Trabajo Práctico N°2

Integrantes: Bautista Crocco, Nicolas Gonzalez

Estructura general de la API

- Autenticación: Token JWT (en cookies)
- Roles:
    - **ADMIN**: puede crear, eliminar mesas, cambiar estado de pedidos.
    - **CLIENTE**: puede ver menús, reservar mesa y hacer pedidos.
- La base de datos utilizada es **SQLite**, gestionada a través del ORM Prisma. El archivo generado tras las migraciones se encuentra en la raíz del proyecto bajo el nombre db.

Endpoints

Para probar todos los endpoints entrar al siguiente enlace de postman: https://web.postman.co/workspace/a78bd3f3-9618-40cc-8197-28f3ccfd38ec

A tener en cuenta: la gran mayoría de las validaciones están realizadas con zod, principalmente el *register* y el *login* del usuario que son realmente importantes.

Nota: los endpoints de *crear-mesa, crear-plato, crear-menu, crear-usuario-admin, crear-pedido, liberar-mesa, cambiar-estado-pedido, eliminar-mesa y eliminar-plato* utilizan un middleware para verificar el rol del usuario, en caso de de que no sea admin, lanza una excepción y no lo deja utilizar dicha ruta.

Nota: los endpoints eliminar-plato y eliminar-mesa no borran definitivamente el registro, sino que hacen un borrado lógico donde un campo indica si están activos o no.

Nota: a los endpoints *liberar-mesa, mostrar-menu-por-id, estado-pedido, eliminar-mesa, eliminar-plato y liberar-mesa* se les pasa como param la id. Salvo en *estado-pedido* que el ID es un string, en el resto es un number.

/usuario/register

```
{
   "nombre":  string,
   "correo":    string,
   "telefono": string,
   "direccion": string,
   "contraseña": string
}
```

Código exitoso: 201

Respuesta :

```
{
  "data": "Usuario creado exitosamente."
}
```

/usuario/login

```
{
   "correo": string,
   "contraseña": string
}
```

A la hora de hacer el login se verifica que la contraseña que se pasa sea igual a la contraseña que se encuentra en la base de datos. Sin embargo, la contraseña está hasheada para más seguridad, por lo que a través de un método de la librería bcrypt se comparan estas contraseñas para verificar el inicio de sesión.

Código exitoso: 200

Respuesta :

```
{
```

```
  "data": "Inicio de sesión correcto."
}
```

POST /usuario/logout

Ruta para cerrar sesión.

Código exitoso: 200

Respuesta :

```
{
  "message": "Sesión cerrada correctamente."
}
```

POST /mesa/crear-mesa

Ruta para crear una mesa. No recibe ningún parámetro, pero tiene la condición de solo tener 15 mesas, no importa si están activas o no. El máximo de mesas creadas es de 15.

Código exitoso: 201

Respuesta :

```
{
  "message": "Mesa creada correctamente.",
  "data": {
     "id": number,
     "estado": "DISPONIBLE",
     "activa": true
  }
}
```

GET /mesa/mesas-disponibles

Ruta para mostrar las mesas que no estén reservadas y activas.

Código exitoso: 200

Respuesta :

```
{
  "mesas_disponibles": [
     {
        "id": number,
        "estado": "DISPONIBLE"
     }
 ]
}
```

DELETE /mesa/eliminar-mesa

Ruta para eliminar una mesa por su ID.

Código exitoso: 200

Respuesta :

```
{
  "message": "Mesa eliminada correctamente.",
  "data": {
     "id": number,
     "estado": "DISPONIBLE",
     "activa": false
  }
}
```

PATCH /mesa/liberar-mesa

Ruta para liberar una mesa que esté reservada.

Código exitoso: 200

Respuesta :

```
{
  "message": "Mesa liberada correctamente.",
  "data": {
     "id": number,
     "estado": "DISPONIBLE",
```

```
    "activa": true
  }
}
```

PATCH /mesa/reservar-mesa

Ruta para reservar una mesa por su ID.

Además de verificar que no esté reservada, también se verifica que esté activa.

Código exitoso: 200

Respuesta :

```
{
  "message": "Mesa reservada correctamente.",
  "data": {
    "id": number,
    "estado": "RESERVADA"
  }
}
```

POST /plato/crear-plato

Ruta para crear un plato.

```
{
  "nombre": string,
  "descripcion": string,
  "precio": string (aunque después se usa como Decimal),
  "categoría": Enum(“ENTRADA”, “PRINCIPAL”, “POSTRE”),
  "menuId": number
}
```

Código exitoso: 201

Respuesta :

```
{
  "message": "Plato creado correctamente.",
  "data": {
```

```
    "id": number,
    "nombre": string,
    "descripcion": string,
    "precio": string(decimal),
    "categoria": Enum,
    "activo": true,
    "menuId": number
  }
}
```

DELETE /plato/eliminar-plato

Ruta para eliminar un plato por su ID.

Código exitoso: 200
Respuesta :

```
{
  "message": "Plato eliminado correctamente.",
  "data": {
    "id": number,
    "nombre": string,
    "descripcion": string,
    "precio": string(decimal),
    "categoria": Enum,
    "activo": false,
    "menuId": number
  }
}
```

POST /menu/crear-menu

Ruta para crear un menu.

```
{
   "nombre": string
}
```

Código exitoso: 201

Respuesta :

```
{
  "message": "Menu creado correctamente.",
  "data": {
    "id": number,
    "nombre": string
  }
}
```

GET /menu/mostrar-menu-por-id

Ruta que muestra los platos de un menú por su ID.

Código exitoso: 200

Respuesta :

```
{
  "message": "Menu devuelto exitosamente.",
  "menus": {
    "id": 1,
    "nombre": "Daily",
    "platos": [
      {
        "id": 1,
        "nombre": string,
        "descripcion": string,
        "precio": string(decimal),
        "categoria": Enum,
        "activo": true,
        "menuId": number
      }
    ]
  }
}
```

GET /menu/listar-menus

Ruta que muestra los menus disponibles.

Código exitoso: 200

Respuesta :

```
{
  "menus": [
    {
      "id": number,
      "nombre": string
    }
  ]
}
```

POST /pedido/crear-pedido

Ruta para crear un pedido, se pasa una lista con los platos a pedir y su cantidad.

```
{
   "platos": [
     {
       "platoId": 2,
       "cantidad": 8
     }
   ]
}
```

Código exitoso: 201

Respuesta :

```
{
  "message": "Pedido creado correctamente.",
  "data": {
    "id": string,
    "usuarioId": string,
    "estado": Enum(“PENDIENTE”, “COCINA”, “ENTREGADO”),
    "monto": decimal,
    "descuento": number,
    "lugar_de_entrega": string
```

```
  }
}
```

GET /pedido/estado-pedido

Ruta para obtener el estado actual del pedido.

Código exitoso: 200

Respuesta :

```
{
  "estado": "PENDIENTE"
}
```

PATCH /pedido/cambiar-estado-pedido

Ruta para cambiar el estado del pedido.

```
{
   "pedidoId": string,
   "estado": Enum(“PENDIENTE”, “COCINA”, “ENTREGADO”)
}
```

Código exitoso: 200

Respuesta :

```
{
  "message": "Estado cambiado correctamente.",
  "nuevo_estado": Enum(“PENDIENTE”, “COCINA”, “ENTREGADO”)
}
```

POST /usuario/crear-usuario-admin

Funciona igual que el registro de un usuario común, solo que en vez de tener rol CLIENTE, va a tener rol admin.

Códigos utilizados

200: OK

201: Recurso creado

400: Error en la solicitud

401: No autenticado

403: No autorizado

404: Recurso no encontrado.

Observaciones

- Las contraseñas están encriptadas con bcrypt.
- El token JWT se almacena en cookies por seguridad.