Examen: Concierto

Archivo: conexion (1).js

js

CopiarEditar

```
const { Sequelize } = require('sequelize');
const sequelize = new Sequelize({
   dialect: 'sqlite',
   storage: './boletos.sqlite'
});

module.exports = sequelize;
```

Explicación:

Este archivo configura la conexión a una base de datos SQLite utilizando Sequelize, un ORM (Object Relational Mapping) para Node.js.

- Se importa Sequelize desde el paquete correspondiente.
- Se crea una instancia de Sequelize configurada para usar SQLite como base de datos, almacenada localmente en el archivo boletos.sqlite.
- Finalmente, la conexión es exportada para ser reutilizada en otros módulos del proyecto.

Archivo: boletos.js

js

CopiarEditar

```
const { DataTypes } = require("sequelize");
const sequelize = require("../conexion");

const boletos = sequelize.define('boletos', {
   id: {
      type: DataTypes.INTEGER,
```

```
        primaryKey: true
    },
    localidad: {
        type: DataTypes.TEXT
    },
    fecha: {
        type: DataTypes.TEXT
    },
    precio: {
        type: DataTypes.DOUBLE
    },
    descuento: {
        type: DataTypes.DOUBLE
    }
}, {
    timestamps: false
});


module.exports = boletos;
```

Explicación:

Este archivo define el modelo de datos boletos, que representa una tabla en la base de datos.

- sequelize.define('boletos', {...}): Crea el modelo y define los campos.
- Campos definidos:
    - id: Identificador único del boleto.
    - localidad: Ubicación del asiento o sector.
    - fecha: Fecha del evento.
    - precio: Costo base del boleto.
    - descuento: Porcentaje de descuento a aplicar.

- timestamps: false: Indica que no se incluirán campos automáticos como createdAt y updatedAt.
- El modelo se exporta para ser utilizado en otras partes del proyecto.

Archivo: app (3).js

js

CopiarEditar

```
const express = require('express');
const boletos = require('./models/boletos');
const bodyParser = require('body-parser');
const app = express();
const puerto = 3000;

app.use(bodyParser.json());

// Iniciar servidor
app.listen(puerto, () => {
    console.log('servidor iniciado');
});

// Ruta POST para calcular total del boleto
app.post('/boletos', async(req, res) => {
    const { localidad, fecha, estudiante } = req.body;

    const data = await boletos.findOne({
        where: {
            localidad,
            fecha
        }
```

```
    });

    let { precio, descuento } = data;

    if (!estudiante) {
        descuento = 0;
    }

    const total = precio - (precio * descuento);

    res.send({
        localidad,
        fecha,
        precio,
        descuento,
        total
    });
});
```

Explicación:

Este archivo contiene la lógica del servidor web utilizando el framework Express.js.

- Se configuran los módulos necesarios (express, body-parser y el modelo boletos).
- Se define el puerto 3000 y se inicializa el servidor.
- Se crea una ruta POST /boletos que:
    - Recibe localidad, fecha y si el usuario es estudiante desde el cuerpo de la solicitud.
    - Busca un boleto que coincida con la localidad y fecha.
    - Calcula el descuento (si no es estudiante, se omite).

    - Calcula el total a pagar y envía una respuesta con toda la información.

funcionamiento del proyecto:

Este sistema es una API sencilla para calcular el precio de boletos con descuento (si aplica). Usa:

- Node.js como entorno de ejecución.
- Express para manejar peticiones HTTP.
- Sequelize como ORM para conectar con una base de datos SQLite.
- SQLite como motor de base de datos local.

Se espera que el cliente envíe una solicitud POST con los datos del boleto, y el sistema devuelve el precio con descuento (si es estudiante) y el total a pagar.