Casos de Uso

Este documento recoge los casos de uso que contempla QRest en sus distintos ámbitos, desde el punto de vista de los usuarios desde la vista de la aplicación.

Diferenciamos en dos catergorías de casos de uso, de alto y bajo nivel:

- **Casos de uso de alto nivel**: Aquellos del alto nivel de abstracción que se compone de otros casos de uso de bajo nivel.
- **Casos de uso de bajo nivel**: Aquellas acciones concretas que puede hacer un usuario en la aplicación.

Se representan en el siguiente diagrama:

```
flowchart LR
  comensal[[Comensal]]
  subgraph casos_de_uso[Casos de uso]
    subgraph casos_de_uso_hacer_pedido[Hacer pedido]
      hacer_pedido([Hacer pedido])
      subgraph casos_de_uso_de_bajo_nivel_hacer_pedido[Casos de uso de bajo nivel]
        anadir_al_pedido([Añadir elemento al pedido])
        eliminar_del_pedido([Eliminar elemento del pedido])
        ver_pedido([Ver Pedido])
        ver_carta([Ver Carta])
        anadir_al_pedido_sin_modificarlo([Añadir elemento al pedido sin modificarlo])
        anadir_al_pedido_modificandolo([Añadir elemento al pedido modificándolo])
        confirmar_pedido([Confirmar pedido])
      end
    end
    subgraph casos_de_uso_pedir_ver_recibo[Ver recibo]
      ver_recibo([Ver recibo])
      subgraph casos_de_uso_de_bajo_nivel_pedir_cuenta[Casos de uso de bajo nivel]
        ver_recibo_individual([Ver recibo individual])
        ver_recibo_total([Ver recibo total])
      end
    end
    pedir_cuenta([Pedir cuenta])
    subgraph casos_de_uso_pedir_ver_recibo[Ver recibo]
      ver_recibo([Ver recibo])
      subgraph casos_de_uso_de_bajo_nivel_pedir_cuenta[Casos de uso de bajo nivel]
        ver_recibo_individual([Ver recibo individual])
        ver_recibo_total([Ver recibo total])
      end
    end
    pagar([Pagar])
  end
  comensal --uses--> hacer_pedido & pedir_cuenta & ver_recibo & pagar
```

```
hacer_pedido -.includes.-> anadir_al_pedido & eliminar_del_pedido & ver_pedido & ver_c
anadir_al_pedido_sin_modificarlo & anadir_al_pedido_modificandolo ==extends==> anadir_
ver_recibo_total & ver_recibo_individual ==extends==> ver_recibo
pedir_cuenta -.includes.-> ver_recibo
pagar -.includes.-> ver_recibo
```

Glosario

Actores

- **Empleado**: Entendido como una persona que opera el restaurante.
- **Comensal**: Dicho de un cliente que está sentado en la mesa que. Comparte pedido con los comensales de la misma mesa.

Pantallas

- **Entrada**: Métodos para acceder a la pantalla en cuestión. Puede ser otra pantalla u otro método.
- **Salida**: Pantallas a las que se puede acceder desde la pantalla en cuestión.
- **Ruta**: Punto de acceso en la API.

Carta

Elemento: Cualquier bebida, entrante, plato, postre, etc. de la carta que puede añadirse al pedido.

Elemento simple: Aquel que no hay que elegir nada. Es decir, no tiene variantes, ni extras opcionales, ni se le pueden quitar ingradientes. Ejemplo: Nestea.

Elemento complejo: Aquel para el que hay que elegir algo. Tiene variantes o extras o se pueden eliminar ingredientes.

Pantallas

```
flowchart LR
    carta[Carta]
    pedido[Pedido]
    recibo_total[Recibo Total]
    recibo_individual[Recibo Individual]
    pago[Pago]
    carta --> pedido
    pedido --> carta
    carta --> recibo_total
    recibo_total --> recibo_individual & pago
    recibo_individual --> recibo_total & pago
    pago --> recibo_total
```

Carta

Se ven los elementos de la carta del restaurante y pueden añadir o eliminar unidades de cualquier elemento de la carta. Todos los comensales de una misma mesa están asociados al mismo pedido y ven en tiempo real cualquer actualización (añadir o eliminar unidades de algún elemento) que haga cualquier otro comensal de la mermisma mesa.

- **Ruta**: /mesa/{Identificador de la mesa}/carta.
- **Entrada**: QR, Pantalla Pedido.
- **Salidas**: Pantalla Pedido, Pantalla Recibo Total.

Pedido

Se ven todos los elementos que han pedido los comensales de la mesa. Esto incluye el elemento concreto; con sus variantes, extras, e ingredientes eliminados, en caso de elementos complejos; y la cantidad.

- **Ruta**: /mesa/{Identificador de la mesa}/pedido
- **Entrada**: Pantalla Carta.
- **Salidas**:
  - Carta.
  - Pedido Confirmado.

Recibo Total

En esta pantalla se ve el recibo del pedido de toda la mesa en todas las comandas. Además del estdo de pago de cada elemento y quíen lo ha pedido.

- **Ruta**: /mesa/{Identificador de la mesa}/recibo/total
- **Entrada**: Pantalla Carta.
- **Salidas**:
  - Recibo Individual.
  - Por Pagar Total

Recibo Individual

En esta pantalla se ve el recibo del total del comensal en todas las comandas. Además del estdo de pago de cada elemento y quíen lo ha pedido.

- **Ruta**: /mesa/{Identificador de la mesa}/recibo/individual
- **Entradas**: Pantalla Recibo total.
- **Salidas**:
  - Recibo Total.
  - Por Pagar Total.

Por Pagar Total

En esta pantall se ve lo que queda por pagar de todo el pedido de todos los comensales en todas las comandas.

- **Ruta**: /mesa/{Identificador de la mesa}/por_pagar/total
- **Entradas**:
  - Pantalla Recibo Total.
  - Pantalla Por Pagar Individual.
  - Carta.
- **Salida**: Por Pagar Individual

Por Pagar Individual

En esta pantall se ve lo que queda por pagar de todo el pedido de todos los comensales en todas las comandas.

- **Ruta**: /mesa/{Identificador de la mesa}/por_pagar/individual
- **Entrada**: Pantalla Por Pagar Total.
- **Salidas**: Por Pagar Total.

Pago Caja

En esta pantalla es en el que el restaurante ve los pagos que se han solicitado pagar en caja. Se puede marcar como pagado un pedido.

- **Ruta**: /mesa/{Identificador de la mesa}/caja
- **Entrada**: Web del restaurante.
- **Salida**: Recibo Total.

Casos de uso

Los casos de uso de alto nivel son:

```
flowchart LR
  comensal[[Comensal]]
  hacer_pedido([Hacer pedido])
  pedir_cuenta([Pedir cuenta])
  ver_recibo([Ver recibo])
  pagar([Pagar])
  comensal --uses--> hacer_pedido & pedir_cuenta & ver_recibo & pagar
```

Que se representan en el siguiente diagrama de flujo:

```
flowchart LR
    hacer_pedido[Hacer pedido]
    pedir_cuenta[Pedir cuenta]
    ver_recibo[Ver recibo]
    pagar[Pagar]
    hacer_pedido --> hacer_pedido & pedir_cuenta
    pedir_cuenta --> ver_recibo
    ver_recibo --> ver_recibo & pagar
```

Casos de uso de *Hacer pedido*

El caso de uso de alto nivel *Hacer pedido* se representa en el siguiente diagrama:

```
flowchart LR
  comensal[[Comensal]]
  hacer_pedido([Hacer pedido])
  anadir_al_pedido([Añadir elemento al pedido])
  eliminar_del_pedido([Eliminar elemento del pedido])
  ver_pedido([Ver Pedido])
  ver_carta([Ver Carta])
  confirmar_pedido([Confirmar pedido])
  anadir_al_pedido_sin_modificarlo([Añadir elemento al pedido sin modificarlo])
  anadir_al_pedido_modificandolo([Añadir elemento al pedido modificándolo])
  comensal --uses--> hacer_pedido
  hacer_pedido --includes--> anadir_al_pedido & eliminar_del_pedido & ver_pedido & ver_c
  anadir_al_pedido_sin_modificarlo & anadir_al_pedido_modificandolo ==extends==> anadir_
```

Hacer pedido

- **Precondición**: Pantalla Carta.
- **Postcondición de éxito**:

Escenario Principal:

1. El comensal navega por la carta.
2. El comensal ejecuta *Añadir un elemento al pedido* o *Eliminar un elemento del pedido*.
3. El comensal ejecuta *Ver pedido*.
4. El comensal ejecuta *Confirmar pedido* de forma satisfactioria.

Escenarios Alternativos:

3a. El comensal vuelve a modificar el peiddo:

1. Vuelve al paso 2.

4a. El comensal vuelve a la carta:

1. El comensal selecciona *Volver a la carta*.
2. El sitema ejecuta *Ver Carta*.
3. Vuelve al paso 1.

4b. El comensal modifica el peido.

1. El comensal ejecuta *Añadir un elemento sin modificar* o *Eliminar un elemento del pedido*.
2. Vuelve al paso 4.

Ver Pedido

- **Precondición**: Pantalla Carta.

- **Postcondición**: Pantalla Pedido.

Escenario principal:

1. El comensal selecciona *Ver pedido*.
2. El sistema redirige a la pantalla *Pedido*.
3. El sistema hace una llamada a la *API* para recoger los datos del *Pedido*.
4. El sistema genera el HTML de la pantalla *Pedido*.
5. El JS del HTML resalta los elementos que han sido pedidos por el comensal.
6. El sistema muestra al comensal la pantalla de *Pedidos*.

Ver Carta

- **Precondición**: Ninguna.
- **Postcondición**: Pantalla Carta.

Escenario principal:

1. El sistema redirige a la pantalla *Carta*.
2. El sistema hace una llamada a la *API* para recoger los datos de la *Carta*.
3. El sistema hace una llamada a la *API* para recoger los datos del *Pedido*.
4. El sistema genera el HTML de la pantalla *Carta*.
5. El JS del HTML filtra segun alérgenos del comensal y resalta las sugerencias.
6. El sistema muestra al comensal la pantalla de *Carta*.

Confirmar Pedido

- **Precondición**: Pantalla Pedido.
- **Postcondición de éxito**: Pantalla Carta.

Escenario principal:

1. El comensal selecciona *Confirmar pedido*.
2. El sistema muestra un modal para pedirle al usuario que confirme la decisión.
3. El comensal selecciona *Aceptar*.
4. El sistema llama a la *API* para confirmar el pedido.
5. El sistema ejecuta *Ver Carta*.

Escenario alternativo:

3a. El comensal cancela la confirmación del pedido.

1. El comensal selecciona *Cancelar*.
2. El sistema cierra el modal. (Sigue en la pantalla *Pedido*).

Añadir elemento al pedido sin modificar

- **Precondición**: Pantalla Carta ó Pedido.
- **Postcondicines**:
  - Se ha añadido una unidad del elemento al pedido.

- Se ha incrementado una unidad de ese elemento en las vistas de los comensales.

Escenario principal:

1. El comensal selecciona '+' sobre el elemento a añadir.
2. El sistema llama a la API para añadir el elemento al pedido con la información del elemento y el identificador del cliente.
3. El JS incrementa en 1 la cantidad de ese elemento.
4. El sistema retransmite al JS del resto de comensales la petición para que incrementen en 1 ese elemento.

Añadir elemento al pedido modificándolo

- **Precondición**: Pantalla Carta.
- **Postcondicines de éxito**:
  - Se ha añadido `cantidad` unidades del elemento al pedido.
  - Se ha incrementado en `cantidad` unidades de ese elemento en las vistas de los comensales.

Escenario principal:

1. El comensal selecciona '+' sobre el elemento a añadir.
2. El sistema muestra un modal para seleccionar las variantes, extras e ingredientes.
3. El comensal selecciona las variantes que quiera (si las hay), selecciona los extras que quiera (si los hay) y deselecciona los ingredientes no deseados (si los hay).
4. El comensal introduce la cantidad.
5. El comensal selecciona *Aceptar*.
6. El sistema llama a la API para añadir el elemento al pedido con la información del elemento, la cantidad y el identificador del cliente.
7. El JS incrementa en `cantidad` la cantidad de ese elemento.
8. El sistema retransmite al JS del resto de comensales la petición para que incrementen en `cantidad` ese elemento.

Escenario alternativo:

[3-5]a. El comensal cierra el modal.

1. El comensal selecciona *Cancelar*.
2. El sistema cierra el modal. (Sigue en la pantalla Carta).

Eliminar elemento del pedido

- **Precondición**: Pantalla Carta ó Pedido.
- **Postcondicines**:
  - Se ha eliminado una unidad del elemento del pedido.
  - Se ha decrementado en una unidad ese elemento en las vistas de los comensales.

Escenario principal:

1. El comensal selecciona '-' sobre el elemento a eliminar.

2. El sistema llama a la API para eliminar el elemento al pedido con la información del elemento y el identificador del cliente.
3. El JS decrementa en 1 la cantidad de ese elemento.
4. El sistema retransmite al JS del resto de comensales la petición para que decrementen en 1 ese elemento.

Casos de uso de *Pedir cuenta*

El caso de uso de alto nivel _Pedir cuenta se representa en el siguiente diagrama:

```
flowchart LR
  comensal[[Comensal]]
  pedir_cuenta([Pedir cuenta])
  ver_recibo_total([Ver recibo total])
  comensal --uses--> pedir_cuenta
  pedir_cuenta -.includes.-> ver_recibo_total
```

Pedir cuenta

- **Precondiciones**:
  - Pantalla Carta.
  - Al menos una comanda (pedido) confirmado.
- **Postcondiciones de éxito**: Se ha generado el recibo.

Escenario principal:

1. El comensal selecciona *Pedir Cuenta*.
2. El sistema comprueba que hay al menos un pedido realizado.
3. El sistema muestra un modal para confirmar la decisión.
4. El comensal selecciona *Aceptar*.
5. El sistema llama a la *API* para generar el recibo.
6. El sistema ejecuta *Ver recibo total*.

Escenarios alternativos:

2a. Se muestra un error de que no se ha pedido nada aún:

1. El sistema detecta que no hay ningún pedido realizado
2. El sistema muestra un mensaje de error avisando de que no se ha hecho ningún pedido aún.

4a. El comensal no pide la cuenta.

1. El comensal selecciona *Cancelar*.
2. El sistema cierra el modal. (Sigue en la pantalla Carta).

Casos de uso de Ver recibo

El caso de uso de alto nivel *Ver recibo* se representa en el siguiente diagrama:

```
flowchart LR
  comensal[[Comensal]]
  ver_recibo([Ver recibo])
  ver_recibo_total([Ver recibo total])
  ver_recibo_individual([Ver recibo individual])
  comensal --uses--> ver_recibo
  ver_recibo_total & ver_recibo_individual ==extends==> ver_recibo
```

Ver recibo total **Precondición**: Se ha ejecutado *Generar recibo*. **Postcondición**: Pantalla Recibo total.

Escenario principal:

1. El sistema llama a la *API* para obtener la información del recibo.
2. El sistema genera el HTML de la pantalla *Recibo total*.
3. El JS del HTML resalta los elementos pagados.
4. El sistema muestra al comensal la pantalla *Recibo total*.

Ver recibo total **Precondición**: Se ha ejecutado *Generar recibo*. **Postcondición**: Pantalla Recibo individual.

Escenario principal:

1. El sistema llama a la *API* para obtener la información del recibo del comensal.
2. El sistema genera el HTML de la pantalla _Recibo individual.
3. El sistema muestra al comensal la pantalla _Recibo inidividual.

Casos de uso de *Pagar*

```
flowchart LR
  comensal[[Comensal]]
  ver_por_pagar([Ver por pagar])
  ver_por_pagar_individual([Ver por pagar individual])
  ver_por_pagar_total([Ver por pagar total])
  pagar([Pagar])
  pagar_online([Pagar online])
  pagar_en_caja([Pagar en caja])
  ver_por_pagar_individual & ver_por_pagar_total ==extends==> ver_por_pagar
  pagar_online & pagar_en_caja ==extends==> pagar
  comensal --uses--> pagar & ver_por_pagar
  pagar -.includes.-> ver_por_pagar
```

Pagar online **Precondición**: Pantalla *Por pagar total o individual*. **Postcondición de éxito**: Se han pagado los elementos correspondientes.

Escenario principal:

1. El comensal selecciona *Pagar*.

2. El comensal introduce los datos de pago.
3. La pasarela de pago comprueba que los datos sean correctos.
4. Los bancarios son correctos y se puede realizar el pago.
5. La pasarela de pago tramita el pago.
6. El sistema marca los elemento correspondientes como pagado.
7. El sistema ejecuta *Ver por pagar*.

Escenario alternativo:

2a. Se cancela el pago.

1. El comensal cancela el pago.
2. El sistema ejecuta *Ver por pagar*.

3a. Los datos bancarios no son correctos.

1. Los datos bancarios introducidos no son correctos.
2. La pasarela de pago informa de los errores.
3. Se vuelve al paso 2.

Pagar en caja **Precondición**: Pantalla *Por pagar total o individual*. **Postcondición de éxito**: Se han pagado los elementos correspondientes.

Escenario principal:

1. El comensal selecciona *Pagar*.
2. El sistema informa al restaurante que se quiere pagar en caja.
3. El sistema redirige al la pantalla de *Esperando pago*.
4. El comensal paga en caja.
5. El camarero marca el pedido como pagado.
6. El sistema marca los elemento correspondientes como pagado.
7. El sistema ejecuta *Ver por pagar*.

Escenario alternativo:

[3-4] Se cancela el pago.

1. El comensal cancela el pago.
2. El sistema ejecuta *Ver por pagar*.