API Hotel Basilio

Base de datos en MySQL con la siguiente linea de config
`'mysql+pymysql://root@localhost/hotel_basilio'`

Endpoints

General

REGISTRO

RUTA:
```
/registro
```

MÉTODO:
```
methods=["POST"]
```

Recibe un json con los siguientes datos: 'nombre', 'pass', 'pass2', 'rol' (este último, debe ser Empleado o Cliente, siendo la primera letra mayúscula). Luego de una serie de validaciones y en caso de que esté todo correcto, registra al usuario en la base de datos.

RETORNA:
```
{'status':400, "message":"El usuario ya existe en la base de datos."}
```
Si el usuario ya está registrado.

```
{"status":200, 'message':'usuario creado'}
```
Si el usuario se creó correctamente.

```
{'status':400, "message":"El usuario contiene carácteres inválidos."}
```
Si el usuario contiene carácteres inválidos

```
{"status":400,"message":"El rol debe ser Empleado o Cliente"}
```
Si el rol es distinto de Empleado o Cliente

```
{'status':400, "message":"Las contraseñas no coinciden."}
```
Si pass y pass2 no coinciden.

LOGIN

RUTA:
```
/login
```

MÉTODO:

```
methods=['POST']
```

Recibe un json con los datos: ‘usuario’, ‘pass’ . Compara los datos ingresados con los datos guardados bajo el nombre de usuario indicado (en caso de existir), si coincide se genera un token de acceso conteniendo los datos ‘usuario’, ‘rol’, ‘id’

RETORNA:

```
{'status':400, 'message':'Hubo un error con los datos, inténtelo nuevamente.'}
```

Si el usuario no existe, o si los datos no coinciden.

```
{'status':200, 'token':token}
```

Si se logueó el usuario con éxito.
Devuelve un token de acceso.

Clientes

- *Todos los campos son obligatorios*
- *Las fechas deben ingresarse en el siguiente formato de String: “yyyy mm dd”*
- *Para estos endpoints, es necesario el header Auth , y que el mismo contenga el token recibido en el login.*

RESERVAR HABITACIÓN

RUTA:

```
/api/cliente/habitacion/reservar
```

MÉTODO:

```
methods=['POST']
```

Recibe un json con los datos: ‘f_inicio’, ‘f_fin’, ‘nro’

f_inicio la fecha de inicio de la reserva (si el usuario desea reservar un rango de fechas) o la única fecha, si el usuario desea reservar un solo día la habitación.
f_fin es la fecha final del rango de fechas que desea reservar el usuario. En caso de querer reservar un solo día, este campo debe dejarse vacío: “”
nro el número de la habitación que desea reservar el usuario.

Verifica que la habitación exista y que esté disponible en la fecha o fechas deseadas. De ser así, se crea la reserva.

RETORNA:

```
{"error":"Esta área es solo para clientes. Por favor, logueese como cliente."}
```
Si en el token está indicado que el rol logueado es Empleado.

```
{"status":400, "message":"La habitacion no existe, o se encuentra inactiva actualmente."}
```
Si no se encontró la habitación, o la misma estaba en estado inactivo.

```
{'status':400,
'message':'habitacion no disponible en las fechas deseadas'}
```
Si la habitación no está disponible en las fechas deseadas.

```
{'status':200,
'message':'habitacion reservada'}
```
Si se realizó la reserva con éxito.

HABITACIONES DISPONIBLES SEGÚN FECHA/FECHAS

RUTA:
```
/api/cliente/habitacion/fecha/disponibles
```

MÉTODO
```
methods=['GET']
```

Recibe un json con los datos: ‘fecha_inicio’, ‘fecha_fin’
‘fecha_inicio’ la fecha de inicio de la búsqueda (si el usuario desea buscar habitaciones disponibles en un rango de fechas) o la única fecha, si el usuario desea buscar habitaciones disponibles por un único día. No dejar vacío este campo.
‘fecha_fin’ es la fecha final del rango de fechas que el usuario indicó para realizar la búsqueda. En caso de querer buscar habitaciones disponibles un solo día, este campo debe dejarse vacío: “”
Hace una búsqueda de todas las habitaciones cuyas fechas de reserva (en caso de tenerlas) no coincidan con las ingresadas por el usuario, y devuelve las habitaciones disponibles, si las hay.

RETORNA:
Json con los datos de las habitaciones disponibles.
Si las hay.

```
{"status":200,
"message":"No hay habitaciones disponibles en la/s fecha/s deseada/s."}
```
Si se realizó la búsqueda con éxito pero no se encontraron coincidencias.

HABITACIONES MENOR A UN PRECIO

RUTA:

```
/api/cliente/habitacion/precio/all
```

MÉTODO:

```
methods=['GET']
```

Recibe un json con el dato 'precio' y realiza validaciones para corroborar que el valor sea un precio válido. Luego, en caso de estar todo en orden, devuelve una lista de habitaciones cuyo precio sea menor al ingresado por el usuario.

RETORNA

```
{"status":400, "message":"Debe ingresar un valor numerico válido."}
```

Si el valor ingresado no es numérico ni positivo.

```
{"message":"No hay resultados."}
```

Si se realizó la búsqueda con éxito pero no se encontraron coincidencias.

Json de habitaciones y sus datos, si se realizó la búsqueda con éxito y hubo resultados.

HABITACIONES OCUPADAS/DISPONIBLES POR FECHA

RUTA

```
/api/cliente/habitacion/fecha/all
```

MÉTODO

```
methods=['GET']
```

Recibe un json con el dato 'fecha' y realiza una búsqueda de todas las habitaciones, discriminando cuales están disponibles en dicha fecha, y cuales están ocupadas.

RETORNA

```
{"disponibles:": (lista de hab. disponibles),
  "ocupadas:": (lista de hab. ocupadas)}
```

Va a retornar un diccionario con 2 listas, una de habitaciones disponibles y otra de habitaciones ocupadas. Puede que una o ambas estén vacías, dependiendo de los resultados de la búsqueda.

RESERVAS PROPIAS

RUTA
```
/api/cliente/reservas
```

MÉTODO
```
methods=['GET']
```

No recibe dato más que el id del usuario logueado, que obtiene a través del token.
Devuelve una lista con las reservas hechas por el cliente activo.

RETORNA
```
{"message":"No posee reservas."}
```
Si el usuario no realizó ninguna reserva.

Json con las reservas del usuario, en caso de haber.

Empleados

LISTADO DE HABITACIONES

RUTA
```
/api/empleado/habitacion/listado
```

MÉTODO
```
methods=['GET']
```

Devuelve una lista con todas las habitaciones cargadas en el sistema, con todos sus datos, en caso de que haya.

RETORNA
```
{"message":"No hay habitaciones."}
```
En caso de que la lista de resultados esté vacía

Json con todas las habitaciones y sus datos.

HABITACIÓN POR ID

RUTA

```
/api/empleado/habitacion/<id>
```

MÉTODO

```
methods=['GET']
```

Recibe el dato ‘id’ por parámetro, devuelve todos los datos de la habitación cuyo id sea el solicitado, si el mismo existe.

RETORNA

```
{"message":"No hay coincidencias con la búsqueda."}
```

Si la búsqueda se realizó con éxito y no encontró la habitación con el id indicado

Json con datos de la habitación si la búsqueda se realizó con éxito y existe una coincidencia.

ALTA DE HABITACIÓN

RUTA

```
/api/empleado/habitacion/alta
```

MÉTODO

```
methods=['POST']
```

Recibe un json con los siguientes datos ‘nro_habitacion’, ‘descripcion_habitacion’, ‘precio_por_dia’, ‘estado’
nro_habitacion el número que se desea darle a la habitación
descripcion_habitacion una descripción de la habitación
precio_por_dia un double/entero indicando el precio de la habitación
estado valor del estado de la habitación. “activo” o “inactivo”.
Verifica que la habitación no exista, luego valida los datos ingresados y finalmente crea el registro con la habitación.

RETORNA

```
{"status": 400, "message": "Por favor ingrese un valor numérico válido."}
```

Si precio o número de habitación no es un número positivo / válido

```
{"error":"El estado no puede ser distinto de activo o inactivo"}
```

Si el campo estado es distinto de activo o inactivo

```
{"status": 400, "message": "La habitación ya existe."}
```

Si el número de habitación ya existe en la base de datos.

```
{"status": 200, "message":"Habitación creada con éxito."}
```

Si la habitación se creó exitosamente

UPDATE DE HABITACIÓN

RUTA

```
/api/empleado/habitacion/editar
```

MÉTODO

```
methods=['PUT']
```

Recibe un json con los datos 'nro_habitacion', 'descripcion_habitacion', 'precio_por_dia', 'id_habitacion'
Busca una habitación que coincida con el id ingresado, y cambia los datos de la misma por los ingresados por el usuario.

RETORNA

```
{"error": "No se encontró la habitación que desea editar. Por favor, inténtelo nuevamente."}
```

Si no encuentra la habitación

```
{"status": 400, "message": "Por favor ingrese un valor numérico válido."}
```

Si el precio o número de habitación no son válidos.

```
{"status":200, "exito": "Datos cambiados con éxito."}
```

Si la habitación se editó correctamente

UPDATE ESTADO DE HABITACIÓN

RUTA

```
/api/empleado/habitacion/estado
```

MÉTODO

```
methods=['PUT']
```

Recibe un json con el dato 'id' siendo el id de la habitación que se desea cambiar el estado, luego procede a cambiar el estado de 1 a 0 y viceversa.

RETORNA

```
{"status":200}
```

Cuando se cambia el estado exitosamente

LISTADO DE RESERVAS

RUTA

```
/api/empleado/reservas
```

MÉTODO

```
methods=['GET']
```

Devuelve una lista de todas las reservas hechas por los clientes, con los datos contenidos en los registros.

RETORNA

```
{"message":"No hay reservas."}
```

Si no hay reservas en los registros

Json con las reservas y los datos, si los hay.