" name="UserId" />
12 </div>
13 <div class="form-group">
14   <label for="Name">Name:</label>
15   <input id="txtUserName" class="form-control" name="Name" />
16 </div>
17 <div class="form-group">
18   <label for="Passwd">Contraseña:</label>
19   <input id="txtUserPasswd" class="form-control" name="Passwd" />
20 </div>
21 <div class="form-group">
22   <label for="Roll">Roll:</label>
23   <input id="txtUserRole" class="form-control" name="Roll" />
24 </div>
25 <div class="text-center panel-body">
26   <button id="btnSave" class="btn btn-sm btn-primary">Save</button>
27 </div>
28
29 <button onclick="GetAllUsers()" class="btn btn-primary">Load User</button>
30
31 <br />
32
33 <h2>User List</h2>
34 <table id="tblUser" class="table table-sm table-striped table-bordered m-2">
35   <thead>
36     <tr>
37       <th>User id</th>
38       <th>User name</th>
39       <th>User password</th>
40       <th>User role</th>
41     </tr>
42   </thead>
43   <tbody>
44   </tbody>
45 </table>
46
47 <script type="text/javascript">
48   $(document).ready(function () {
49   });
50
51   function GetAllUsers() {
52     $.getJSON("/Users/GetAllUsers", function (users) {
53       $("#tblUser tbody tr").remove();
54       $.map(users, function (user) {
55         $("#tblUser tbody").append("<tr>"
56           + " <td>" + user.userId + "</td>"
57           + " <td>" + user.UserName + "</td>"
58           + " <td>" + user.passwd + "</td>"
59           + " <td>" + user.role + "</td>"
60           + " <td>"
61             + " <button class='btn-success' onclick='Edit(" + user.userId + ")' style='margin-right:5px;'>Edit</button>"
62             + " <button class='btn-danger' onclick='Delete(" + user.userId + ")'>Delete</button>"
63           + " </td>"
64           + " </tr>");
65       });
66     });
67   }
68
69   function Edit(userId) {
70     if (userId > 0) {
71       $.getJSON("/Users/GetById?userId=" + userId, function (user) {
72         $("#txtUserId").val(user.userId);
73         $("#txtUserName").val(user.UserName);
74         $("#txtUserPasswd").val(user.passwd);
75         $("#txtUserRole").val(user.role);
76       });
77     }
78   }
79 </script>
```

Enseguida, se le dio función a los botones y cajas de texto para el envío de los parámetros a los métodos GET, POST, DELETE Y PUT.

```
Index.cshtml  UsersController.cs  User.cs
37   <th>User id</th>
38   <th>User name</th>
39   <th>User password</th>
40   <th>User role</th>
41 </tr>
42 </thead>
43 <tbody></tbody>
44 </table>
45
46 <script type="text/javascript">
47   $(document).ready(function () {
48   });
49
50   function GetAllUsers() {
51     $.getJSON("/Users/GetAllUsers", function (users) {
52       $("#tblUser tbody tr").remove();
53       $.map(users, function (user) {
54         $("#tblUser tbody").append("<tr>"
55           + " <td>" + user.userId + "</td>"
56           + " <td>" + user.UserName + "</td>"
57           + " <td>" + user.passwd + "</td>"
58           + " <td>" + user.role + "</td>"
59           + " <td>"
60             + " <button class='btn-success' onclick='Edit(" + user.userId + ")' style='margin-right:5px;'>Edit</button>"
61             + " <button class='btn-danger' onclick='Delete(" + user.userId + ")'>Delete</button>"
62           + " </td>"
63           + " </tr>");
64       });
65     });
66   }
67
68   function Edit(userId) {
69     if (userId > 0) {
70       $.getJSON("/Users/GetById?userId=" + userId, function (user) {
71         $("#txtUserId").val(user.userId);
72         $("#txtUserName").val(user.UserName);
73         $("#txtUserPasswd").val(user.passwd);
74         $("#txtUserRole").val(user.role);
75       });
76     }
77   }
78 </script>
```

```
Index.cshtml  UsersController.cs  User.cs
67
68
69 function Edit(userId) {
70     if (userId > 0) {
71         $.getJSON("/Users/GetById?userId=" + userId, function (user) {
72             $("#txtUserId").val(user.userId);
73             $("#txtUserName").val(user.UserName);
74             $("#txtUserPasswd").val(user.passwd);
75             $("#txtUserRole").val(user.role);
76         });
77     }
78 }
79
80 function Delete(userId) {
81     if (userId > 0) {
82         $.ajax({
83             url: '/User/Delete?userId=' + userId,
84             type: 'DELETE',
85             dataType: 'json',
86             success: function (ex) {
87                 console.log('Error in Operation');
88             }
89         })
90     }
91 }
92
93 $( "btnSave" ).click(function () {
94     var oUser = {
95         userId: $("#txtUserId").val(),
96         name: $("#txtUserName").val(),
97         passwd: $("#txtUserPasswd").val(),
98         role: $("#txtUserRole").val(),
99     };
100
101     $.post("/Users/AddUpdateUser", oUser)
102         .done(function (data) {
103             GetAllUsers
104         });
105 });
106 </script>
```

Finalmente, la aplicación se ejecutó para comprobar los ajustes realizados, en este caso, mediante la realización de las operaciones CRUD con la tabla Usuarios:

1.- Prueba select con el cliente:

ASPCienteAPI Home Privacy

Add a User

Name:

Contraseña:

Roll:

Save

Load User

User List

User id	User name	User password	User role
---------	-----------	---------------	-----------

© 2021 - ASPCienteAPI - Privacy

Como se puede observar, al dar clic en el botón “load user” se muestran los registros de la tabla usuarios:

Save

Load User

User List

User id	User name	User password	User role	
1	usuario1	81dc9bdb52d04dc20036dbd8313ed055	Administrador	<div>EditDelete</div>
1	usuario2	81dc9bdb52d04dc20036dbd8313ed055	Usuario	<div>EditDelete</div>

2.- Inserción de datos con el método PUT.

También se comprobó la inserción de datos:

Add a User

Name:

Usuario3

Contraseña:

Roll:

Usuario

Save

Al pulsar nuevamente el botón “Load Users” se comprobó que los cambios fueron realizados correctamente:

Save

Load User

User List

User id	User name	User password	User role	
1	usuario1	81dc9bdb52d04dc20036dbd8313ed055	Administrador	<div>EditDelete</div>
2	usuario2	81dc9bdb52d04dc20036dbd8313ed055	Usuario	<div>EditDelete</div>
3	usuario3	81dc9bdb52d04dc20036dbd8313ed055	Usuario	<div>EditDelete</div>

3.- Se comprobó la función de eliminar al dar clic en el respectivo botón del tercer registro de la tabla:

Load User

Save

User List

User id	User name	User password	User role	
1	usuario1	81dc9bdb52d04dc20036dbd8313ed055	Administrador	<div>EditDelete</div>
2	usuario2	81dc9bdb52d04dc20036dbd8313ed055	Usuario	<div>EditDelete</div>
3	usuario3	81dc9bdb52d04dc20036dbd8313ed055	Usuario	<div>EditDelete</div>

Load User

Save

User List

User id	User name	User password	User role	
1	usuario1	81dc9bdb52d04dc20036dbd8313ed055	Administrador	<div>EditDelete</div>
1	usuario2	81dc9bdb52d04dc20036dbd8313ed055	Usuario	<div>EditDelete</div>