

Universidad de Nariño.

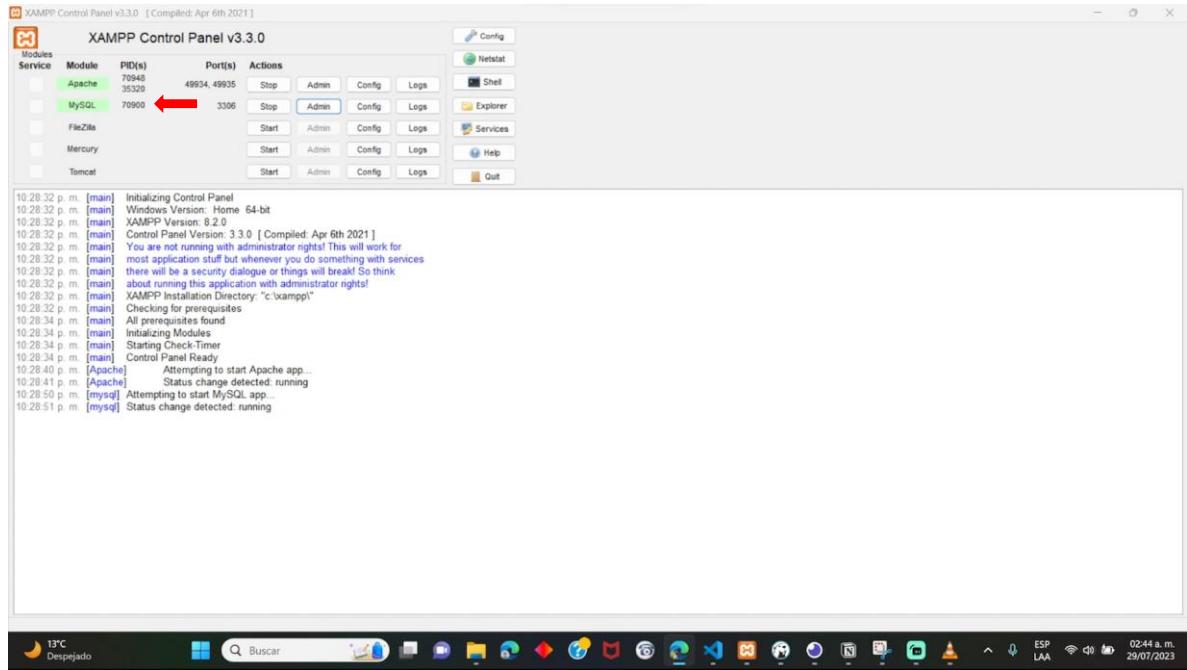
Ingeniería de Sistemas.

Diplomado de actualización en nuevas tecnologías para el desarrollo de Software.

Taller Unidad 2 Backend

Presentado por: Dana Criollo.

1. Crear una base de datos MYSQL/MARIADB que permita llevar el registro de un FRUVER (FRUTAS VERDURAS), así como también el proceso de solicitud de compra de estas.
- Como primer paso, inicié el servicio de MySQL desde XAMPP:



- Desde el phpMyAdmin creé una base de datos con el nombre 'dcfruver':

The screenshot shows the phpMyAdmin interface on a Windows desktop. In the top navigation bar, there are tabs for 'Bases de datos', 'SQL', 'Estado actual', 'Cuentas de usuarios', 'Exportar', 'Importar', 'Configuración', 'Replicación', 'Variables', and 'Más'. Below the tabs, a search bar contains 'utf8mb4_general_ci' and a red arrow points to the 'Crear' (Create) button. A modal window titled 'Crear base de datos' is open, showing a text input field with 'dcfruver' and a dropdown menu also set to 'utf8mb4_general_ci'. Another red arrow points to the 'Crear' button in this modal. The main list of databases on the left includes 'academica', 'ej1laravel', 'ej2laravel', 'fruver', 'fruver1', 'information_schema', 'laboratorios', 'laravel', 'laravelprueba', 'mysql', 'paciente', 'performance_schema', 'phpmyadmin', 'pruebalaravel', 'registro', 'serviteca', 'sesion', 'terapias', 'test', and 'trabajos_fin_carrera'. The bottom status bar shows the date and time as '29/07/2023 02:44 a.m.'.

- Me conecté a la base de datos 'dcfruver' desde DBeaver:

The screenshot shows the DBeaver interface on a Windows desktop. The top menu bar includes 'Archivo', 'Editar', 'Navegar', 'Buscar', 'Editor SQL', 'Base de Datos', 'Ventana', and 'Ayuda'. The 'Base de Datos' tab is selected. On the left, a 'Navegador de Bases de Datos' panel shows a tree structure with 'Proyectos' and 'dcfruver - localhost:3306'. A connection dialog box is open in the center, titled 'Conectar a base de datos'. It shows 'MySQL' as the driver and has tabs for 'General', 'Driver properties', 'SSH', 'Proxy', and 'SSL'. Under 'General', the 'Server' section is shown with 'Connect by:' set to 'Host' (radio button selected), 'URL:' set to 'jdbc:mysql://localhost:3306/dcfruver', 'Server Host' set to 'localhost' (with a red double-headed arrow below it), and 'Database' set to 'dcfruver' (with a red double-headed arrow below it). The 'Authentication (Database Native)' section shows 'Nombre de usuario:' as 'root' and 'Contraseña:' as empty. The 'Advanced' section includes 'Server Time Zone' (set to 'Auto-detect'), 'Local Client' (set to 'MySQL Binaries'), and a note about using variables. At the bottom of the dialog are buttons for 'Probar conexión...' (Test connection...), 'Anterior' (Previous), 'Siguiente' (Next), 'Finalizar' (Finish), and 'Cancelar' (Cancel). The bottom status bar shows the date and time as '29/07/2023 02:45 a.m.'

- En la base de datos 'dcfruver' creé las siguientes tablas.

Tabla 'usuario' con los atributos 'idusuario', 'nombreusu', 'contrasenausu' y 'rol':

The screenshot shows the DBVisualizer interface with the following details:

- Project:** dcfruver - localhost:3306
- Table Name:** usuario (highlighted by a red arrow)
- Engine:** InnoDB
- Auto Increment:** 1
- Charset:** latin1
- Collation:** latin1_swedish_ci
- Columns:**

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
idusuario	1	int(11)	[x]	[x]	PRI		auto_increment		
nombreusu	2	varchar(100)	[x]						
contrasenausu	3	varchar(255)	[x]						
rol	4	varchar(20)	[x]						

Tabla 'productos' con los atributos 'idproducto', 'nomproducto', 'categoriaproducto', 'descripcionproducto', 'precioproducto', 'stockproducto' e 'imgproducto':

The screenshot shows the DBVisualizer interface with the following details:

- Project:** dcfruver - localhost:3306
- Table Name:** productos (highlighted by a red arrow)
- Engine:** InnoDB
- Auto Increment:** 1
- Charset:** latin1
- Collation:** latin1_swedish_ci
- Columns:**

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
idproducto	1	int(11)	[x]	[x]	PRI		auto_increment		
nomproducto	2	varchar(100)	[x]						
categoriaproducto	3	varchar(50)	[x]						
descripcionproducto	4	varchar(100_)	[x]						
precioproducto	5	int(6)	[x]						
stockproducto	6	int(11)	[x]						
imgproducto	7	longblob	[x]						

Tabla 'clientes' con los atributos 'cedulacliente', 'nomcliente', 'dircliente', 'telcliente' y 'correocliente':

DBVisualizer 23.0.5 - clientes

Navegador de Bases de Datos > Proyectos

Table Name: clientes

Engine: InnoDB

Auto Increment: 0

Charset: latin1

Collation: latin1_swedish_ci

Columns

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
cedulacliente	1	varchar(10)	[x]		PRI				
nomcliente	2	varchar(100)	[x]						
dircliente	3	varchar(100)	[x]						
telcliente	4	varchar(10)	[x]						
correocliente	5	varchar(100)	[x]						

Project - General >

5 elementos

Save... Revert Refresh

Tabla 'pedidos' con los atributos 'idpedido', 'cedulacliente', 'fechapedido', 'total' y 'estado':

DBVisualizer 23.0.5 - pedidos

Navegador de Bases de Datos > Proyectos

Table Name: pedidos

Engine: InnoDB

Auto Increment: 1

Charset: latin1

Collation: latin1_swedish_ci

Columns

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
idpedido	1	int(11)	[x]	[x]	PRI		auto_increment		
cedulacliente	2	varchar(10)	[x]		MUL				
fechapedido	3	date	[x]						
total	4	int(11)	[x]						
estado	5	varchar(1)	[x]						

Project - General >

5 elementos

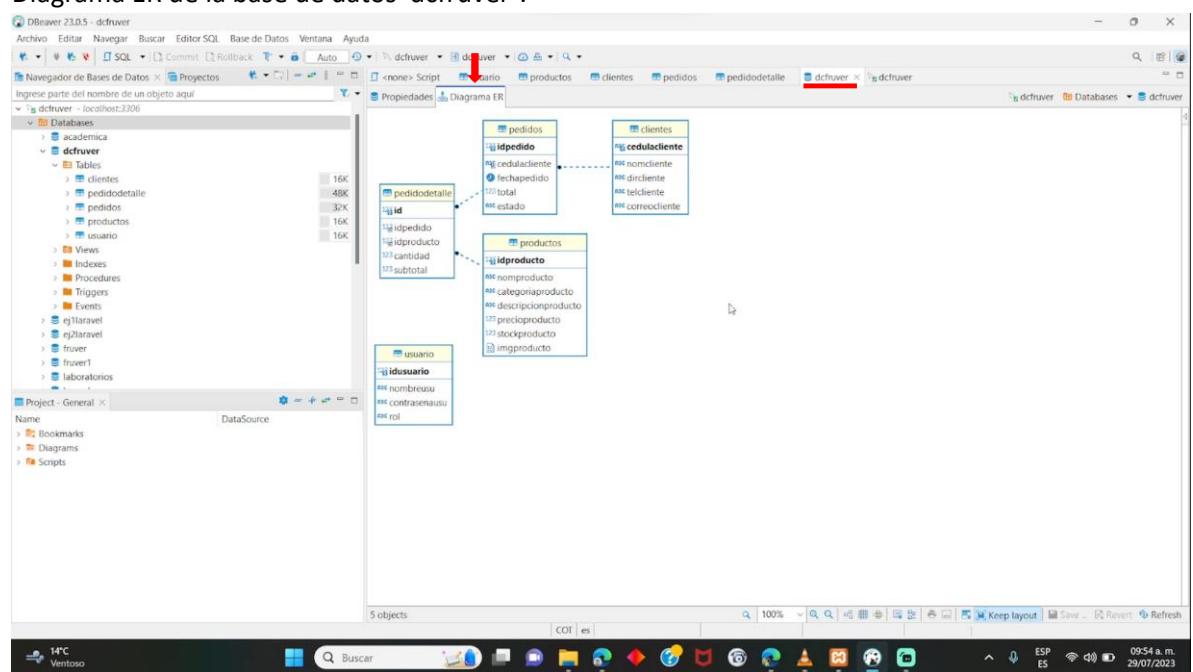
Save... Revert Refresh

Tabla 'pedidodetalie' con los atributos 'id', 'idpedido', 'idproducto', 'cantidad' y 'subtotal':

The screenshot shows the DBVisualizer interface for creating a table named 'pedidodetalie'. The table has the following structure:

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
id	1	int(11)	[x]	[x]	PRI		auto increment		
idpedido	2	int(11)	[x]	[]	MUL				
idproducto	3	int(11)	[x]	[]	MUL				
cantidad	4	int(11)	[x]	[]					
subtotal	5	int(11)	[x]	[]					

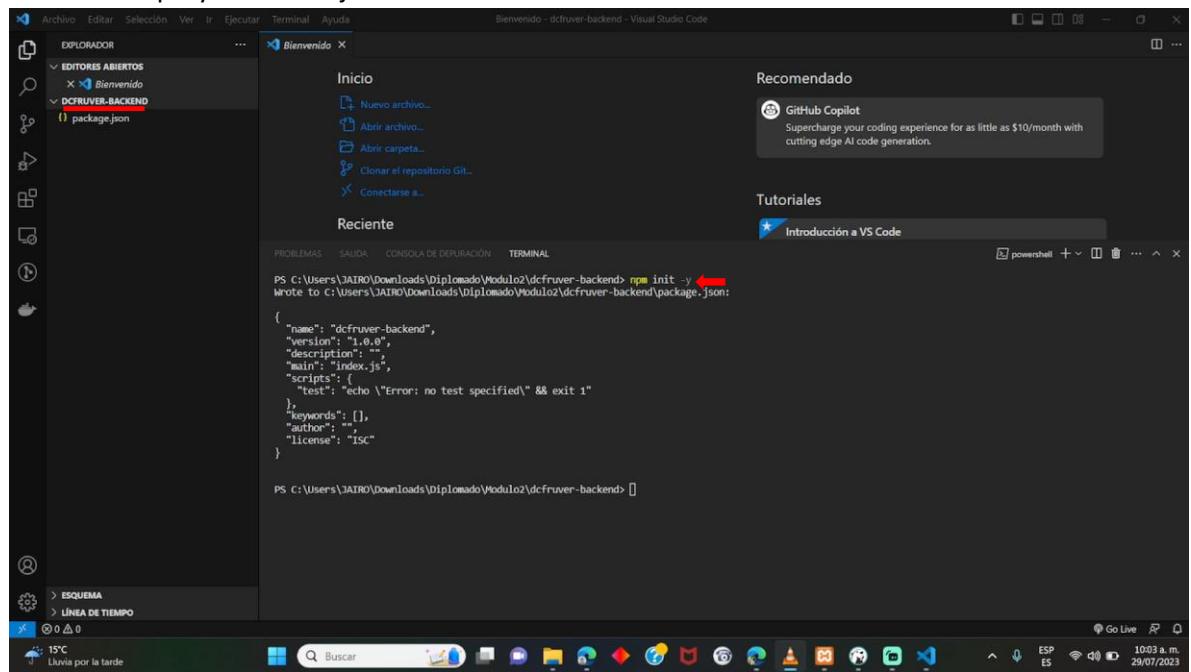
Diagrama ER de la base de datos 'dcfruver':



2. Desarrollar una aplicación Backend implementada en NodeJS y ExpressJS que haga uso de a base de datos del primer punto y que permita el desarrollo de todas las tareas asociadas al registro y administración de las frutas y verduras (La empresa debe contar con un nombre).

- Creé la carpeta del proyecto 'dcfruver-backend' y desde visual empecé a instalar los paquetes necesarios para el desarrollo de la aplicación:

Incialicé un proyecto Node.js en el directorio actual:

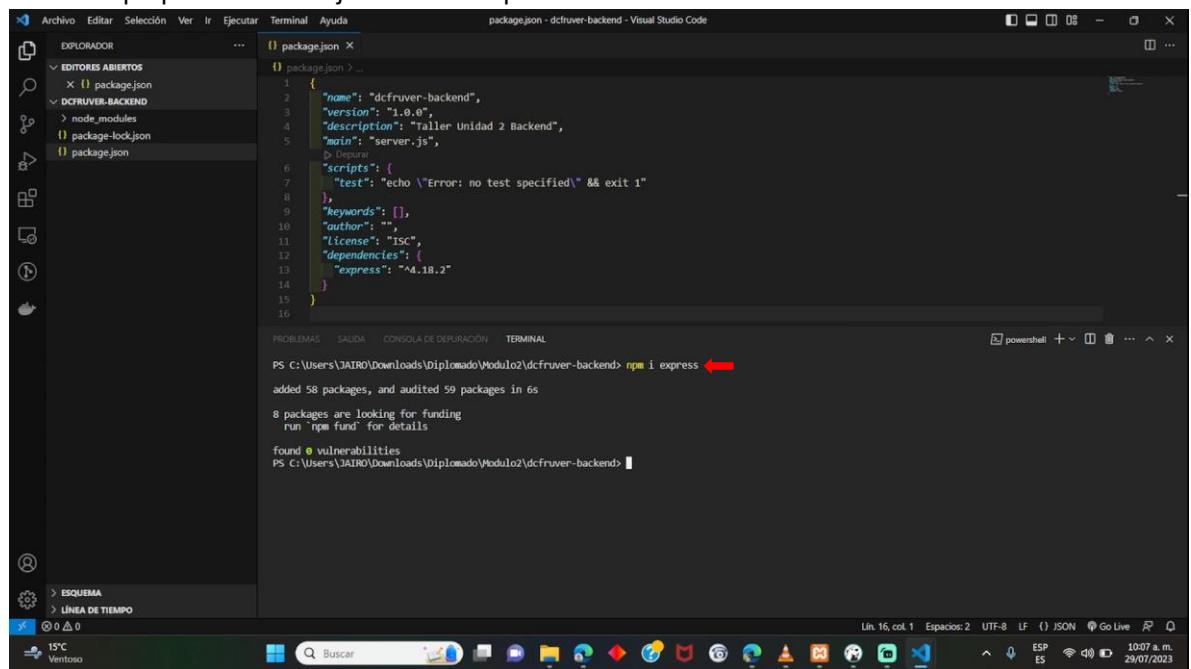


The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, a folder named 'DCFCLUVER-BACKEND' is expanded, showing a 'package.json' file. The terminal at the bottom has the following content:

```
PS C:\Users\JAIRO\Downloads\diplomado\Modulo2\dcfruver-backend> npm init -y
Write to C:\Users\JAIRO\Downloads\diplomado\Modulo2\dcfruver-backend\package.json:
{
  "name": "dcfruver-backend",
  "version": "1.0.0",
  "description": "Taller Unidad 2 Backend",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

PS C:\Users\JAIRO\Downloads\diplomado\Modulo2\dcfruver-backend> [REDACTED]
```

Instalé el paquete de Node.js llamado "express":



The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, a folder named 'DCFCLUVER-BACKEND' is expanded, showing a 'node_modules' folder which contains a 'package-lock.json' and a 'package.json' file. The terminal at the bottom has the following content:

```
PS C:\Users\JAIRO\Downloads\diplomado\Modulo2\dcfruver-backend> npm i express
added 58 packages, audited 59 packages in 6s
8 packages are looking for funding
  run 'npm fund' for details
found 0 vulnerabilities
PS C:\Users\JAIRO\Downloads\diplomado\Modulo2\dcfruver-backend> [REDACTED]
```

Instalé los paquetes "mysql" y "mysql2" de Node.js, que proporciona funcionalidades para conectarse y comunicarse con bases de datos MySQL desde una aplicación Node.js:

The screenshot shows a Visual Studio Code interface with the following details:

- Terminal:** The terminal window displays the command `npm i mysql` being run, followed by the output: "added 12 packages, and audited 71 packages in 3s". It also shows the command `npm i mysql2` being run, followed by the output: "added 11 packages, and audited 82 packages in 4s". Both commands have red arrows pointing to them.
- File Explorer:** The sidebar shows an open folder named "DCFCLUVER-BACKEND" containing several files: package.json, package-lock.json, node_modules, and server.js.
- Code Editor:** The main editor area shows the content of the package.json file, which includes dependencies for express (4.18.2), mysql (2.18.1), and mysql2 (3.5.2).
- Bottom Status Bar:** The status bar shows the current line (Líne. 18) and column (col. 1), as well as other settings like Espacios: 2, UTF-8, LF, JSON, Go Live, and file paths.

Instalé el paquete "nodemon" como una dependencia de desarrollo:

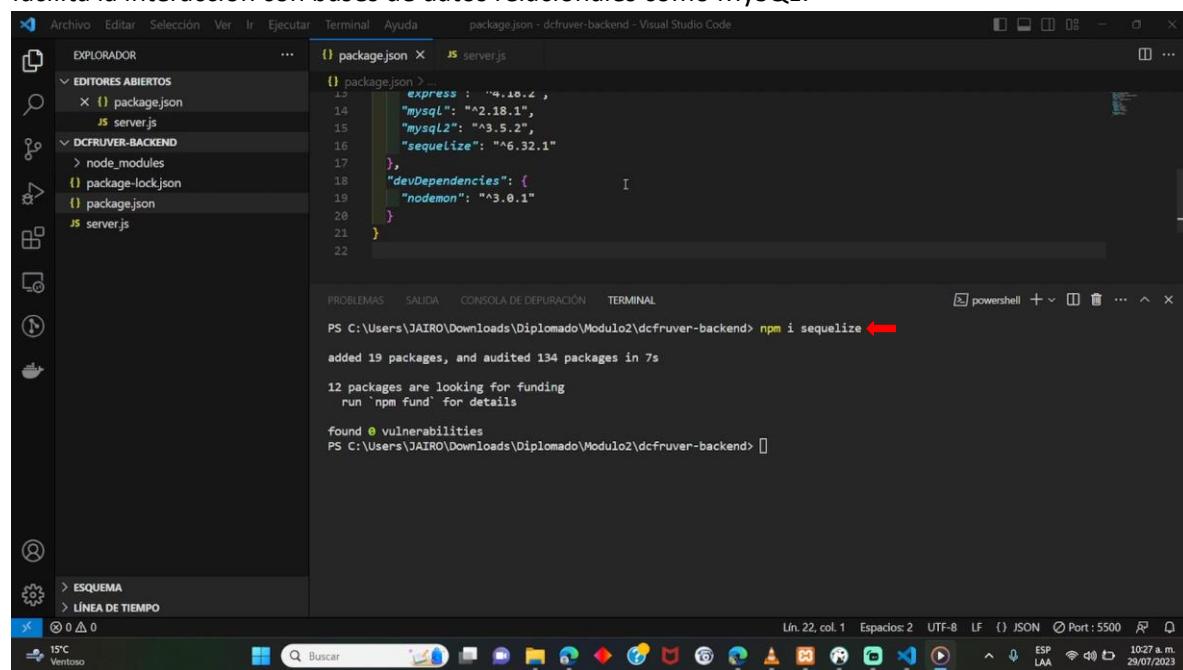
The screenshot shows a Visual Studio Code interface. The left sidebar has 'EXPLORADOR' open, showing a tree view with 'EDITORES ABIERTOS' containing 'package.json', 'DCFNUVER-BACKEND' containing 'node_modules', 'package-lock.json', 'package.json', and 'server.js'. The main editor area shows the 'package.json' file with the following content:

```
1 {  
2   "name": "dcfnuver-backend",  
3   "version": "1.0.0",  
4   "description": "taller Unidad 2 Backend",  
5   "main": "server.js",  
6   "scripts": {  
7     "test": "echo \\"$Error: no test specified\\" && exit 1"  
8   },  
9   "keywords": [],  
10  "author": "",  
11  "license": "ISC",  
12  "dependencies": {  
13    "express": "4.18.2",  
14    "mysql": "2.18.1",  
15    "mysql2": "3.5.2"  
16  },  
17}
```

Below the editor are tabs for 'PROBLEMAS', 'SALIDA', 'CONSOLA DE DEPURACIÓN', and 'TERMINAL'. The 'TERMINAL' tab is active, showing the command: `PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfnuver-backend> npm i nodeon --save-dev`. A red arrow points to the 'nodeon' package name. The output of the command is:
added 33 packages, and audited 115 packages in 4s
11 packages are looking for funding
 run 'npm fund' for details
found 0 vulnerabilities
PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfnuver-backend> []

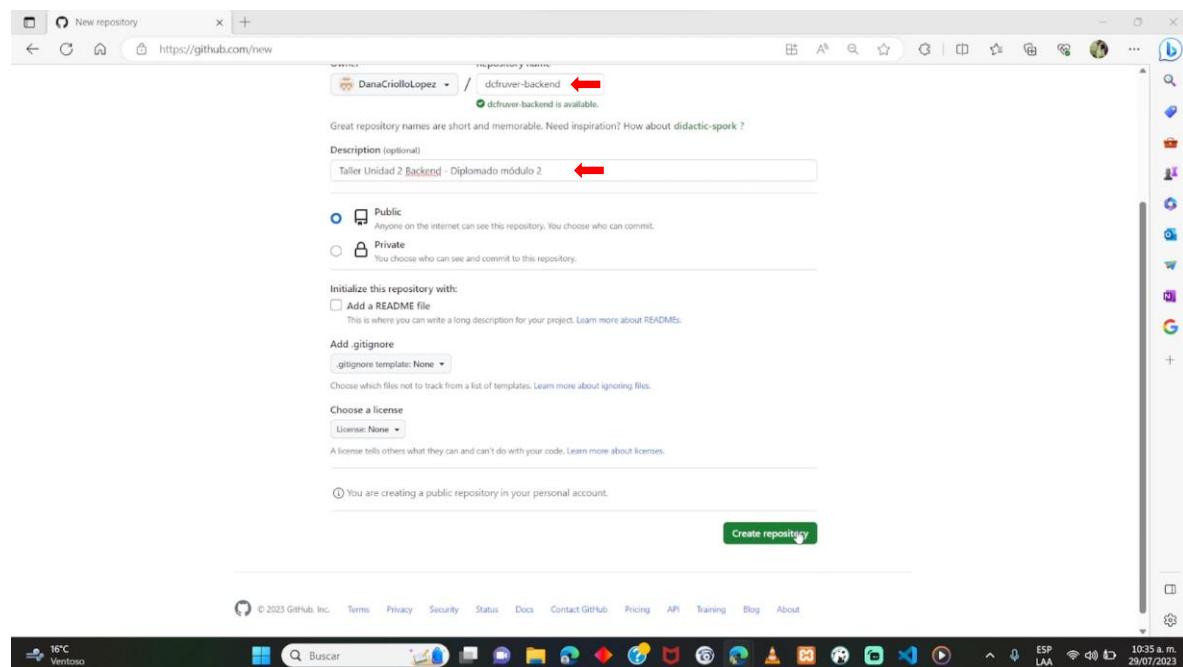
The bottom status bar shows 'Lin. 21, col. 1 Espacios: 2 UTF-8 LF () JSON Go Live' and system icons for battery, signal, and date.

Instalé el paquete "sequelize" de Node.js, que es un ORM (Object-Relational Mapping) que facilita la interacción con bases de datos relacionales como MySQL:

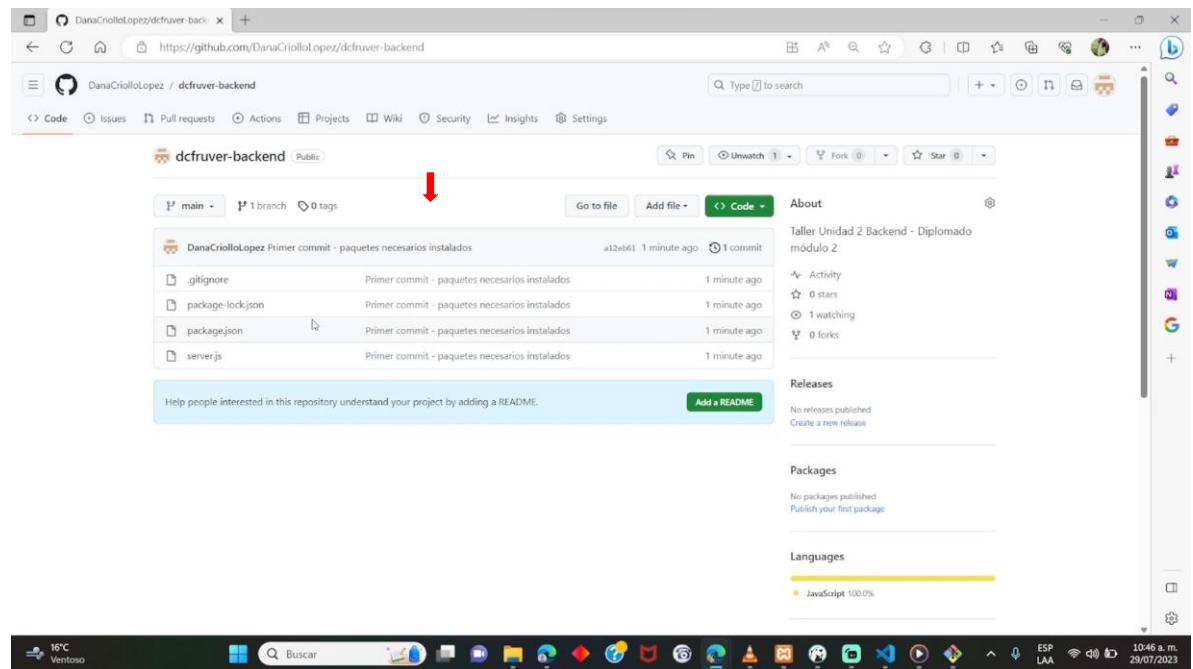


The screenshot shows the Visual Studio Code interface. In the center, the terminal window displays the command `npm i sequelize` being run, along with its output: "added 19 packages, and audited 134 packages in 7s", "12 packages are looking for funding", "run 'npm fund' for details", and "found 0 vulnerabilities". The file `package.json` is open in the editor, showing dependencies for express, mysql, mysql2, and sequelize. The status bar at the bottom right shows the date and time as 10:35 a.m. on 29/07/2023.

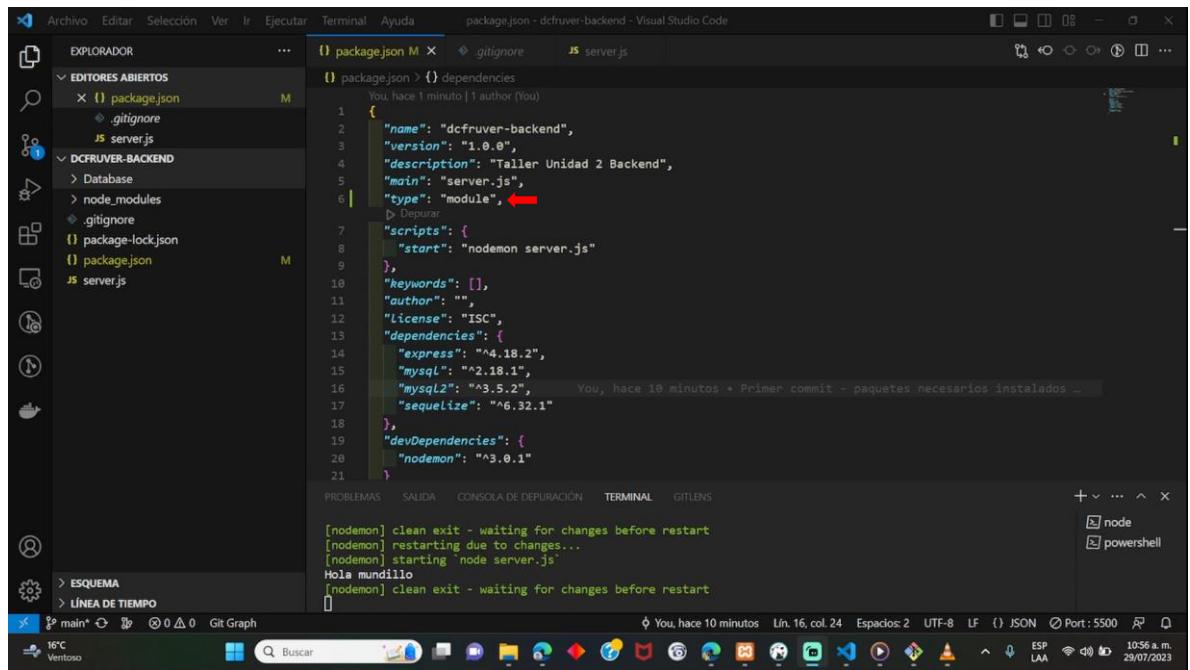
- Creé el repositorio remoto de este proyecto en mi cuenta de GitHub e hice el primer commit llamado “Primer commit – paquetes necesarios instalados”:



```
MINGW64:/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend
$ git init ←
Initialized empty Git repository in c:/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend/.git/
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (master)
$ nano .gitignore ←
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (master)
$ git add . ←
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (master)
$ git commit -m "Primer commit - paquetes necesarios instalados" ←
[master (root-commit) a12eb61] Primer commit - paquetes necesarios instalados
 4 files changed, 1381 insertions(+)
    create mode 100644 .gitignore
    create mode 100644 package-lock.json
    create mode 100644 package.json
    create mode 100644 server.js
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (master)
$ git remote add origin https://github.com/DanaCriolloLopez/dcfruver-backend.git ←
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (master)
$ git branch -M main ←
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push -u origin main ←
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 14.84 KiB | 7.42 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/DanaCriolloLopez/dcfruver-backend.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$
```



- Especifique en el archivo ‘package.json’ que iba a trabajar en el proyecto utilizando el sistema de módulos:

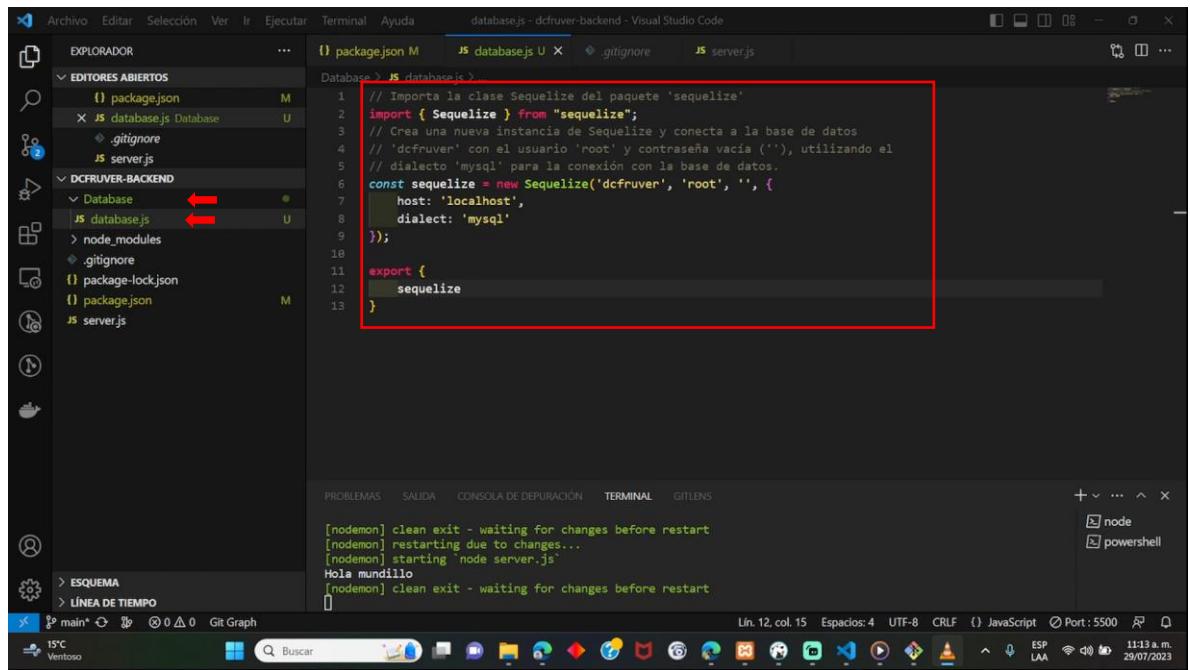


```

{
  "name": "dcfruver-backend",
  "version": "1.0.0",
  "description": "Taller Unidad 2 Backend",
  "main": "server.js",
  "type": "module", -----^
  "scripts": {
    "start": "nodemon server.js"
  },
  "keywords": [],
  "author": "",
  "License": "ISC",
  "dependencies": {
    "express": "^4.18.2",
    "mysql": "^2.18.1",
    "mysql2": "^3.5.2",
    "sequelize": "^6.32.1"
  },
  "devDependencies": {
    "nodemon": "^3.0.1"
  }
}

```

- Creé la carpeta ‘Database’ y dentro de esta carpeta creé el archivo ‘database.js’ donde programé la conexión de mi proyecto con la base de datos ‘dcfruver’ creada en el punto anterior:

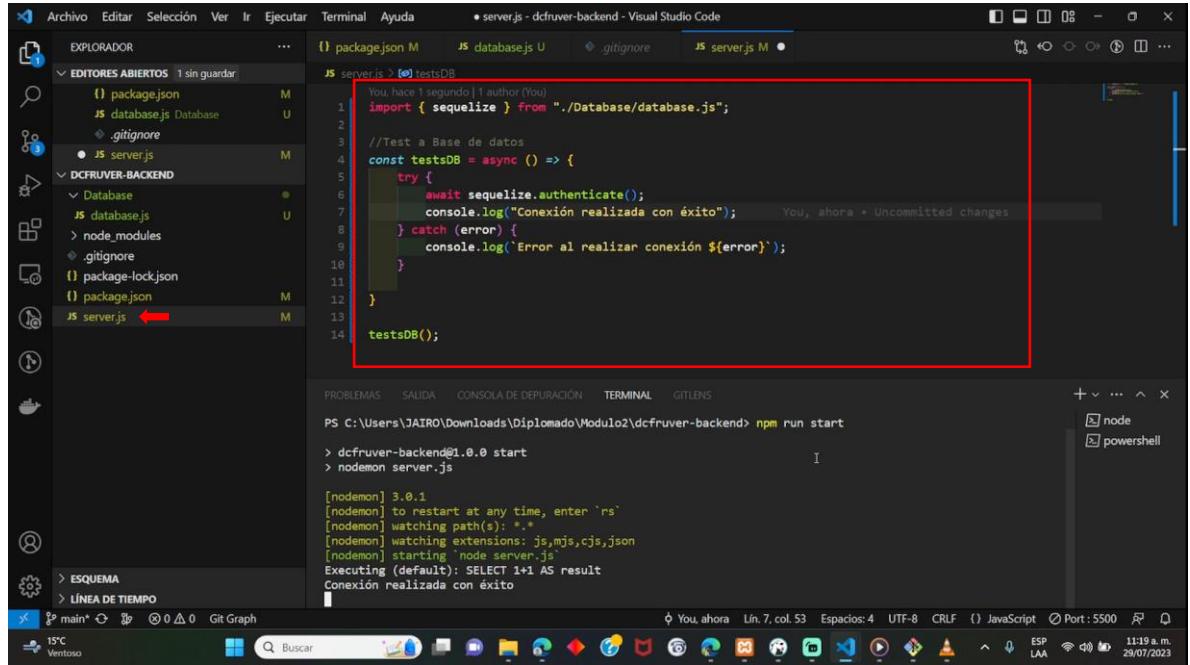


```

// Importa la clase Sequelize del paquete 'sequelize'
import { Sequelize } from "sequelize";
// Crea una nueva instancia de Sequelize y conecta a la base de datos
// 'dcfruver' con el usuario 'root' y contraseña vacía (''), utilizando el
// dialecto 'mysql' para la conexión con la base de datos.
const sequelize = new Sequelize('dcfruver', 'root', '', {
  host: 'localhost',
  dialect: 'mysql'
});
export {
  sequelize
}

```

- Desde el archivo 'server.js' hice el test de conexión a la base de datos:



The screenshot shows the Visual Studio Code interface with the 'server.js' file open. A red box highlights the following code in the editor:

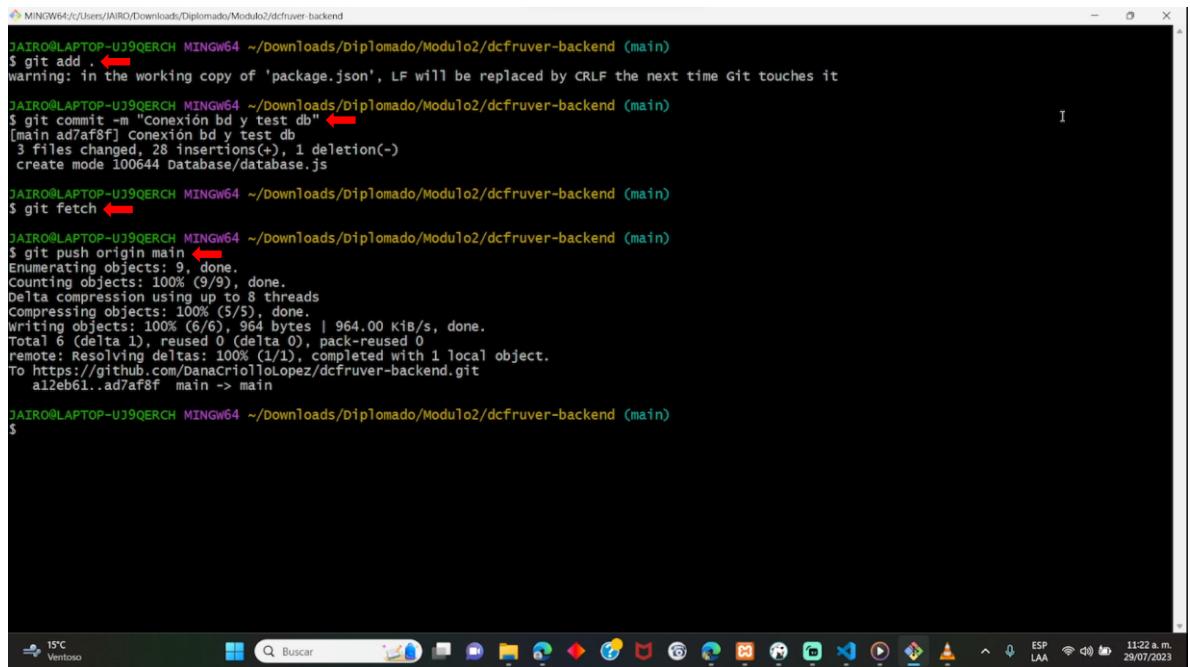
```

You hace 1 segundo | f.author (You)
1 import { Sequelize } from "./Database/database.js";
2
3 //Test a la Base de datos
4 const testsDB = async () => {
5   try {
6     await Sequelize.authenticate();
7     console.log("Conexión realizada con éxito"); You, ahora + Uncommitted changes
8   } catch (error) {
9     console.log(`Error al realizar conexión ${error}`);
10  }
11 }
12
13 testsDB();

```

The terminal below shows the command `npm run start` being executed, and the output indicates the connection was successful.

- Hice el segundo commit guardando los anteriores cambios llamado "Conexión b y test db":

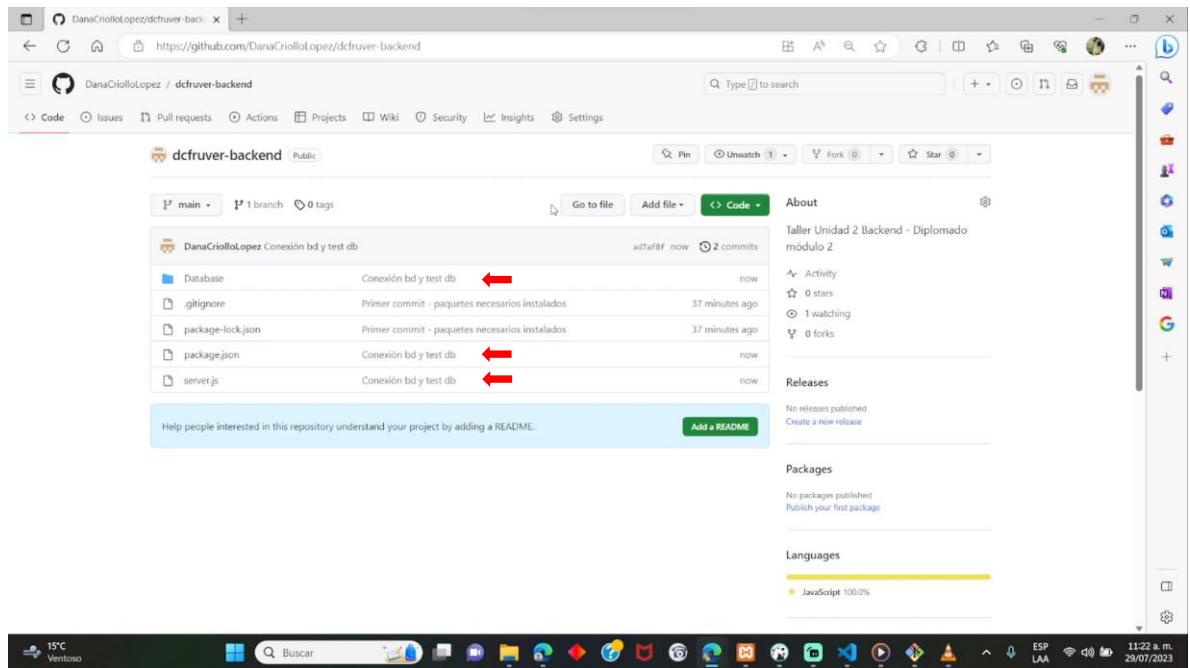


The terminal window shows the following sequence of commands:

```

MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend
$ git add .
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "Conexión bd y test db"
[main ad7af8f] conexión bd y test db
 3 files changed, 28 insertions(+), 1 deletion(-)
  create mode 100644 Database/database.js
MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git fetch
MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 964 bytes | 964.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DanaCriolloLopez/dcfruver-backend.git
 a12eb61..ad7af8f main --> main
MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ 

```



- Creé la carpeta Models y empecé a crear los modelos de todas las tablas de la base de datos ‘dcfuruver’:

Modelo para la tabla ‘usuario’:

```

const Usuario = sequelize.define('usuario', {
  // Model attributes are defined here
  idusuario: {
    type: DataTypes.INTEGER,
    primaryKey: true,
    autoIncrement: true,
    allowNull: false
  },
  nombreusu: {
    type: DataTypes.STRING(100),
    allowNull: false
  },
  contrasenausu: {
    type: DataTypes.STRING(255),
    allowNull: false
  },
  rol: {
    type: DataTypes.STRING(28),
    allowNull: false
  },
  {
    timestamps: false
  }
});

```

Modelo para la tabla 'productos':

```
import { DataTypes } from 'sequelize';
import { sequelize } from '../Database/database.js';

const Productos = sequelize.define('productos', {
  idproducto: {
    type: DataTypes.INTEGER,
    primaryKey: true,
    autoIncrement: true,
    allowNull: false
  },
  nomproducto: {
    type: DataTypes.STRING(100),
    allowNull: false
  },
  categoriaproducto: {
    type: DataTypes.STRING(50),
    allowNull: false
  },
  descripcióncionproducto: {
    type: DataTypes.STRING(1000),
    allowNull: false
  },
  precioproducto: {
    type: DataTypes.INTEGER,
    allowNull: false
  },
  stockproducto: {
    type: DataTypes.INTEGER,
    allowNull: false
  }
}, {
  timestamps: false
});
```

Modelo para la tabla 'clientes':

```
import { DataTypes } from 'sequelize';
import { sequelize } from '../Database/database.js';

const Clientes = sequelize.define('clientes', {
  cedulacliente: {
    type: DataTypes.STRING(10),
    primaryKey: true,
    allowNull: false
  },
  nomcliente: {
    type: DataTypes.STRING(100),
    allowNull: false
  },
  dircliente: {
    type: DataTypes.STRING(100),
    allowNull: false
  },
  telcliente: {
    type: DataTypes.STRING(10),
    allowNull: false
  },
  correocliente: {
    type: DataTypes.STRING(100),
    allowNull: false
  }
}, {
  timestamps: false
});
```

Modelo para la tabla 'pedidos.js':

```
import { DataTypes } from 'sequelize';
import { sequelize } from '../Database/database.js';
import { Clientes } from './clientes.js';

const Pedidos = sequelize.define('pedidos', {
  id: {
    type: DataTypes.INTEGER,
    primaryKey: true,
    autoIncrement: true,
    allowNull: false
  },
  cedulacliente: {
    type: DataTypes.STRING(18),
    allowNull: false
  },
  fechapedido: {
    type: DataTypes.DATE,
    allowNull: false
  },
  total: {
    type: DataTypes.INTEGER,
    allowNull: false
  },
  estado: {
    type: DataTypes.STRING(1),
    allowNull: false
  }
},
{
  timestamps: false
});
```

Modelo para la tabla 'pedidodetalle':

```
import { DataTypes } from 'sequelize';
import { sequelize } from '../Database/database.js';
import { Pedidos } from './pedidos.js';
import { Productos } from './productos.js';

const PedidoDetalle = sequelize.define('pedidodetalle', {
  id: {
    type: DataTypes.INTEGER,
    primaryKey: true,
    autoIncrement: true,
    allowNull: false
  },
  idpedido: {
    type: DataTypes.INTEGER,
    allowNull: false
  },
  idproducto: {
    type: DataTypes.INTEGER,
    allowNull: false
  },
  cantidad: {
    type: DataTypes.INTEGER,
    allowNull: false
  },
  subtotal: {
    type: DataTypes.INTEGER,
    allowNull: false
  }
});
```

- Sincronicé los modelos creados con la base de datos desde el archivo server.js:

```

    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    server.js - dcfruver-backend - Visual Studio Code
    EXPLORADOR ... rbojs U JS productos.js U JS clientes.js U JS pedidos.js U JS pedidodetalle.js U JS server.js M
    EDITORES ABIERTOS
    DCFRUVER-BACKEND
    > Database
    > Models
        JS clientes.js U
        JS pedidodetalle.js U
        JS pedidos.js U
        JS productos.js U
        JS usuario.js U
    > node_modules
    .gitignore
    package-lock.json
    package.json
    JS server.js M
    JS testsDB
    You, hace 1 segundo || 1 author (You)
    1 import { Sequelize } from './Database/database.js';
    2
    3 // Test a la Base de datos
    4 const testsDB = async () => {
    5     try {
    6         // Sincroniza los modelos con la base de datos
    7         await sequelize.sync(); ←
    8
    9         console.log('Modelos sincronizados correctamente con la base de datos.');
    10    } catch (error) {
    11        console.error(`Error al sincronizar modelos con la base de datos: ${error}`);
    12    }
    13
    14 }
    15
    16 testsDB();
  
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

PS C:\Users\JAIRO\Downloads\Diplomado\Modules\Modulo2\dcfuruver-backend> npm run start

> dcfuruver-backend@1.0.0 start

> nodemon server.js

[nodemon] 3.0.1

[nodemon] to restart at any time, enter `rs`

[nodemon] watching path(s): *.*

[nodemon] watching extensions: js,mjs,cjs,json

[nodemon] starting 'node server.js'

Executing (default): SELECT 1+1 AS result

Modelos sincronizados correctamente con la base de datos. ←

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

Φ You, hace 47 minutos Lín. 12, col. 5 Espacios: 4 UTF-8 CRLF {} JavaScript ⚡ Port: 5500 ⚡

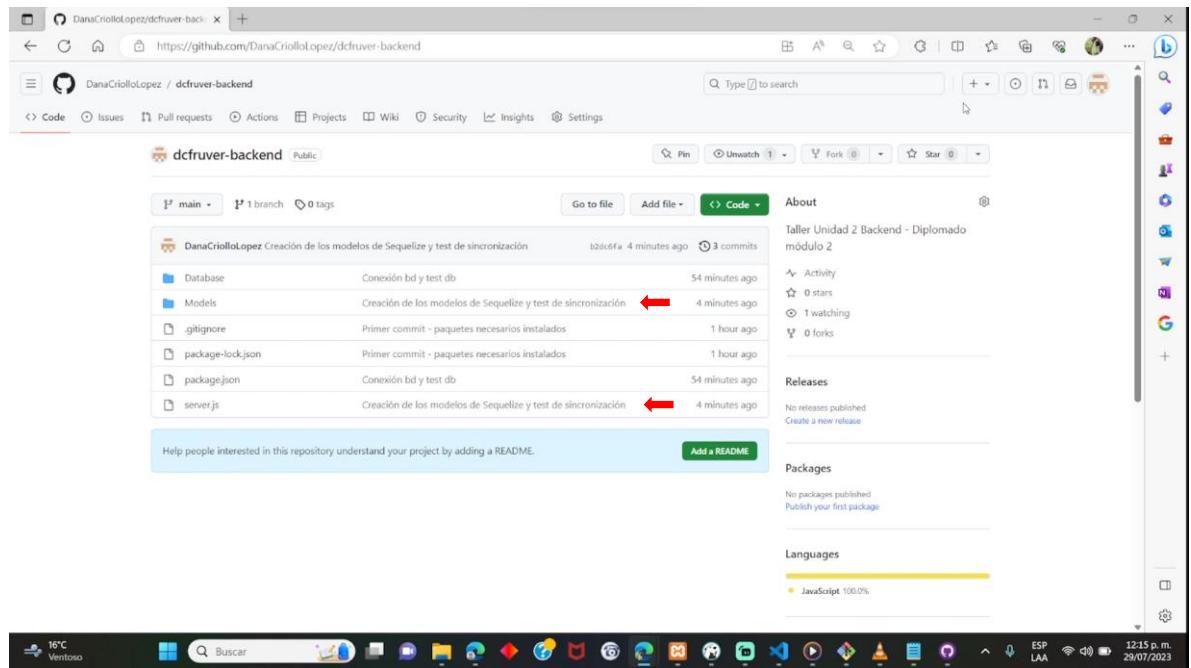
18°C Ventoso Buscar

- Hice un commit guardando los anteriores cambios llamado “Creación de los modelos de Sequelize y test de sincronización”:

```

MINGW64:/t/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfuruver-backend
$ git add .
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfuruver-backend (main)
$ git commit -m "Creación de los modelos de Sequelize y test de sincronización" ←
[main b2dc6fa] creación de los modelos de Sequelize y test de sincronización
 6 files changed, 196 insertions(+), 3 deletions(-)
   create mode 100644 Models/clientes.js
   create mode 100644 Models/pedidodetalle.js
   create mode 100644 Models/pedidos.js
   create mode 100644 Models/productos.js
   create mode 100644 Models/usuario.js
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfuruver-backend (main)
$ git fetch ←
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfuruver-backend (main)
$ git push origin main ←
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 1.85 KiB | 1.85 MiB/s, done.
Total 9 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 1 local object.
To https://github.com/DanaCriollolopez/dcfuruver-backend.git
 ad7af8f..b2dc6fa main -> main
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfuruver-backend (main)
$ |
  
```

18°C Ventoso Buscar



- Configure y ejecute el servicio utilizando Node.js y Express.

Creé el archivo 'routes.js' dentro de la carpeta 'Routes' y empecé a configurar las rutas:

```

import Router from 'express';
import { getUsuario } from '../Controllers/controller_table_usuario.js';

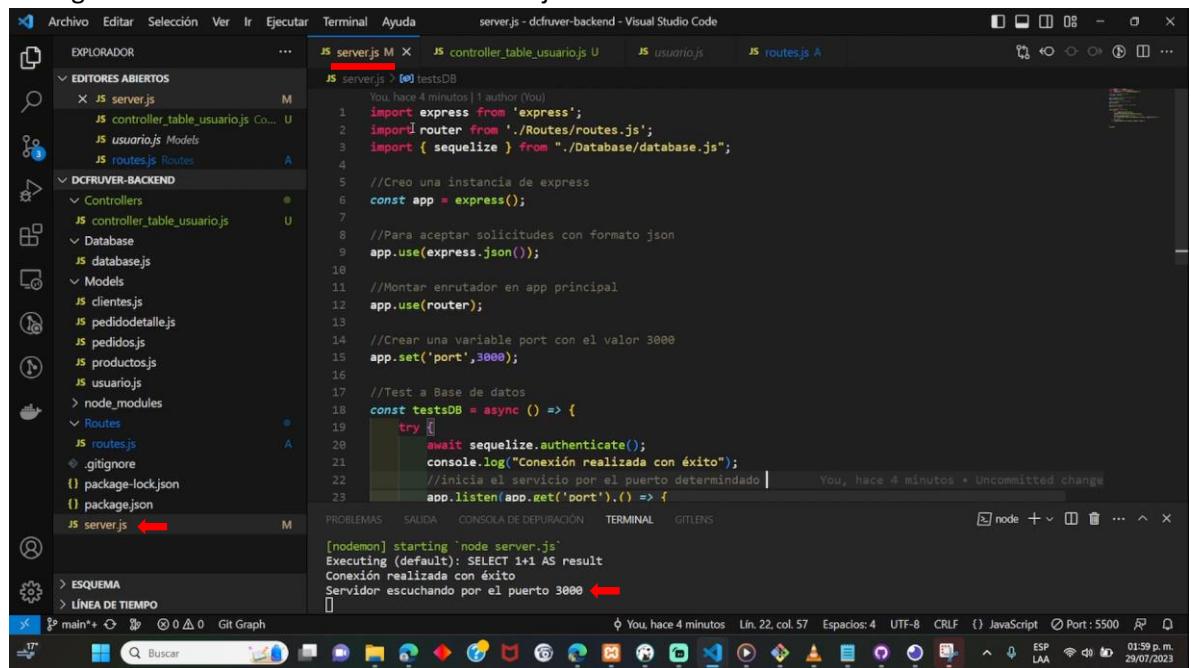
const router= Router();

// GET method route
router.get('/', (req, res) => {
  res.send('GET página principal express taller backend');
});

export default router;

```

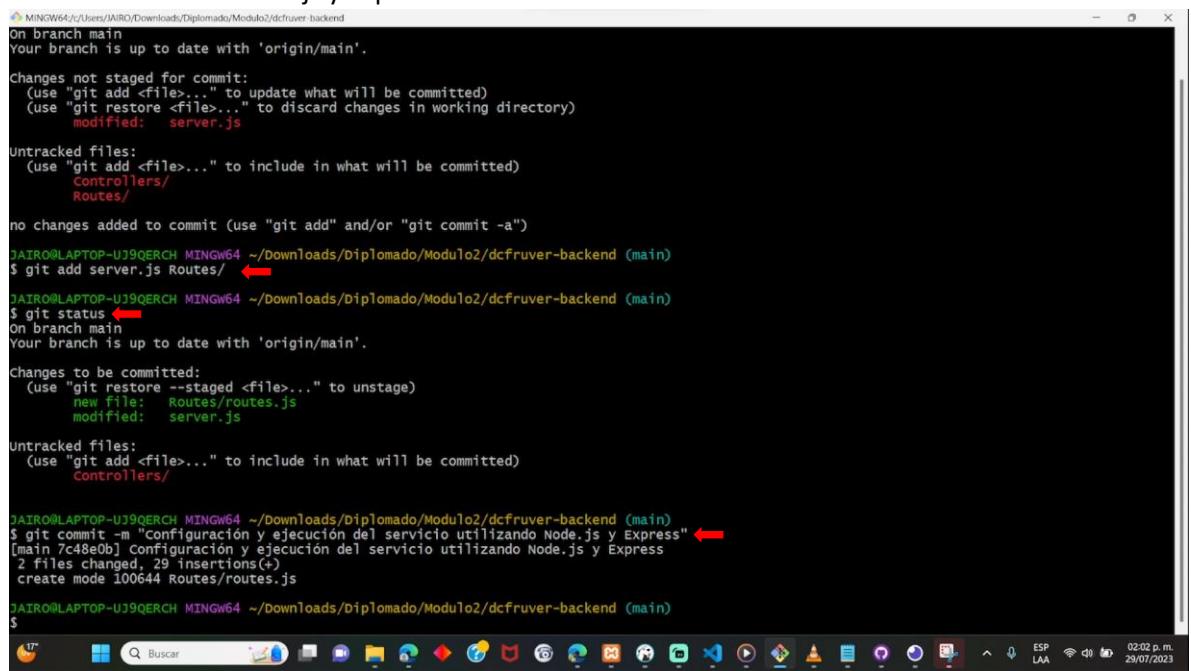
Configure el servicio desde el archivo 'server.js':



```
You, hace 4 minutos | 1 author (You)
1 import express from 'express';
2 import router from './Routes/routes.js';
3 import { sequelize } from "./Database/database.js";
4
5 //Creo una instancia de express
6 const app = express();
7
8 //Para aceptar solicitudes con formato json
9 app.use(express.json());
10
11 //Montar enrutador en app principal
12 app.use(router);
13
14 //Crear una variable port con el valor 3000
15 app.set('port', 3000);
16
17 //Test a Base de datos
18 const testsDB = async () => {
19   try {
20     await sequelize.authenticate();
21     console.log("Conexión realizada con éxito");
22     //Inicia el servicio por el puerto determinado
23     app.listen(app.get('port'), () => {
      You, hace 4 minutos + Uncommitted change
      [nodemon] starting `node server.js`
      Executing (default): SELECT 1+1 AS result
      Conexión realizada con éxito
      Servidor escuchando por el puerto 3000
    })
  }
}

```

Hice un commit guardando los anteriores cambios llamado "Configuración y ejecución del servicio utilizando Node.js y Express":



```
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   server.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Controllers/
    Routes/

no changes added to commit (use "git add" and/or "git commit -a")

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git add server.js Routes/ ←
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git status ←
On branch main
Your branch is up to date with 'origin/main'.

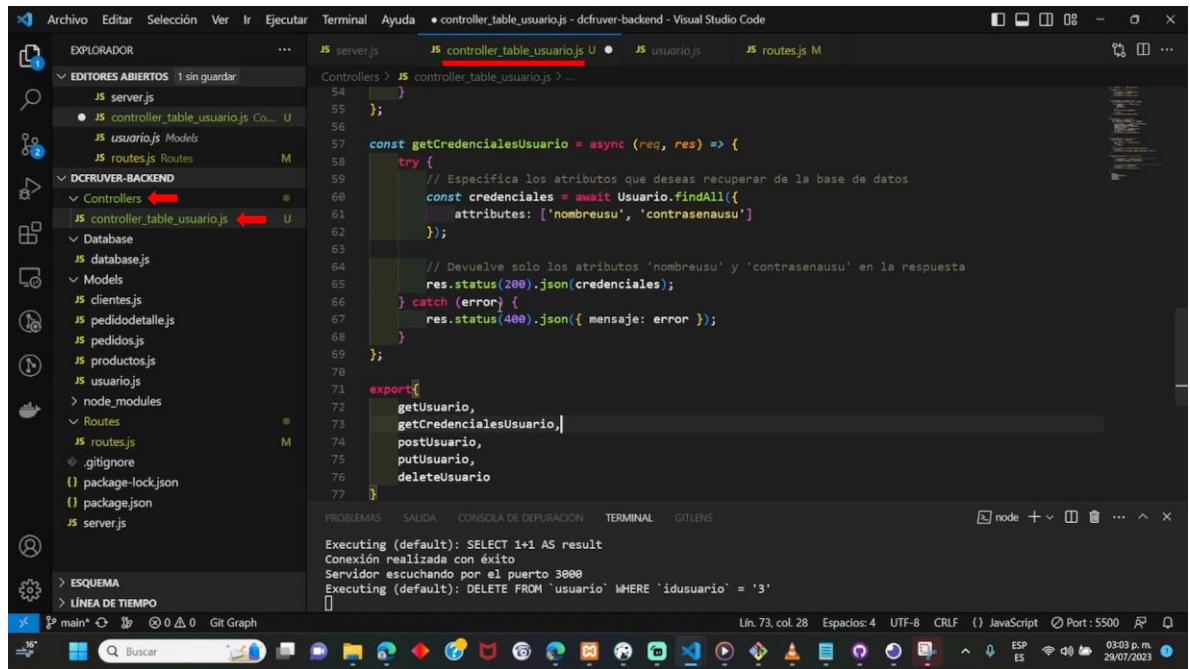
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   Routes/routes.js
    modified:   server.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    controllers/

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "Configuración y ejecución del servicio utilizando Node.js y Express" ←
[main 7c48e0b] Configuración y ejecución del servicio utilizando Node.js y Express
  2 files changed, 29 insertions(+)
  create mode 100644 Routes/routes.js

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$
```

- Creé la carpeta ‘Controllers’ y en ella creé los controladores para cada tabla de mi base de datos ‘dcfruver’, empezando con el controlador para la tabla ‘usuario’ en el archivo llamado “controller_table_usuario.js”, en él creé las funciones getUsuario, postUsuario, putUsuario, deleteUsuario y getCredencialesUsuario (retorna únicamente los atributos de ‘nombreusu’ y ‘contrasenausu’):



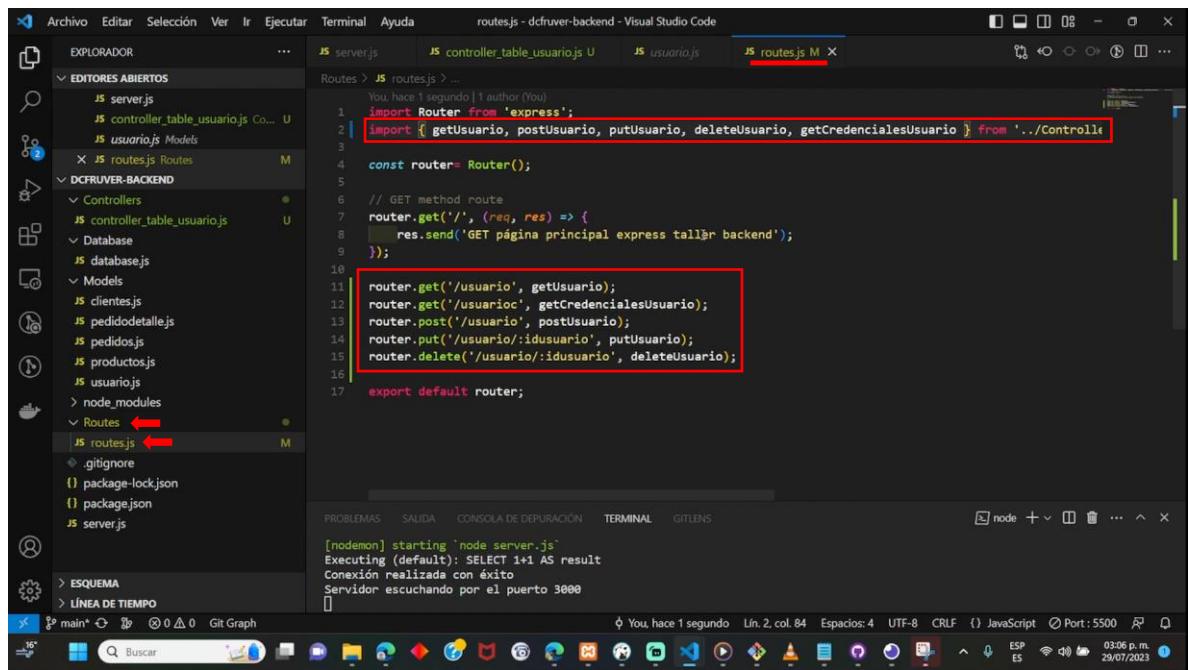
```

    const getCredencialesUsuario = async (req, res) => {
      try {
        // Especifica los atributos que deseas recuperar de la base de datos
        const credenciales = await Usuario.findAll({
          attributes: ['nombreusu', 'contrasenausu']
        });

        // Devuelve solo los atributos 'nombreusu' y 'contrasenausu' en la respuesta
        res.status(200).json(credenciales);
      } catch (error) {
        res.status(400).json({ mensaje: error });
      }
    };

    export {
      getUsuario,
      getCredencialesUsuario,
      postUsuario,
      putUsuario,
      deleteUsuario
    };
  
```

- Creé las rutas a cada una de las anteriores funciones en el archivo ‘routes.js’:



```

    import Router from 'express';
    import { getUsuario, postUsuario, putUsuario, deleteUsuario, getCredencialesUsuario } from '../Controllers/controller_table_usuario.js';

    const router= Router();

    // GET method route
    router.get('/', (req, res) => {
      res.send('GET página principal express tallar backend');
    });

    router.get('/usuario', getUsuario);
    router.get('/usuario/:idusuario', getCredencialesUsuario);
    router.post('/usuario', postUsuario);
    router.put('/usuario/:idusuario', putUsuario);
    router.delete('/usuario/:idusuario', deleteUsuario);

    export default router;
  
```

- Hice un commit guardando los anteriores cambios llamado “Controller y rutas de la tabla usuario”:

```

MINGW64:/c/Users/AIRO/Downloads/Diplomado/Modulo2/dcfruver-backend
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified:   Routes/routes.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Controllers/

no changes added to commit (use "git add" and/or "git commit -a")

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git add .

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "Controller y rutas de la tabla usuario"
[main b9959e2] controller y rutas de la tabla usuario
 2 files changed, 88 insertions(+), 2 deletions(-)
  create mode 100644 controllers/controller_table_usuario.js

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git fetch

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push origin main

```

- Creé el controlador para la tabla ‘productos’ en el archivo llamado “controller_table_productos.js”, en él creé las funciones getProductos, postProductos, putProductos, deleteProductos, getStockProducto (retorna el valor de la variable ‘stockproducto’ de un producto específico) y putStockProductos (modifica el valor de la variable ‘stockproducto’ de un producto específico):

```

controller_table_productos.js  controller_table_usuario.js

// Actualizamos solo la variable stockproducto
producto.stockproducto = stockproducto;

// Guardamos los cambios en la base de datos
await producto.save();

res.status(200).json({ mensaje: 'Producto actualizado exitosamente', producto });
} catch (error) {
  res.status(400).json({ mensaje: error });
}

export {
  getProductos,
  postProductos,
  putProductos,
  deleteProductos,
  getStockProducto,
  putStockProductos
}

```

- Creé las rutas a cada una de las anteriores funciones en el archivo ‘routes.js’:

```

    routes.js
    10 //rutas para la consultas de la tabla usuario
    11 router.get('/usuario', getUsuario);
    12 router.post('/usuario', postUsuario);
    13 router.put('/usuario/:idusuario', putUsuario);
    14 router.delete('/usuario/:idusuario', deleteUsuario);
    15 router.get('/usuario/:idusuario', getUsuario);
    16 router.get('/usuario/:idusuario', getCredentialsUsuario);

    17 //rutas para las consultas de la tabla productos
    18 router.get('/productos', getProductos);
    19 router.post('/productos', postProductos);
    20 router.put('/productos/:idproducto', putProductos);
    21 router.delete('/productos/:idproducto', deleteProductos);
    22 router.get('/productos/:idproducto', getStockProducto);
    23 router.put('/productos/:idproducto', putStockProductos);
    24
    25
    26
    27
    28

    export default router;
  
```

- Hice un commit guardando los anteriores cambios llamado “Controller y rutas de la tabla productos”:

```

MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Routes/routes.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Controllers/controller_table_productos.js
        img/
no changes added to commit (use "git add" and/or "git commit -a")

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git add .

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "controller y rutas de la tabla productos"
[main 8ee73bd] controller y rutas de la tabla productos
 4 files changed, 124 insertions(+), 1 deletion(-)
 create mode 100644 controllers/controller_table_productos.js
 create mode 100644 img/manzanas.jpg
 create mode 100644 img/tomates.jpg

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git fetch
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push origin main
  
```

- Creé el controlador para la tabla ‘clientes’ en el archivo llamado “controller_table_clientes.js”, en él creé las funciones getClientes, postClientes, putClientes, deleteClientes y getCorreoCliente(retorna el valor de la variable ‘correocliente’ de un cliente específico):

```

    JS server.js JS routes.js JS controller_table_clientes.js U JS clientes.js .gitignore
Controllers > JS controller_table_clientes.js > ...
71     } else {
72         res.status(404).json({ mensaje: 'Cliente no encontrado' });
73     }
74 } catch (error) {
75     res.status(400).json({ mensaje: error });
76 }
77 }
78
79 export{
80     getClientes,
81     postClientes,
82     putClientes,
83     deleteClientes,
84     getCorreoCliente
85 }

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

Servidor escuchando por el puerto 3000
Executing (default): SELECT `idproducto`, `nomproducto`, `categoriasproducto`, `descripcionproducto`, `precioproducto`, `stockproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '1';
Executing (default): SELECT `idproducto`, `nomproducto`, `categoriasproducto`, `descripcionproducto`, `precioproducto`, `stockproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '2';
Executing (default): SELECT `idproducto`, `nomproducto`, `categoriasproducto`, `descripcionproducto`, `precioproducto`, `stockproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '1';
Executing (default): UPDATE `productos` SET `stockproducto`=? WHERE `idproducto` = ?

Lín. 84, col. 21 Espacios: 4 UTF-8 CRLF () JavaScript () Port: 5500

- Creé las rutas a cada una de las anteriores funciones en el archivo ‘routes.js’:

```

    JS server.js JS routes.js M JS controller_table_clientes.js U JS clientes.js .gitignore
Routes > JS routes.js > ...
23 router.put('/productos/:idproducto', putProductos);
24 router.delete('/productos/:idproducto', deleteProductos);
25 router.get('/productos/:idproducto', getStockProducto);
26 router.put('/productos/:idproducto', putStockProductos);
27
28 //rutas para las consultas de la tabla clientes
29
30 router.get('/clientes', getClientes);
31 router.post('/clientes', postClientes);
32 router.put('/clientes/:cedulacliente', putClientes);
33 router.delete('/clientes/:cedulacliente', deleteClientes);
34 router.get('/clientes/:cedulacliente', getCorreoCliente);
35
36 export default router;
37

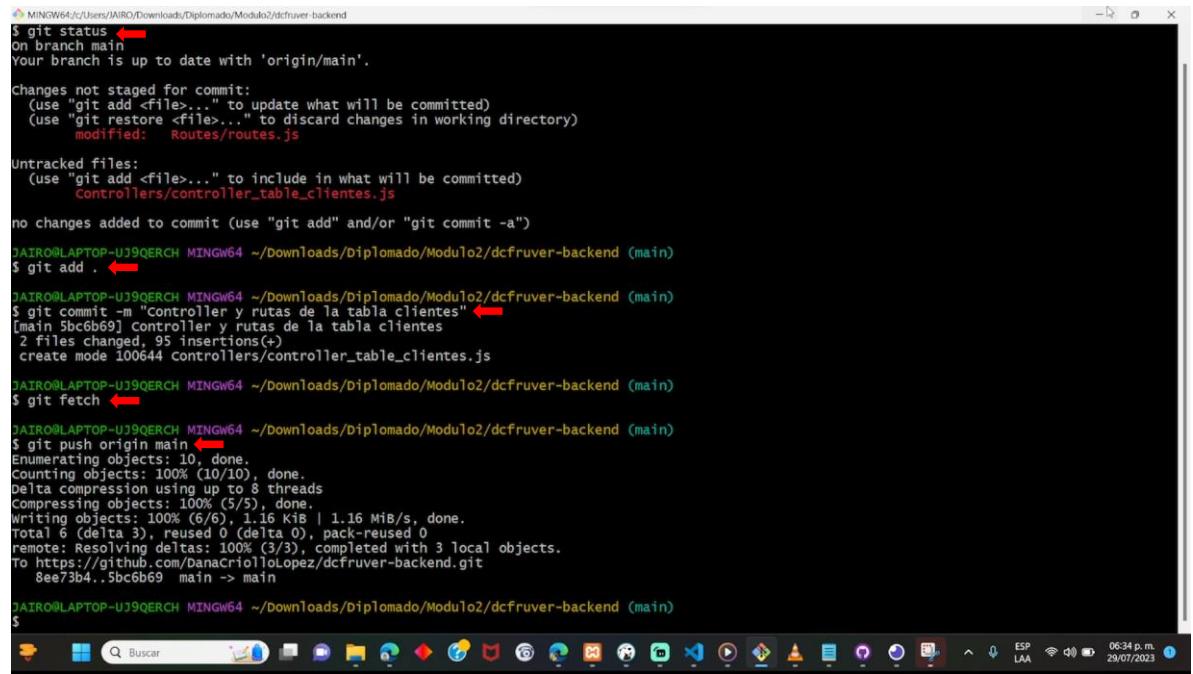
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

Servidor escuchando por el puerto 3000
Executing (default): SELECT `idproducto`, `nomproducto`, `categoriasproducto`, `descripcionproducto`, `precioproducto`, `stockproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '1';
Executing (default): SELECT `idproducto`, `nomproducto`, `categoriasproducto`, `descripcionproducto`, `precioproducto`, `stockproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '2';
Executing (default): SELECT `idproducto`, `nomproducto`, `categoriasproducto`, `descripcionproducto`, `precioproducto`, `stockproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '1';
Executing (default): UPDATE `productos` SET `stockproducto`=? WHERE `idproducto` = ?

3 selecciones Espacios: 4 UTF-8 CRLF () JavaScript () Port: 5500

- Hice un commit guardando los anteriores cambios llamado “Controller y rutas de la tabla clientes”:



```

MINGW64:/c/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend
$ git status
on branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   Routes/routes.js

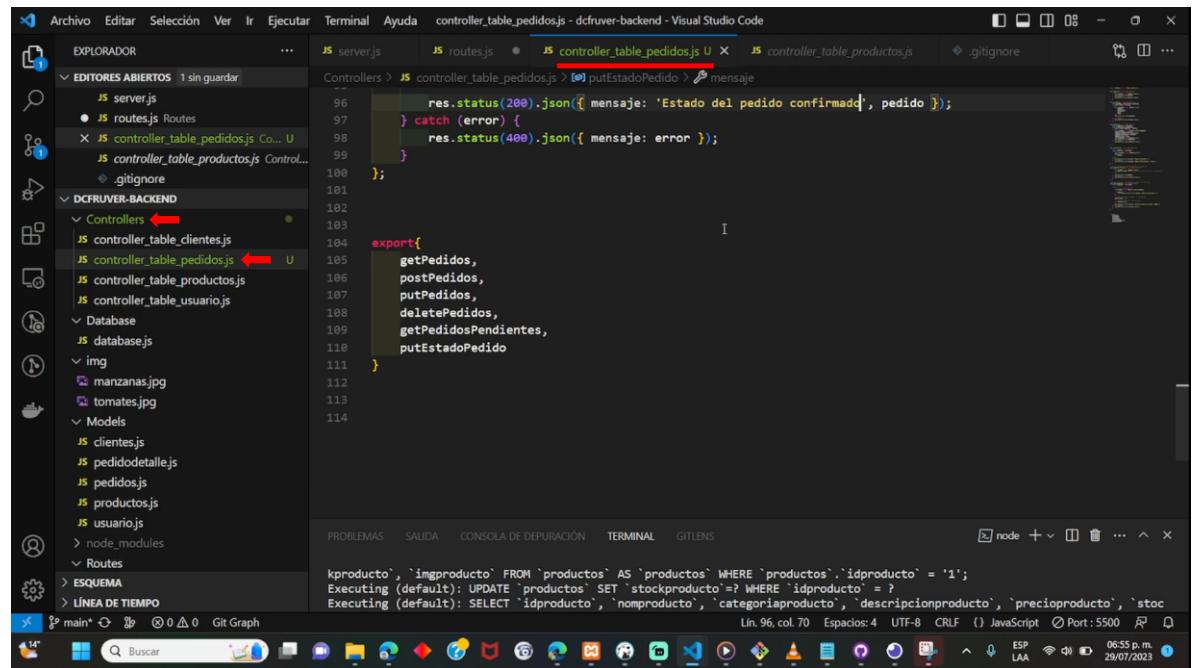
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Controllers/controller_table_clientes.js

no changes added to commit (use "git add" and/or "git commit -a")

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git add .
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "Controller y rutas de la tabla clientes"
[main 5bc6b69] controller y rutas de la tabla clientes
 2 files changed, 95 insertions(+)
 create mode 100644 controllers/controller_table_clientes.js

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git fetch
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push origin main
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 1.16 KiB | 1.16 MiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/DanaCriolloLopez/dcfruver-backend.git
  8ee73b4..5bc6b69 main -> main
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ 
```

- Creé el controlador para la tabla ‘pedidos’ en el archivo llamado “controller_table_pedidos.js”, en él creé las funciones getPedidos, postPedidos, putPedidos, deletePedidos, getPedidosPendientes (retorna todos los pedidos con ‘estado’= ‘P’ que significan los pedidos pendientes) y putEstadoPedido (Modifica el ‘estado’ a ‘C’, para decir que un pedido específico ya no está en estado pendiente sino en confirmado):



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under 'DCFRUVER-BACKEND' with files like 'server.js', 'routes.js', 'controller_table_clientes.js', 'controller_table_pedidos.js' (which is the active file), 'controller_table_productos.js', and '.gitignore'.
- Code Editor:** The active file is 'controller_table_pedidos.js'. The code contains several functions:
 - A catch block for errors: `} catch (error) { res.status(400).json({ mensaje: error }) }`
 - An export block defining methods: `export{ getPedidos, postPedidos, putPedidos, deletePedidos, getPedidosPendientes, putEstadoPedido }
 - SQL queries in the comments: `kproducto`, `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '1';` and `UPDATE `productos` SET `stockproducto`=? WHERE `idproducto` = ?`
- Terminal:** Shows command-line output related to database queries.

- Creé las rutas a cada una de las anteriores funciones en el archivo ‘routes.js’:

```

    routes.js - dcfruver-backend - Visual Studio Code
    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    EXPLORADOR ... JS server.js JS routes.js JS controller_table_pedidos.js U JS controller_table_productos.js .gitignore
    EDITORES ABIERTOS 1 sin guardar
    JS server.js
    ● JS routes.js Routes
    JS controller_table_pedidos.js Controller...
    JS controller_table_productos.js Control...
    .gitignore
    DCFRUVER-BACKEND
    JS database.js
    img
    manzanas.jpg
    tomates.jpg
    Models
    JS clientes.js
    JS pedidodetalles.js
    JS pedidos.js
    JS productos.js
    JS usuario.js
    > node_modules
    > Routes
    JS routes.js
    .gitignore
    package-lock.json
    package.json
    JS server.js
    PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
    node + ... ^ x
    kproducto', `imgproducto` FROM `productos` AS `productos` WHERE `productos`.`idproducto` = '1';
    Executing (default): UPDATE `productos` SET `stockproducto`=? WHERE `idproducto` = ?
    Executing (default): SELECT `idproducto`, `nombreproducto`, `categoriacproducto`, `descripcionproducto`, `precioproducto`, `stoc
    Lin. 43, col. 37 Espacios: 4 UTF-8 CRLF () JavaScript Port:5500 R
    05:56 p. m. 29/07/2023
  
```

- Hice un commit guardando los anteriores cambios llamado “Controller y rutas de la tabla pedidos”:

```

MINGW64:/Users/JAIRO/Downloads/Diplomado/Modulo2/dcfruver-backend
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   Routes/routes.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Controllers/controller_table_pedidos.js

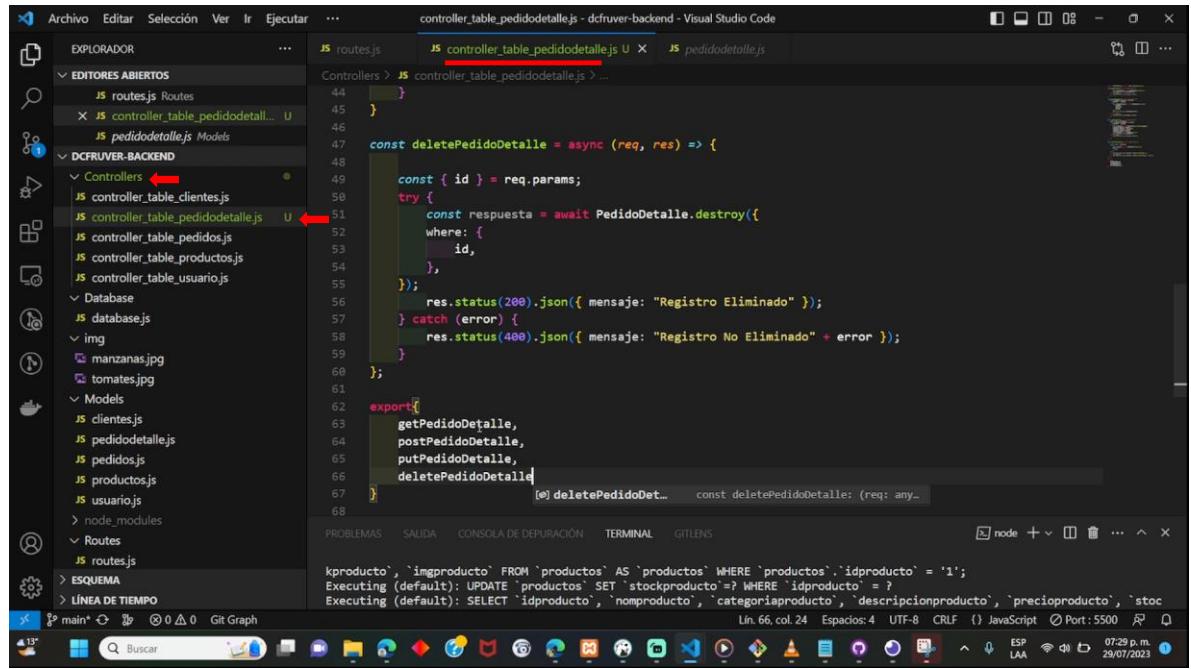
no changes added to commit (use "git add" and/or "git commit -a")

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git add .
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "Controller y rutas de la tabla pedidos"
[main 4389263] controller y rutas de la tabla pedidos
2 files changed, 124 insertions(+)
  create mode 100644 controllers/controller_table_pedidos.js

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git fetch
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push origin main
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 1.34 KiB | 1.34 MiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/DanaCrioitoLopez/dcfruver-backend.git
  9bc6069..4389263  main -> main

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ |
  
```

- Creé el controlador para la tabla ‘pedidodetalles’ en el archivo llamado “controller_table_pedidodetalles.js”, en él creé las funciones getPedidoDetalle, postPedidoDetalle y deletePedidoDetalle:



```

Archivo Editar Selección Ver Ir Ejecutar ...
JS routes.js JS controller_table_pedidodetalles.js U JS pedidodetalles.js
Explorador ...
EDTORES ABIERTOS
JS routes.js Routes
X JS controller_table_pedidodetalles.js U
JS pedidodetalles.js Models
DCFSTRUER-BACKEND
Controller <-->
JS controller_table_clientes.js
JS controller_table_pedidodetalles.js U <-->
JS controller_table_pedidos.js
JS controller_table_productos.js
Database
JS database.js
img
manzanas.jpg
tomates.jpg
Models
JS clientes.js
JS pedidodetalles.js
JS pedidos.js
JS productos.js
JS usuario.js
node_modules
Routes
JS routes.js
ESQUEMA
LÍNEA DE TIEMPO
main* Git Graph
Buscar

```

```

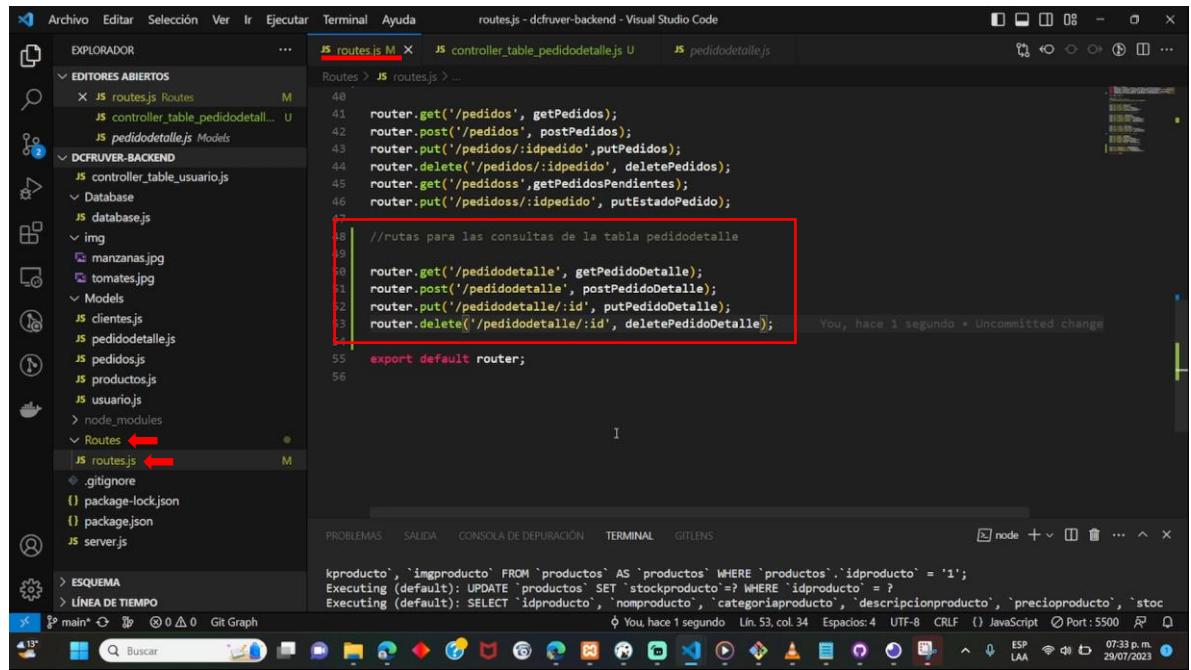
const deletePedidoDetalle = async (req, res) => {
  const { id } = req.params;
  try {
    const respuesta = await PedidoDetalle.destroy({
      where: {
        id,
      },
    });
    res.status(200).json({ mensaje: "Registro Eliminado" });
  } catch (error) {
    res.status(400).json({ mensaje: "Registro No Eliminado" + error });
  }
};

export [
  getPedidoDetalle,
  postPedidoDetalle,
  putPedidoDetalle,
  deletePedidoDetalle
];

```

Lín. 66, col. 24 Espacios: 4 UTF-8 CRLF () JavaScript Port: 5500 R Q 07:29 p.m. 29/07/2023

- Creé las rutas a cada una de las anteriores funciones en el archivo ‘routes.js’:



```

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda routes.js - dcftruver-backend - Visual Studio Code
Explorador ...
EDTORES ABIERTOS
JS routes.js M JS controller_table_pedidodetalles.js U JS pedidodetalles.js
Routes > JS routes.js ...
40 router.get('/pedidos', getPedidos);
41 router.post('/pedidos', postPedidos);
42 router.put('/pedidos/:idpedido', putPedidos);
43 router.delete('/pedidos/:idpedido', deletePedidos);
44 router.get('/pedidos', getPedidosPendientes);
45 router.put('/pedidos/:idpedido', putEstadoPedido);

46 //rutas para las consultas de la tabla pedidodetalles
47 router.get('/pedidodetalles', getPedidoDetalle);
48 router.post('/pedidodetalles', postPedidoDetalle);
49 router.put('/pedidodetalles/:id', putPedidoDetalle);
50 router.delete('/pedidodetalles/:id', deletePedidoDetalle);

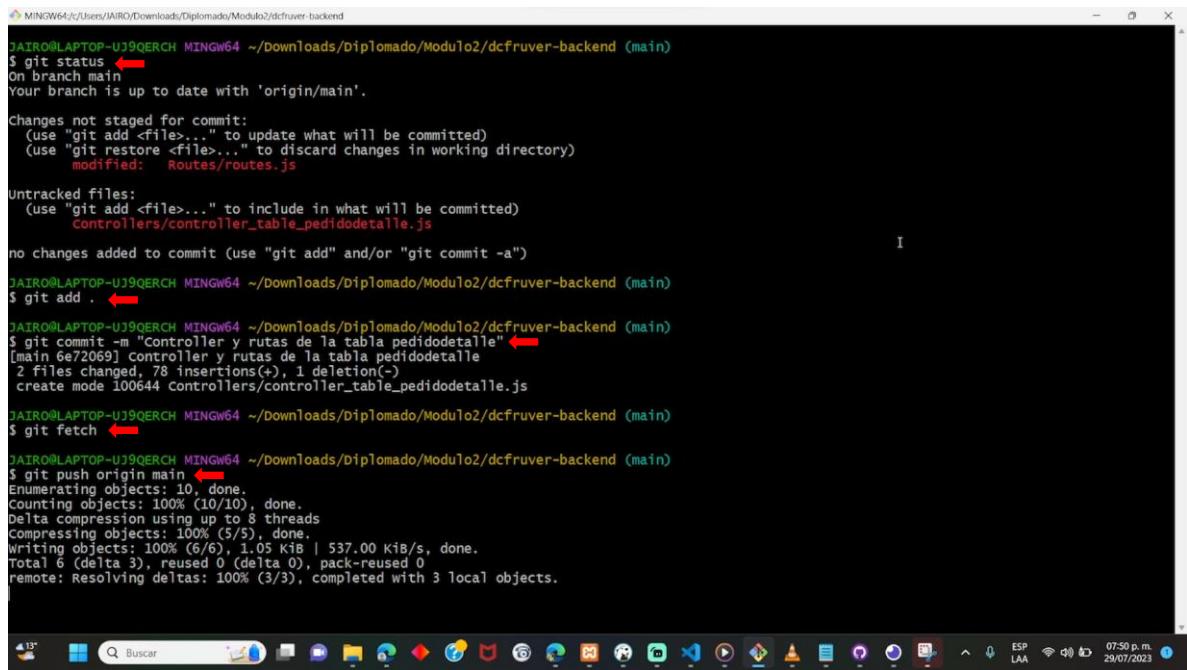
55 export default router;

```

You, hace 1 segundo * Uncommitted change

Lín. 53, col. 34 Espacios: 4 UTF-8 CRLF () JavaScript Port: 5500 R Q 07:33 p.m. 29/07/2023

- Hice un commit guardando los anteriores cambios llamado “Controller y rutas de la tabla pedidodetalie”:



```

MINGW64:/c/Users/AIRO/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified:   Routes/routes.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    controllers/controller_table_pedidodetalie.js

no changes added to commit (use "git add" and/or "git commit -a")

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git add .

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git commit -m "Controller y rutas de la tabla pedidodetalie"
[main 6e72069] controller y rutas de la tabla pedidodetalie
 2 files changed, 78 insertions(+), 1 deletion(-)
 create mode 100644 controllers/controller_table_pedidodetalie.js

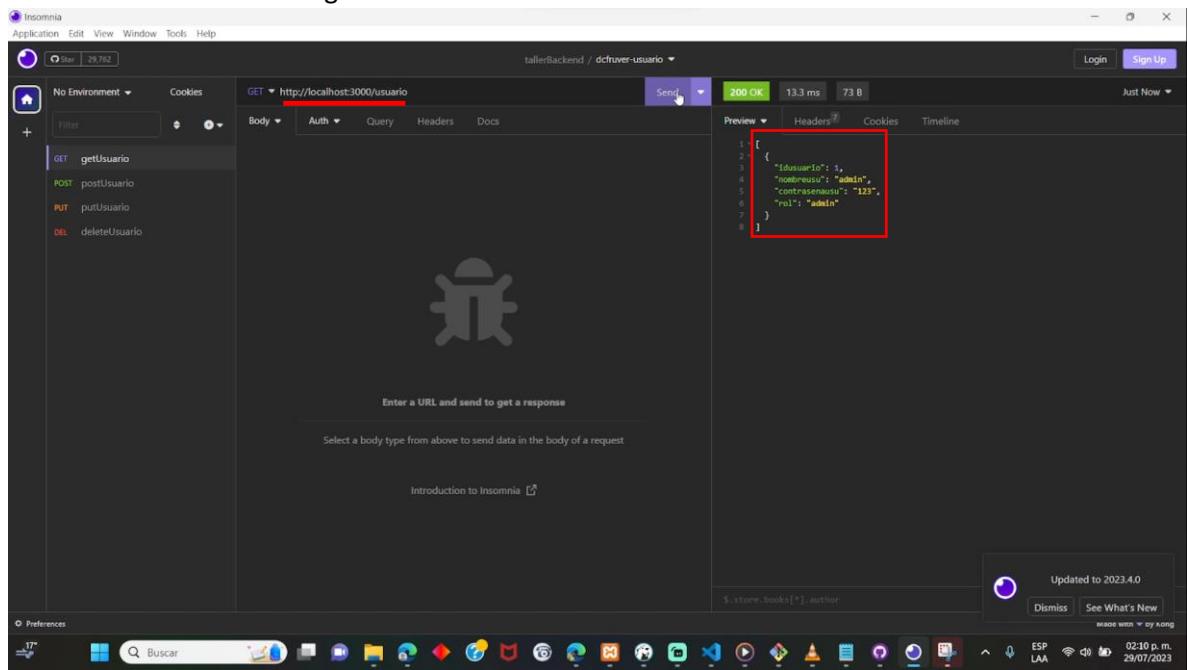
JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git fetch

JAIRO@LAPTOP-UJ9QERCH MINGW64 ~/Downloads/Diplomado/Modulo2/dcfruver-backend (main)
$ git push origin main
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 1.05 KiB | 537.00 KiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.

  
```

3. Realizar verificación de las diferentes operaciones a través de un cliente grafico (Postman, Imnsomia, etc.), tomar capturas de pantalla que evidencien el resultado de las solicitudes realizadas.

- Verificación de la solicitud getUsuario a la tabla ‘usuario’:

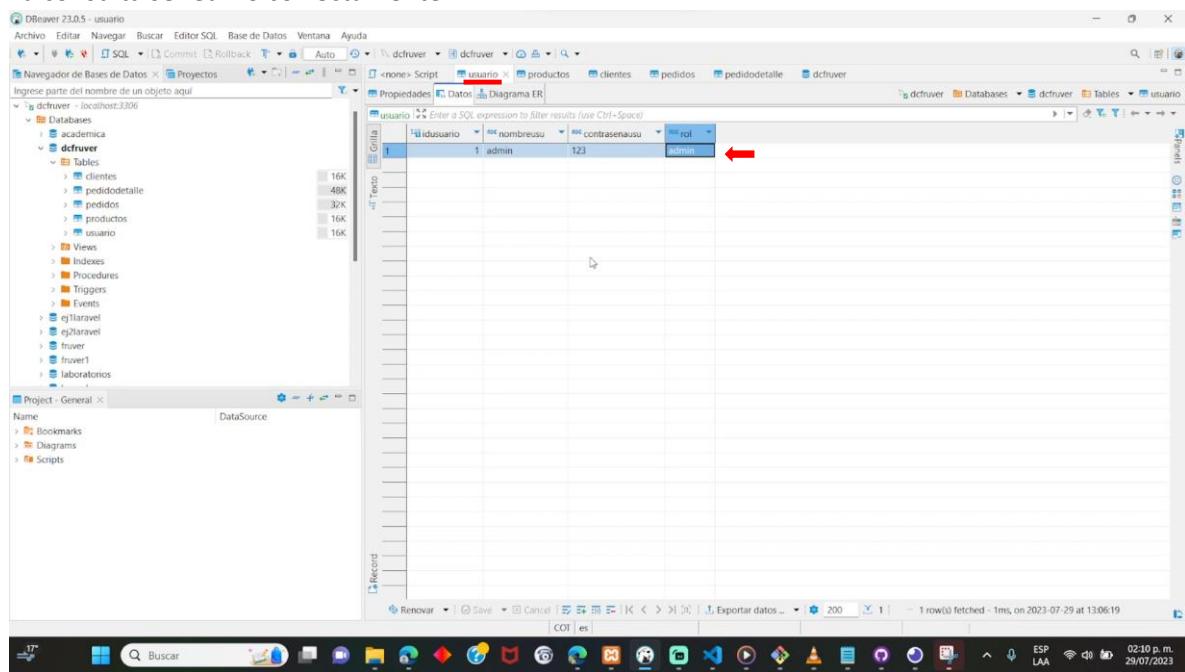


The screenshot shows the Insomnia REST client interface. A GET request is made to `http://localhost:3000/usuario`. The response is a 200 OK status with a response time of 13.3 ms and a body size of 73 B. The response body is a JSON array containing one element:

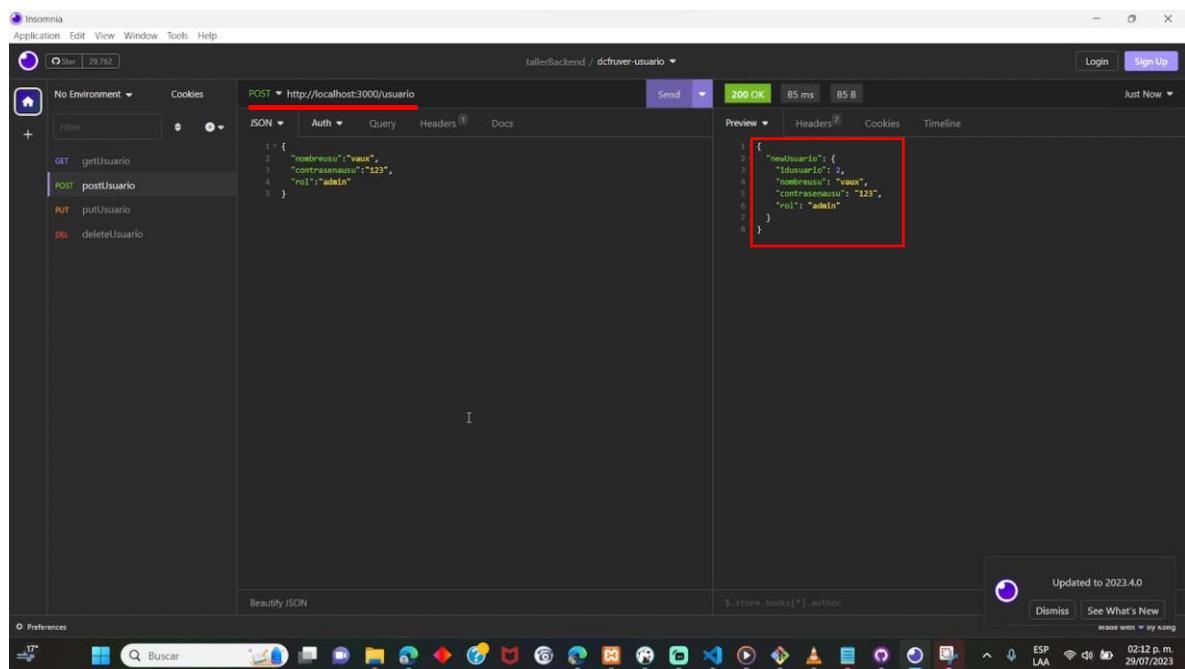
```

[
  {
    "id": 1,
    "nombreusu": "admin",
    "contrasenau": "123",
    "rol": "admin"
  }
]
  
```

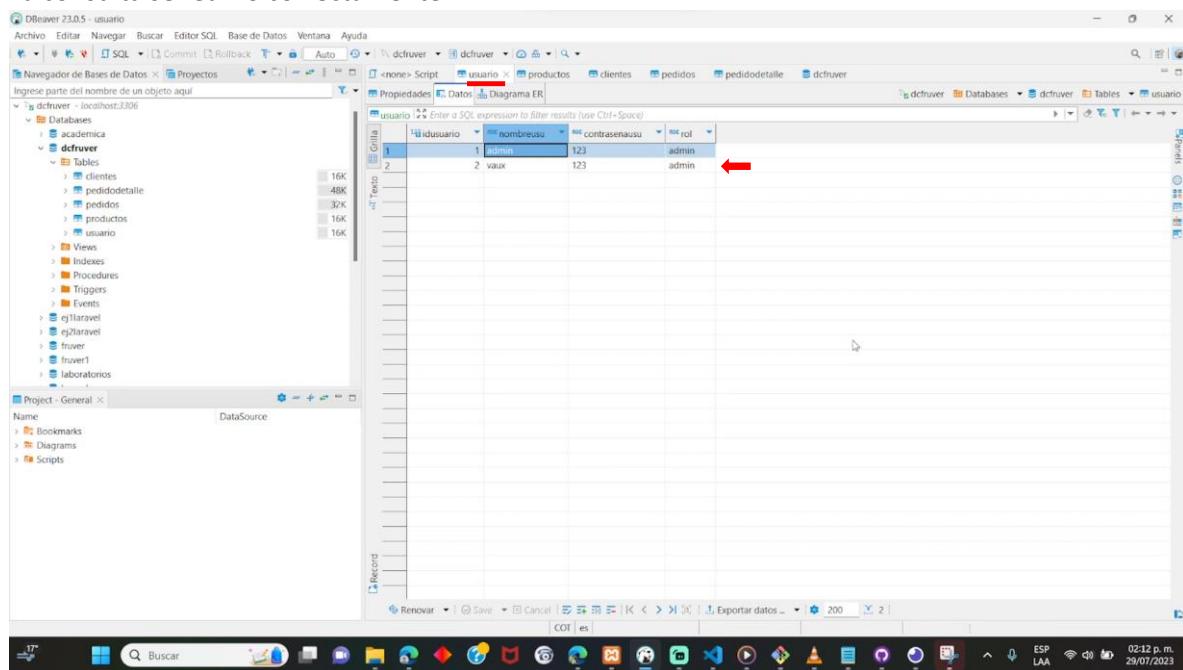
La consulta se realizó correctamente:



- Verificación de la solicitud postUsuario a la tabla ‘usuario’:



La consulta se realizó correctamente:

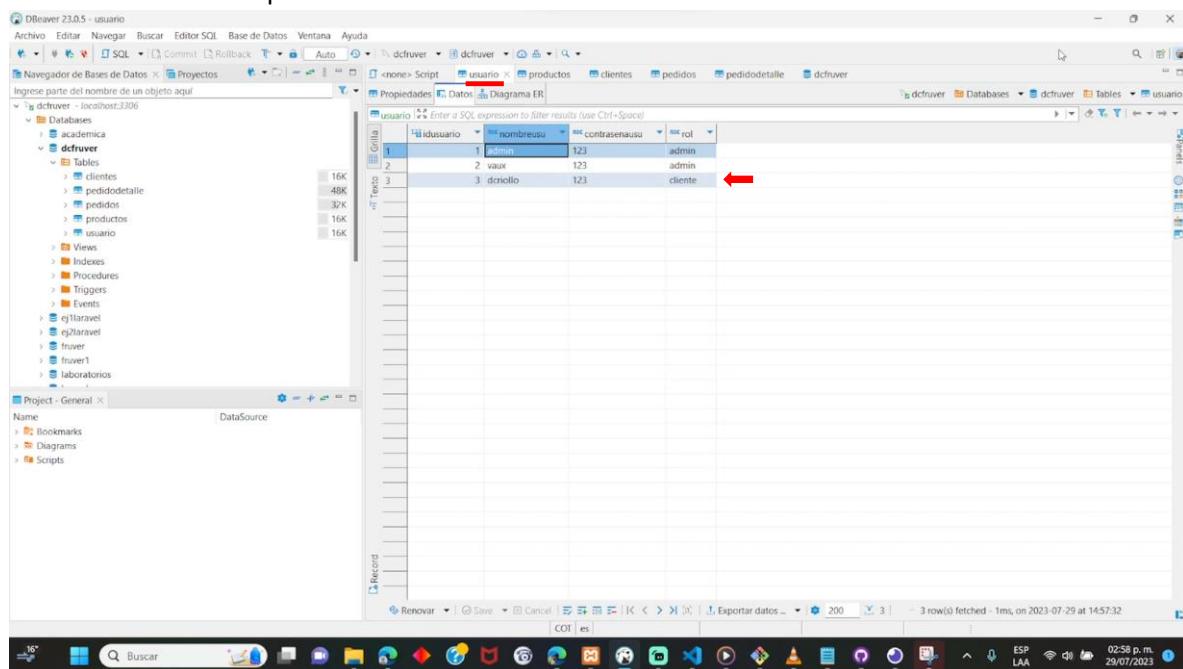


The screenshot shows the DBeaver interface with the 'usuario' table selected. The table has four columns: idusuario, nombreusu, contrasenausu, and rol. There are three rows of data:

	idusuario	nombreusu	contrasenausu	rol
1	1	admin	123	admin
2	2	vaux	123	admin
3	3	driollo	123	cliente

- Verificación de la solicitud putUsuario a la tabla ‘usuario’.

Para esta verificación primero añadí un 3 dato a la tabla ‘usuario’:



The screenshot shows the DBeaver interface with the 'usuario' table selected. The table has four columns: idusuario, nombreusu, contrasenausu, and rol. There are four rows of data:

	idusuario	nombreusu	contrasenausu	rol
1	1	admin	123	admin
2	2	vaux	123	admin
3	3	driollo	123	cliente
4	4	luis	123	admin

Hice la consulta desde Insomnia:

The screenshot shows the Insomnia REST Client interface. A PUT request is made to `http://localhost:3000/usuario/3`. The request body contains the following JSON:

```
[{"modusuario": {"idusuario": 3, "nombreusu": "dcriollomod", "contrasenau": "123mod", "rol": "clientemod"}}]
```

The response is a 200 OK status with a response time of 134 ms and 100 B. The response body is identical to the request body.

La consulta se realizó correctamente:

The screenshot shows the DBVisualizer interface connected to the `dcrfuer` database. The `usuario` table is selected, displaying three rows of data:

ID	Nombre	Contraseña	Rol
1	vaux	123	admin
2	dcriollomod	123mod	clientemod

An arrow points to the third row, indicating the modified user data.

- Verificación de la solicitud deleteUsuario a la tabla ‘usuario’:

Eliminé el registro 3 que modifiqué en la anterior verificación:

The screenshot shows the Insomnia REST Client interface. A DELETE request is made to `http://localhost:3000/usuario/3`. The response status is `200 OK` with a response time of `53.4 ms` and a size of `32 B`. The response body is displayed as a JSON object: `{ "mensaje": "Registro Eliminado" }`, which is highlighted with a red box.

La consulta se realizó correctamente:

The screenshot shows the DBeaver database client. The left sidebar shows the project structure and the `dclruver` database. The `Tables` section is expanded, showing the `usuario` table. The table data grid displays two rows:

	idusuario	nombreusu	contrasenusu	rol
1	1	admin	123	admin
2	2	valix	123	admin

- Verificación de la solicitud getCredencialesUsuario (retorna únicamente los atributos de ‘nombreusu’ y ‘contrasenausu’ a la tabla ‘usuario’):

The screenshot shows the Insomnia REST client interface. The URL in the header is `http://localhost:3000/usuario`. The response status is `200 OK` with a duration of `49.3 ms` and a size of `88 B`. The response body is a JSON array:

```
[{"nombreusu": "admin", "contrasenausu": "123"}, {"nombreusu": "vaux", "contrasenausu": "123"}]
```

- Verificación de la solicitud getProductos a la tabla ‘productos’:

The screenshot shows the Insomnia REST client interface. The URL in the header is `http://localhost:3000/productos`. The response status is `200 OK` with a duration of `13 ms` and a size of `219 B`. The response body is a JSON array:

```
[{"idproducto": 1, "nombresproducto": "Manzanas", "categoriasproducto": "Frutas", "descripcionproducto": "Manzanas frescas y jugosas. Caja de 10 kg.", "precioproducto": 25000, "stockproducto": 60, "Imagenproducto": ".\\img\\manzanas.jpg"}]
```

La consulta se realizó correctamente:

Screenshot of DBBeaver 23.0.5 showing the 'productos' table in the 'dfruver' database. The table has columns: idproducto, nomproducto, categoriaproducto, descripcionproducto, precioproducto, stockproducto, and imgproducto. A single row is selected and highlighted with a red box, containing the values: 1, Manzanas, Frutas, Manzanas frescas y jugosas, 25.000, 60, and a file path to an image.

	idproducto	nomproducto	categoriaproducto	descripcionproducto	precioproducto	stockproducto	imgproducto
1	1	Manzanas	Frutas	Manzanas frescas y jugosas	25.000	60	..\img\manzanas.jpg

- Verificación de la solicitud postProductos a la tabla ‘productos’:

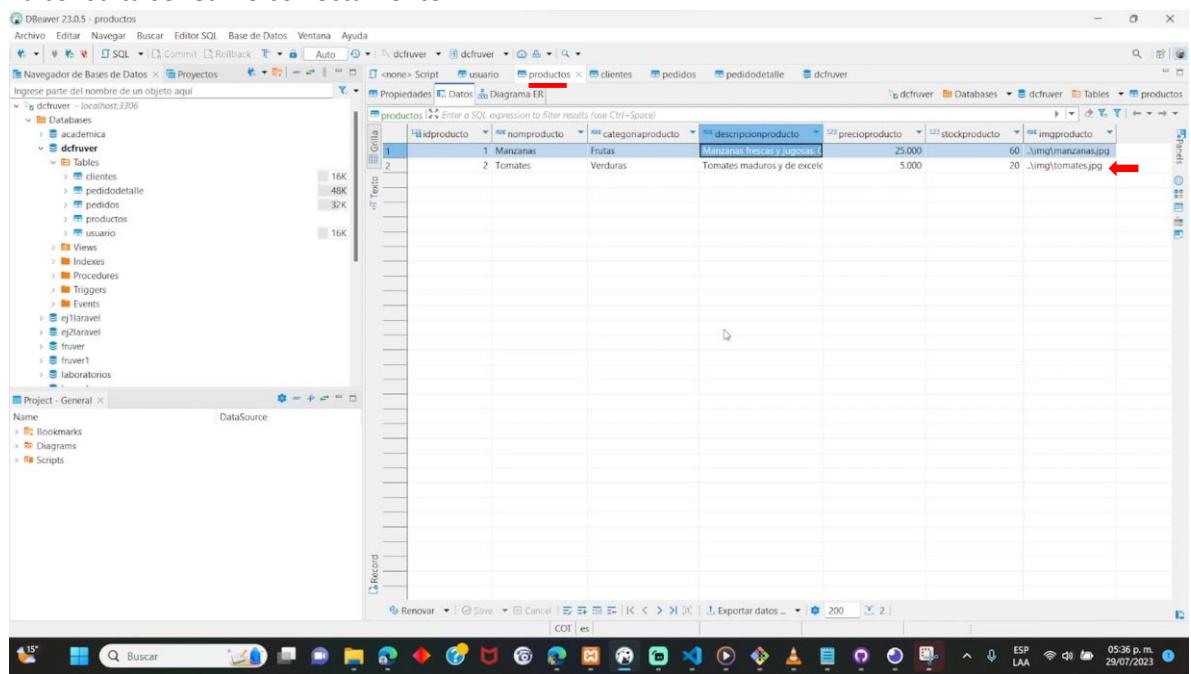
Screenshot of Insomnia Rest Client showing a POST request to 'localhost:3000/productos'. The response status is 200 OK with a response time of 162 ms and a body size of 244 B. The response body is a JSON object containing a new product entry with the same details as the one in the DBBeaver screenshot.

```

{
  "newProducto": {
    "idproducto": 1,
    "nomproducto": "Tomates",
    "categoriaproducto": "Verduras",
    "descripcionproducto": "Tomates maduros y de excelente calidad. Bolsa de 1 kg.",
    "precioproducto": 5000,
    "stockproducto": 20,
    "imgproducto": "..\\img\\tomates.jpg"
  }
}

```

La consulta se realizó correctamente:

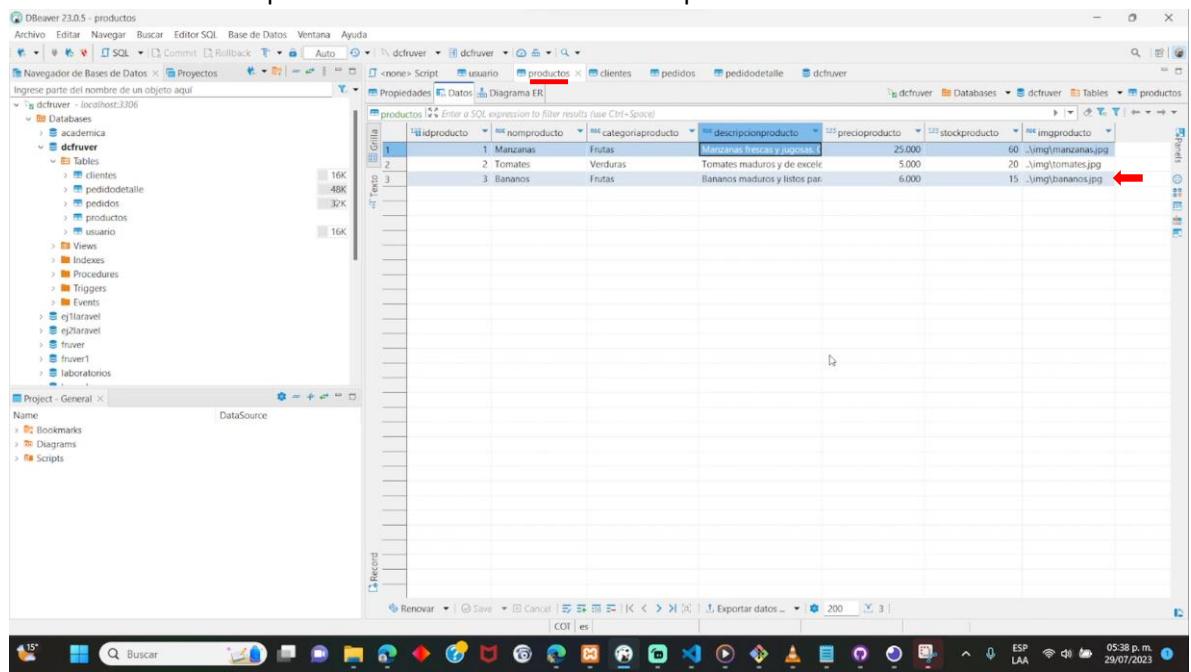


The screenshot shows the DBVisualizer interface with the 'productos' table selected. The table has columns: idproducto, nomproducto, categoria producto, descripcion producto, precio producto, stock producto, and img producto. There are two rows of data:

	idproducto	nomproducto	categoría producto	descripcion producto	precio producto	stock producto	img producto
1	1	Manzanas	Frutas	Manzanas frescas y jugosas	25.000	60	.\img\manzanas.jpg
2	2	Tomates	Verduras	Tomates maduros y de excele	5.000	20	.\img\tomates.jpg

- Verificación de la solicitud putProductos a la tabla 'productos'.

Para esta verificación primero añadí un 3 dato a la tabla 'productos':



The screenshot shows the DBVisualizer interface with the 'productos' table selected. The table now has three rows of data:

	idproducto	nomproducto	categoría producto	descripcion producto	precio producto	stock producto	img producto
1	1	Manzanas	Frutas	Manzanas frescas y jugosas	25.000	60	.\img\manzanas.jpg
2	2	Tomates	Verduras	Tomates maduros y de excele	5.000	20	.\img\tomates.jpg
3	3	Bananas	Frutas	Bananos maduros y listos par	6.000	15	.\img\bananas.jpg

Hice la consulta desde Insomnia:

The screenshot shows the Insomnia REST Client interface. The top bar displays "tallerBackend / dcfrover-productos" and "200 OK". The left sidebar lists several API endpoints under "GET getProductos" and one under "PUT getProductos". The main area shows a JSON response for the PUT request to "/productos/3". The response object contains a single product entry with the following fields:

```
1: {
  "idproducto": 3,
  "nomproducto": "Bananos mod",
  "categoria producto": "Frutas mod",
  "descripcionproducto": "Bananos maduros y listos para comer. Racimo de 5 unidades. mod.",
  "precioproducto": 0,
  "stockproducto": 1,
  "imgproducto": "...\\img\\bananos.jpg"
}
```

The entire JSON response is highlighted with a red box.

La consulta se realizó correctamente:

The screenshot shows the DBVisualizer tool interface. The left sidebar shows a tree view of the database structure, including databases like "dcfrover", tables like "clientes", "pedidos", and "productos", and other objects like "Views", "Indexes", and "Procedures". The main area displays the "productos" table in a grid format. The table has columns: idproducto, nomproducto, categoria producto, descripcionproducto, precioproducto, stockproducto, and imgproducto. There are three rows of data:

	idproducto	nomproducto	categoría producto	descripcionproducto	precioproducto	stockproducto	imgproducto
1	1	Manzanas	Frutas	Manzanas frescas y jugosas. Caja de 6 unidades.	25.000	60	...\\img\\manzana.jpg
2	2	Tomates	Verduras	Tomates maduros y de excelente calidad.	5.000	20	...\\img\\tomates.jpg
3	3	Bananos mod	Frutas mod	Bananos maduros y listos para comer. Racimo de 5 unidades. mod.	0	1	...\\img\\bananos.jpg

A red arrow points to the last row of the grid, indicating the updated banana record.

- Verificación de la solicitud deleteProductos a la tabla ‘productos’:

Eliminé el registro 3 que modifiqué en la anterior verificación:

The screenshot shows the Insomnia REST Client interface. The URL is `localhost:3000/productos/3`. The response status is `200 OK` with a response time of `19.6 ms` and a size of `32 B`. The response body is a JSON object: `{ "mensaje": "Registro Eliminado" }`.

La consulta se realizó correctamente:

The screenshot shows the DBVisualizer tool interface. The database selected is `dclfruver` and the table is `productos`. The table structure includes columns: `idproducto`, `nomproducto`, `categoria producto`, `descripcion producto`, `precioproducto`, `stockproducto`, and `imgproducto`. There are two rows of data: Manzanas and Tomates. The row for Manzanas is highlighted with a red box.

	<code>idproducto</code>	<code>nomproducto</code>	<code>categoria producto</code>	<code>descripcion producto</code>	<code>precioproducto</code>	<code>stockproducto</code>	<code>imgproducto</code>
1	1	Manzanas	Frutas	Manzanas frescas y jugosas	25.000	60	./img/manzanas.jpg
2	2	Tomates	Verduras	Tomates maduros y de excele	5.000	20	./img/tomates.jpg

- Verificación de la solicitud getStockProducto (retorna el valor de la variable ‘stockproducto’ de un producto específico) a la tabla ‘productos’:

The screenshot shows the Insomnia REST client interface. The URL in the header is `localhost:3000/productos/1`. The response status is `200 OK` with a time of `29 ms` and a size of `20 B`. The response body is a JSON object:

```
1: {  
2:   "stockproducto": 60  
3: }
```

. The entire response body is highlighted with a red box.

La consulta se realizó correctamente porque como vemos en el anterior pantallazo de la tabla ‘productos’, el producto con ‘idproducto’=1, sí tiene como ‘stockproducto’=60.

- Verificación de la solicitud putStockProductos (modifica el valor de la variable ‘stockproducto’ de un producto específico) a la tabla ‘productos’:

The screenshot shows the Insomnia REST client interface. The URL in the header is `PUT localhost:3000/productos/1`. The response status is `200 OK` with a time of `22.1 ms` and a size of `276 B`. The response body is a JSON object:

```
1: {  
2:   "mensaje": "Producto actualizado exitosamente",  
3:   "producto": {  
4:     "idproducto": 1,  
5:     "nombresproducto": "Manzanas",  
6:     "categoriasproducto": "Frutas",  
7:     "descripcionproducto": "Manzanas frescas y jugosas. Caja de 10 kg.",  
8:     "precioproducto": 25000,  
9:     "stockproducto": 55,  
10:    "legoproducto": "..\\Img\\Manzanas.jpg"  
11:  }  
12: }
```

. Both the request body and the response body are highlighted with red boxes.

La consulta se realizó correctamente, ya que al producto con 'idproducto'=1, se le actualizó el atributo 'stockproducto' con el valor 55:

	idproducto	nomproducto	categoría producto	descripcion producto	precio producto	stock producto	img producto
1	1	Manzanas	Frutas	Manzana blanca y jugosa	25.000	55	\img\manzanas.jpg
2	2	Tomates	Verduras	Tomates maduros y de excelente calidad	5.000	20	\img\tomates.jpg

- Verificación de la solicitud getClientes a la tabla 'clientes':

```

[{"id": 1, "cedulacliente": "1111", "nombcliente": "luis Torres", "dircliente": "Pasto - Nari", "telcliente": "7282028", "correocliente": "luis.torres@example.com"}]

```

La consulta se realizó correctamente:

The screenshot shows the DBeaver interface with the following details:

- Project:** tallerBackend / dcfruver - pedidos
- Table:** clientes
- Result Grid:** One row is displayed with the following data:

cedulacliente	nomcliente	dircliente	telcliente	correocliente
1111	Luis Torres	Pasto - Nar	7202020	LuisTorres@example.com
- Status Bar:** Inserted: 1 / Deleted: 0 / Updated: 0

- Verificación de la solicitud postClientes a la tabla ‘clientes’:

The screenshot shows the Insomnia REST Client interface with the following details:

- Method:** POST
- URL:** localhost:3000/clientes
- Headers:** Content-Type: application/json
- Body (JSON):**

```
1: {
2:   "cedulacliente": "2222",
3:   "nomcliente": "Carmen López",
4:   "dircliente": "Linares - Nar",
5:   "telcliente": "7182020",
6:   "correocliente": "carmen.lopez@example.com"
7: }
```
- Response:** 200 OK | 177 ms | 163 B
- Preview:** Shows the JSON response from the previous step.
- Bottom Status Bar:** Updated to 2023.4.0

La consulta se realizó correctamente:

DBeaver 23.0.5 - clientes

Navegador de Bases de Datos | Proyectos

Ingresar parte del nombre de un objeto aquí

Propiedades | Datos | Diagrama ER

	cedulacliente	nomcliente	dircliente	telcliente	correocliente
1	1111	Luis Torres	Pasto - Nar	7202020	luis.torres@example.com
2	2222	Carmen López	Linares - Nar	7102020	carmen.lopez@example.com

Record | Renovar | Save | Cancel | Exportar datos ... | 200 | COT | es | 06:29 p.m. 29/07/2023

- Verificación de la solicitud putClientes a la tabla ‘clientes’:

Para esta verificación primero añadí un 3 dato a la tabla ‘productos’:

DBeaver 23.0.5 - clientes

Navegador de Bases de Datos | Proyectos

Ingresar parte del nombre de un objeto aquí

Propiedades | Datos | Diagrama ER

	cedulacliente	nomcliente	dircliente	telcliente	correocliente
1	1111	Luis Torres	Pasto - Nar	7202020	luis.torres@example.com
2	2222	Carmen López	Linares - Nar	7102020	carmen.lopez@example.com
3	3333	Juan Martínez	Bogotá	7302020	juan.martinez@example.com

Record | Renovar | Save | Cancel | Exportar datos ... | 200 | COT | es | 06:30 p.m. 29/07/2023

Hice la consulta desde Insomnia:

The screenshot shows the Insomnia REST Client interface. The URL is `PUT localhost:3000/clients/3333`. The response status is `200 OK` with a time of `51.1 ms` and a size of `162 B`. The response body is highlighted with a red box and contains the following JSON data:

```
1: {
2:   "nombcliente": "Juan Martinez mod",
3:   "dirccliente": "Bogotá mod",
4:   "telcliente": "00000",
5:   "correocliente": "juan.martinez@mod.com"
6: }
```

La consulta se realizó correctamente:

The screenshot shows the DBVisualizer 23.0.5 interface. The database is `dfriver` and the table is `clients`. The data grid shows the following rows:

	cedulacliente	nombcliente	dirccliente	telcliente	correocliente
1	1111	Luis Torres	Patio - Nar	7202020	luis.torres@exams
2	2222	Carmen Lopez	Linares - Nar	7102020	carmelopez@exams
3	3333	Juan Martinez mod	Bogota mod	00000	juan.martinez@mod.com

A red arrow points to the last row (id 3) which has been updated.

- Verificación de la solicitud deleteClientes a la tabla ‘clientes’:

Eliminé el registro con ‘cedulacliente’=3333 que modifiqué en la anterior verificación:

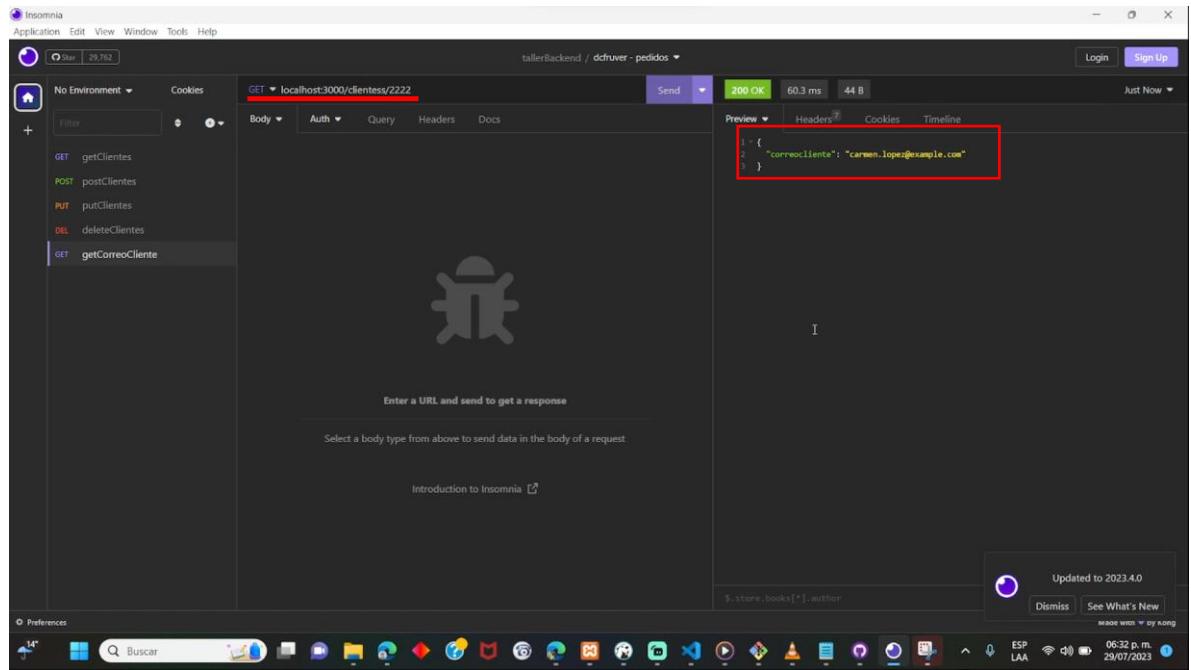
The screenshot shows the Insomnia REST Client interface. The URL is `localhost:3000/clientes/3333`. The response status is `200 OK` with a response time of `20.3 ms` and a size of `32 B`. The response body is a JSON object: `{ "mensaje": "Registro Eliminado" }`. The left sidebar shows various API endpoints: GET getClientes, POST postClientes, PUT putClientes, and DELETE deleteClientes.

La consulta se realizó correctamente:

The screenshot shows the DBeaver database client. The left sidebar shows the database structure with tables like `academica`, `dcliver`, `clientes`, `pedidodetalle`, `pedidos`, `productos`, and `usuario`. The main window displays the `clientes` table with the following data:

	cedulacliente	nombrecliente	dircliente	telcliente	correocliente
1	1111	Luis Torres	Paito - Nar	7202020	luis.torres@example.com
2	2222	Carmen Lopez	Linares - Nar	7102020	carmen.lopez@example.com

- Verificación de la solicitud getCorreoCliente (retorna el valor de la variable 'correocliente' de un cliente específico) a la tabla 'clientes':



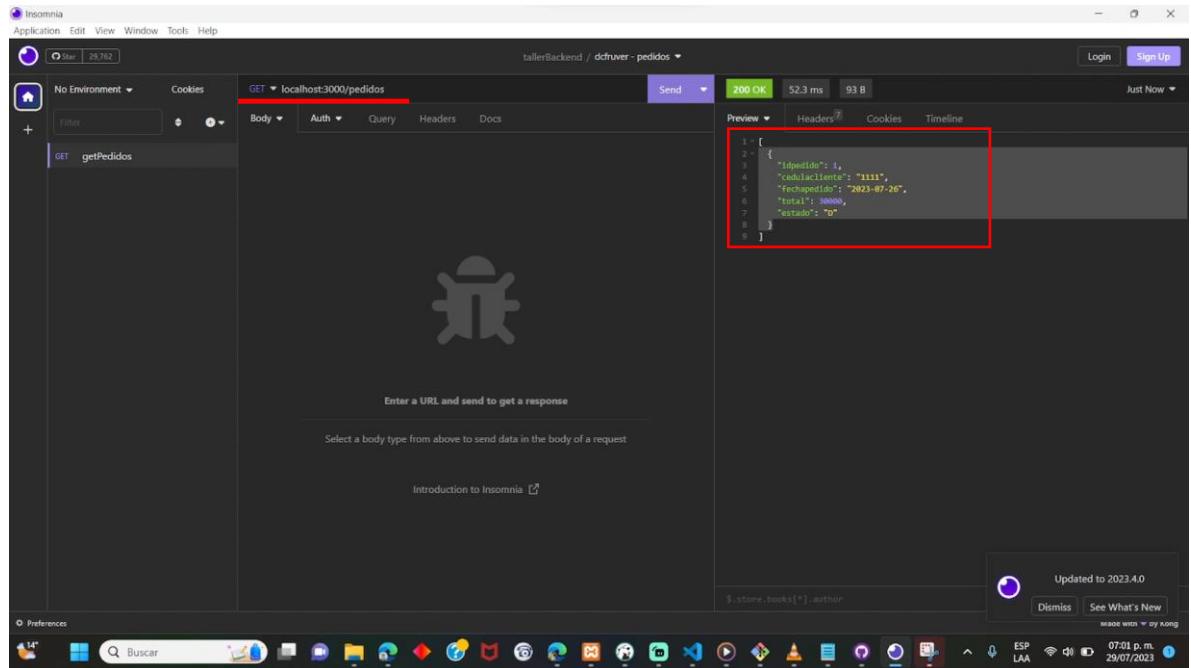
```

{
  "correocliente": "carmen.lopez@example.com"
}

```

La consulta se realizó correctamente, ya que al cliente con 'cedulacliente'=2222, sí tiene como atributo 'correocliente' con el valor 'carmen.lopez@example.com', como lo podemos ver en el anterior pantallazo de la tabla 'clientes'.

- Verificación de la solicitud getPedidos a la tabla 'pedidos':



```

[{"idpedido": 1, "cedulacliente": "1111", "fechapedido": "2023-07-26", "total": 30000, "estado": "0"}]

```

La consulta se realizó correctamente:

The screenshot shows the DBeaver 23.0.5 interface with the 'pedidos' table selected. A red arrow points to the last row of the results table, which contains the following data:

idpedido	cedulacliente	fechapedido	total	estado
1	1111	2023-07-26	30.000	P

- Verificación de la solicitud postPedidos a la tabla ‘pedidos’, en esta tabla agregare 2 registros:

The screenshot shows the Insomnia Rest Client interface with a successful POST request to the 'pedidos' endpoint. A red box highlights the JSON response body, which contains the following data:

```
1 + {
2 +   "newpedido": {
3 +     "idpedido": 2,
4 +     "cedulacliente": "1111",
5 +     "fechapedido": "2023-07-27",
6 +     "total": 60000,
7 +     "estado": "P"
8 +   }
9 }
```

Screenshot of the Insomnia REST Client showing a successful POST request to `localhost:3000/pedidos`. The response body is highlighted with a red box:

```
1: {
2:   "cedulacliente": "2222",
3:   "fechapedido": "2023-07-27",
4:   "total": 5000,
5:   "estado": "P"
6: }
```

The response status is 200 OK with a time of 22.6 ms and 118 B.

La consulta se realizó correctamente:

Screenshot of the DBVisualizer 23.0.5 database client showing the results of a query on the `pedidos` table. The results are highlighted with red arrows:

idpedido	cedulacliente	fechapedido	total	estado
1	1111	2023-07-26	30.000	P
2	1111	2023-07-27	60.000	P
3	2222	2023-07-27	5.000	P

- Verificación de la solicitud putPedidos a la tabla ‘pedidos’:

Hice la consulta desde Insomnia, modifique el pedido con ‘idpedido’=2 :

The screenshot shows the Insomnia REST Client interface. The URL is `PUT localhost:3000/pedidos/2`. The response status is `200 OK` with a response time of `94.5 ms` and a size of `119 B`. The response body is highlighted with a red box and contains the following JSON data:

```

1+ {
2+   "idpedido": 1,
3+   "idpedido": 2,
4+   "cadulacliente": "1111",
5+   "fechapedido": "2023-07-28",
6+   "total": 60000,
7+   "estado": "P"
8+
9}

```

La consulta se realizó correctamente:

The screenshot shows the DBeaver database client. The left sidebar shows the database structure for the `dfruver` database, including tables like `academic`, `clientes`, `pedidos`, `pedidodetalle`, and `usuario`. The main window displays the `pedidos` table with the following data:

	idpedido	cadulacliente	fechapedido	total	estado
1	1	1111	2023-07-26	30.000	P
2	2	1111	2023-07-28	65.000	P
3	3	2222	2023-07-27	5.000	P

- Verificación de la solicitud deletePedidos a la tabla ‘pedidos’:

Eliminé el registro con ‘idpedido’=3:

The screenshot shows the Insomnia REST Client interface. The URL is `localhost:3000/pedidos/3`. The response status is `200 OK` with a time of `49.7 ms` and a size of `32 B`. The response body is a JSON object: `{ "mensaje": "Registro Eliminado" }`.

La consulta se realizó correctamente:

The screenshot shows the DBVisualizer tool. The left sidebar shows the database structure with the `dcfruer` database selected. The `pedidos` table is open, displaying the following data:

idpedido	cedulacliente	fechapedido	total	estado
1	1111	2023-07-26	30.000	P
2	1111	2023-07-28	65.000	P

- Verificación de la solicitud putEstadoPedido (Modifica el ‘estado’ de ‘P’ a ‘C’, para decir que un pedido específico ya no está en estado pendiente sino en confirmado) a la tabla ‘pedidos’:

Modifico el registro con ‘idpedido’=1:

```

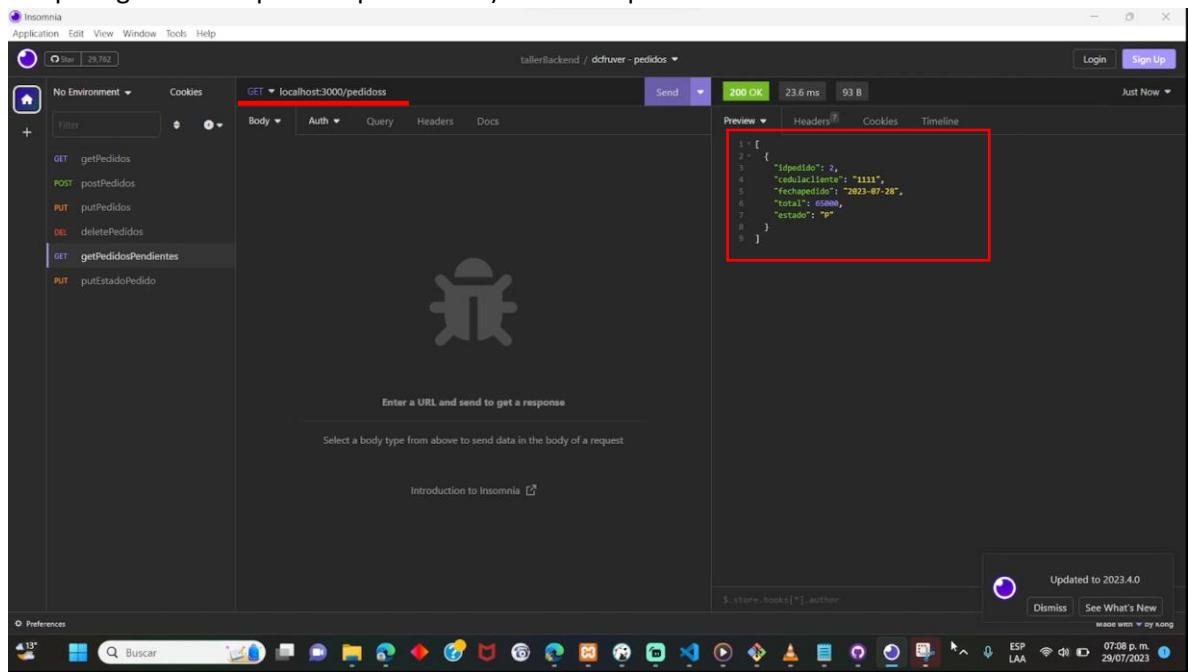
1: {
2:   "mensaje": "Estado del pedido confirmado",
3:   "pedido": {
4:     "idpedido": 1,
5:     "cedulacliente": "1111",
6:     "fechapedido": "2023-07-26",
7:     "total": 30000,
8:     "estado": "C"
9:   }
10: }

```

La consulta se realizó correctamente:

	idpedido	cedulacliente	fechapedido	total	estado
1	1111		2023-07-26	30.000	C
2	1111		2023-07-28	65.000	P

- Verificación de la solicitud `getPedidosPendientes` (retorna todos los pedidos con 'estado'='P' que significan los pedidos pendientes) a la tabla 'pedidos':



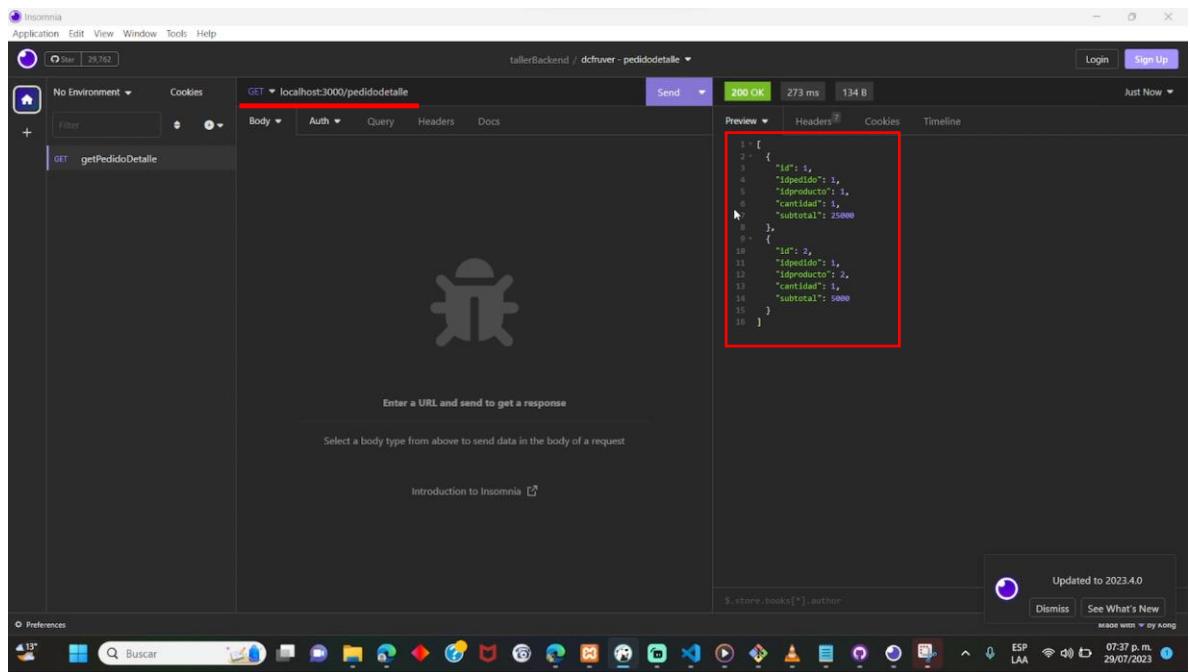
```

[{"idpedido": 1, "idcliente": "1111", "fechadetalle": "2023-07-28", "total": 6500, "estado": "P"}, {"idpedido": 2, "idcliente": "2222", "fechadetalle": "2023-07-28", "total": 5000, "estado": "P"}]

```

La consulta se realizó correctamente, en la anterior consulta nos podemos dar cuenta que solo el pedido con 'idpedido'=2 quedo con 'estado'='P'.

- Verificación de la solicitud `getPedidoDetalle` a la tabla 'pedidodetalle':



```

[{"id": 1, "idpedido": 1, "idproducto": 1, "cantidad": 1, "subtotal": 25000}, {"id": 2, "idpedido": 1, "idproducto": 2, "cantidad": 2, "subtotal": 50000}]

```

La consulta se realizó correctamente:

DBeaver 23.0.5 - pedidodetalie

Navegador de Bases de Datos × Proyectos × <none> Script usuario productos clientes pedidos pedidodetalie dcfriver

Ingresar parte del nombre de un objeto aquí

pedidodetalie Enter a SQL expression to filter results (use Ctrl+Space)

id	idpedido	idproducto	cantidad	subtotal
1	1	1	1	25.000
2	2	1	2	5.000

Project - General

Name: DataSource

Bookmarks

Diagrams

scripts

Renovar Save Cancel Exportar datos 200 2 Inserted: 2 / Deleted: 0 / Updated: 0

07:36 p. m. 29/07/2023

- Verificación de la solicitud postPedidoDetalle a la tabla ‘pedidodetalie’, en esta tabla agregare 3 registros:

Insomnia

Application Edit View Window Tools Help

tallerBackend / dcfriver - pedidodetalie

POST localhost:3000/pedidodetalie

Send 200 OK 129 ms 87 B Just Now

No Environment Cookies

Preview Headers Cookies Timeline

```
1: {
2:   "idpedido": 2,
3:   "idproducto": 1,
4:   "cantidad": 2,
5:   "subtotal": 50000
6: }
```

Updated to 2023.4.0

Dismiss See What's New

Beauty JSON \$ store.books[*].author

Preferences

07:46 p. m. 29/07/2023

Insomnia

Application Edit View Window Tools Help

localhost:3000/pedidodetalle

POST 200 OK 24.9 ms 87 B Just Now

No Environment Cookies

Filter

GET getPedidoDetalle

POST postPedidoDetalle

PUT New Request

Send Headers Preview Cookies Timeline

```
1 + {
2 +   "idpedido": 2,
3 +   "idproducto": 2,
4 +   "cantidad": 3,
5 +   "subtotal": 15000
6 + }
```

Updated to 2023.4.0 Dismiss See What's New

Beauty JSON \$ store.books[*].author

13° Buscar ESP LAA 07:46 p.m. 29/07/2023

Insomnia

Application Edit View Window Tools Help

localhost:3000/pedidodetalle

POST 200 OK 11.8 ms 87 B Just Now

No Environment Cookies

Filter

GET getPedidoDetalle

POST postPedidoDetalle

PUT New Request

Send Headers Preview Cookies Timeline

```
1 + {
2 +   "idpedido": 3,
3 +   "idproducto": 3,
4 +   "cantidad": 4,
5 +   "subtotal": 15000
6 + }
```

Updated to 2023.4.0 Dismiss See What's New

Beauty JSON \$ store.books[*].author

13° Buscar ESP LAA 07:46 p.m. 29/07/2023

La consulta se realizó correctamente:

The screenshot shows the DBeaver interface with the 'pedidodetalle' table selected. A red arrow points to the last row of the table, which has an id of 5. The table structure includes columns: id, idpedido, idproducto, cantidad, and subtotal. The data for row 5 is: id=5, idpedido=1, idproducto=2, cantidad=4, subtotal=160000.

	id	idpedido	idproducto	cantidad	subtotal
1	1	1	1	1	25.000
2	2	2	2	1	5.000
3	3	2	2	2	50.000
4	4	2	2	3	15.000
5	5	1	2	4	15.000

- Verificación de la solicitud `putPedidoDetalle` a la tabla ‘pedidodetalle’, modificar el registro con ‘id’=5:

The screenshot shows the Insomnia Rest Client interface. A red box highlights the JSON payload sent in the PUT request to 'localhost:3000/pedidodetalle/5'. The payload is a JSON object with fields: idpedido (2), idproducto (1), cantidad (4), and subtotal (100000).

```
1 {
2   "idpedido": 2,
3   "idproducto": 1,
4   "cantidad": 4,
5   "subtotal": 100000
6 }
```

La consulta se realizó correctamente:

DBeaver 23.0.5 - pedidodetalle

Navegador de Bases de Datos | Proyectos

Ingresar parte del nombre de un objeto aquí

pedidodetalle

Enter a SQL expression to filter results (use Ctrl+Space)

id	idpedido	idproducto	cantidad	subtotal
1	1	1	1	25,000
2	2	2	2	50,000
3	3	2	1	25,000
4	4	2	3	75,000
5	5	2	4	100,000

Project - General

Name: Data Source: dcfuvver

Renovar Save Cancel | Exportar datos ... | 200 | 5 | 5 row(s) fetched - 1ms, on 2023-07-29 at 19:48:20

COT es

ESP LAA 07:48 p.m. 29/07/2023

- Verificación de la solicitud deletePedidoDetalle a la tabla 'pedidodetalle', eliminaré el registro con 'id'=5:

Insomnia

Application Edit View Window Tools Help

DELETE /localhost:3000/pedidodetalle/5

Send 200 OK 28.8 ms 32 B Just Now

Preview Headers Cookies Timeline

```
{ "mensaje": "Registro Eliminado" }
```

No Environment Cookies

Body Auth Query Headers Docs

GET getPedidoDetalle

POST postPedidoDetalle

PUT putPedidoDetalle

DELETE deletePedidoDetalle

Enter a URL and send to get a response

Select a body type from above to send data in the body of a request

Introduction to Insomnia

Updated to 2023.4.0 Dismiss See What's New

Preferences

13" Buscar

ESP LAA 07:49 p.m. 29/07/2023

La consulta se realizó correctamente:

The screenshot shows the DBeaver 23.0.5 interface with the title bar "DBeaver 23.0.5 - pedidodetalle". The menu bar includes Archivo, Editar, Navegar, Buscar, Editor SQL, Base de Datos, Ventana, Ayuda. The toolbar has icons for New, Open, Save, Import, Export, and others. The main window displays a table titled "pedidodetalle" with the following data:

#	id	idpedido	idproducto	cantidad	subtotal
1	1	1	1	1	25.000
2	2	1	2	1	5.000
Totales	3	3	2	2	30.000
	4	4	2	2	15.000

The table is highlighted with a red border. The bottom status bar shows "COT es" and the system tray indicates it's 07:49 p.m. on 29/07/2023.