Pruebas Documentadas

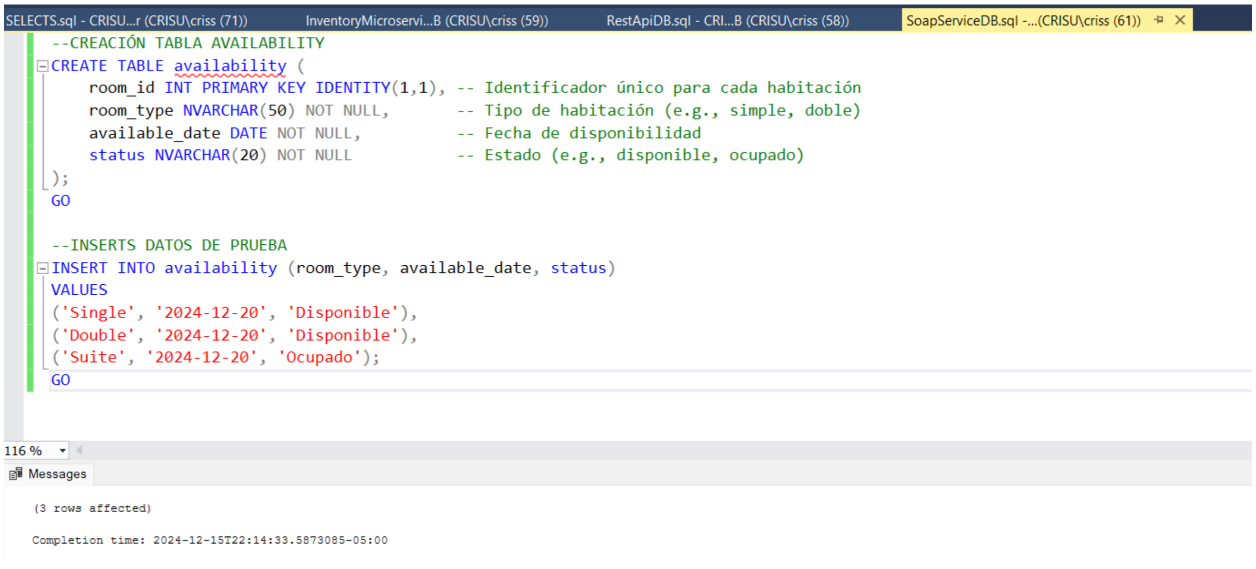
Nombre: Chicaiza Cristopher

Materia: Integración de Sistemas

Tarea: Examen Practico Progreso 2

Ejecución SQL

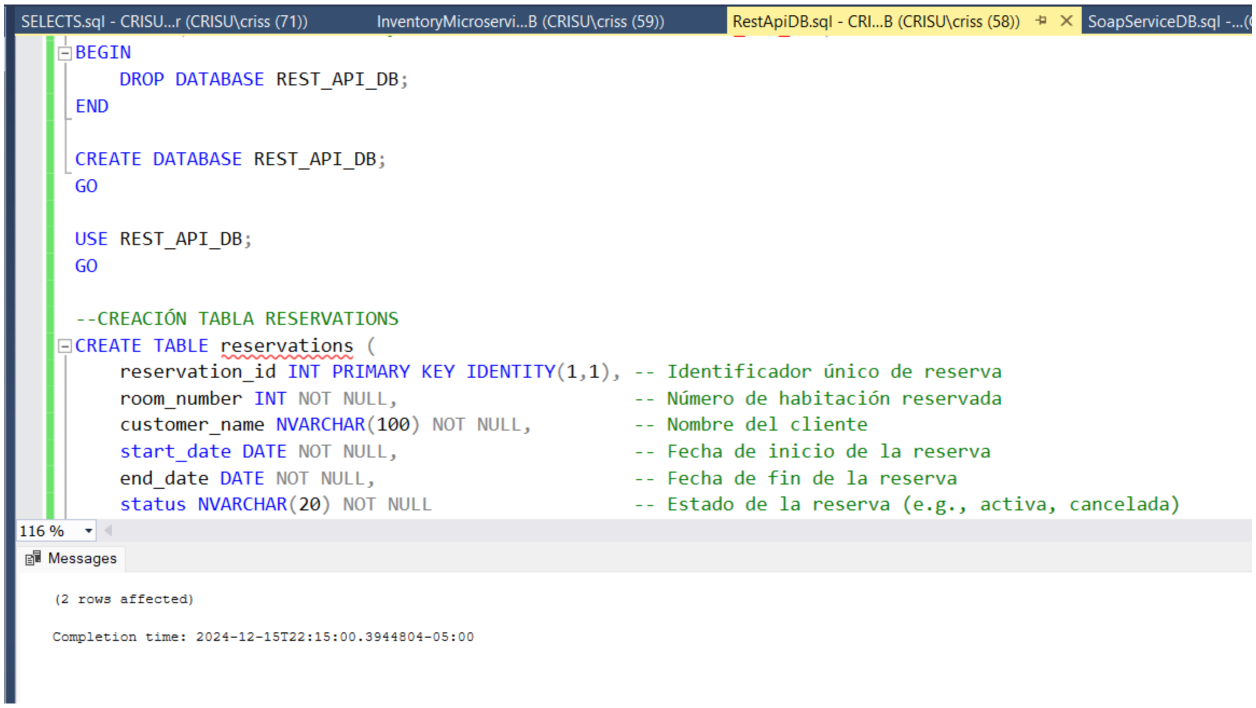
SoapService


SELECTS.sql - CRISU...r (CRISU\criss (71))
InventoryMicroservi...B (CRISU\criss (59))
RestApiDB.sql - CRI...B (CRISU\criss (58))
SoapServiceDB.sql -...(CRISU\criss (61))
116 %
Messages

```sql
--CREACIÓN TABLA AVAILABILITY
CREATE TABLE availability (
    room_id INT PRIMARY KEY IDENTITY(1,1), -- Identificador único para cada habitación
    room_type NVARCHAR(50) NOT NULL,       -- Tipo de habitación (e.g., simple, doble)
    available_date DATE NOT NULL,          -- Fecha de disponibilidad
    status NVARCHAR(20) NOT NULL           -- Estado (e.g., disponible, ocupado)
);
GO

--INSERTS DATOS DE PRUEBA
INSERT INTO availability (room_type, available_date, status)
VALUES
('Single', '2024-12-20', 'Disponible'),
('Double', '2024-12-20', 'Disponible'),
('Suite', '2024-12-20', 'Ocupado');
GO
```

```
(3 rows affected)

Completion time: 2024-12-15T22:14:33.5873085-05:00
```

RestApi


SELECTS.sql - CRISU...r (CRISU\criss (71)) InventoryMicroservi...B (CRISU\criss (59)) RestApiDB.sql - CRI...B (CRISU\criss (58)) SoapServiceDB.sql -...(

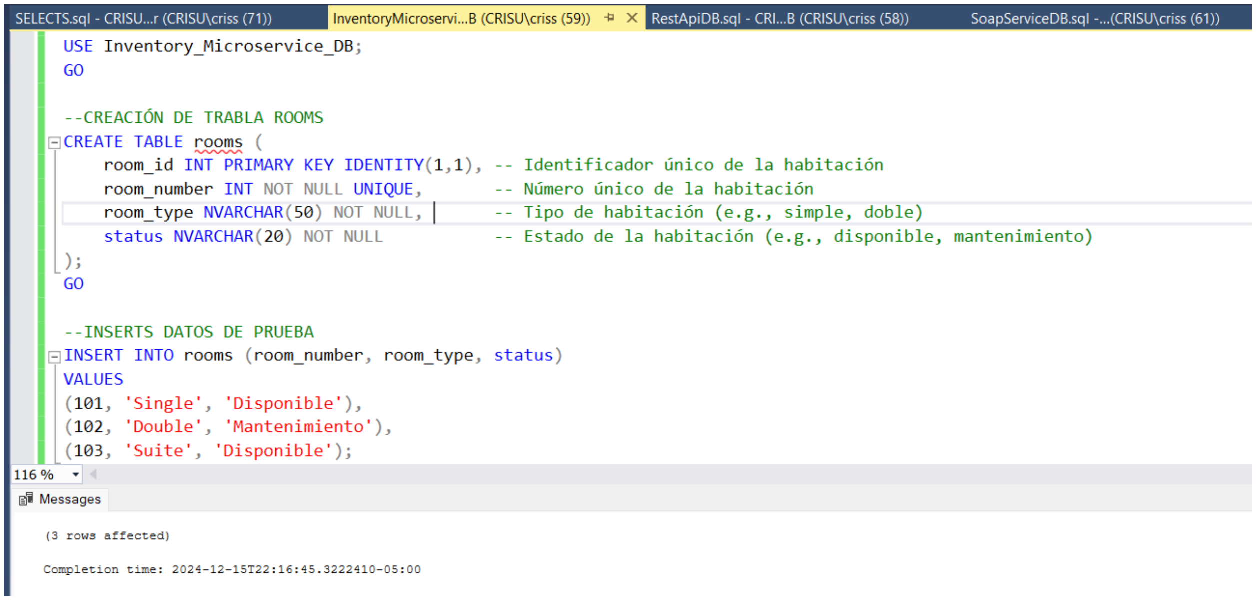
```
BEGIN
    DROP DATABASE REST_API_DB;
END

CREATE DATABASE REST_API_DB;
GO

USE REST_API_DB;
GO

--CREACIÓN TABLA RESERVATIONS
CREATE TABLE reservations (
    reservation_id INT PRIMARY KEY IDENTITY(1,1), -- Identificador único de reserva
    room_number INT NOT NULL,                     -- Número de habitación reservada
    customer_name NVARCHAR(100) NOT NULL,         -- Nombre del cliente
    start_date DATE NOT NULL,                     -- Fecha de inicio de la reserva
    end_date DATE NOT NULL,                       -- Fecha de fin de la reserva
    status NVARCHAR(20) NOT NULL                  -- Estado de la reserva (e.g., activa, cancelada)
```

116 %

Messages

```
(2 rows affected)

Completion time: 2024-12-15T22:15:00.3944804-05:00
```

Inventory Microservice

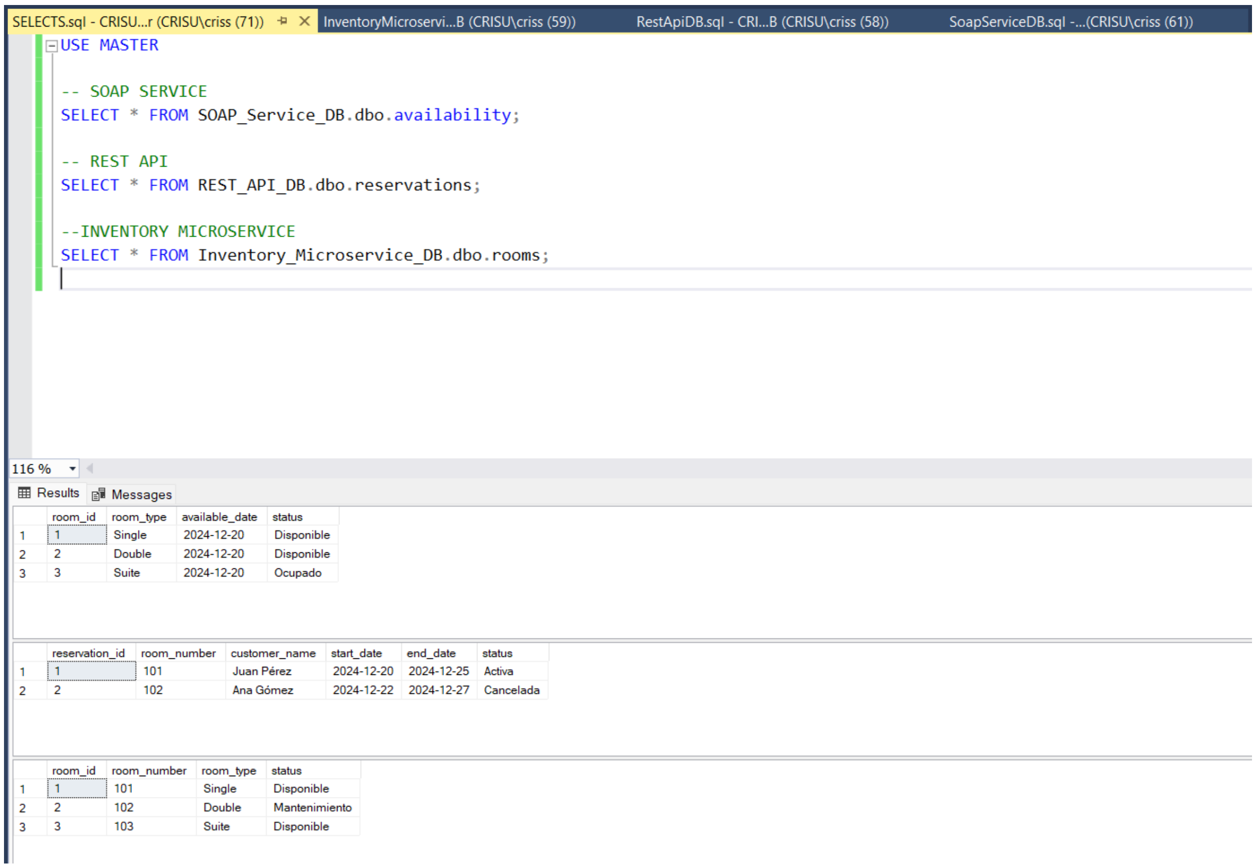

SELECTS.sql - CRISU...r (CRISU\criss (71)) InventoryMicroservi...B (CRISU\criss (59)) RestApiDB.sql - CRI...B (CRISU\criss (58)) SoapServiceDB.sql -...(CRISU\criss (61))

```
USE Inventory_Microservice_DB;
GO

--CREACIÓN DE TRABLA ROOMS
CREATE TABLE rooms (
    room_id INT PRIMARY KEY IDENTITY(1,1), -- Identificador único de la habitación
    room_number INT NOT NULL UNIQUE,       -- Número único de la habitación
    room_type NVARCHAR(50) NOT NULL,       -- Tipo de habitación (e.g., simple, doble)
    status NVARCHAR(20) NOT NULL           -- Estado de la habitación (e.g., disponible, mantenimiento)
);
GO

--INSERTS DATOS DE PRUEBA
INSERT INTO rooms (room_number, room_type, status)
VALUES
(101, 'Single', 'Disponible'),
(102, 'Double', 'Mantenimiento'),
(103, 'Suite', 'Disponible');
```

116 %

Messages

```
(3 rows affected)

Completion time: 2024-12-15T22:16:45.3222410-05:00
```



SELECTS.sql - CRISU...r (CRISU\criss (71)) | InventoryMicroservi...B (CRISU\criss (59)) | RestApiDB.sql - CRI...B (CRISU\criss (58)) | SoapServiceDB.sql -...(CRISU\criss (61))

```sql
USE MASTER

-- SOAP SERVICE
SELECT * FROM SOAP_Service_DB.dbo.availability;

-- REST API
SELECT * FROM REST_API_DB.dbo.reservations;

--INVENTORY MICROSERVICE
SELECT * FROM Inventory_Microservice_DB.dbo.rooms;
```

116 %

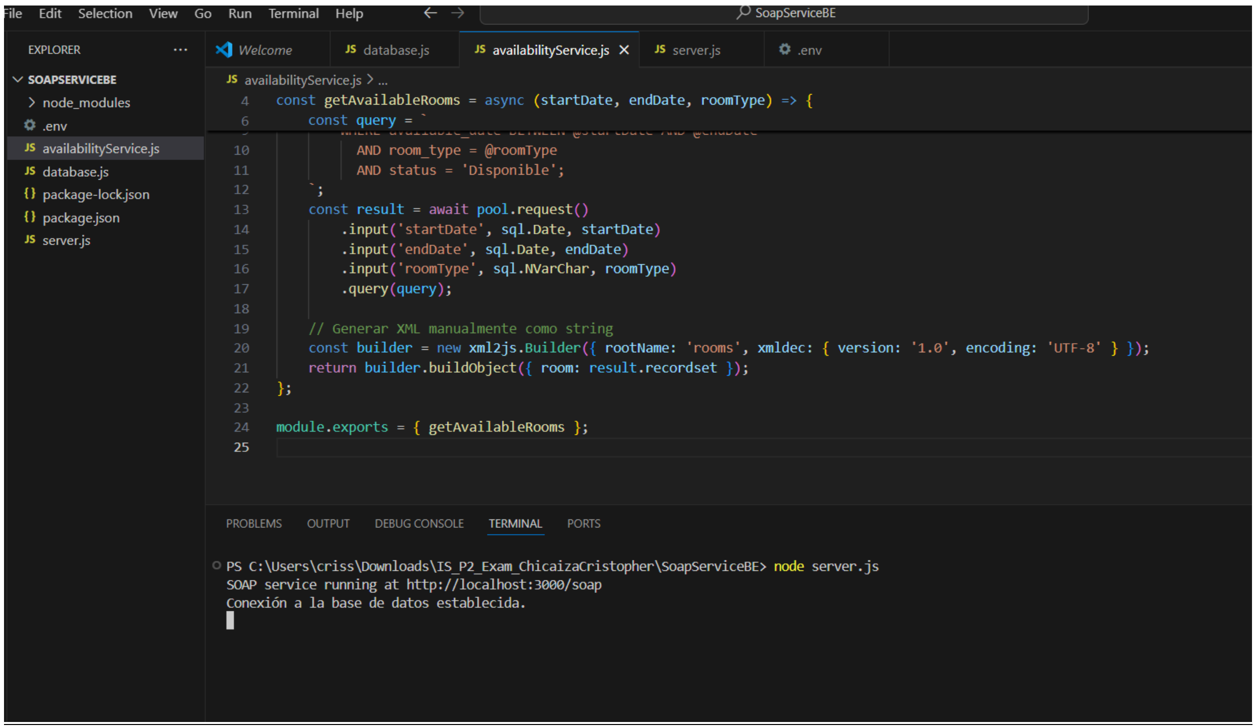
Results | Messages

	room_id	room_type	available_date	status
1	1	Single	2024-12-20	Disponible
2	2	Double	2024-12-20	Disponible
3	3	Suite	2024-12-20	Ocupado

	reservation_id	room_number	customer_name	start_date	end_date	status
1	1	101	Juan Pérez	2024-12-20	2024-12-25	Activa
2	2	102	Ana Gómez	2024-12-22	2024-12-27	Cancelada

	room_id	room_number	room_type	status
1	1	101	Single	Disponible
2	2	102	Double	Mantenimiento
3	3	103	Suite	Disponible

Soap Backend Service



```javascript
const getAvailableRooms = async (startDate, endDate, roomType) => {
    const query = `
            AND room_type = @roomType
            AND status = 'Disponible';
    `;
    const result = await pool.request()
        .input('startDate', sql.Date, startDate)
        .input('endDate', sql.Date, endDate)
        .input('roomType', sql.NVarChar, roomType)
        .query(query);

    // Generar XML manualmente como string
    const builder = new xml2js.Builder({ rootName: 'rooms', xmldec: { version: '1.0', encoding: 'UTF-8' } });
    return builder.buildObject({ room: result.recordset });
};

module.exports = { getAvailableRooms };
```

```
PS C:\Users\criss\Downloads\IS_P2_Exam_ChicaizaCristopher\SoapServiceBE> node server.js
SOAP service running at http://localhost:3000/soap
Conexión a la base de datos establecida.
```

Test
Datos:

URL: http://localhost:3000/soap

HEADER:

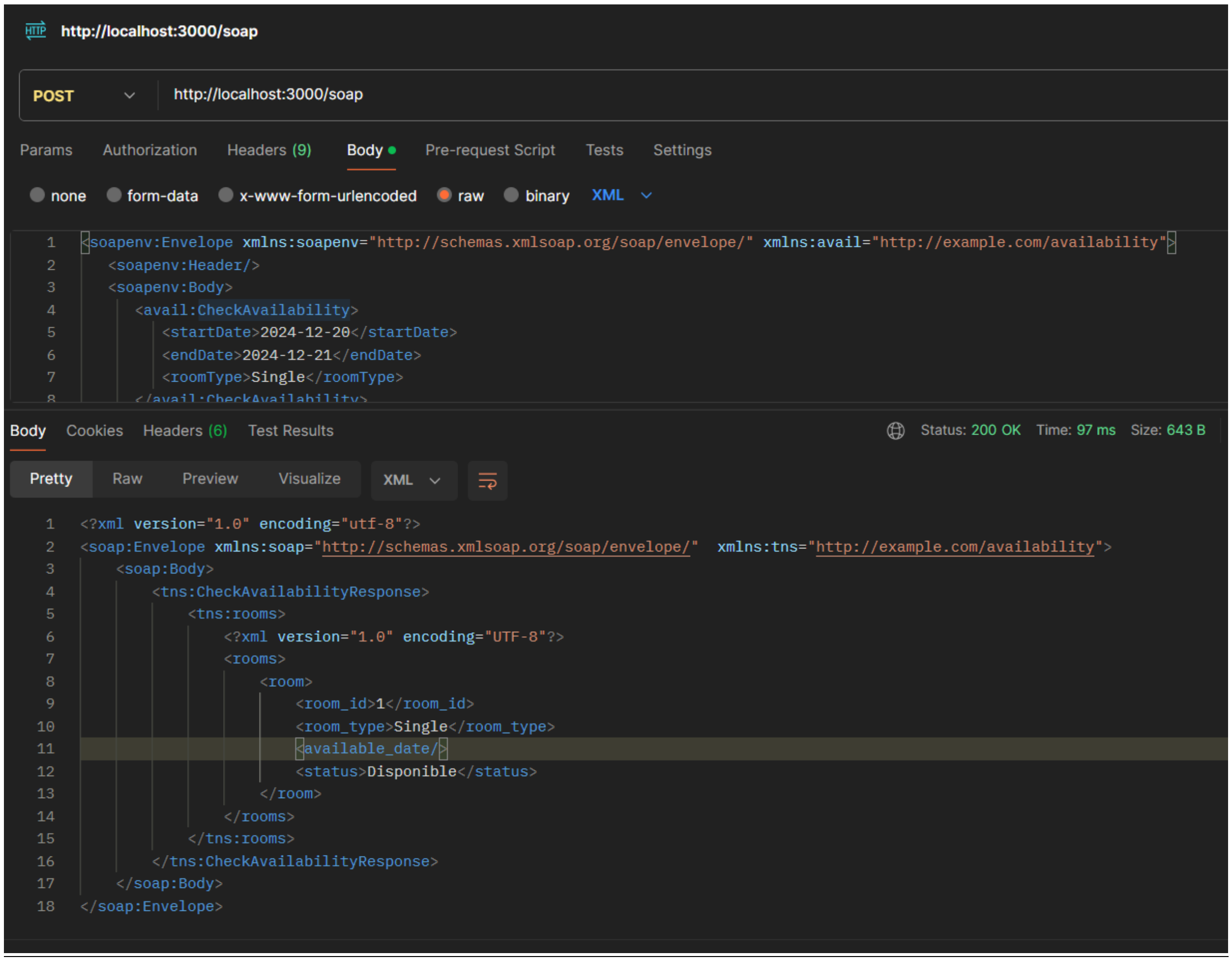
Key: Content-Type

Value: text/xml

Body XML:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:
avail="http://example.com/availability">

  <soapenv:Header/>

  <soapenv:Body>

    <avail:CheckAvailability>

      <startDate>2024-12-20</startDate>

      <endDate>2024-12-21</endDate>

      <roomType>Single</roomType>

    </avail:CheckAvailability>

  </soapenv:Body>

</soapenv:Envelope>
```

Resultado:


http://localhost:3000/soap
POST
Params Authorization Headers (9) Body Pre-request Script Tests Settings
none form-data x-www-form-urlencoded raw binary XML
Body Cookies Headers (6) Test Results
Pretty Raw Preview Visualize XML

POST http://localhost:3000/soap

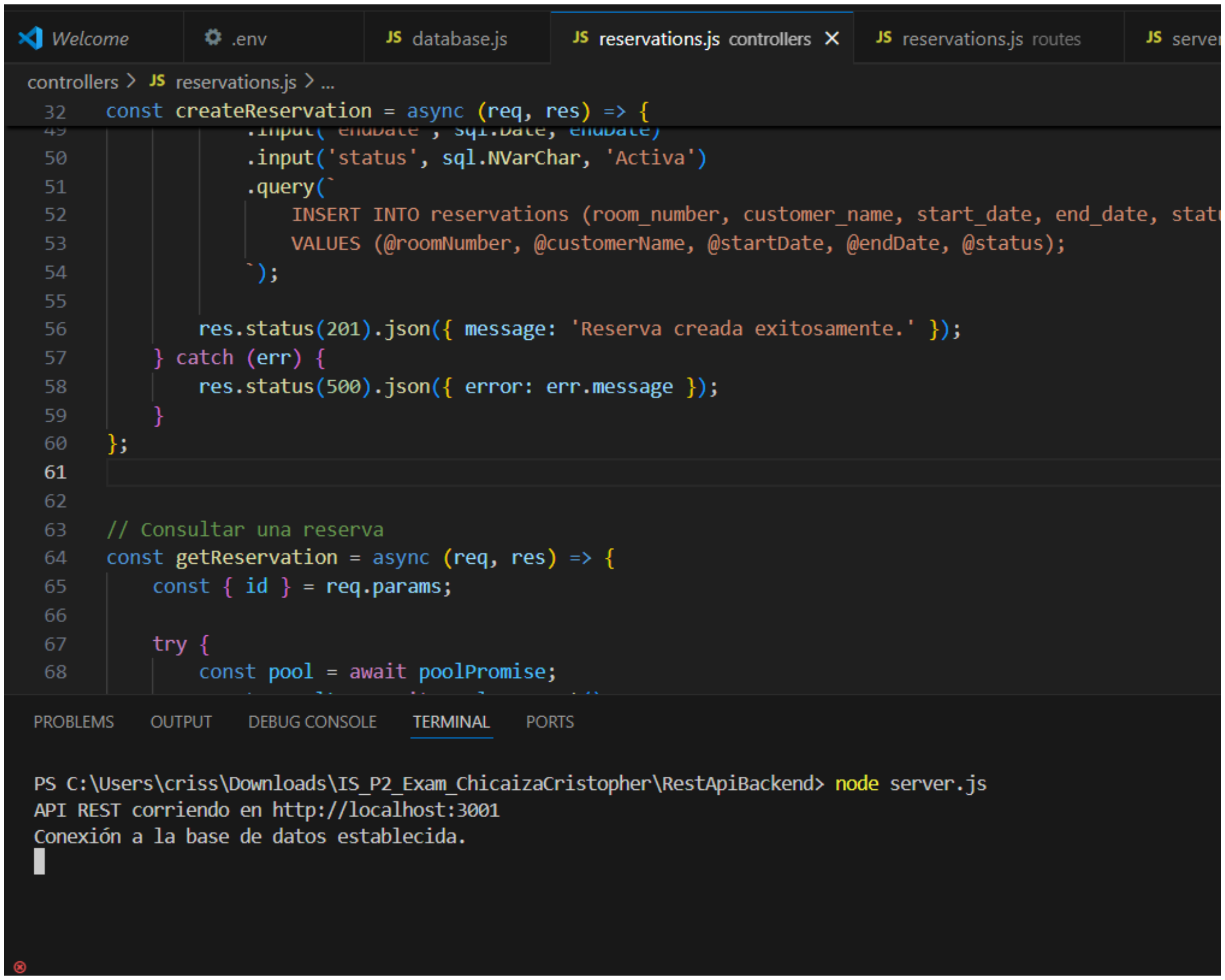
Body: raw, XML

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:avail="http://example.com/availability">
  <soapenv:Header/>
  <soapenv:Body>
    <avail:CheckAvailability>
      <startDate>2024-12-20</startDate>
      <endDate>2024-12-21</endDate>
      <roomType>Single</roomType>
    </avail:CheckAvailability>
```

Status: 200 OK Time: 97 ms Size: 643 B

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"  xmlns:tns="http://example.com/availability">
    <soap:Body>
        <tns:CheckAvailabilityResponse>
            <tns:rooms>
                <?xml version="1.0" encoding="UTF-8"?>
                <rooms>
                    <room>
                        <room_id>1</room_id>
                        <room_type>Single</room_type>
                        <available_date/>
                        <status>Disponible</status>
                    </room>
                </rooms>
            </tns:rooms>
        </tns:CheckAvailabilityResponse>
    </soap:Body>
</soap:Envelope>
```

Rest API Backend



```
const createReservation = async (req, res) => {
            .input('endDate', sql.Date, endDate)
            .input('status', sql.NVarChar, 'Activa')
            .query(`
                INSERT INTO reservations (room_number, customer_name, start_date, end_date, stat
                VALUES (@roomNumber, @customerName, @startDate, @endDate, @status);
            `);

        res.status(201).json({ message: 'Reserva creada exitosamente.' });
    } catch (err) {
        res.status(500).json({ error: err.message });
    }
};


// Consultar una reserva
const getReservation = async (req, res) => {
    const { id } = req.params;

    try {
        const pool = await poolPromise;
```

```
PS C:\Users\criss\Downloads\IS_P2_Exam_ChicaizaCristopher\RestApiBackend> node server.js
API REST corriendo en http://localhost:3001
Conexión a la base de datos establecida.
```

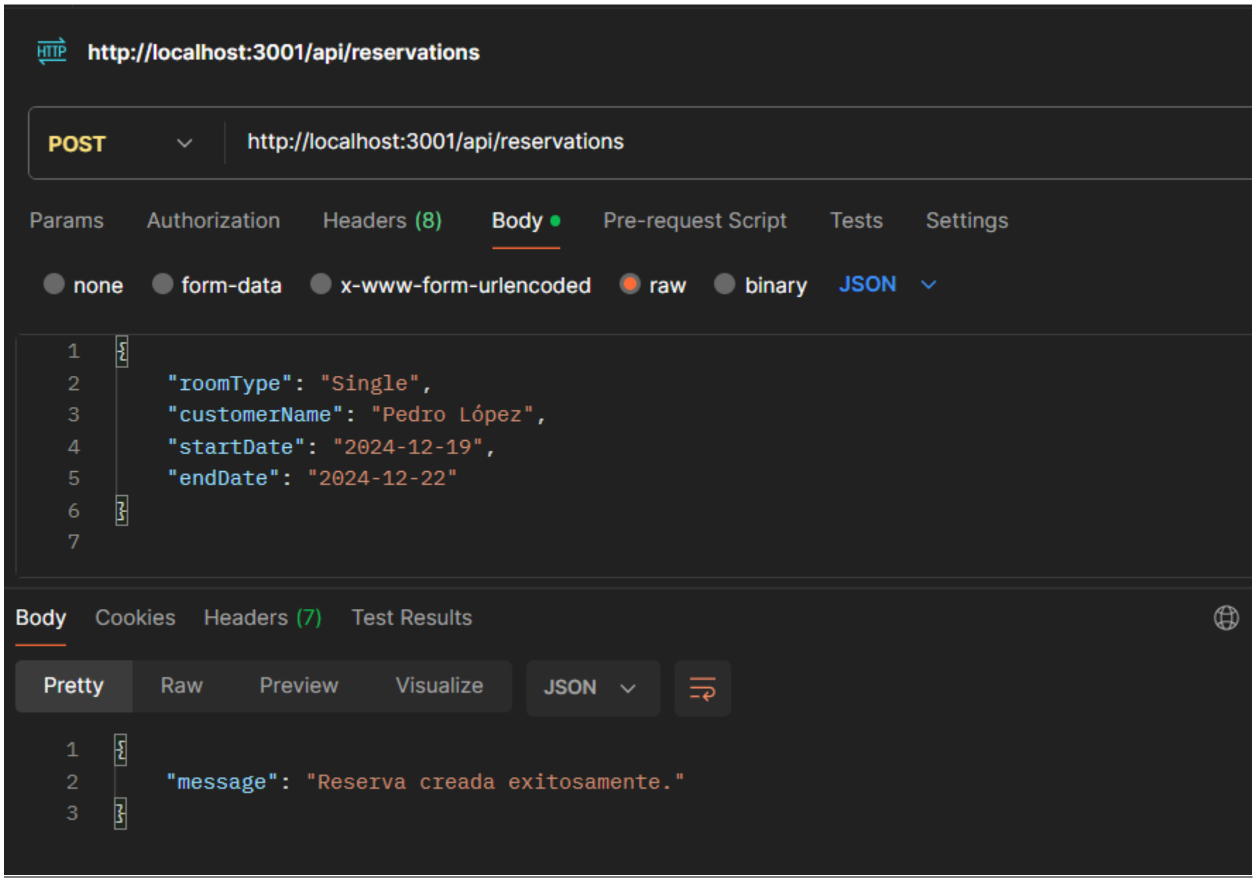
Pruebas:
POST /reservations

URL: http://localhost:3001/api/reservations

BODY:

```
{
  "roomType": "Single",
  "customerName": "Pedro López",
  "startDate": "2024-12-19",
  "endDate": "2024-12-22"
}
```

Resultados



http://localhost:3001/api/reservations

POST http://localhost:3001/api/reservations

```
{
    "roomType": "Single",
    "customerName": "Pedro López",
    "startDate": "2024-12-19",
    "endDate": "2024-12-22"
}
```

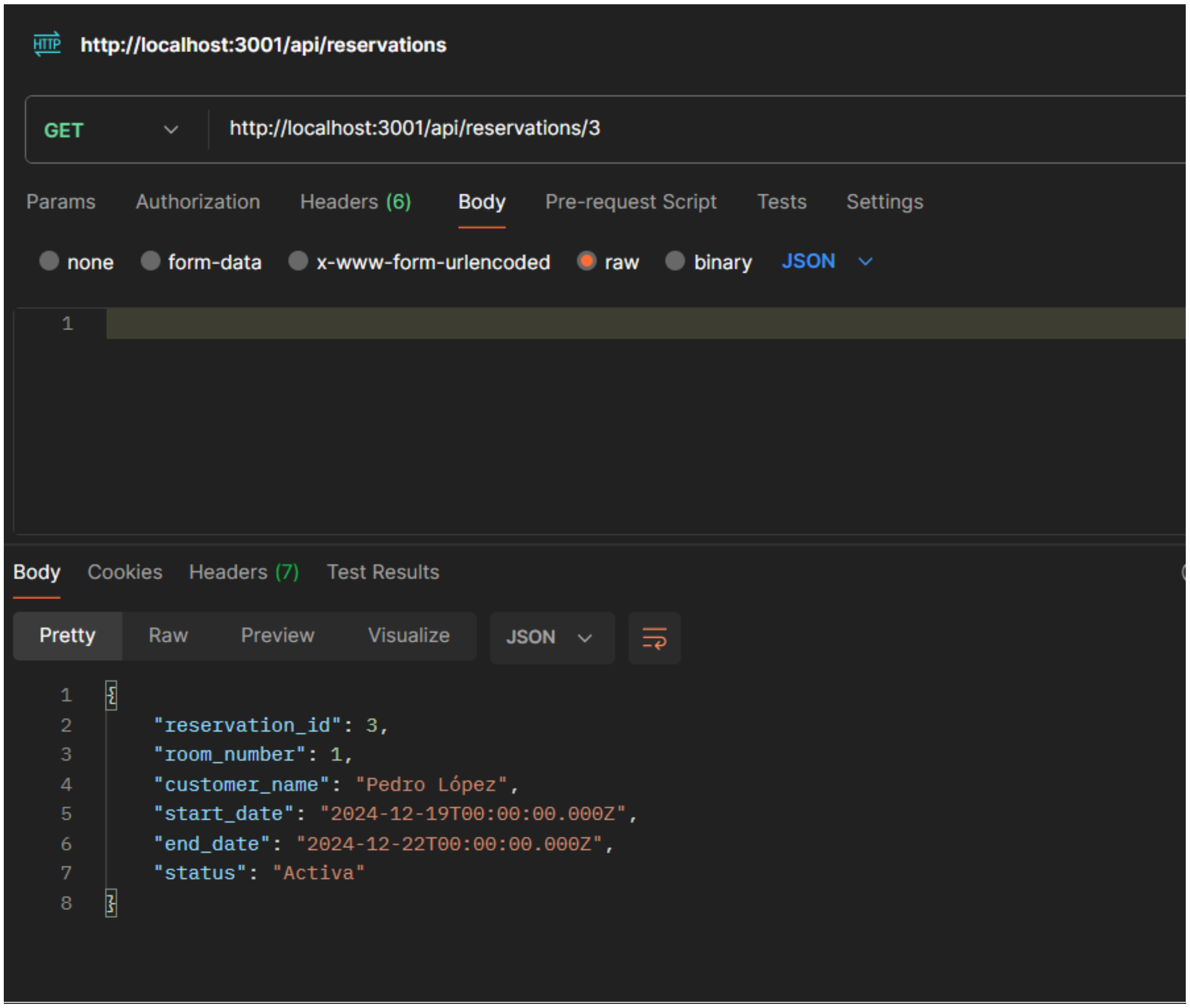
Response Body (Pretty, JSON):

```
{
    "message": "Reserva creada exitosamente."
}
```

GET /reservations/{id}

URL: http://localhost:3001/api/reservations/3

Resultados:


http://localhost:3001/api/reservations

GET http://localhost:3001/api/reservations/3

Params Authorization Headers (6) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded [✓ raw] binary JSON

Body Cookies Headers (7) Test Results

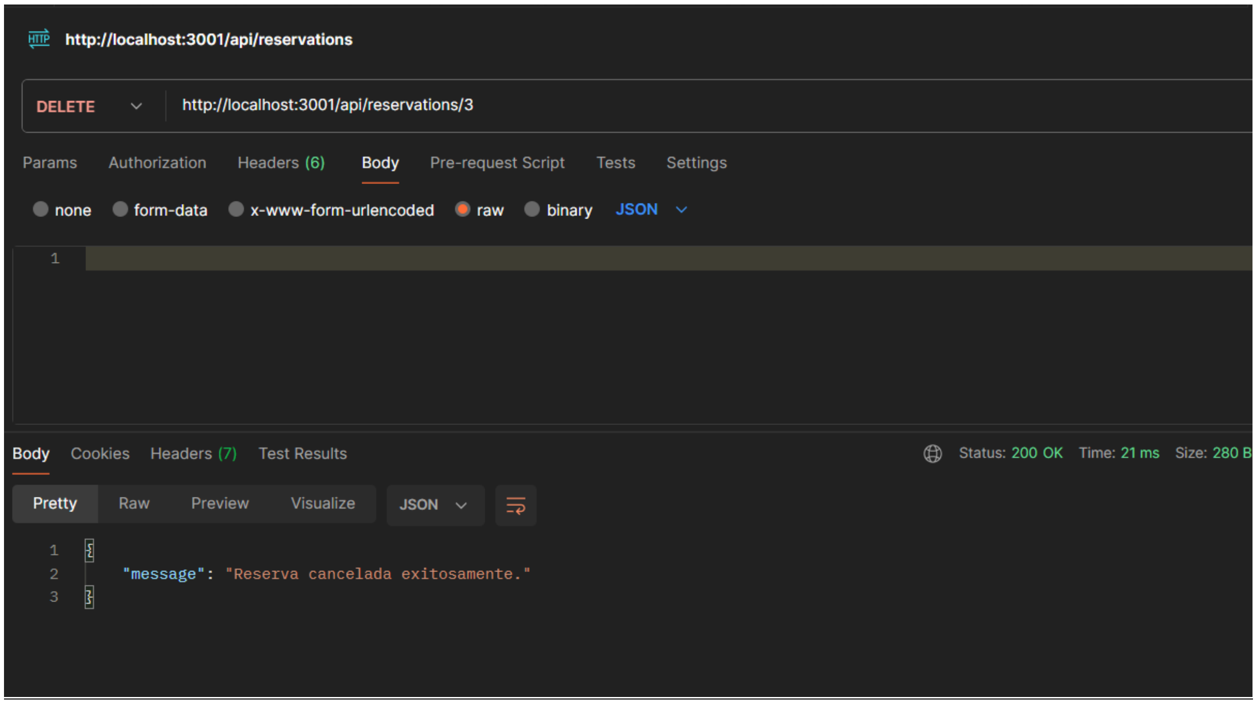
Pretty Raw Preview Visualize JSON

```
{
    "reservation_id": 3,
    "room_number": 1,
    "customer_name": "Pedro López",
    "start_date": "2024-12-19T00:00:00.000Z",
    "end_date": "2024-12-22T00:00:00.000Z",
    "status": "Activa"
}
```

DELETE /reservations/{id}

URL: http://localhost:3001/api/reservations/3

Resultados:

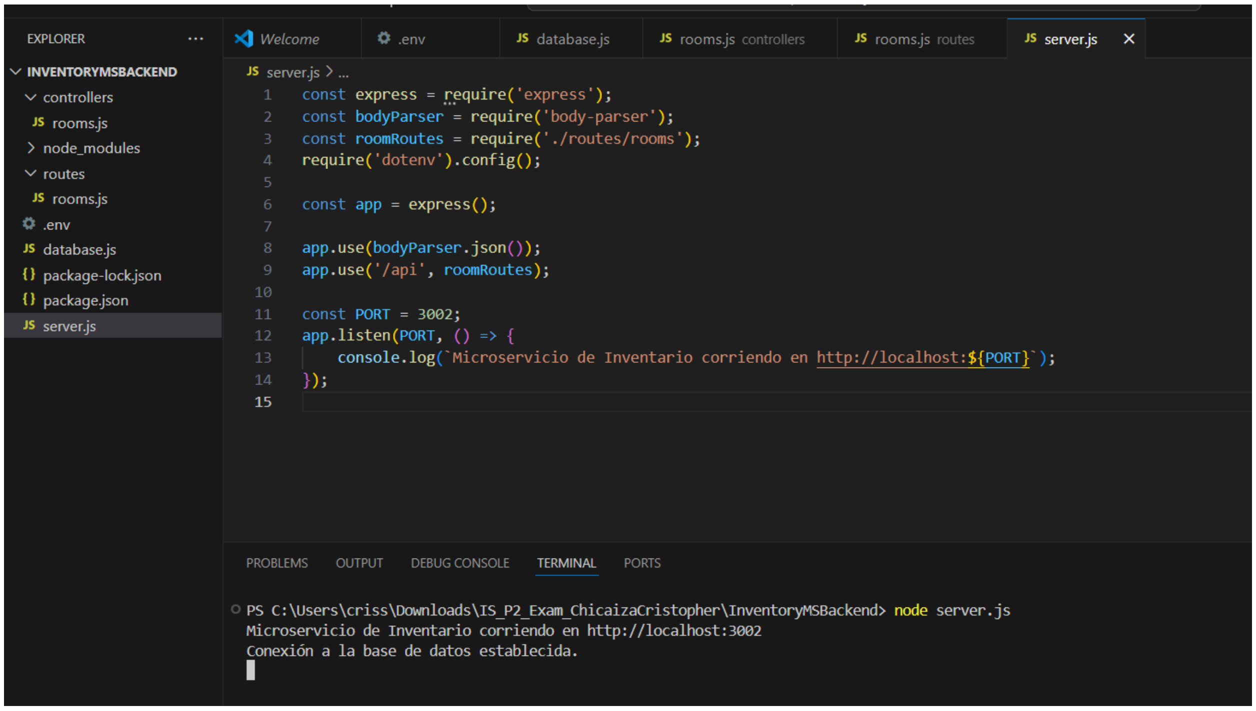


DELETE http://localhost:3001/api/reservations/3

Status: 200 OK Time: 21 ms Size: 280 B

```
{
    "message": "Reserva cancelada exitosamente."
}
```

Microservicio



```javascript
const express = require('express');
const bodyParser = require('body-parser');
const roomRoutes = require('./routes/rooms');
require('dotenv').config();

const app = express();

app.use(bodyParser.json());
app.use('/api', roomRoutes);

const PORT = 3002;
app.listen(PORT, () => {
    console.log(`Microservicio de Inventario corriendo en http://localhost:${PORT}`);
});
```

```
PS C:\Users\criss\Downloads\IS_P2_Exam_ChicaizaCristopher\InventoryMSBackend> node server.js
Microservicio de Inventario corriendo en http://localhost:3002
Conexión a la base de datos establecida.
```

Pruebas:

POST /rooms

URL: http://localhost:3002/api/rooms

Body:

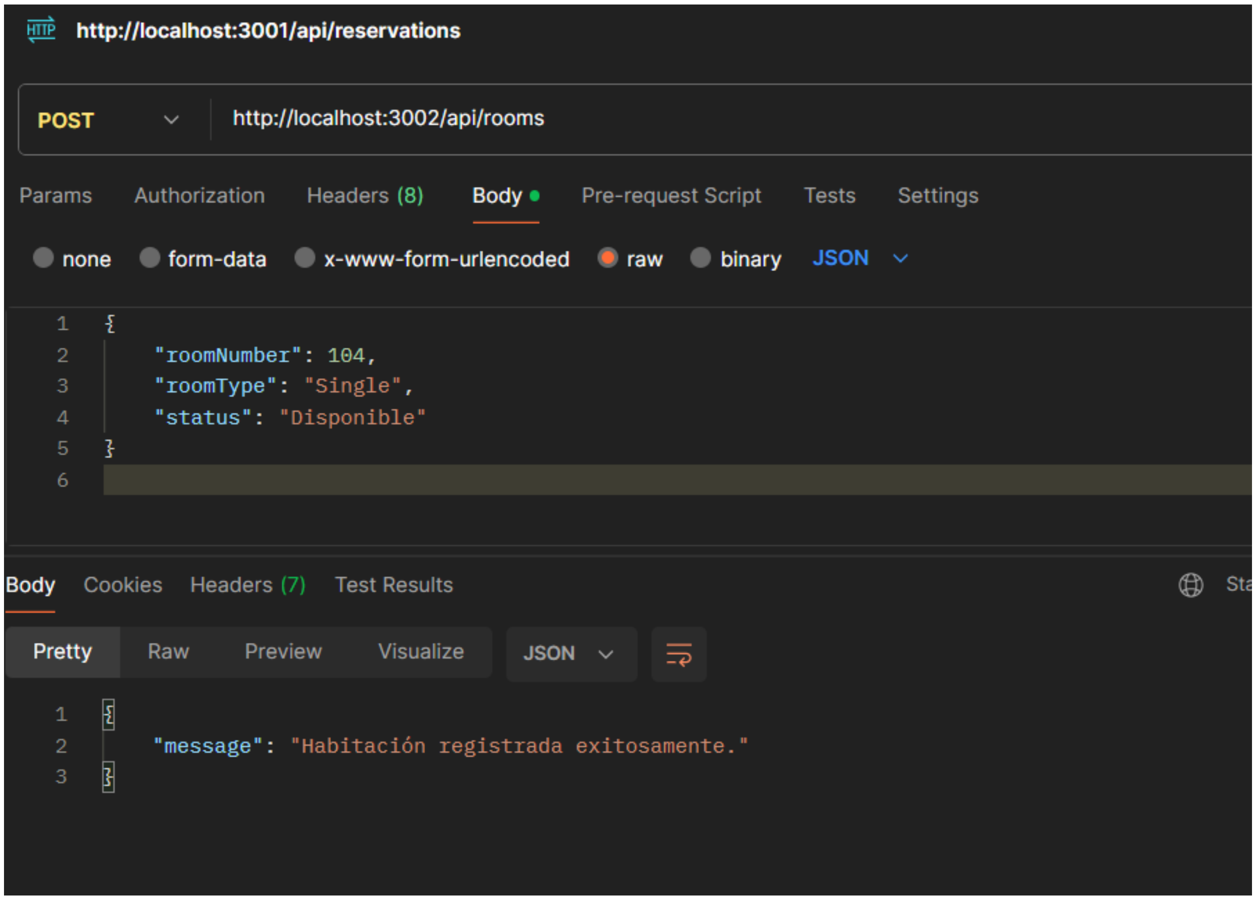
{

  "roomNumber": 104,

  "roomType": "Single",

  "status": "Disponible"

}



POST http://localhost:3002/api/rooms

```
{
    "roomNumber": 104,
    "roomType": "Single",
    "status": "Disponible"
}
```

Body — Pretty — JSON

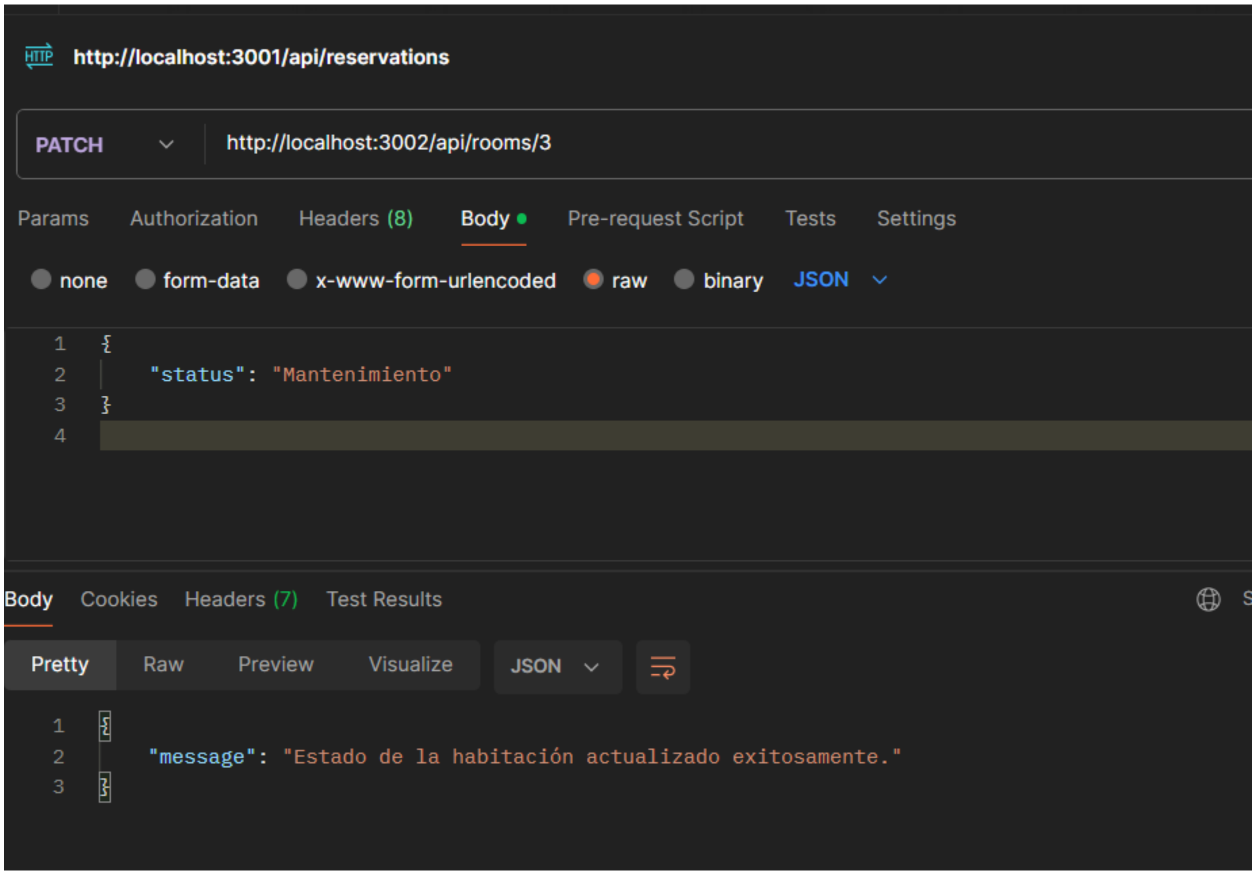
```
{
    "message": "Habitación registrada exitosamente."
}
```

PATCH /rooms/{id}

URL: http://localhost:3002/api/rooms/3

Body:

```
{
  "status": "Mantenimiento"
}
```


http://localhost:3001/api/reservations

PATCH http://localhost:3002/api/rooms/3

```
{
    "status": "Mantenimiento"
}
```

```
{
    "message": "Estado de la habitación actualizado exitosamente."
}
```