Trabajo Práctico N°2

Materia: Desarrollo de Sistemas

Curso: 6to AO

Integrante: Lucía Saint Martin

Profesor: Manuel Corales

Contenidos:

Introducción

En este trabajo práctico desarrollé una API REST para la gestión de un restaurante, cumpliendo con los requerimientos funcionales propuestos por la consigna. El proyecto fue realizado utilizando TypeScript, Express y SQLite, con Prisma ORM como herramienta de acceso a la base de datos. Se implementaron funcionalidades para manejar usuarios (administradores y clientes), platos, pedidos, mesas y reservaciones, con enfoque en la autenticación, autorización y manejo de errores estructurado. La arquitectura está organizada por capas (routers, controllers, services y middlewares) para asegurar un código mantenible y escalable.

Dependencias

El proyecto utiliza varias dependencias clave:

- **Express**: es el framework de Node.js que permite definir rutas, middlewares y manejar peticiones HTTP de manera organizada y eficiente.

- **jsonwebtoken (JWT)**: se emplea para generar y verificar tokens que permiten identificar y autorizar a los usuarios (clientes o administradores) en distintas rutas protegidas del sistema.

- **Prisma ORM**: es el ORM que facilita la comunicación con la base de datos SQLite. Permite definir los modelos, relaciones y realizar operaciones como consultas, inserciones o actualizaciones de forma sencilla y segura, evitando escribir SQL directamente.

Autenticación y Autorización

La autenticación se basa en JSON Web Tokens. Cuando un usuario inicia sesión (ya sea administrador o cliente), se le genera un token JWT firmado, que se devuelve en la respuesta. Este token debe ser incluido en los headers de futuras solicitudes a rutas protegidas como prueba de identidad. La autorización se realiza verificando que el rol del usuario (administrador o cliente) tenga los permisos adecuados para acceder a cada ruta. Si no tiene los permisos necesarios, se devuelve un error correspondiente. Este mecanismo garantiza que solo usuarios válidos y con los permisos adecuados puedan utilizar funcionalidades sensibles del sistema.

Manejo de Errores

Implemente un sistema de manejo de errores personalizados. Esto permite devolver respuestas específicas al cliente, además de facilitar el mantenimiento del código. Mi objetivo era evitar respuestas genéricas (500 Internal Server Error) para todos los fallos e identificar y categorizar errores esperados (como falta de autenticación, datos inválidos o recursos no encontrados).

La estructura que hice fue en una carpeta errors cree el archivo *BaseError.ts* donde esta la clase base BaseError (hereda de Error) con estos atributos:

- **statusCode** (código HTTP)
- **name**
- **message**

Además, creé subclases específicas para los tipos de error más comunes, en el mismo archivo.

Después en la carpeta middleware, cree el archivo *errorHandler.ts*, donde detecta si el error es una instancia de BaseError, y devuelve el statusCode, name y message apropiados. Si no lo es, devuelve un 500.

En el archivo *index.ts* luego de las rutas se registra el errorHandler al final: **app.use(errorHandler);**

Los códigos de errores que implemente son:

Código HTTP	Nombre de Clase	Descripción	Ejemplo
400	BadRequestError	datos requeridos no enviados	falta contraseña
401	UnauthorizedError	usuario no autenticado	login sin token válido
403	ForbiddenError	usuario autenticado pero sin permisos	rol incorrecto
404	NotFoundError	recurso no encontrado	admin inexistente
409	ConflictError	conflicto de datos	mail duplicado en base de datos
500	InternalServerError	error inesperado del servidor	catch

Endpoints y Métodos HTTP

Endpoints: Método HTTP - Ruta - Header - Body - HTTP Status - Respuesta
Métodos HTTP utilizados: GET - POST - PATCH - DELETE

Admin:

- ➢ **Obtener todos los administradores:**
  - **Método**: GET
  - **URL**: http://localhost:3000/admin/
  - **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
  - **Body**: No lleva body
  - **HTTP Status**:
    - 200 OK
    - 401 Unauthorized → "Token no proporcionado."
    - 403 Forbidden → "Token invalido o expirado." o "No tenes permisos suficientes."
    - 500 Internal Server Error
  - **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Admins obtenidos con éxito"
    - data: admins → todos los administradores de la base de datos.
  - **Respuesta Error**:
    - ok: false
    - mensaje: <mensaje de error>

- ➢ **Registrar administrador:**
  - **Método**: POST
  - **URL**: http://localhost:3000/admin/register
  - **Header**: No requiere autenticación
  - **Body JSON**:
    - email: <email_string>
    - password: <password_string>
  - **HTTP Status**:
    - 200 OK
    - 400 Bad Request → "No se ingresó la contraseña."
    - 409 Conflict → "El email ya está registrado."
    - 500 Internal Server Error
  - **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Admin registrado con éxito"

        - data: admin
            - adminId: number
            - email: string
            - password: string
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Login administrador:**
- **Método**: POST
- **URL**: http://localhost:3000/admin/login
- **Header**: No requiere autenticación
- **Body JSON**:
    - email: <email_string>
    - password: <password_string>
- **HTTP Status**:
    - 200 OK
    - 404 Not Found → “Credenciales incorrectas. Verifique el email y la contraseña.”
    - 500 Internal Server Error
- **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Login exitoso"
    - data: token → hash del token, para usar en los headers que necesitan acceso de usuario.
- **Respuesta Error**:
    - ok: false
    - mensaje: <mensaje de error>

➢ **Eliminar administrador:**
- **Método**: DELETE
- **URL**: http://localhost:3000/admin/:id
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:
    - 200 OK
    - 401 Unauthorized → “Token no proporcionado.”
    - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
    - 404 Not Found → “No se encontró ningún administrador con ID: ${id}”
    - 500 Internal Server Error

- **Respuesta 200 0K**:
  - ok: true
  - mensaje: "Admin eliminado con éxito"
  - data: deletedAdmin → datos del administrador eliminado.
- **Respuesta Error**:
  - ok: false
  - mensaje: <mensaje de error>

Cliente:

➢ **Obtener todos los clientes:**
- **Método**: GET
- **URL**: http://localhost:3000/client
- **Header**:
  - Key: Authorization
  - Value: Bearer <token_admin>
- **Body**: No tiene body
- **HTTP Status**:
  - 200 OK
  - 401 Unauthorized → "Token no proporcionado."
  - 403 Forbidden → "Token invalido o expirado." o "No tenes permisos suficientes."
  - 500 Internal Server Error
- **Respuesta 200 0K**:
  - ok: true
  - mensaje: "Clientes obtenidos con éxito"
  - data: clients → todos los clientes de la base de datos.
- **Respuesta Error**:
  - ok: false
  - mensaje: <mensaje de error>

➢ **Registrar cliente:**
- **Método**: POST
- **URL**: http://localhost:3000/client/register
- **Header**: No requiere autenticación
- **Body JSON**:
  - name: <name_string>
  - email: <email_string>
  - phone: <phone_number>
  - password: <password_string>
  - address: <address_string>
- **HTTP Status**:
  - 200 OK
  - 400 Bad Request → "No se ingresó la contraseña."

        - 409 Conflict → “El email ya está registrado.” o “El teléfono ya está registrado.”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Cliente registrado con éxito"
        - data: client → datos del cliente registrado.
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Login cliente:**
    - **Método**: POST
    - **URL**: http://localhost:3000/client/login
    - **Header**: No requiere autenticación
    - **Body JSON**:
        - email: <email_string>
        - password: <password_string>
    - **HTTP Status**:
        - 200 OK
        - 404 Not Found → “No se encontró ningún cliente con los datos ingresados”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Login exitoso"
        - data: token → hash del token
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Eliminar cliente:**
    - **Método**: DELETE
    - **URL**: http://localhost:3000/client/:id
    - **Header**:
        - Key: Authorization
        - Value: Bearer <token_admin/client>
    - **Body**: No lleva body
    - **HTTP Status:**
        - 200 OK
        - 401 Unauthorized → “Token no proporcionado.”
        - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
        - 404 Not Found → “No se encontró ningún cliente con ID: ${id}”

        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Cliente eliminado con éxito"
        - data: deletedClient → datos del cliente eliminado.
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

Platos:

- **Obtener todos los platos:**
    - **Método:** GET
    - **URL**: http://localhost:3000/menu/
    - **Header:** No requiere autenticación
    - **Body:** No lleva body
    - **HTTP Status posibles:**
        - 200 OK
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Menú obtenido con éxito"
        - data: menu. → lista con todos los platos.
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

- Crear Plato:
    - **Método**: POST
    - **URL**: http://localhost:3000/menu/createDish
    - **Header**:
        - Key: Authorization
        - Value: Bearer <token_admin>
    - **Body JSON**:
        - name: <name_string>
        - description: <description_string>
        - price: <price_number>
        - category: <category_string>
    - **HTTP Status**:
        - 200 OK
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Plato creado con éxito"

        - data: dish
            - dishId: number
            - name: string
            - description: string
            - price: number
            - category: string
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Cambiar precio del Plato:**

- **Método**: PATCH
- **URL**: http://localhost:3000/menu/changePrice/:id
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body JSON**:
    - price: <price_number>
- **HTTP Status**:
    - 200 OK
    - 404 Not Found → “No hay ningún plato con el ID: ${id}”
    - 500 Internal Server Error
- **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Precio cambiado con éxito"
    - data: changedDish
        - dishId: number
        - name: string
        - description: string
        - price: number
        - category: string
- **Respuesta Error**:
    - ok: false
    - mensaje: <mensaje de error>

➢ **Eliminar Plato:**

- **Método**: DELETE
- **URL**: http://localhost:3000/menu/:id
- **Header**:
    - Key: Authorization
    - **Value**: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:
    - 200 OK

        - 404 Not Found → “No hay ningún plato con el ID: ${id}”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Plato eliminado con éxito"
        - data: deletedDish
            - dishId: number
            - name: string
            - description: string
            - price: number
            - category: string
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

Pedido - Platos:

- ➢ **Obtener todos los registros de platos por pedido:**
    - **Método**: GET
    - **URL**: http://localhost:3000/orderDish/
    - **Header**:
        - Key: Authorization
        - Value: Bearer <token_admin>
    - **Body**: No lleva body
    - **HTTP** Status posibles:
        - 200 OK
        - 401 Unauthorized → “Token no proporcionado.”
        - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Datos obtenidos con éxito"
        - data: orderDish
            - orderId: number
            - dishId: number
            - qty: number
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

Pedido:

- ➢ **Obtener todos los pedidos:**

- **Método**: GET
- **URL**: http://localhost:3000/order/
- **Header**:
  - Key: Authorization
  - Value: Bearer <token_admin>
- **Body**: No tiene body
- **HTTP** Status posibles:
  - 200 OK
  - 401 Unauthorized → “Token no proporcionado.”
  - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
  - 500 Internal Server Error
- **Respuesta 200 0K**:
  - ok: true
  - mensaje: "Pedidos obtenidos con éxito"
  - data: orders → lista de pedidos.
- **Respuesta Error**:
  - ok: false
  - mensaje: <mensaje de error>

➢ **Ver estado de un pedido:**

- **Método**: GET
- **URL**: http://localhost:3000/order/seeStatus/:id
- **Header**:
  - Key: Authorization
  - Value: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:
  - 200 OK
  - 401 Unauthorized → “Token no proporcionado.”
  - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
  - 404 Not Found → “No se encontró ningun pedido con ID: ${id}”
  - 500 Internal Server Error
- **Respuesta 200 0K**:
  - ok: true
  - mensaje: "Estado obtenido con éxito"
  - data: status?.status → string
- **Respuesta Error**:
  - ok: false
  - mensaje: <mensaje de error>

➢ **Crear pedido:**

- **Método**: POST

- **URL**: http://localhost:3000/order/createOrder
- **Header**:
  - Key: Authorization
  - Value: Bearer <token_client>
- **Body JSON**:
  - clientId: <clientId_number>
  - dishes: <dishes_number[]>
- **HTTP Status**:
  - 200 OK
  - 400 Bad Request
  - 401 Unauthorized → “Token no proporcionado.”
  - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
  - 404 Not Found → “No se encontró ningún cliente con ID: ${id}” o “No hay ningún plato con el ID: ${id}”
  - 500 Internal Server Error
- **Respuesta 200 0K**:
  - ok: true
  - mensaje: "Pedido creado con éxito"
  - data: newOrder
    - orderId: number
    - clientId: number
    - totalAmount: number
    - discount: string (tipo de descuento)
    - finalAmount: number
    - status: string
    - deliveryAddress: string
- **Respuesta Error**:
  - ok: false
  - mensaje: <mensaje de error>

➢ **Eliminar pedido:**

- **Método**: DELETE
- **URL**: http://localhost:3000/order/:id
- **Header**:
  - Key: Authorization
  - Value: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:
  - 200 OK
  - 401 Unauthorized → “Token no proporcionado.”
  - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
  - 404 Not Found → “No se encontró ningun pedido con ID: ${id}”

        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Pedido eliminado con éxito"
        - data: deletedOrder
            - orderId: number
            - clientId: number
            - totalAmount: number
            - discount: string (tipo de descuento)
            - finalAmount: number
            - status: string
            - deliveryAddress: string
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Cambiar estado del Pedido:**

- **Método**: PATCH
- **URL**: http://localhost:3000/order/changeStatus
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body JSON**:
    - orderId:
    - status: <status_string>
- **HTTP Status**:
    - 200 OK
    - 400 Bad Request → “Estado incorrecto, debería ser: pendiente, en cocina o enviado.”
    - 404 Not Found → “No se encontró ningún pedido con ID: ${id}”
    - 500 Internal Server Error
- **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Estado cambiado con éxito"
    - data: changedOrder
        - orderId: number
        - clientId: number
        - totalAmount: number
        - discount: string (tipo de descuento)
        - finalAmount: number
        - status: string
        - deliveryAddress: string
- **Respuesta Error**:
    - ok: false

        - mensaje: <mensaje de error>

Reservación:

- ➢ **Obtener todas las reservas:**
    - **Método**: GET
    - **URL**: http://localhost:3000/reservation/
    - **Header**:
        - Key: Authorization
        - Value: Bearer <token_admin>
    - **Body**: No lleva body
    - **HTTP Status**:
        - 200 OK
        - 401 Unauthorized → “Token no proporcionado.”
        - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Reservaciones obtenidas con éxito"
        - data: reservations → lista de reservaciones.
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

- ➢ **Crear una reserva:**
    - **Método**: POST
    - **URL**: http://localhost:3000/reservation/createReservation
    - **Header**:
        - Key: Authorization
        - Value: Bearer <token_client>
    - **Body**: No lleva body
    - **HTTP Status**:
        - 200 OK
        - 400 Bad Request
        - 401 Unauthorized → “Token no proporcionado.”
        - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
        - 404 Not Found → “No se encontró la reserva a eliminar”
        - 409 Conflict → “La mesa ${id} ya esta reservada.”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Reservación creada con éxito"

        - data: reservation
            - reservationId: number
            - tableId: number
            - clientId: number
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Eliminar Reserva**

- **Método**: DELETE
- **URL**: URL: http://localhost:3000/reservation/:id
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:
    - 200 OK
    - 401 Unauthorized → “Token no proporcionado.”
    - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
    - 404 Not Found → “No se encontró la reserva a eliminar”
    - 409 Conflict → “La mesa ${id} ya esta disponible.”
    - 500 Internal Server Error
- **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Reserva eliminada con éxito"
    - data: deletedReservation
        - reservationId: number
        - tableId: number
        - clientId: number
- **Respuesta Error**:
    - ok: false
    - mensaje: <mensaje de error>

Mesas:

➢ **Obtener todas las mesas:**

- **Método**: GET
- **URL**: http://localhost:3000/table/
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:

        - 200 OK
        - 401 Unauthorized → “Token no proporcionado.”
        - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Mesas obtenidas con éxito"
        - data: tables → lista de las mesas
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Obtener Mesas disponibles:**
- **Método**: GET
- **URL**: http://localhost:3000/table/disponibility
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin/cliente>
- **Body**: No lleva body
- **HTTP Status**:
    - 200 OK
    - 401 Unauthorized → “Token no proporcionado.”
    - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
    - 500 Internal Server Error
- **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Mesas disponibles obtenidas con éxito"
    - data: tables → lista de las mesas disponibles
- **Respuesta Error**:
    - ok: false
    - mensaje: <mensaje de error>

➢ **Crear mesa:**
- **Método**: POST
- **URL**: http://localhost:3000/table/createTable
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body JSON**:
    - tableId: <tableId_number>
    - status: <status_string>
- **HTTP Status**:

        - 200 OK
        - 400 Bad Request → “No se pueden crear más de 15 mesas.”
        - 401 Unauthorized → “Token no proporcionado.”
        - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
        - 500 Internal Server Error
    - **Respuesta 200 0K**:
        - ok: true
        - mensaje: "Mesa creada con éxito"
        - data: table
            - tableId: number
            - status: string
    - **Respuesta Error**:
        - ok: false
        - mensaje: <mensaje de error>

➢ **Eliminar mesa:**

- **Método**: DELETE
- **URL**: http://localhost:3000/table/:id
- **Header**:
    - Key: Authorization
    - Value: Bearer <token_admin>
- **Body**: No lleva body
- **HTTP Status**:
    - 200 OK
    - 401 Unauthorized → “Token no proporcionado.”
    - 403 Forbidden → “Token invalido o expirado.” o “No tenes permisos suficientes.”
    - 404 Not Found → “No se encontró ninguna mesa con ID: ${id}`”
    - 500 Internal Server Error
- **Respuesta 200 0K**:
    - ok: true
    - mensaje: "Mesa eliminada con éxito"
    - data: deletedTable
        - tableId: number
        - status: string
- **Respuesta Error**:
    - ok: false
    - mensaje: <mensaje de error>