TP 2 Desarrollo de Sistemas: “Restaurante Lo de Migue”

Alumnos: Santino Nicolas ANDREATTA & Kiara Micaela KOO

Requisitos

Cliente

- Los clientes deben poder registrarse, hacer login e interactuar con el sistema a través de un método de autenticación que dé cuenta de su identidad.
- Tendrán la posibilidad de reservar una mesa en el restaurante o efectuar un pedido.
- Debe existir la posibilidad de consultar el menú del restaurante
- Los clientes deberán poder consultar la disponibilidad de las mesas
    - podrán reservar tan solo una que se encuentre disponible.
- Los clientes tienen la posibilidad de seleccionar varios platos del menú.
- Tanto el administrador como el cliente podrán consultar el estado del pedido.

Admin

- El administrador podrá eliminar o agregar platos al menú
- El administrador tendrá la posibilidad de disponer una mesa que estaba previamente reservada.
- Tanto el administrador como el cliente podrán consultar el estado del pedido.
- El administrador manejara las transiciones entre los estados del pedido
- El administrador podrá ser autenticado por el sistema mediante algún método de identificación.

Otras cosas

- Hay 15 mesas en total
- Una vez que el cliente solicita un pedido, este se pondrá en estado “pendiente”, luego pasará a “en cocina” y por último pasará al estado “enviado”.
- Se calculará el monto total del @s a partir de la suma de los distintos platos
- Habrá un sistema de descuentos en función a la cantidad de pedidos (descuento aplicado al monto total)
    - 0-3 = no descuento
    - +3 = “Habitue” (%10)
    - +5 = “Premium” (%20)
    - +7 = “TopPremium” (%50)

Middleware

Logged In

Chequea si el usuario está logueado.

Códigos de estado

- 401: “No inició sesión”

Logged Out

Chequear que no esté logueado

Códigos de estado

- 401: “Ya inició sesión”

Admin

Chequea si es admin (requiere estar logueado):

Códigos de estado

- 401: "El usuario no es administrador"

End points

/menu

GET

Devuelve los platos.

Códigos de estado

- 200: { platos }
- 500: “Error al obtener platos”

/register

POST

Permite registrarse a un usuario

Estructura del body

```
{
  "nombre": string,
  "correo": string,
  "teléfono": string,
  "contraseña": string,
  "dirección": string
}
```

ejemplo:

```
{
  "nombre": "Juan",
  "correo": "juan@juan",
  "teléfono": "20",
  "contraseña": "123",
  "dirección": "hola"
}
```

Códigos de estado

- 201: { usuario }
- 500: “Error al registrar usuario”

/login

POST

Login del usuario, requiere no estar logueado

Estructura del body

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

ejemplo:

```
{
    "nombre": “Juan”,
    "contraseña": “123”
}
```

Códigos de estado

- 200: “Login exitoso”
- 400: “Nombre y contraseña son requeridos”
- 401: “Contraseña incorrecta”
- 404: “'Usuario no encontrado”
- 500: “Error al encontrar usuario”
- 500: “Error en el login:'

/logout

GET

Desloguea al usuario

Códigos de estado

- 200: “Logout exitoso”
- 500: “Error en cerrar sesión”

/mesas_disponibles

GET

Devuelve las mesas disponibles

Códigos de estado

- 200: {mesas}
- 500: "Error al obtener mesas"

/mis-reservas

GET

Devuelve las reservas del usuario logueado

Códigos de estado

- 200: { reservas }
- 500: "Error al obtener reservas del usuario"

POST

Crea una reserva

Estructura del body

```
{
  "numeroMesa": int
}
```

ejemplo:

```
{
   "numeroMesa": 3
}
```

Códigos de estado

- 201: { reserva }
- 400: "Falta el numero de la mesa"
- 400: "La mesa no está disponible"
- 500: "Error al crear reserva"

ps. El error 400: “La mesa no está disponible” sólo aparece si el usuario logueado es un cliente. En caso de ser un administrador, este error no aparece y se crea la reserva sin problema alguno.

/mis-reservas/:id

DELETE

Elimina/Cancela la reserva del usuario

Códigos de estado

- 200: "Reserva eliminada correctamente"
- 400: "ID inválido"
- 403: "No tenés permiso para eliminar esta reserva"
- 404: "Reserva no encontrada"
- 500: "Error al eliminar reserva"

/mis_pedidos

GET

Devuelve los pedidos del usuario logueado

Códigos de estado

- 200: { pedidos }
- 400: "ID de usuario inválido"
- 500: "Error al obtener pedidos del usuario"

POST

Crea un pedido

Estructura del body

```
{
  "platos": [
    { "platoId": int, "cantidad": int },
    { "platoId": int, "cantidad": int }
  ]
}
```

ejemplo:

```
{
  "platos": [
    { "platoId": 1 , "cantidad": 2 },
    { "platoId": 3 , "cantidad": 4 }
  ]
}
```

Códigos de estado

- 201: { pedido }
- 400: "Datos incompletos o inválidos"
- 500: "Error al crear pedido"

/mis_pedidos/estados

GET

Devuelve los estados de los pedidos del usuario logueado

Códigos de estado

- 200: { estados_pedidos }
- 400: "ID de usuario inválido"
- 500: "Error al obtener estados de pedidos del usuario"

Admin exclusive

/usuarios

GET

Devuelve los usuarios

Códigos de estado

- 200: { usuarios }
- 500: "Error al obtener usuarios"

/usuarios/:id

DELETE

Elimina a un usuario

Códigos de estado

- 200: “Usuario eliminado correctamente”
- 400: "ID inválido"
- 404: "Usuario no encontrado"
- 500: "Error al eliminar usuario"

/mesas

GET

Devuelve todas las mesas (disponibles y no disponibles)

Códigos de estado

- 200: { mesas }
- 500: "Error al obtener mesas"

POST

Crea una mesa

Estructura del body

```
{
  “numero”: int
}
```

ejemplo:

```
{
  "numero": 6
}
```

Códigos de estado

- 201: { mesa }
- 400: "Datos inválidos"
- 500: "Error al crear mesa"

/mesas/:id

DELETE

Elimina una mesa

Códigos de estado

- 200: “Mesa eliminada correctamente”
- 400: "ID inválido"
- 404: "Mesa no encontrada"
- 500: "Error al eliminar mesa"

/platos

POST

Crea un plato

Estructura del body

```
{
  "nombre": string,
  "descripción": string,
  "precio": integer,
  "categoría": "ENTRADA" | "PRINCIPAL" | "POSTRE"
}
```

ejemplo:

```
{
    "nombre": “Tiramisu”,
    "descripción": “postre italiano”,
    "precio": 3000,
    "categoría": “POSTRE”
}
```

Códigos de estado

- 201: { plato }
- 400: "Datos inválidos"
- 500: "Error al crear plato"

/platos/:id

DELETE

Elimina un plato

Códigos de estado

- 200: “Plato eliminado correctamente”
- 400: "ID inválido"
- 404: "Plato no encontrado"
- 500: "Error al eliminar plato"

/reservas

GET

Devuelve todas las reservas de todos los usuarios

Códigos de estado

- 200: { reservas }
- 500: "Error al obtener reservas"

/reservas/:id

DELETE

Elimina una reserva

Códigos de estado

- 200: "Reserva eliminada correctamente"
- 400: "ID inválido"
- 404: "Reserva no encontrada"
- 500: "Error al eliminar reserva"

/pedidos

GET

Devuelve todos los pedidos de todos los usuarios

Códigos de estado

- 200: { pedidos }
- 500: "Error al obtener pedidos"

/pedidos/:id

PATCH

Actualiza el estado de un pedido

Códigos de estado

- 200: { pedido }
- 404: "Pedido no encontrado"
- 400: "El pedido ya fue entregado"
- 500: "Error al actualizar estado del pedido"

DELETE

Elimina un pedido

Códigos de estado

- 200: "Pedido eliminado correctamente"
- 400: "ID inválido"
- 404: "Pedido no encontrado"
- 500: "Error al eliminar pedido"