**Andrés Felipe Pérez – 202215659**
**Rubén Darío Franco – 202012119**
**Mario Velásquez – 202020502**

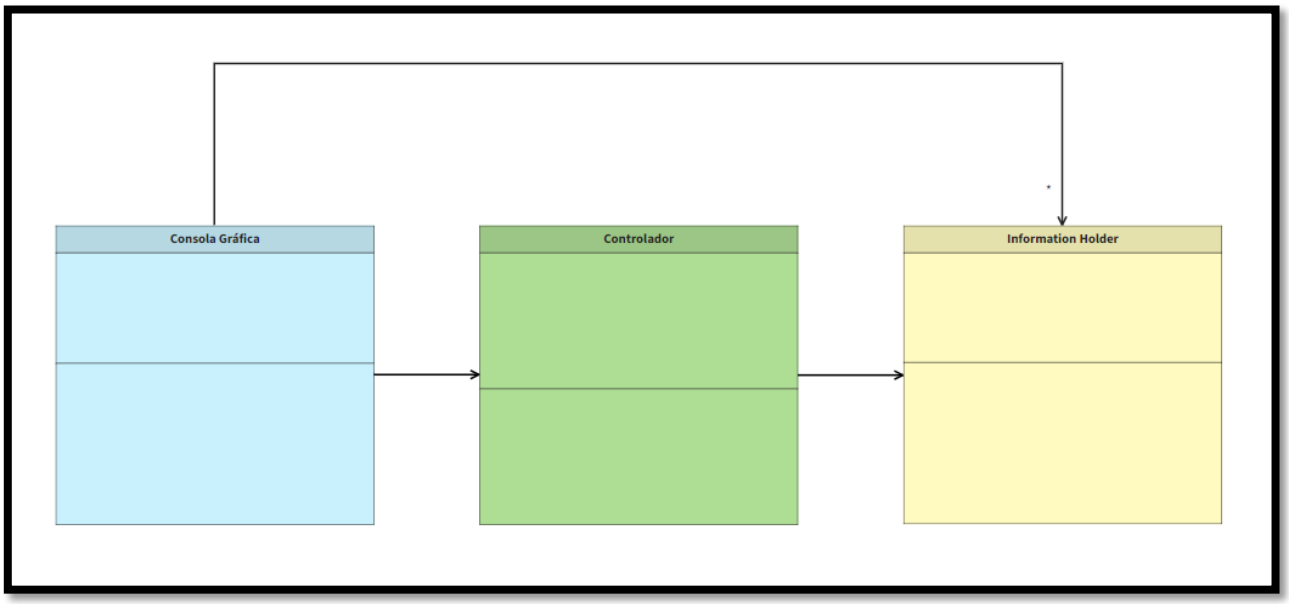
# PROYECTO 3 – DPOO
# ENTREGA
# DOCUMENTO DE DISEÑO E IMPLEMENTACIÓN DE LA INTERFAZ

## DIAGRAMA DE CLASES:

Se desarrolló el siguiente diagrama de clases, se puede acceder a él a través del siguiente enlace:

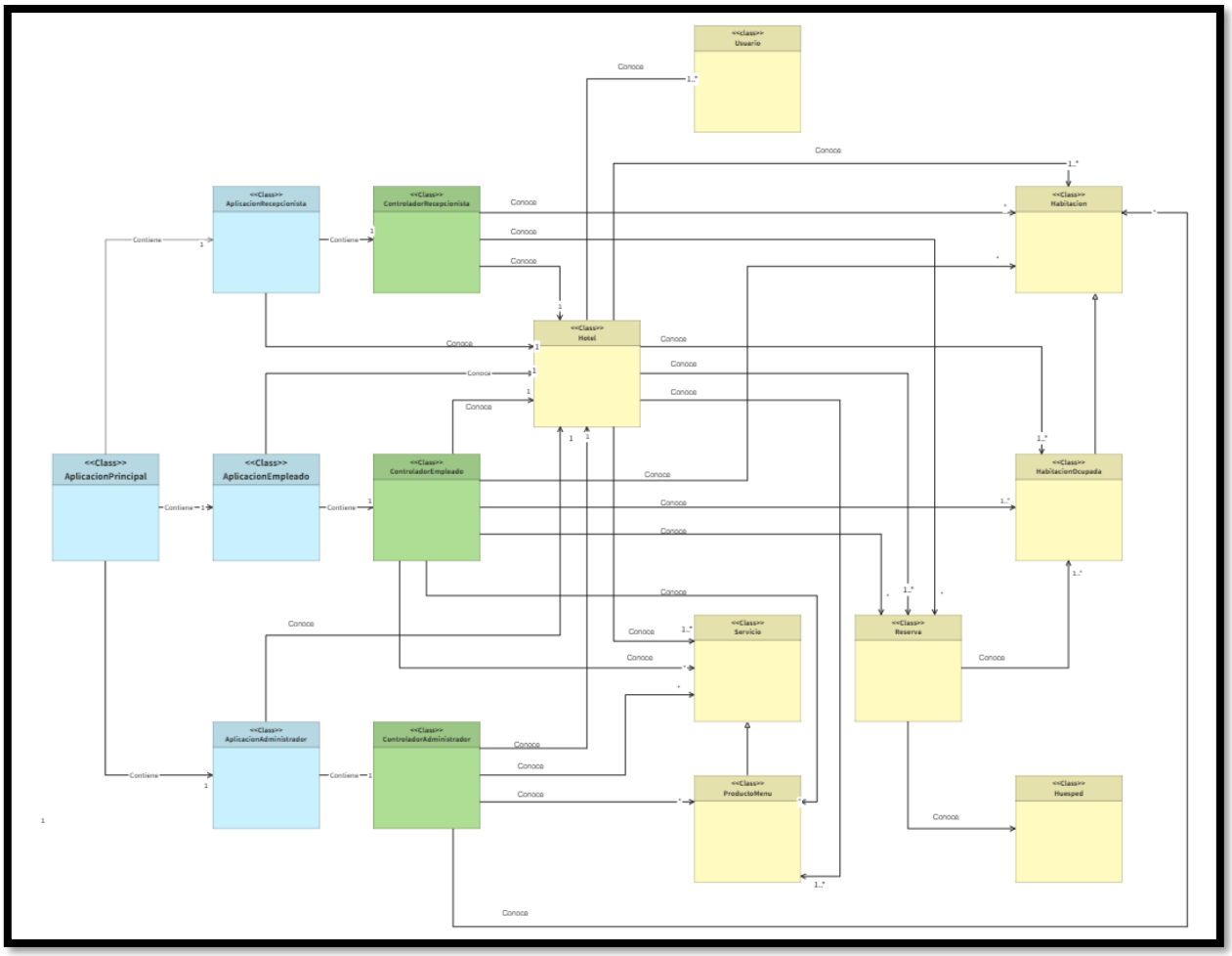
https://app.moqups.com/ChEzWXm6MHnCurrl2Oqh4tlMnBExc1oC/view/page/a8bfb0d3e

**Diagrama de alto nivel:**



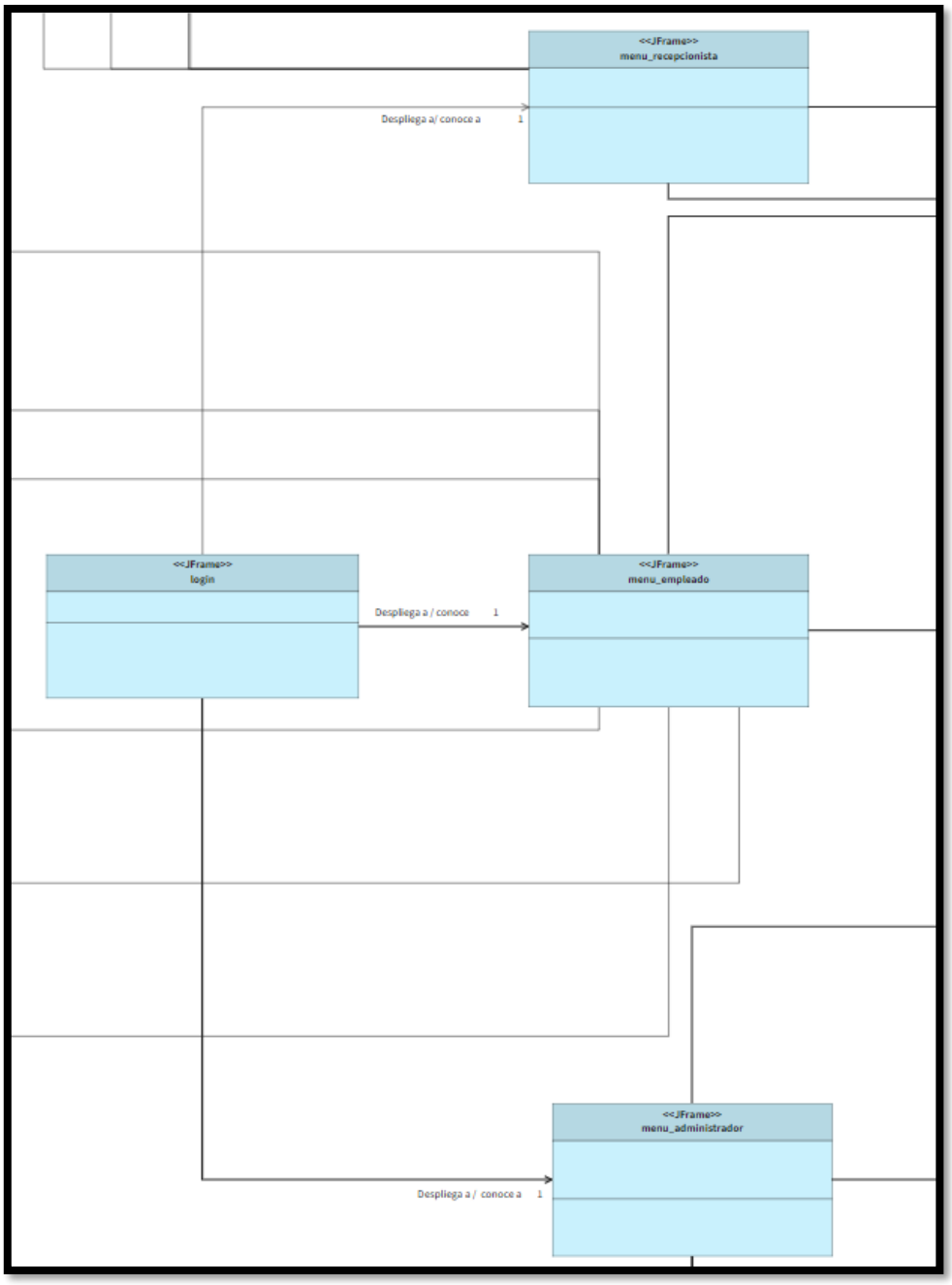

Nota: los colores de este diagrama se mantienen en los siguientes.

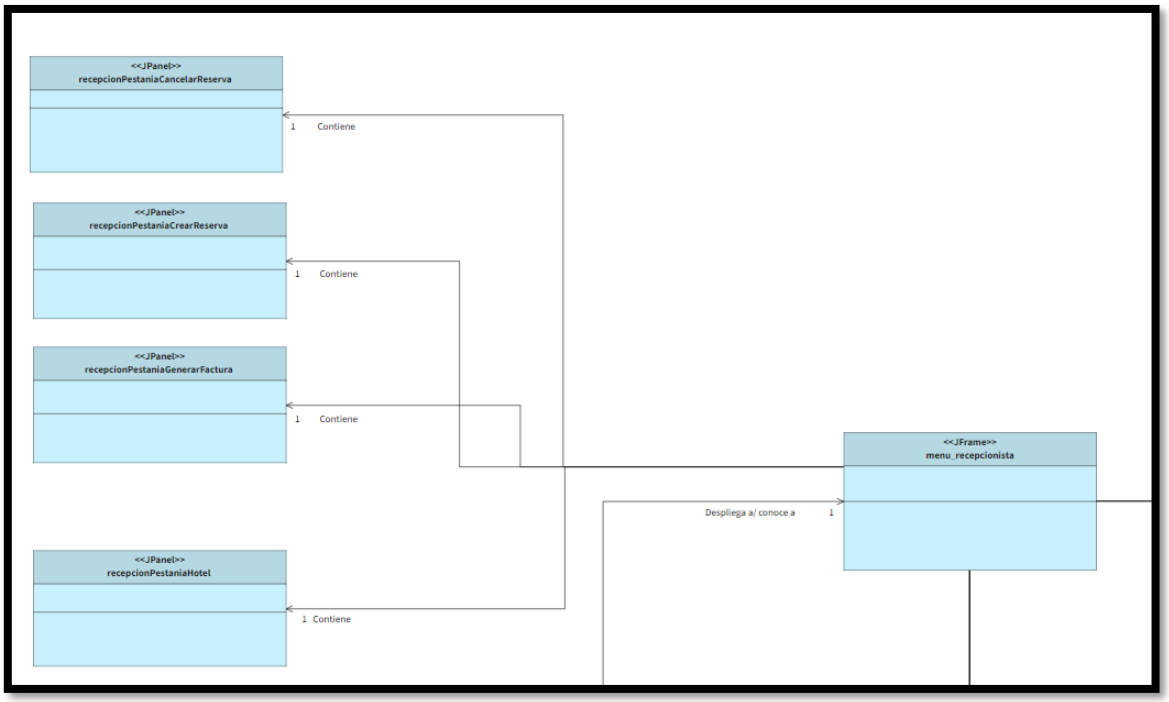
**Diagrama de relaciones:**

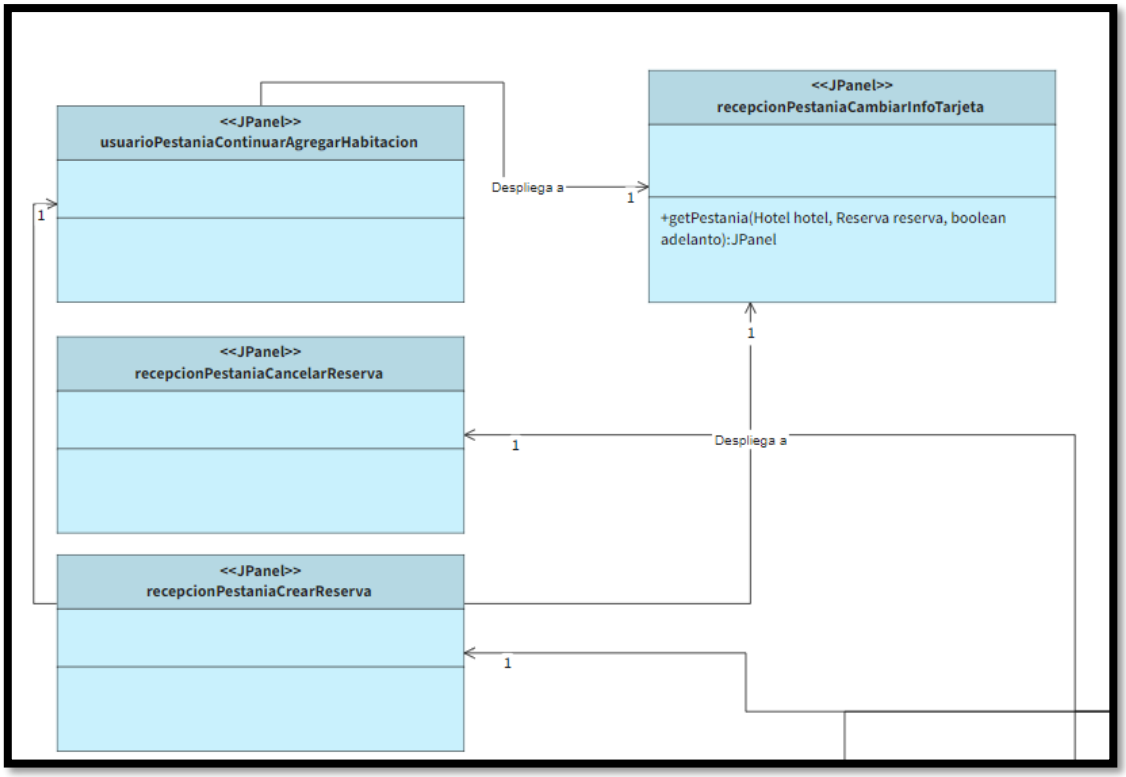


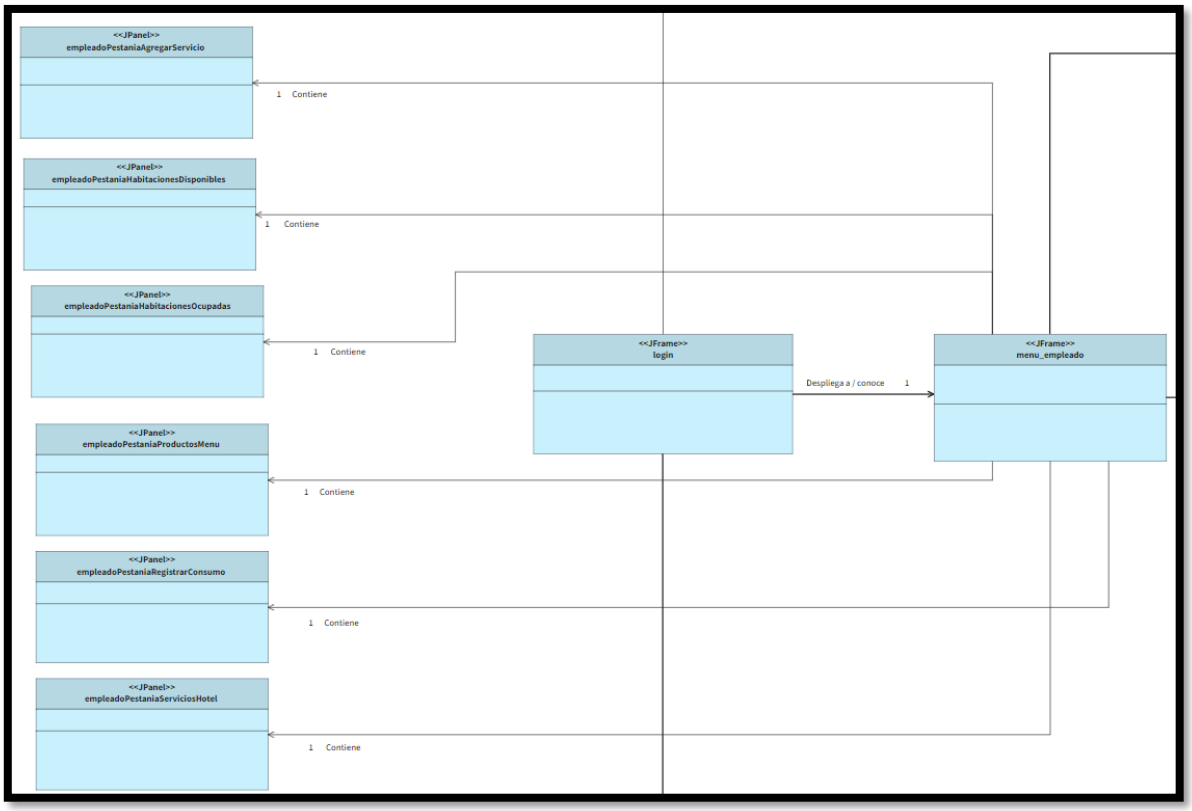

Este diagrama de relaciones mantiene el mismo tipo de relaciones que el nuevo diagrama que implementa la interfaz gráfica de la aplicación. Por esta razón, se sigue conservando esta imagen y se le complementa con las imágenes a continuación, que se encargan de mostrar las relaciones presentes en la interfaz. Igualmente, en la explicación de cada una de las clases, se puede deducir algunas de las relaciones que existen entre las distintas clases presentes.

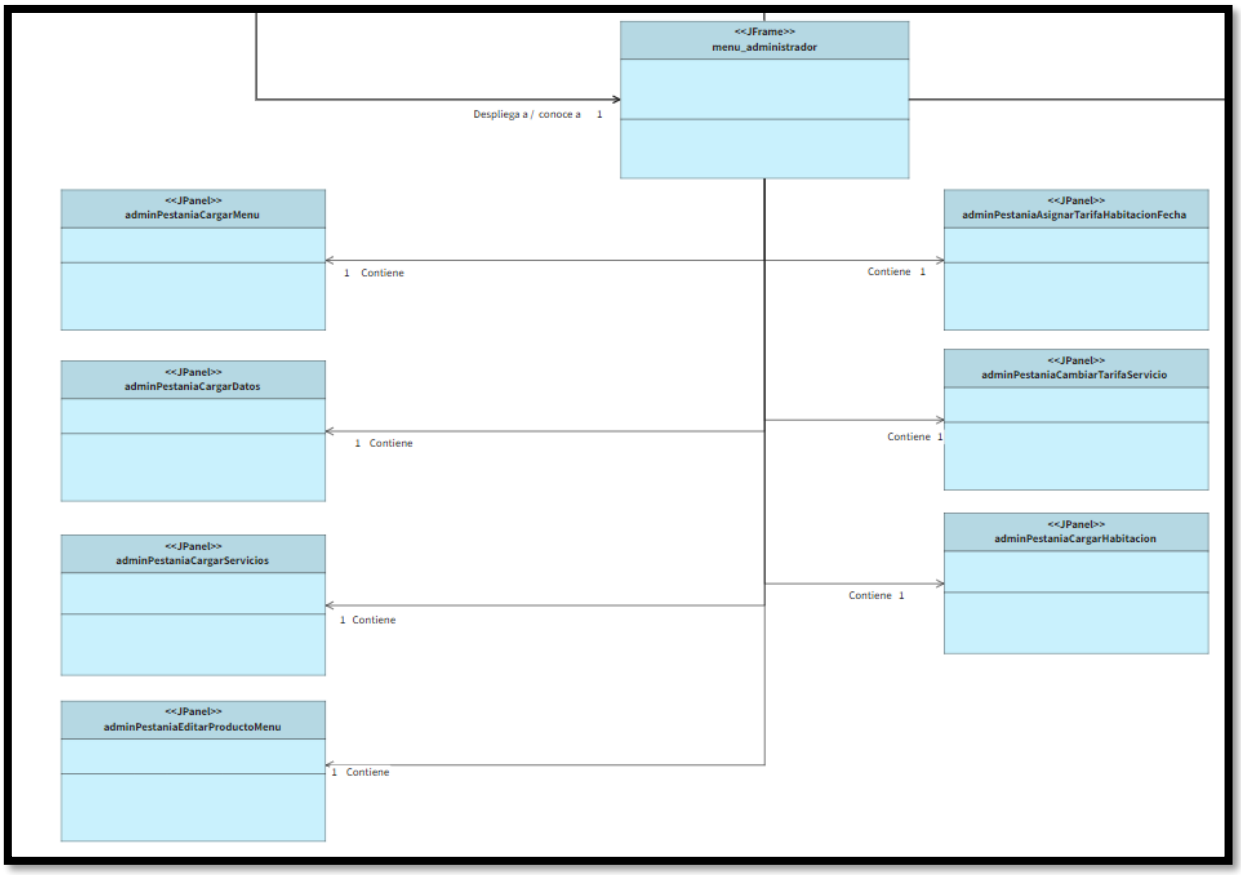
## Relaciones en la interfaz



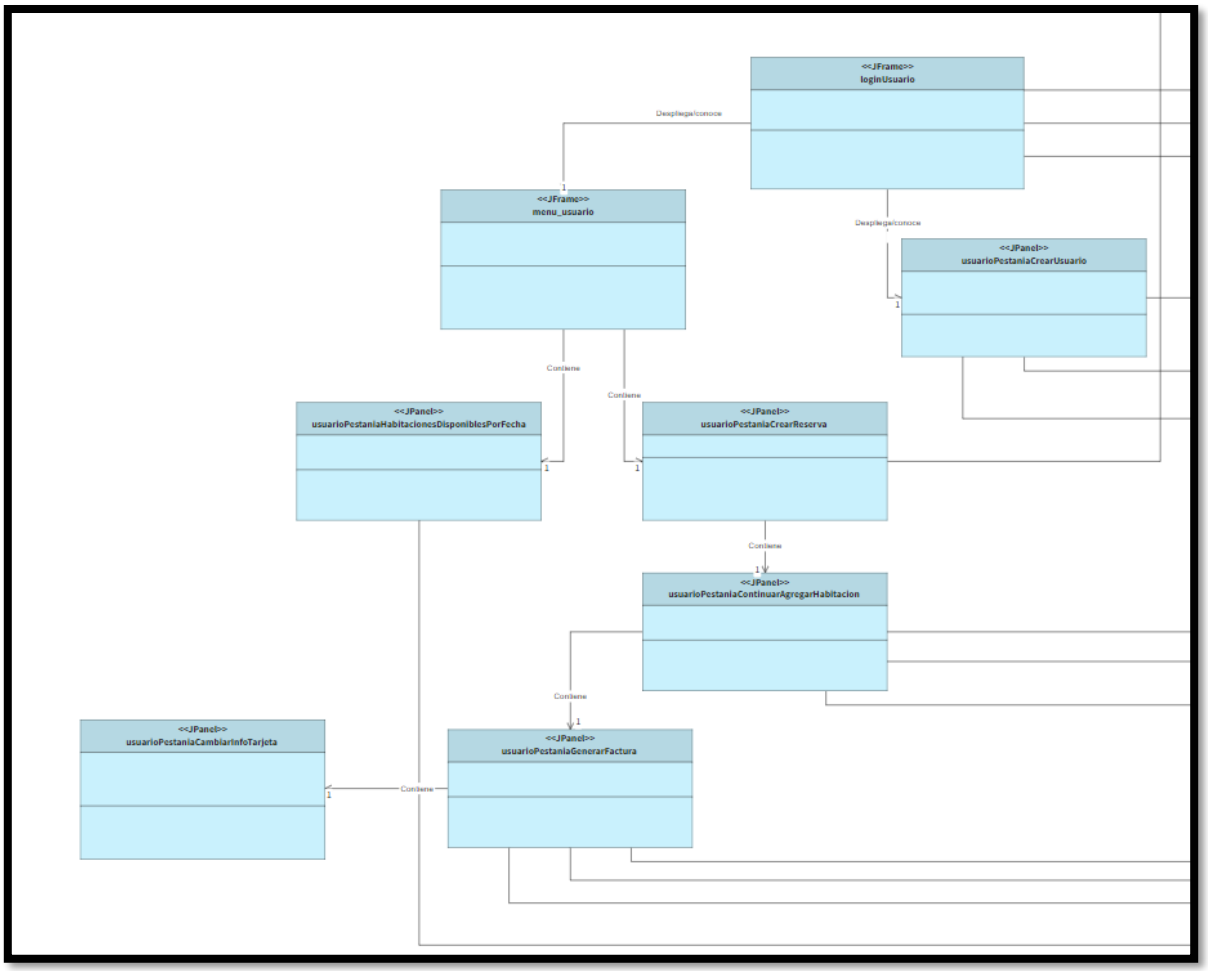

<<JPanel>>
recepcionPestaniaCancelarReserva
1 Contiene
<<JPanel>>
recepcionPestaniaCrearReserva
1 Contiene
<<JPanel>>
recepcionPestaniaGenerarFactura
1 Contiene
<<JPanel>>
recepcionPestaniaHotel
1 Contiene
<<JFrame>>
menu_recepcionista
Despliega a/ conoce a
1


<<JPanel>>
usuarioPestaniaContinuarAgregarHabitacion
Despliega a
1
<<JPanel>>
recepcionPestaniaCambiarInfoTarjeta
+getPestania(Hotel hotel, Reserva reserva, boolean adelanto):JPanel
1
<<JPanel>>
recepcionPestaniaCancelarReserva
1
Despliega a
<<JPanel>>
recepcionPestaniaCrearReserva
1


<<JPanel>>
empleadoPestaniaAgregarServicio
1 Contiene
<<JPanel>>
empleadoPestaniaHabitacionesDisponibles
1 Contiene
<<JPanel>>
empleadoPestaniaHabitacionesOcupadas
1 Contiene
<<JFrame>>
login
Despliega a / conoce 1
<<JFrame>>
menu_empleado
<<JPanel>>
empleadoPestaniaProductosMenu
1 Contiene
<<JPanel>>
empleadoPestaniaRegistrarConsumo
1 Contiene
<<JPanel>>
empleadoPestaniaServiciosHotel
1 Contiene


<<JFrame>>
menu_administrador
Despliega a / conoce a 1
<<JPanel>>
adminPestaniaCargarMenu
1 Contiene
<<JPanel>>
adminPestaniaAsignarTarifaHabitacionFecha
Contiene 1
<<JPanel>>
adminPestaniaCargarDatos
1 Contiene
<<JPanel>>
adminPestaniaCambiarTarifaServicio
Contiene 1
<<JPanel>>
adminPestaniaCargarServicios
1 Contiene
<<JPanel>>
adminPestaniaCargarHabitacion
Contiene 1
<<JPanel>>
adminPestaniaEditarProductoMenu
1 Contiene

**Relaciones en la Interfaz de la aplicación dispuesta para los huéspedes del hotel**



**Relaciones entre las clases que componen la Pasarela de pago de la aplicación**

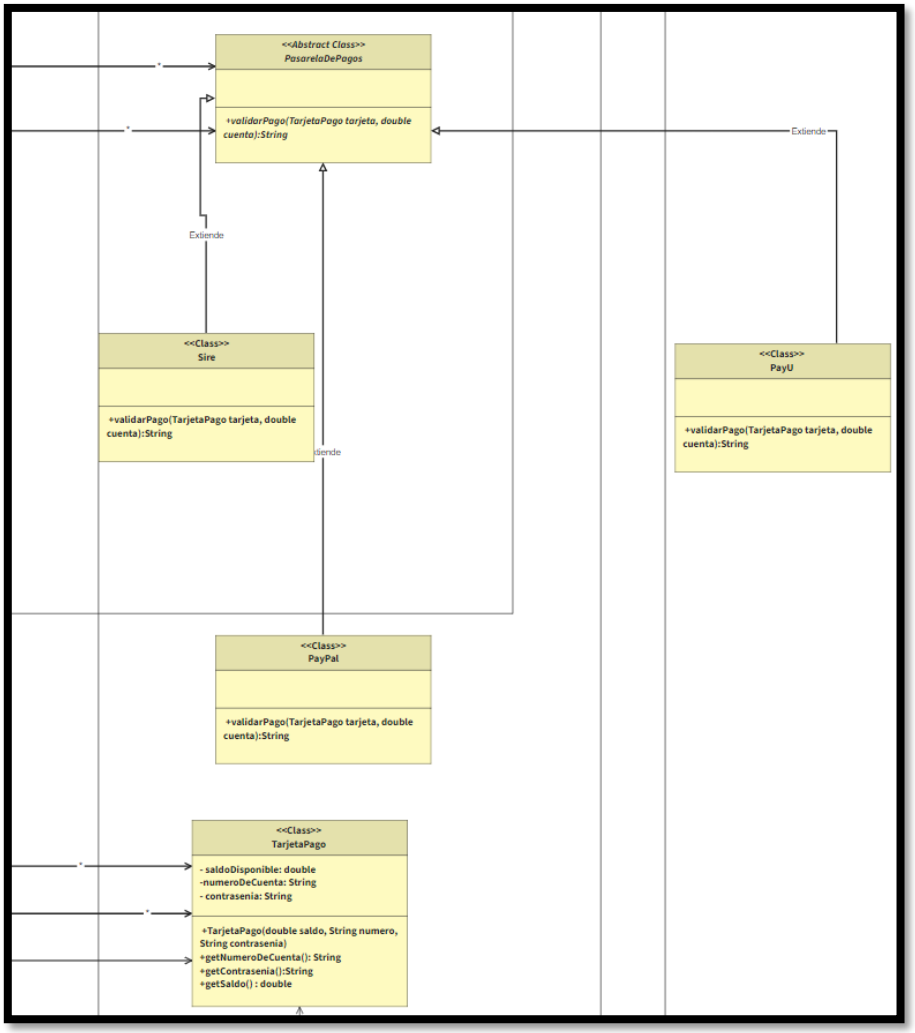
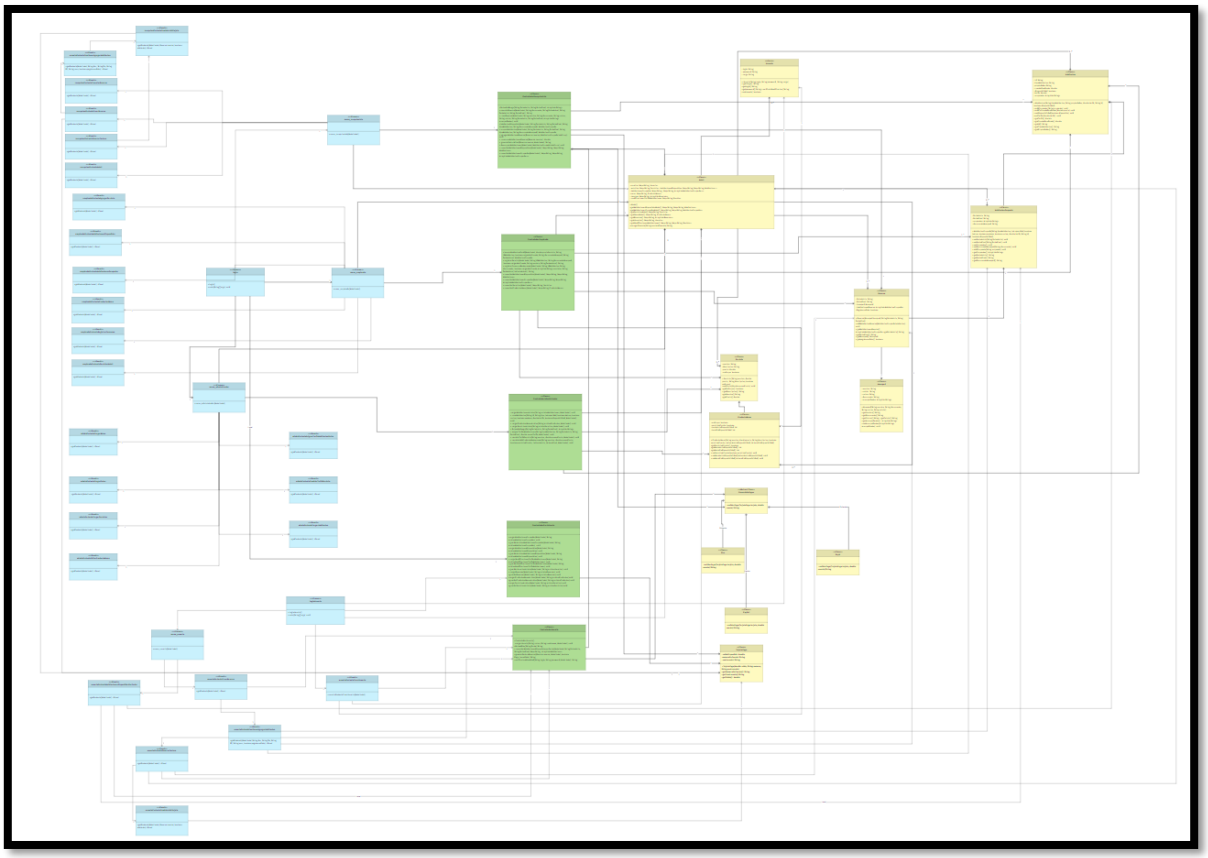


**Diagrama de clases:**



Recomendamos acceder al enlace que dejamos al principio del documento, para observar detalladamente el diagrama de clases construido, ya que se nos hace muy difícil mostrar todas las relaciones presentes.

**Decisiones de diseño:**

Se han implementado diversas decisiones al diseño de la solución. Entra ellas están el número de restaurantes en la aplicación, el tener distintos controladores y aplicaciones para cada tipo de usuario, o el uso de archivos .txt para el manejo de la información.

En primer lugar, se decidió crear un solo restaurante dentro de la aplicación para el hotel a pesar de que en la realidad cada hotel puede tener más de un hotel. Es decir, la aplicación mostrará todos los productos de los menús de todos los restaurantes. Lo anterior se da por facilidad de implementación con el valor agregado que desde cualquier restaurante del hotel se pueden pedir y facturar productos de cualquier otro restaurante.

Por otro lado, se decidió que cada tipo de usuario tendría su propia aplicación y controlador. Lo anterior permite que mayor facilidad en el sistema de autenticación de usuario y evita que un usuario no administrador tenga acceso a funcionalidades de administrador, por ejemplo. Lo anterior dificulta un poco la implementación, especialmente si se añaden más tipos de usuarios. Sin embargo, esta decisión facilita la autenticación de usuarios y la seguridad del sistema.

Finalmente se tiene la decisión del uso de archivos .txt que se da para facilitar la carga de los archivos por parte del administrador. Este tipo de archivos son de baja complejidad para el administrador y evitan la necesidad de desarrollar un sistema binario de archivos, que puede llegar a ser muy complejo.

## PERSISTENCIA

Para garantizar que la información que se genere en cada ejecución de la aplicación quede guardada en la base de datos, de modo que los datos persistan a lo largo del tiempo, se utilizarán archivos de texto en el formato .txt para almacenar y cargar la información. El motivo por el que se usarán archivos .txt es que su manipulación es bastante sencilla y porque podremos darle a la información el formato que más se ajuste a nuestras necesidades.

La información que decidimos guardar en estos archivos fue la que corresponde a las reservas activas del hotel, las habitaciones disponibles del hotel, las habitaciones ocupadas en ciertas fechas, los productos del menú del restaurante (esto tiene en cuenta las modificaciones realizadas por los administradores del hotel), los servicios ofrecidos por el hotel (también tiene en cuenta las modificaciones efectuadas por los administradores) y las modificaciones que se le hagan a las tarifas de cada uno de los tipos de habitaciones para unas fechas dadas.

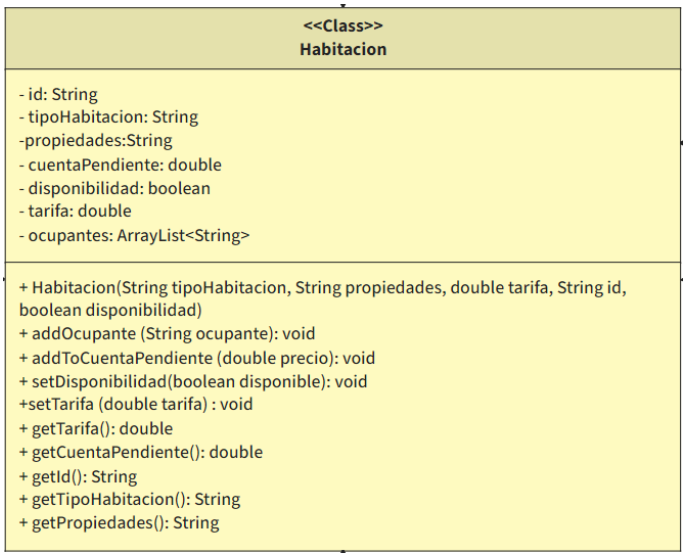
Cada una de estas estará almacenada en archivos independientes ubicados en una carpeta diferente, una carpeta externa, a dónde se encuentra la implementación de la aplicación para evitar modificaciones por parte de los usuarios de la aplicación (Administradores, recepcionistas y otros empleados).

La manera en la que hará la manipulación de estos archivos será a través de un controlador encargado exclusivamente de estas funcionalidades, el cual cargará información pasada y la actualizará con los cambios que se realicen en cada ejecución de la aplicación. De este modo disminuiremos el acoplamiento entre los controladores correspondientes a los usuarios de la aplicación y el controlador de persistencia.

## DESCRIPCIÓN DETALLA (RESPONSABILIDAD Y JUSTIFICACIÓN) DE LOS COMPONENTES DEL DISEÑO QUE COMPONEN LAS DIFERENTES FUNCIONALIDADES DEL SISTEMA

## MODEL

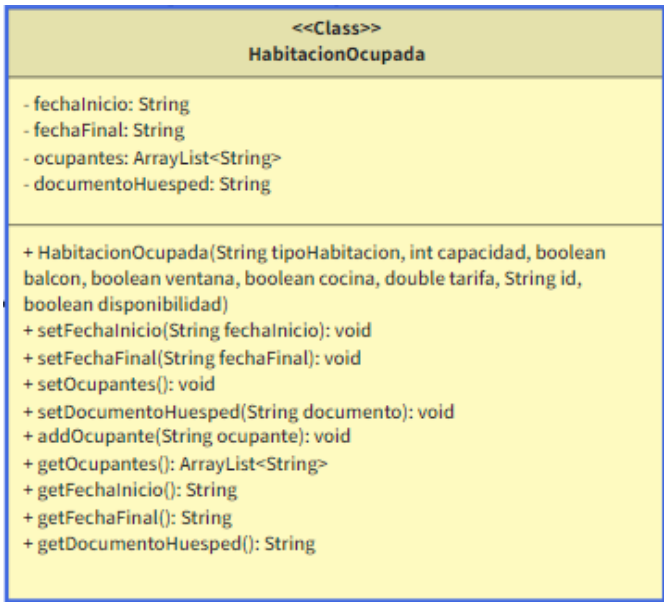
## CLASS HABITACIÓN




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

La clase Habitación se establece con el fin de crear y administrar (los diferentes atributos con los que cuenta) por separado cada una de las habitaciones (objetos de la clase) de las que estará compuesto el hotel. Esto se hará por medio de los atributos y métodos asignados (como los podemos ver en la imagen), que nos permitirán, primero que todo, acceder al valor de ciertos atributos por medio de los get presentes. Así mismo, se cuenta con un constructor, que nos permitirá crear un objeto de tipo habitación; algunos set, que nos facilitan la cambiar el valor de algunos atributos de las habitaciones, como es su disponibilidad y tarifa, y; algunos add, que nos permiten agregar información a atributos como “ocupantes” (ocupantes de la habitación) y “cuentaPendiente” (se refiere a la cuenta que se le asigna a una habitación cuando se realizan algunos consumos que no son pagados de forma inmediata).

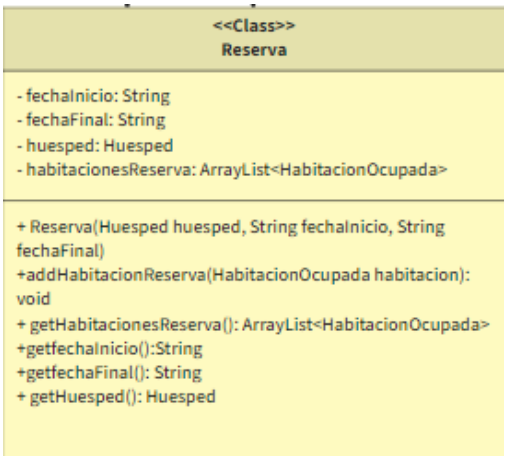
## CLASS HABITACIÓN OCUPADA




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

La clase Habitación Ocupada extiende de la clase Habitación y se establece con el fin de crear y administrar por separado cada una de las habitaciones ocupadas (objetos de la clase) de las que estará compuesto el hotel. Esto se hará por medio de los atributos y métodos asignados (como los podemos ver en la imagen), que nos permiten, con el constructor, crear un objeto de tipo habitación ocupada; con los set, cambiar el valor de algunos atributos de las habitaciones, como es su fecha de inicio y fin de ocupación, sus ocupantes y el documento del responsable de la habitación ocupada; con los add, agregar información a atributos como “ocupante” (ocupantes de la habitación), y; con los gets, acceder al valor actualizado de los atributos de la clase.

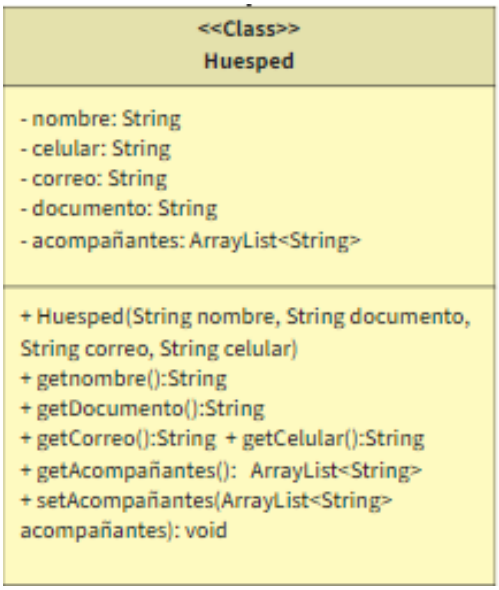
## CLASS RESERVA




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

La clase Reserva, se establece con el fin de crear y administrar por separado las reservas que tenga y pueda llegar a tener el hotel. De esta forma, esta clase nos facilitará conocer la fecha de inicio y fin de una reserva, las habitaciones de las que se compone la reserva y que serán ocupadas (las cuales pueden ser agregadas a disposición del huésped), y el responsable de la misma.

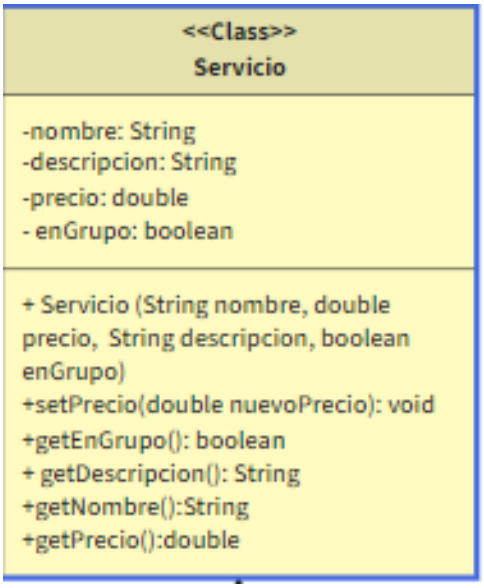
**CLASS HUESPED**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el fin de crear el objeto Huésped (que en otras palabras es un cliente del hotel), que será usado para el registro de los ocupantes del hotel y de cada una de las habitaciones. Así mismo, será utilizado en el registro y asignación de consumos, servicios y cuentas pendientes, que es fundamental para conocer el responsable de consumo y uso de los diferentes servicios que ofrece el hotel.

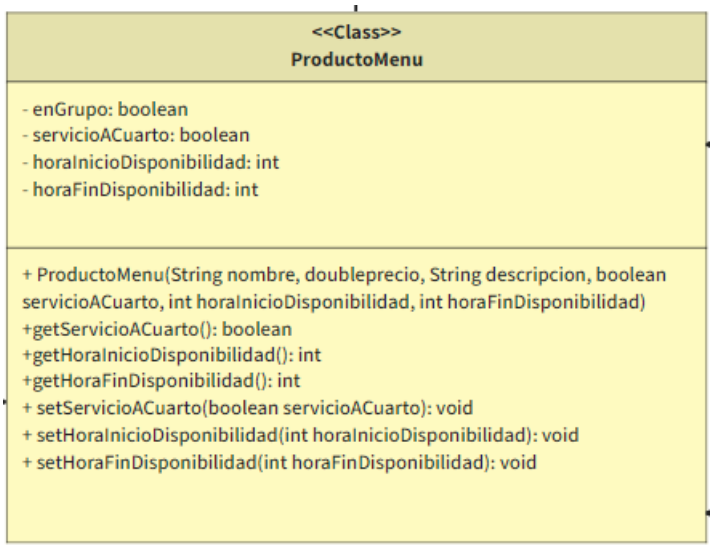
**CLASS SERVICIO**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el fin de crear y administrar cada uno de los servicios y consumos con los que cuenta el Hotel. Es así como, por medio de esta clase, se nos facilitará el acceso y la comunicación de la información, pues podremos conocer las características de cada servicio/ consumo como es su nombre, descripción, precio y si es de pago en grupo o por persona.

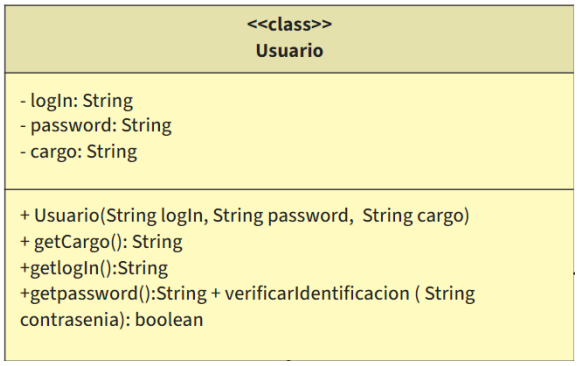
## CLASS PRODUCTO MENÚ




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase extiende de la clase Servicio y se establece con el fin de crear y administrar los diferentes productos de los que estará compuesto del menú que se ofrecerá en el restaurante del hotel. De esta forma, esta clase nos facilitará el acceso y la comunicación de información, ya que nos permitirá conocer y modificar las características de cada uno de estos productos, como su hora inicio y fin de disponibilidad, y si están ponibles para ser consumidos en el cuarto.

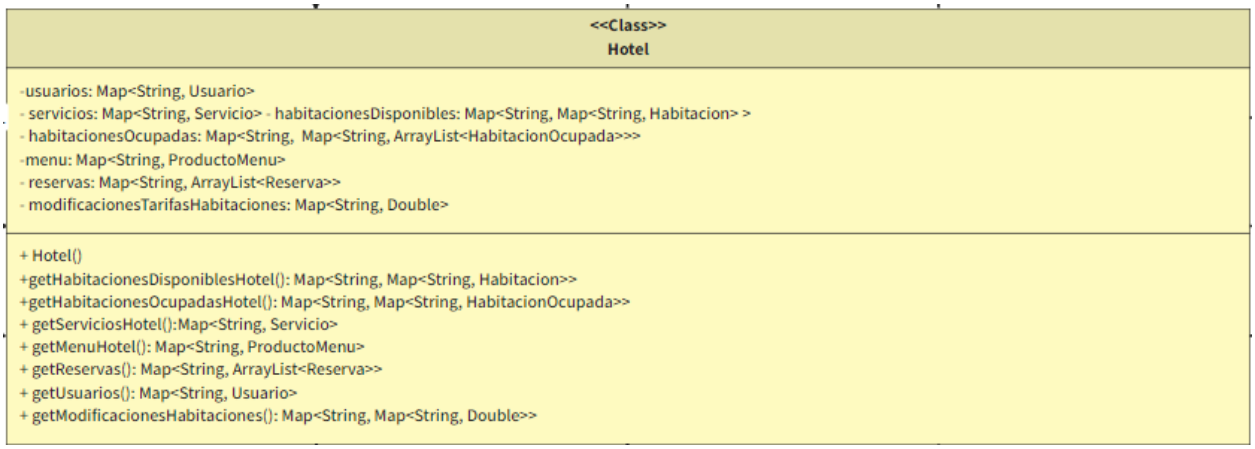
## CLASS USUARIO



## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el fin de crear los usuarios del sistema construido. De esta forma, esta clase nos permitirá construir un usuario, que tendrá un login, una contraseña y un cargo, que será utilizado posteriormente para redirigirlo al menú de opciones correspondiente a su cargo. Igualmente, la clase Usuario, nos permitirá hacer un control de acceso y seguridad al sistema, ya que nos facilitará hacer la verificación de contraseña cuando un usuario quiera ingresar y ejecutar acciones permitidas según su cargo.

## CLASS HOTEL




## ACLARACIÓN DE LOS ATRIBUTOS

Usuarios: Map<Login del usuario, objeto de la clase usuario>

Servicios: Map<Nombre del servicio, objeto de la clase servicio>

HabitacionesOcupadas: Map<Tipo de Habitación, Map<Id de habitación, ArrayList<registro de las habitaciones ocupadas (una habitación puede estar ocupada en diferentes fechas)>>>

HabitacionesDisponibles: Map<Tipo de Habitación, Map<Id de habitación, ArrayList<registro de las habitaciones ocupadas (una habitación puede estar ocupada en diferentes fechas)>>>

Menu: Map<Nombre del producto, objeto de la clase ProductoMenu>

Reservas: Map<documento del huésped que hace la reserva, <objeto de la clase Reserva>

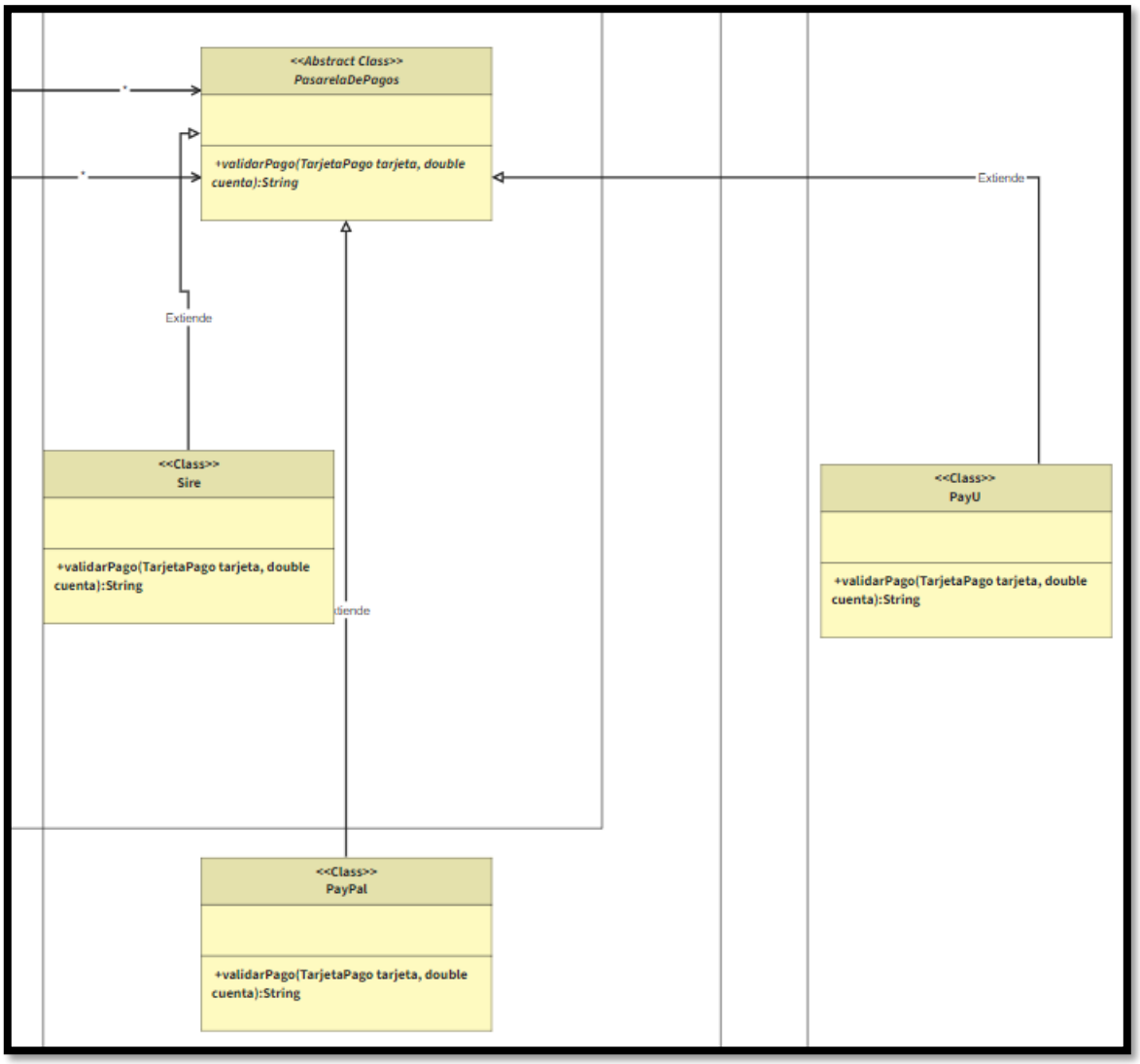
modificaciones Tarifas Habitaciones: Map<Tipo de habitación, Map<fecha, tarifa asignada por el administrador>>

## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

La clase Hotel, puede ser considerada como una de las más importantes en esta etapa, ya que es con la que conectan todas las clases y el eje central del sistema. Es así como, esta clase se establece con el fin de crear el Hotel, almacenar y acceder a toda la información del

hotel, que se compone de las habitaciones, habitaciones ocupadas, reservas, menú, usuarios y las modificaciones de las tarifas de las habitaciones. Así, podemos ver como esta clase actúa como el gran INFORMATION HOLDER de la aplicación, siendo eje primordial para su funcionamiento.

## ABSTRACT CLASS PASARELADEPAGOS Y CLAS SIRE, PAYU, PAYPAL



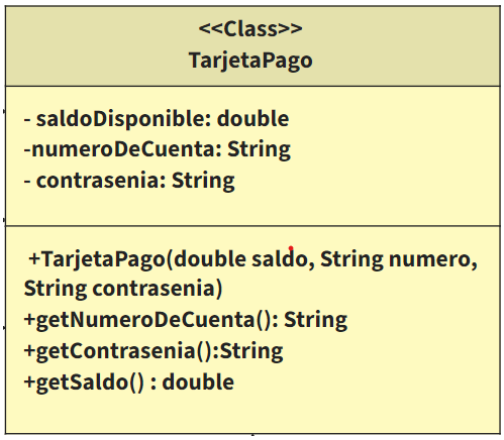

## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

La clase abstracta PasarelaDePagos se crea con el objetivo de implementar la pasarela de pagos con la que contará el hotel. Esto hará posible el pago por medio de tarjeta crédito/débito de todos los servicios consumidos por uno o varios huéspedes.

Esta clase cuenta con un método abstracto (validarPago) que es implementado por sus clases hijas (PayPal, PayU y Sire) y que tiene la funcionalidad de verificar que el número de la cuenta bancaria que se le ingrese por parámetro sea válido (es decir, que sea de 10 dígitos) y que, por otro lado, que la cuenta tenga saldo suficiente para hacer efectivo el cobro de algún servicio consumido.

Esta clase es utilizada principalmente desde la aplicación dispuesta para los huéspedes (cuando estos deseen hacer pago inmediato de su reserva) y desde la aplicación dispuesta para los recepcionistas del hotel (cuando hacen el checkout de algún cliente).

## CLASS TARJETA DE PAGO

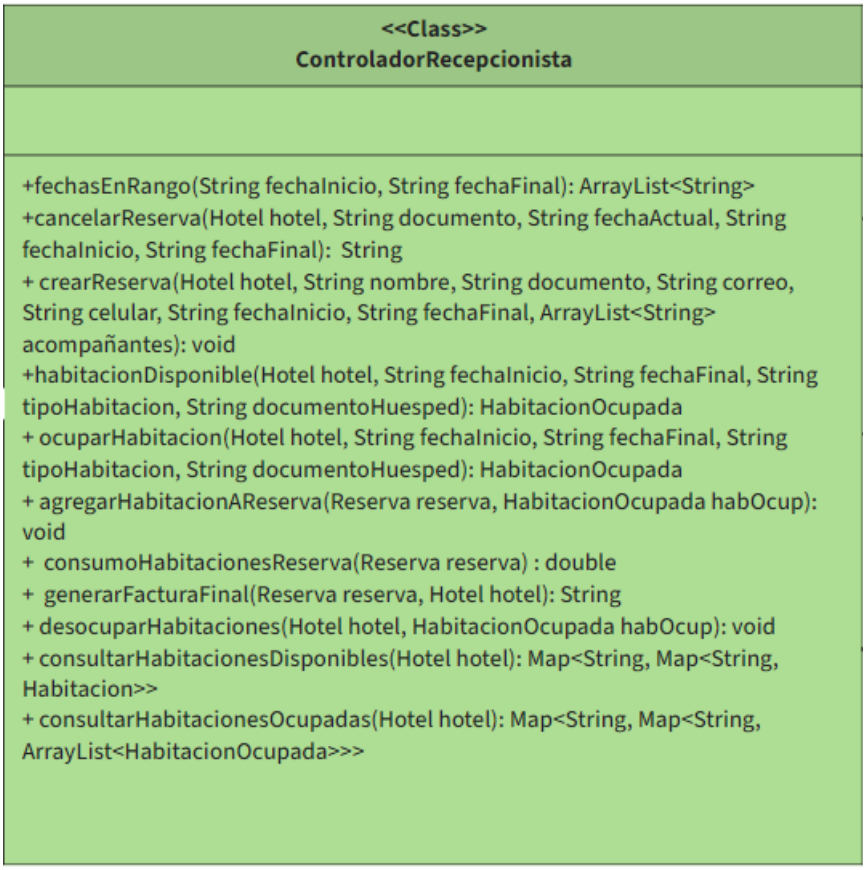



## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se crea con el objetivo de ser la representación de la cuenta bancaria con la que cuenta un huésped. Debido a esto, esta clase tiene atributos que corresponden a la información necesaria para realizar cualquier transacción que sea necesaria dentro del sistema de cobro del hotel (contraseña, número de cuenta y saldo disponible). Así, cuenta con algunos métodos, además del constructor, que nos permiten extraer la información necesaria para hacer posible la autenticación de una cuenta y verificar que cuenta con el saldo suficiente para hacer una determinada transacción.

## CONTROLADOR

## CLASS CONTROLADOR RECEPCIONISTA



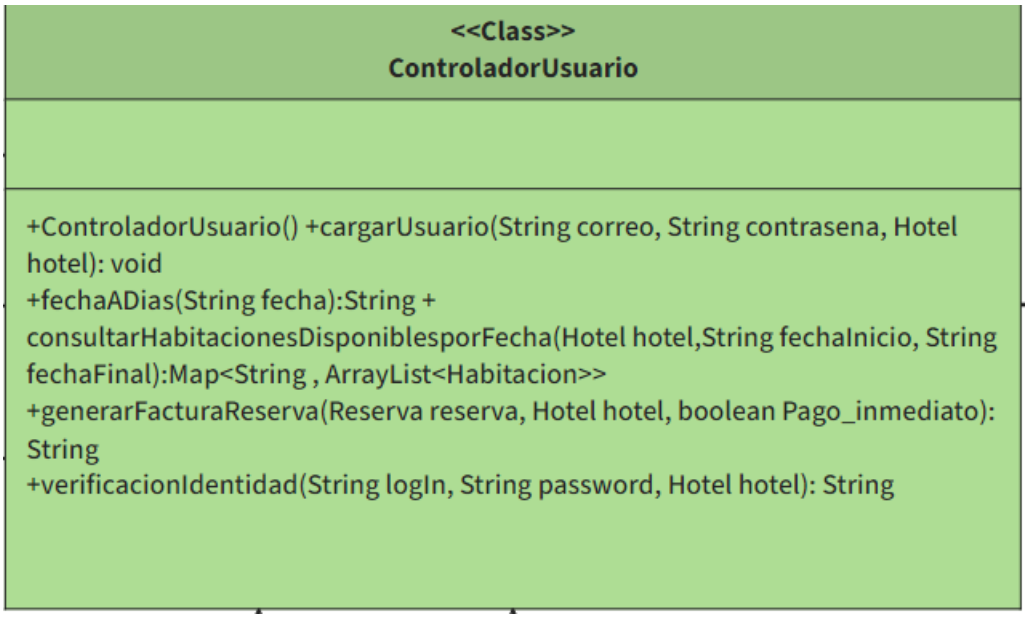

## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de implementar la ejecución de cada una de las opciones con las que cuenta el menú disponible para el recepcionista del hotel. De esta forma, tenemos a los métodos más importantes que nos permiten comprender como se ejecuta cada una de las funcionalidades disponibles para un recepcionista:

- El método fechasEnRango, se encarga a partir de un rango de fechas en formato “DD/MM” determinar el conjunto de fechas que está dentro del mismo. Esto nos servirá, para estalecer las fechas de reserva, disponibilidad y ocupación de las diferentes habitaciones con las que cuenta el hotel.
- El método cancelarReserva, se encarga de cancelar una reserva a partir del documento del huésped responsable de una reserva, una fecha actual (para conocer los días de anticipación con los que se está realizando la cancelación) y de la fecha de inicio y fin de la reserva que se quiere cancelar. Este método, nos permite garantizar, por medio del uso de la fecha actual y la fecha de inicio, que para realizar una cancelación, un huésped debe realizarlo antes de las 48 horas antes de que inicie la reserva.

- El método habitacionDisponible y ocuparHabitación, se encarga de verificar si en un rango de fechas y para un tipo de habitación, existe una habitación disponible para reservar. Si es el caso, entonces ocupará la primera habitación disponible. Sin embargo, aún si no hay habitación disponible, verificará dentro de las habitaciones ocupadas si hay alguna que no se cruce con la fecha dispuesta de reserva, y si no lo hace la reservará y la ocupará, actualizando los datos del huésped (documentoHuesped) responsable al que se le asignará la habitación.
- El método agregarHabitaciónAReserva, se encarga de agregar una habitación a una reserva recibida por parámetro.
- El método generarFacturaFinal, se encarga de contabilizar el consumo por cada habitación perteneciente a una reserva recibida por parámetro (sumando la tarifa por noche y la cuenta pendiente asignada) y así obtener un saldo total, que será reflejado en la factura junto con el nombre del huésped responsable, la fecha de inicio y fin de la reserva, el número de noches y, el número de identificación de cada habitación ocupada.
- El método desocuparHabitaciones, se encarga de desocupar una habitación recibida por parámetro y ponerla nuevamente disponible para su reserva y ocupación.

**CLASS CONTROLADOR USUARIO**

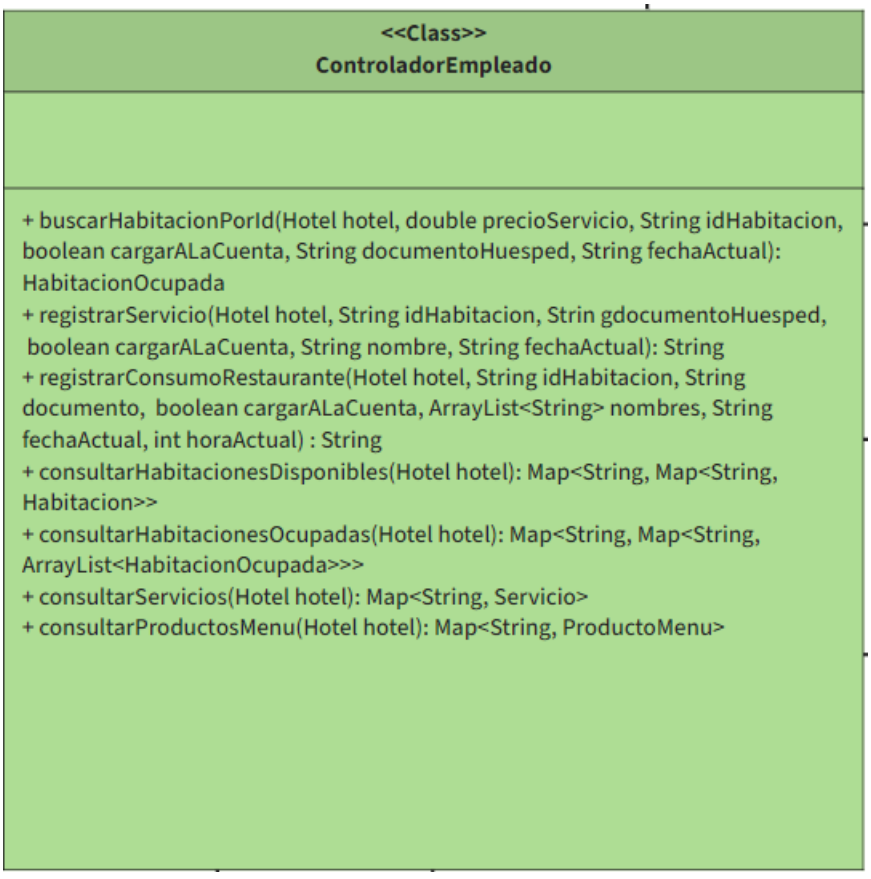



**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase extiende de la clase ControladorRecepcionista (con el fin de reutilizar código, ya que comparten algunas funcionalidades como la de crear reserva) y se establece con el objetivo de implementar la ejecución de cada una de las opciones con las que cuenta el menú disponible para un usuario huésped del hotel. De esta forma, tenemos a los métodos más importantes que nos permiten comprender como se ejecuta cada una de las funcionalidades disponibles para un usuario huésped:

- El método consultarHabitacionesDisponiblesporFecha, se encarga de extraer las habitaciones disponibles para unas fechas ingresadas por parámetro. Esto lo hace por medio de un mapa en el que las llaves son los tipos de habitación y los valores una lista de las habitaciones disponibles según su tipo.
- El método generarFacturaReserva, se encarga de contabilizar el consumo por cada habitación perteneciente a una reserva recibida por parámetro (sumando la tarifa por noche) y así obtener un saldo total, que será reflejado en la factura junto con el nombre del huésped responsable, la fecha de inicio y fin de la reserva, el número de noches y, el número de identificación de cada habitación ocupada. Además, este método cuenta con un atributo booleano (pagoInmediato) que nos permitirá saber si un usuario hizo pago inmediato de su reserva y si es el caso, hacerle un descuento del 10% sobre el valor total de esa reserva.
- El método cargarUsuario, se encarga de crear un Usuario huésped en el caso de que no esté en la base de datos del hotel.

**CLASS CONTROLADOR EMPLEADO**

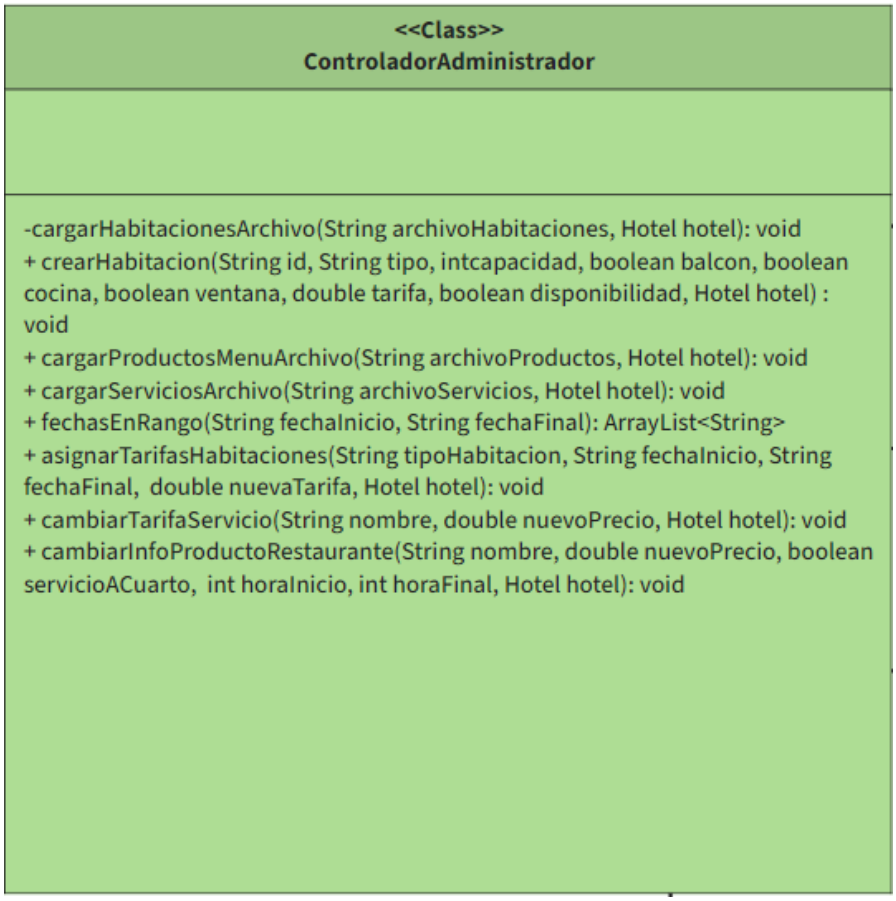


**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de implementar la ejecución de cada una de las opciones con las que cuenta el menú disponible para un empleado del hotel. De esta forma, tenemos a los métodos más importantes que nos permiten comprender como se ejecuta cada una de las funcionalidades disponibles para un recepcionista:

- El método registrarServicio junto con el método buscarHabitacionPorId, se encarga de registrar un servicio disponible en el hotel, a una habitación que, por medio del método buscarHabitacionPorId, se encarga de buscarla, y si es el caso que el huésped no quiera pagar de forma inmediata el servicio consumido, se lo cargará a una cuenta pendiente. Por otro lado, así el pago se haga de forma inmediata o no, se le generará una factura al huésped/des en donde se registra la información pertinente del servicio consumido.
- El método registrarConsumoRestaurante junto con el método buscarHabitacionPorId, se encarga de registrar el consumo de algún producto de Restaurante, a una habitación que, por medio del método buscarHabitacionPorId, nos permite buscarla, y si es el caso que el huésped no quiera pagar de forma inmediata el servicio consumido, se lo cargará a una cuenta pendiente. Así mismo, para el registro del consumo, contabilizará, si el producto está dentro de la hora de disponibilidad, el precio de cada producto consumido (ProductoMenu) y por último, así el pago se haga de forma inmediata o no, se le generará una factura al huésped/des en donde se registra la información pertinente de cada producto consumido y del precio total del servicio.
- El método consultarHabitacionesDisponibles se encarga de retornar el mapa con la información de las habitaciones disponibles en el hotel.
- El método consultarHabitacionesOcupadas se encarga de retornar el mapa con la información de las habitaciones ocupadas en el hotel.
- El método consultarServicios se encarga de retornar el mapa con la información de servicios disponibles en el hotel.
- El método consultarProductosMenu se encarga de retornar el mapa con la información de los productosMenú disponibles en el restaurante del hotel.

## CLASS CONTROLADOR ADMINISTRADOR




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de implementar la ejecución de cada una de las opciones con las que cuenta el menú disponible para el administrador del hotel. De esta forma, tenemos a los métodos más importantes que nos permiten comprender como se ejecuta cada una de las funcionalidades disponibles para un administrador:

- El método cargarHabitacionesArchivo se encarga de leer un archivo .txt en el que se almacena la información de las habitaciones. Es así como, cada línea del archivo constituye una habitación que pasará a ser un objeto de la clase habitación y a hacer parte del mapa de habitaciones disponibles. El archivo .txt está compuesto de esta forma:

  Id; tipo de habitación (s: suite, e: estándar, sd: suite doble); capacidad; balcón (boolean); ventana (boolean); cocina (boolean); tarifa; disponibilidad (boolean)

- El método crearHabitacion se encarga de crear una habitación una por una de forma directa sin necesidad de leer un archivo.

- El método cargarProductosMenuArchivo se encarga de leer un archivo .txt en el que se almacena la información de las productos del menú del restaurante del hotel. Es así como, cada línea del archivo constituye un producto del menú que pasará a ser un objeto de la clase ProductoMenu y a hacer parte del mapa de menú. El archivo .txt está compuesto de esta forma:

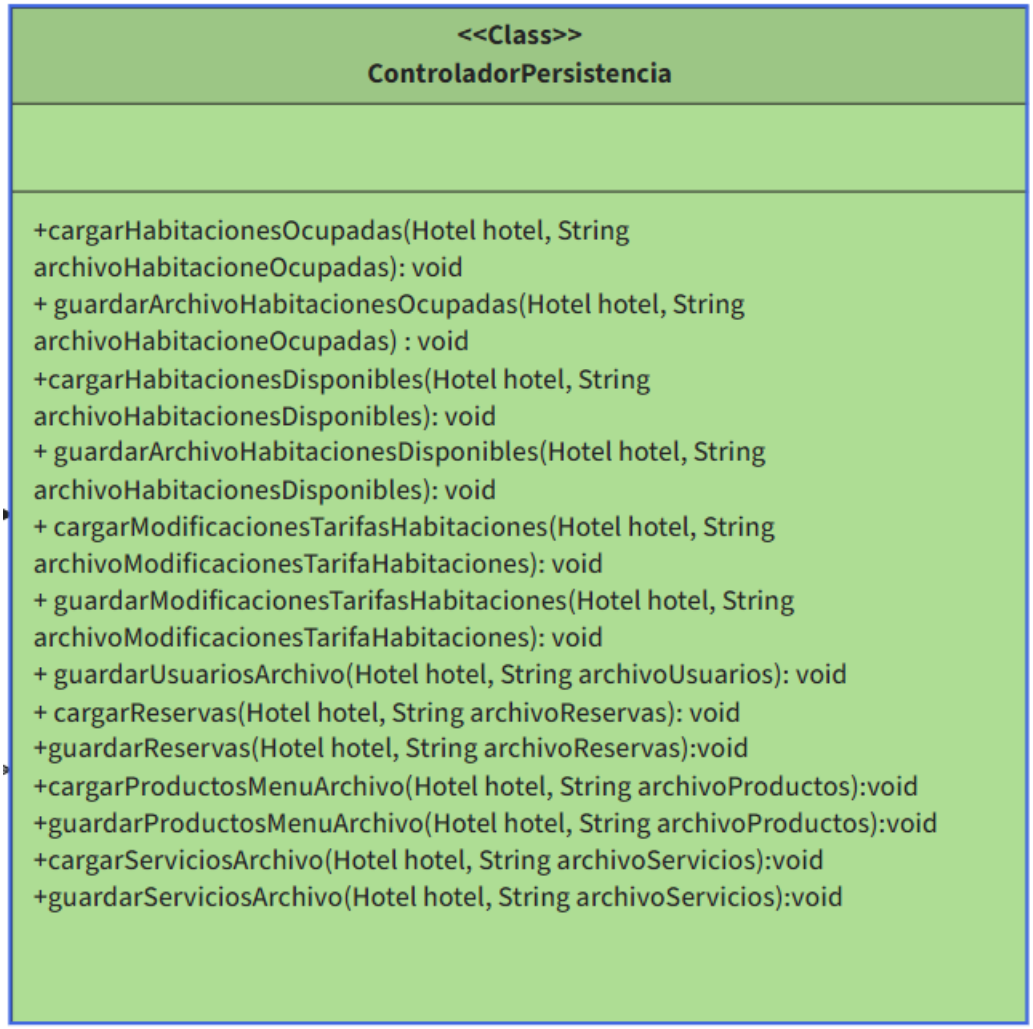
  nombre; costo; capacidad; descripción; servicio al Cuarto (boolean); horaInicio; horaFin

- El método cargarServiciosArchivo se encarga de leer un archivo .txt en el que se almacena la información de los servicios disponibles en el hotel. Es así como, cada línea del archivo constituye un servicio que pasará a ser un objeto de la clase Servicio y a hacer parte del mapa de servicios. El archivo .txt está compuesto de esta forma:

  nombre; costo; grupo (boolean); descripción

- El método asignarTarifasHabitaciones se encarga actualizar la tarifa (ingresada por parámetro) de las habitaciones correspondientes a un tipo de habitación (ingresado por parámetro) durante un rango de fechas (ingresadas por parámetro)
- El método cambiarTarifaServicio se encarga de actualizar el precio (ingresado por parámetro) de un servicio ingresado por parámetro, que es buscado por medio de algunos get.
- El método cambiarInfoProductoRestaurante se encarga de actualizar la información de un productoMenu ingresado por parámetro.

## CLASS CONTROLADOR PERSISTENCIA




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de implementar la ejecución de cada una de las funcionalidades relacionadas a la persistencia. De esta forma, los métodos que posee esta clase se encargan de cargar la información de archivos .txt (por medio de su lectura) a los diferentes Maps y ArrayList () que tienen la funcionalidad de almacenar la información relacionada a:

- Las habitaciones ocupadas.
- Las habitaciones disponibles.
- Las modificaciones de tarifas de las habitaciones.
- Los usuarios huésped del hotel.
- Las reservas.
- Los productos del menú del restaurante.
- Los servicios del hotel.

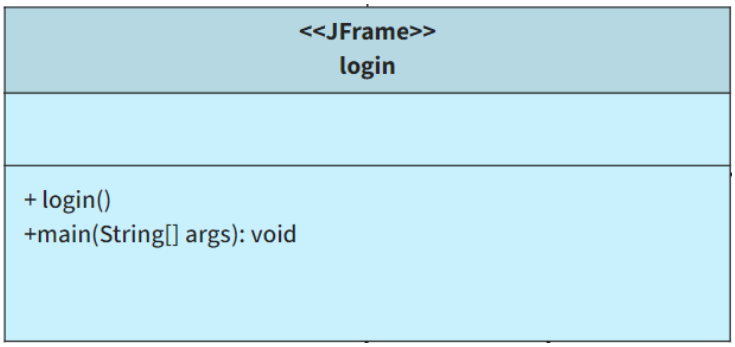
Así mismo, algunos de estos métodos se encargan de cargar la información que fue añadida o modificada en los diferentes Maps y ArrayList() (que contienen toda la información administrativa y operativa del hotel) a los archivos .txt (por medio de su escritura) correspondientes según el tipo de datos a almacenar.

Los métodos cargar y guardar correspondientes entre sí, utilizan el mismo archivo para guardar y extraer la información necesaria.

Finalmente, esta clase nos permite garantizar la persistencia de la aplicación y que de esta manera no existan alteraciones en la información que permite la correcta operación del sistema y del hotel.

## CONSOLA GRÁFICA

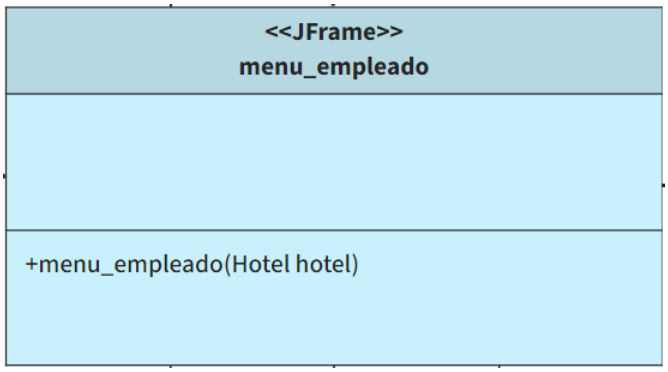
## CLASS LOGIN




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JFrame (Ventana) principal con la que interactúan todos los usuarios del sistema y por medio de la cual son redirigidos a aplicaciones, que se adaptan a las funcionalidades disponibles para cada usuario dependiendo su cargo. En esta ventana podrá encontrar dos campos de texto, en donde ingresará su correo y contraseña, que será a partir de lo que se podrá autenticar su ingreso.

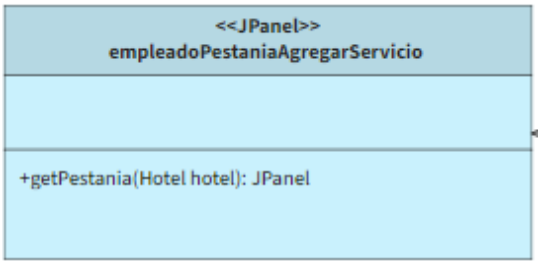
## CLASS MENU_EMPLEADO




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JFrame (Ventana) principal con la que interactúan todos los empleados del hotel y por medio de la cual tendrán acceso a todas las funcionalidades disponibles para su cargo. En esta ventana podrán encontrar un JButton correspondiente a cada funcionalidad disponible para un empleado. De esta forma, este JFrame también actuará como un ActionListener al desplegar, según el botón seleccionado, el JPanel correspondiente a la funcionalidad seleccionada.

## CLASS EMPLEADOPESTANIAAGREGARSERVICIO




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

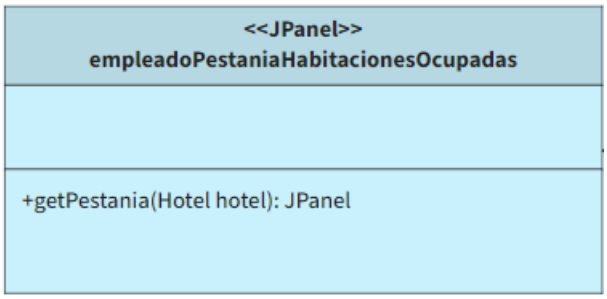
Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los empleados del hotel y por medio de la cual podrán ejecutar la función de agregar un Servicio consumido a la cuenta de un cliente/habitación del hotel. En esta ventana se podrán encontrar algunos JRadioButton, JLabel, JButton y JTextField, que le permitirán a un empleado ingresar la información necesaria para el cobro y registro de un Servicio consumido. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

## CLASS EMPLEADOPESTANIAHABITACIONESDISPONIBLES


## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los empleados del hotel y por medio de la cual podrán ejecutar la función de consultar habitaciones disponibles según el tipo de habitación. En esta ventana se podrán encontrar algunos JButton y JLabel, que le permitirán a un empleado seleccionar el tipo de habitación y mostrar la información correspondiente a las habitaciones disponibles. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

## CLASS EMPLEADOPESTANIAHABITACIONESOCUPADAS




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los empleados del hotel y por medio de la cual podrán ejecutar la función de consultar habitaciones ocupadas según el tipo de habitación. En esta ventana se podrán encontrar algunos JButton y JLabel, que le permitirán a un empleado seleccionar el tipo de habitación y observar la información correspondiente a las habitaciones disponibles. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

## CLASS EMPLEADOPESTANIAPRODUCTOSMENU


## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

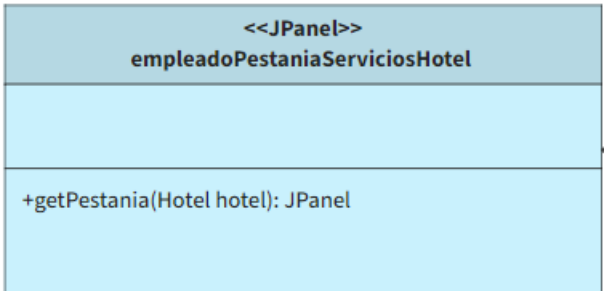
Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los empleados del hotel y por medio de la cual podrán ejecutar la función de consultar los productos del menú del restaurante. En esta ventana se podrán encontrar algunos JButton, JLabel y JComboBox, que le permitirán a un empleado seleccionar un producto menú de una lista de productos y observar su información correspondiente. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

## CLASS EMPLEADOPESTANIAREGISTRARCONSUMO


## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los empleados del hotel y por medio de la cual podrán ejecutar la función de registrar el consumo de alguno de productos del menú del restaurante. En esta ventana se podrán encontrar algunos JButton, JLabel y JComboBox, que le permitirán a un empleado seleccionar un producto menú de una lista de producto y recibir la información necesaria para registrar el cobro por el consumo. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

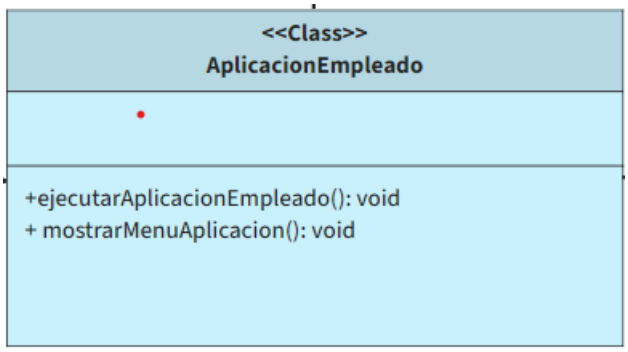
## CLASS EMPLEADOPESTANIASERVICIOSHOTEL




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los empleados del hotel y por medio de la cual podrán ejecutar la función de consultar los servicios del hotel. En esta ventana se podrán encontrar algunos JButton, JLabel y JComboBox, que le permitirán a un empleado seleccionar un servicio de una lista de servicios y observar su información correspondiente. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

## CLASS APLICACIÓN EMPLEADO




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear la interfaz principal con la que interactúan todos los empleados del hotel y por medio de la cual se muestran las opciones que se adaptan a las funcionalidades disponibles para su cargo. De esta forma, los métodos más importantes de la clase controlador Empleado que se relacionan con cada opción son:

**Menú de opciones**

```
"1. Registrar servicio"
"2. Registrar consumo en restaurante"
"3. Consultar habitaciones disponibles"
"4. Consultar habitaciones ocupadas"
"5. Consultar servicios del hotel"
"6. Consultar productos del menú"
"7. Cerrar aplicación empleado"
```

Opción 1:

- controlador.registrarServicio()

Opción 2: Adicionalmente, en esta opción se le muestra una serie de opciones sobre los productos que se desean consumir del menú ofrecido en el restaurante.

- controlador.registrarConsumoRestaurante()

Opción 3:

- controlador.consultarHabitacionesDisponibles()

Opción 4:

- controlador.consultarHabitacionesOcupadas()

Opción 5:

- controlador.consultarServicios ()

Opción 6:

- controlador.consultarProductosMenu()

**CLASS MENU_RECEPCIONISTA**


**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

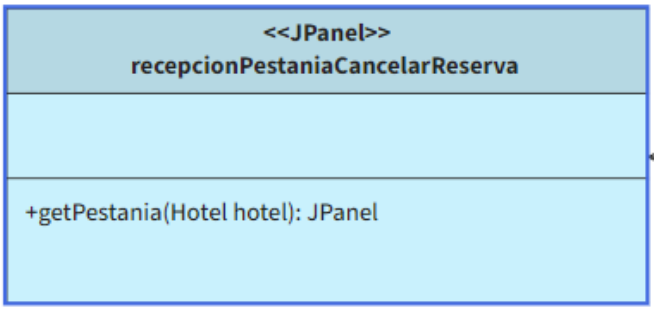
Esta clase se establece con el objetivo de crear el JFrame (Ventana) principal con la que interactúan los recepcionistas del hotel y por medio de la cual tendrán acceso a todas las funcionalidades disponibles para su cargo. En esta ventana podrán encontrar un JBoton correspondiente a cada funcionalidad disponible para un recepcionista. De esta forma, este JFrame también actuará como un ActionListener al desplegar, según el botón seleccionado, el Jpanel correspondiente a la funcionalidad seleccionada.

**CLASS RECEPCIONPESTANIACAMBIARINFOTARJETA**


**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los recepcionistas del hotel y por medio de la cual podrán ejecutar la función de actualizar la información de la tarjeta bancaria de alguno de los huéspedes del hotel. Estos cambios, son principalmente del saldo de la cuenta (que no se actualiza automáticamente) y del número de la tarjeta (que puedes ser inválido por haber, posiblemente, sido ingresado erróneamente). En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un recepcionista ingresar la información necesaria para actualizar la información de la tarjeta correspondiente. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS RECEPCIONPESTANIACANCELARRESERVA**

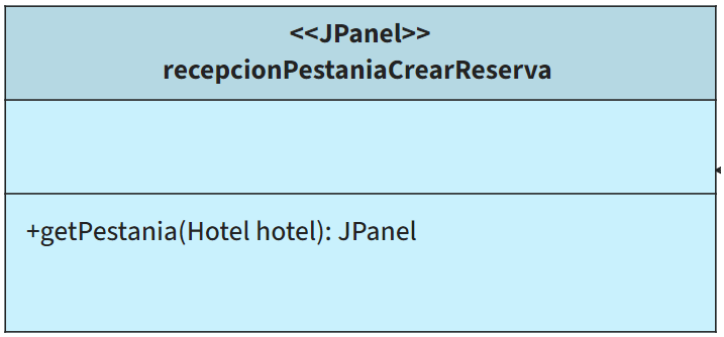



**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los recepcionistas del hotel y por medio de la cual podrán ejecutar la función de cancelar una reserva. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un recepcionista ingresar la información necesaria de una reserva para

cancelarla. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

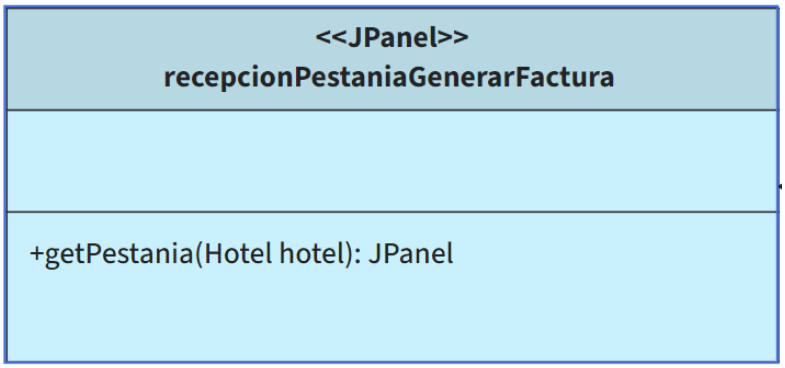
## CLASS RECEPCIONPESTANIACREARRESERVA




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los recepcionistas del hotel y por medio de la cual podrán ejecutar la función de crear una reserva. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un recepcionista ingresar la información necesaria de una reserva para crearla. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

## CLASS RECEPCIONPESTANIAGENERARFACTURA

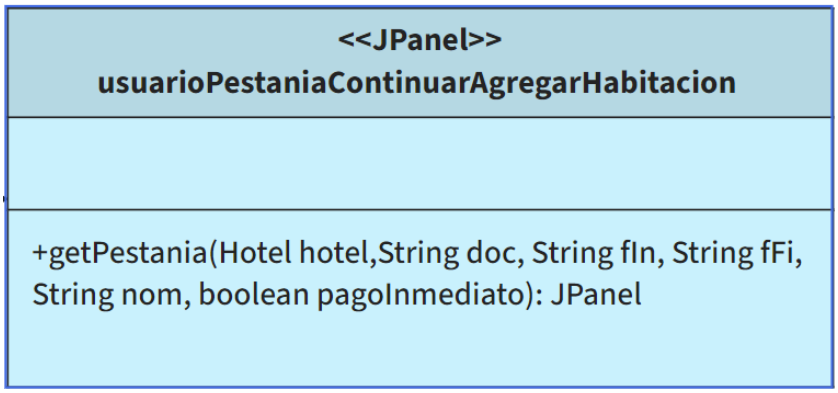



## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los recepcionistas del hotel y por medio de la cual podrán ejecutar la función de generar la factura correspondiente al consumo de algún servicio o producto. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un recepcionista

ingresar la información necesaria de los clientes para generar la factura. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

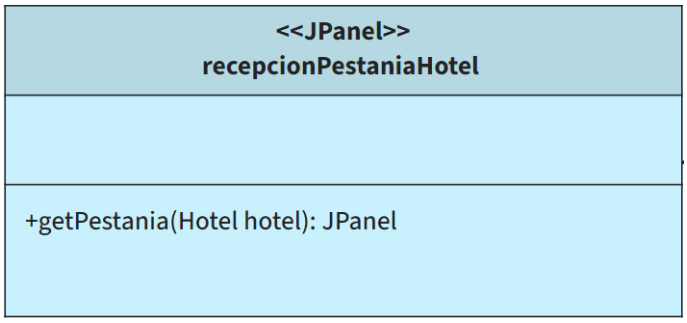
**CLASS RECEPCIONPESTANIACONTINUARAGREGARHABITACION**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los recepcionistas del hotel y por medio de la cual podrán ejecutar la función de agregar habitaciones a una reserva en creación, a la vez que también les permite finalizarla. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un recepcionista ingresar la información necesaria para agregar habitaciones según su capacidad y tipo. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

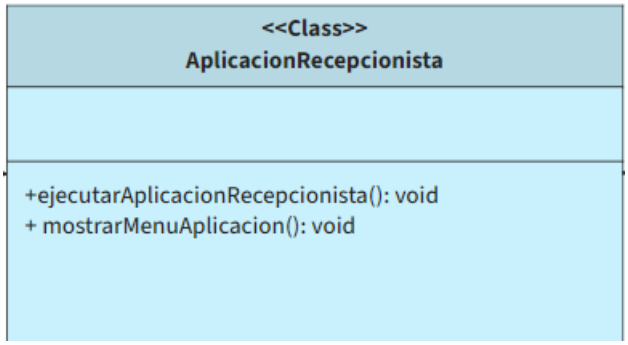
**CLASS RECEPCIONPESTANIAHOTEL**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los recepcionistas del hotel y por medio de la cual podrán observar la ocupación del hotel haciendo uso de un calendario presente el Panel.

## CLASS APLICACIÓN RECEPCIONISTA




## RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE

Esta clase se establece con el objetivo de crear la interfaz principal con la que interactúan todos los recepcionistas del hotel y por medio de la cual se muestran las opciones que se adaptan a las funcionalidades disponibles para su cargo. De esta forma, los métodos más importantes de la clase controlador Recepcionista que se relacionan con cada opción son:

**Menú de opciones**

```
"1. Crear reserva"
"2. Hacer check-out”
"3. Cancelar reserva"
"4. Consultar habitaciones disponibles"
"5. Consultar habitaciones ocupadas"
"6. Cerrar aplicación recepcionista"
```

Opción 1:

- controlador.crearReserva()

Opción 2:

- controlador.generarFacturaFinal()
- desocuparHabitaciones()

Opción 3:

- controlador.cancelarReserva()

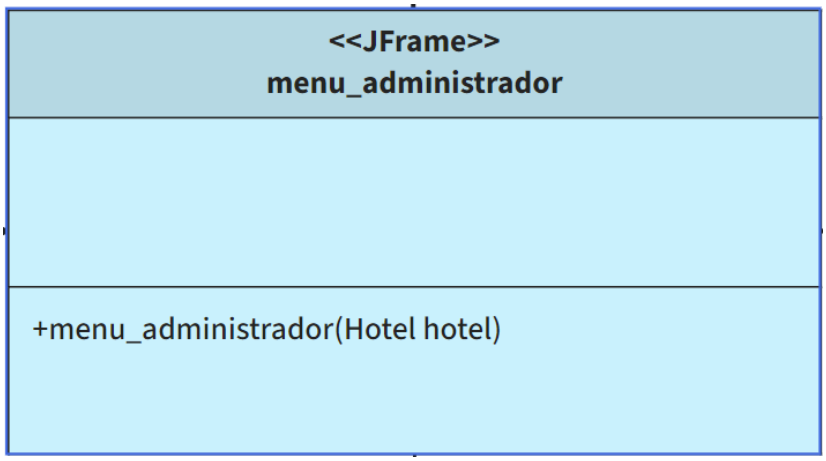
Opción 4:

- controlador.consultarHabitacionesDisponibles()

Opción 5:

- controlador. consultarHabitacionesOcupadas()

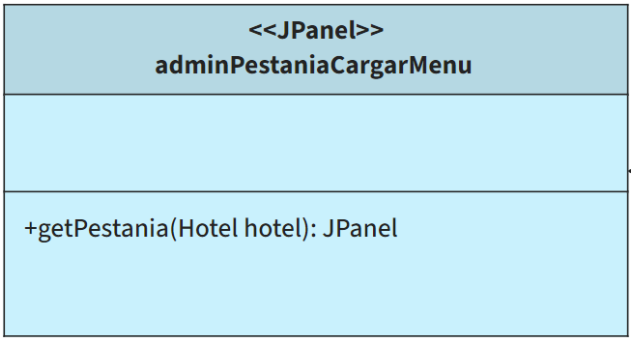
**CLASS MENU_ADMINISTRADOR**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JFrame (Ventana) principal con la que interactúa el administrador del hotel y por medio de la cual tendrán acceso a todas las funcionalidades disponibles para su cargo. En esta ventana podrá encontrar un JBoton correspondiente a cada funcionalidad disponible para un administrador. De esta forma, este JFrame también actuará como un ActionListener al desplegar, según el botón seleccionado, el Jpanel correspondiente a la funcionalidad seleccionada

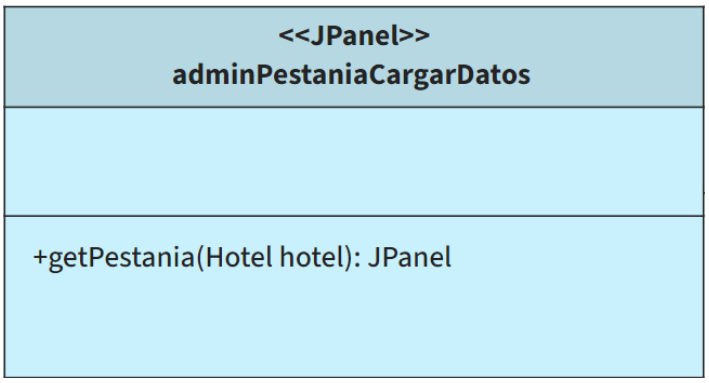
**CLASS ADMINPESTANIACARGARMENU**



**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de cargar el menú (productos menú) del restaurante del hotel. En esta ventana se podrán encontrar algunos JButton, JLabel y JRadioButton, que le permitirán al administrador seleccionar si desea o no cargar el archivo .txt que contiene toda la información de cada uno de los productos que compone el menú. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

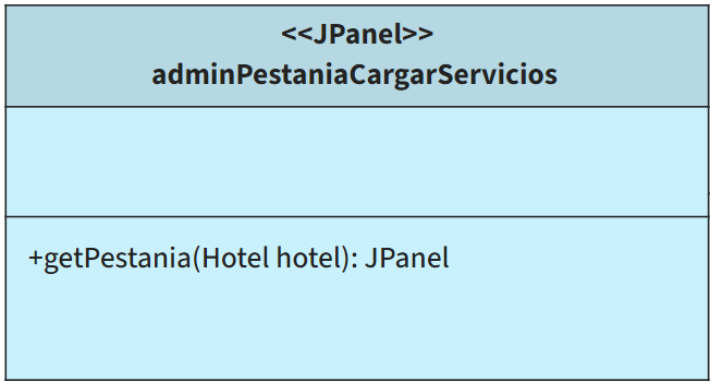
**CLASS ADMINPESTANIACARGARDATOS**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de cargar los datos de los usuarios del sistema del hotel. En esta ventana se podrán encontrar algunos JButton, JLabel y JRadioButton, que le permitirán al administrador seleccionar si desea o no cargar el archivo .txt que contiene toda la información de cada uno de los usuarios (empleados) del sistema del hotel. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

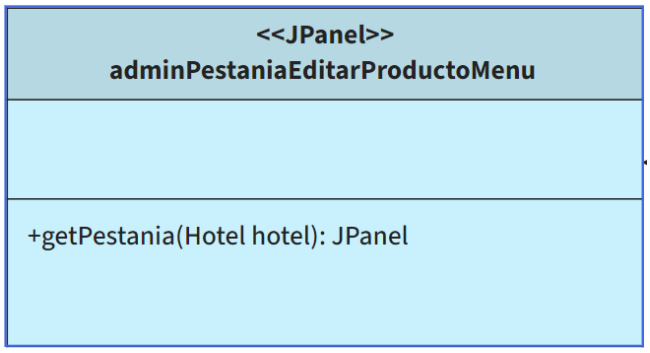
**CLASS ADMINPESTANIACARGARSERVICIOS**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de cargar cada uno de los servicios del hotel. En esta ventana se podrán encontrar algunos JButton, JLabel y JRadioButton, que le permitirán al administrador seleccionar si desea o no cargar el archivo .txt que contiene toda la información de cada uno de los servicios disponibles para el Hotel. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS ADMINPESTANIAEDITARPRODUCTOMENU**

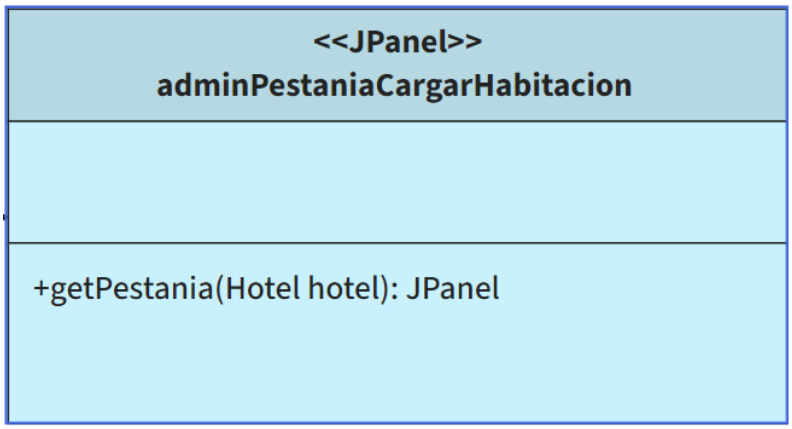



**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de modificar las características de un producto menú seleccionado. En esta ventana se podrán encontrar algunos JButton, JLabel, JTextField, JComboBox y JRadioButton, que le permitirán al administrador modificar manualmente las características de un producto menú seleccionado. De esta forma, este JPanel también actuará como un ActionListener al

reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

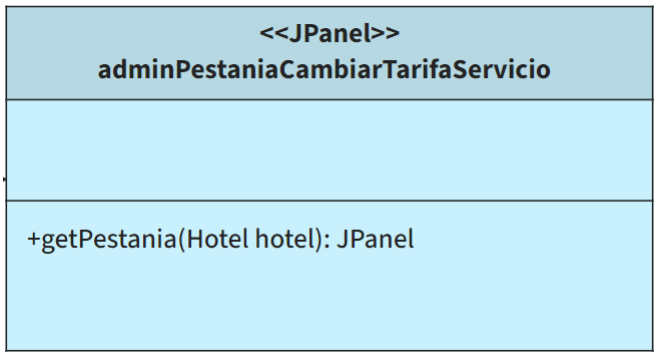
**CLASS ADMINPESTANIACARGARHABITACION**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de crear una habitación, modificando cada una sus características. En esta ventana se podrán encontrar algunos JButton, JLabel, JTextField y JRadioButton, que le permitirán al administrador modificar manualmente las características de la habitación que quiere crear. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS ADMINPESTANIACAMBIARTARIFASERVICIO**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de cambiar la tarifa a un servicio seleccionado. En esta ventana se podrán encontrar algunos JButton, JLabel y JComboBox, que le permitirán al administrador modificar manualmente la tarifa de un servicio seleccionado. De esta forma, este JPanel también actuará como un ActionListener

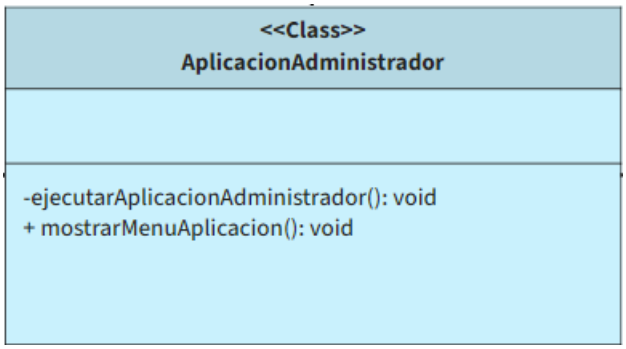
al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS ADMINPESTANIACAMBIARTARIFAHABITACIÓNFECHA**


**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúa el administrador y por medio de la cual podrá ejecutar la función de cambiar la tarifa de un tipo de habitación seleccionado para una rango de fechas determinado. En esta ventana se podrán encontrar algunos JButton, JLabel, JRadioButton y JComboBox, que le permitirán al administrador seleccionar el tipo de habitación, para posteriormente modificar manualmente su tarifa, la cual será aplicada durante el rango de fechas que también se ingresa por medio de los JTextField dispuestos en el JPanel. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS APLICACIÓN ADMINISTRADOR**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear la interfaz principal con la que interactúa el administrador y por medio de la cual se muestran las opciones que se adaptan a las funcionalidades disponibles para su cargo. De esta forma, los métodos más importantes de la clase controlador Recepcionista que se relacionan con cada opción son:

**Menú de opciones**

```
"1. Cargar habitaciones"
"2. Crear habitación"
"3. Cargar productos del menú"
"4. Cargar servicios del hotel"
"5. Asignar tarifa por tipo de habitación en rango de fechas"
"6. Cambiar tarifa de un servicio"
"7. Cambiar información de un plato/bebida del menú"
"8. Cerrar aplicación administrador"
```

Opción 1:

- controlador.cargarHabitacionesArchivo()

Opción 2:

- controlador.crearHabitacion()

Opción 3:

- controlador.cargarProductosMenuArchivo()

Opción 4:

- controlador.cargarServiciosArchivo()

Opción 5:

- controlador.asignarTarifasHabitaciones()

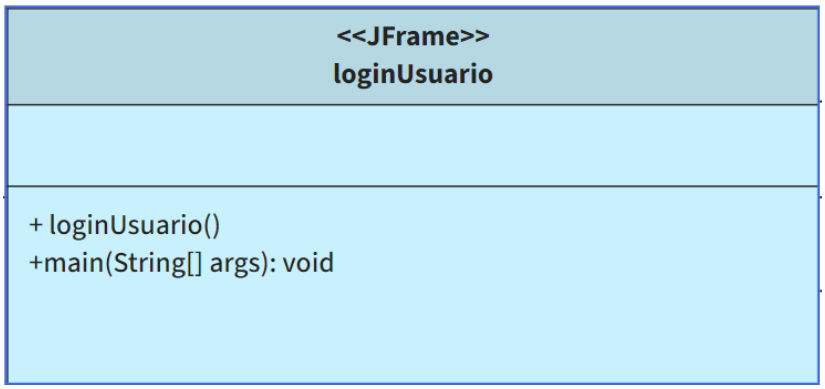
Opción 6:

- controlador. cambiarTarifaServicio()

Opción 7:

- controlador.cambiarInfoProductoRestaurante()

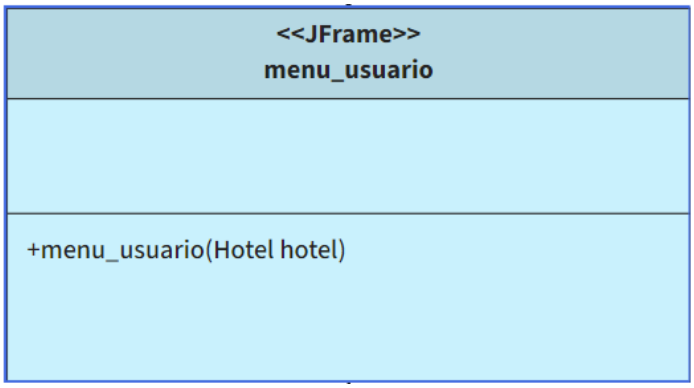
**CLASS LOGINUSUARIO**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JFrame (Ventana) principal con la que interactúan todos los huéspedes del hotel y por medio de la cual son redirigidos a un menú principal, que se adapta a las funcionalidades disponibles para los huéspedes del hotel. En esta ventana podrá encontrar dos campos de texto, en donde ingresará su correo y contraseña, que será a partir de lo que se podrá autenticar su ingreso.

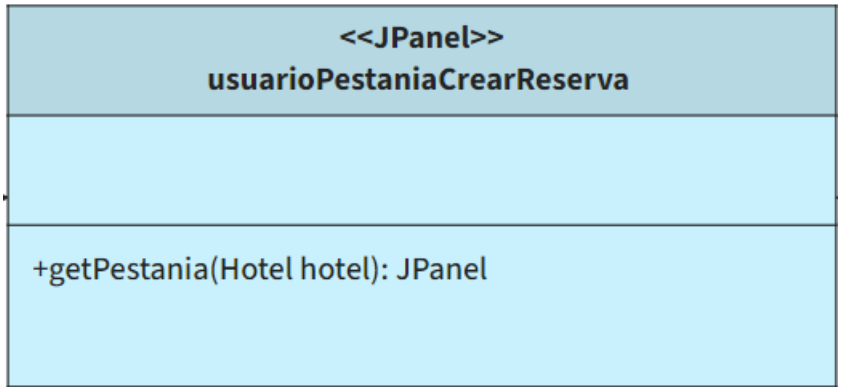
**CLASS MENU_USUARIO**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JFrame (Ventana) principal con la que interactúan los huéspedes del hotel y por medio de la cual tendrán acceso a todas las funcionalidades disponibles para ellos. En esta ventana podrán encontrar un JBoton correspondiente a cada funcionalidad disponible para un huésped. De esta forma, este JFrame también actuará como un ActionListener al desplegar, según el botón seleccionado, el Jpanel correspondiente a la funcionalidad seleccionada.

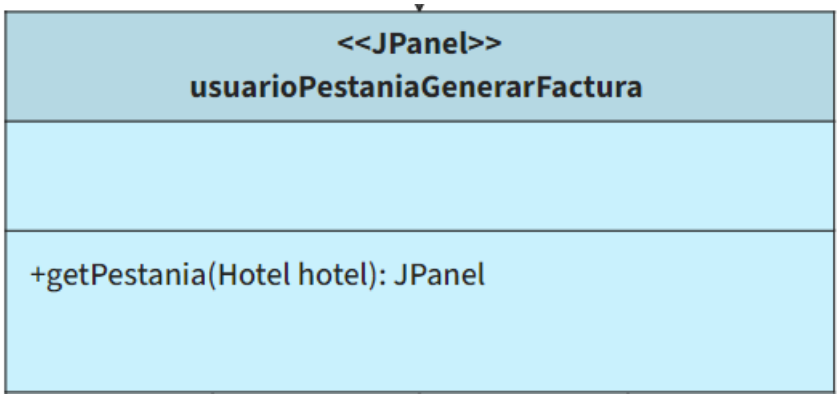
**CLASS USUARIOPESTANIACREARRESERVA**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los huéspedes del hotel y por medio de la cual podrán ejecutar la función de crear una reserva. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un huésped ingresar la información necesaria de una reserva para crearla. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS USUARIOPESTANIAGENERARFACTURA**

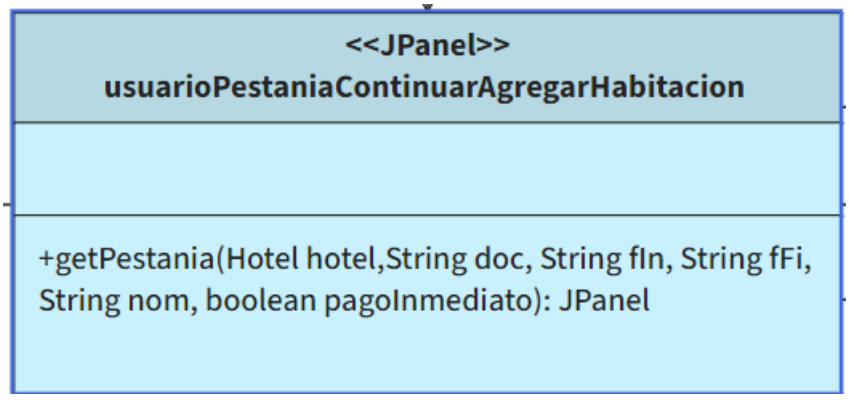



**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los huéspedes del hotel y por medio de la cual podrán ejecutar la función de generar la factura correspondiente a la reserva, si es el caso de que hicieron pago inmediato por tarjeta crédito. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un huésped ingresar la información necesaria para generar la factura. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes

componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

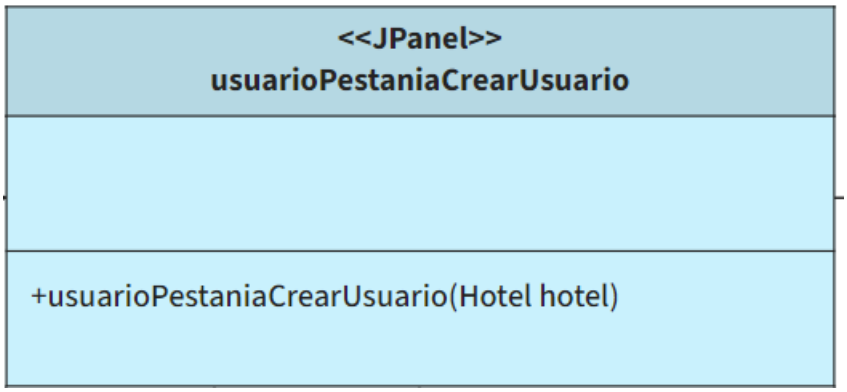
**CLASS USUARIOPESTANIACONTINUARAGREGARHABITACION**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los huéspedes del hotel y por medio de la cual podrán ejecutar la función de agregar habitaciones a una reserva en creación, a la vez que también les permite finalizarla. En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un huésped ingresar la información necesaria para agregar habitaciones según su capacidad y tipo. Igualmente, esta clase tiene un método getPestania () que tiene un parámetro de tipo booleano (pagoInmediato) que si es true, permite redirigir al húesped, cuando ya no quiera agregar más habitaciones, a un nuevo espacio dedicado a la pasarela de pago con la que cuenta el hotel. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS USUARIOPESTANIACREARUSUARIO**

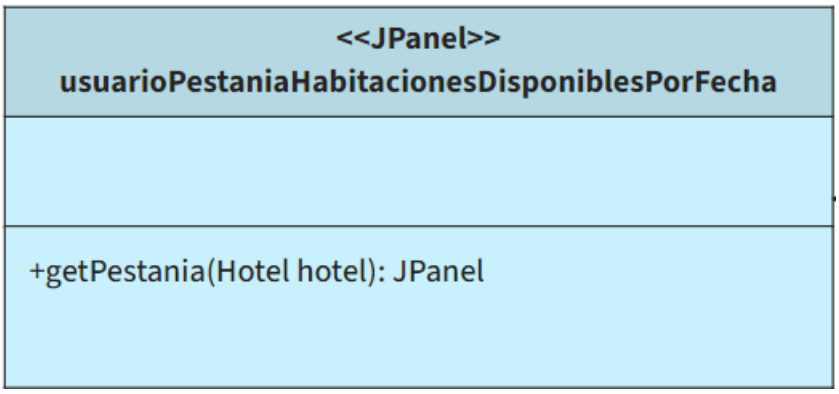



**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JFrame (ventana) con el que interactúan todos los huéspedes del hotel y por medio de la cual podrán registrarse en el caso de que no

lo estuvieran. En esta ventana podrá encontrar dos campos de texto, en donde ingresarán su correo y contraseña de registro, que será a partir de lo que se podrá autenticar su ingreso posteriormente cuando inicien sesión.

**CLASS USUARIO_PESTANIA_HABITACIONES_DISPONIBLES_POR_FECHA**




**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los huéspedes del hotel y por medio de la cual podrán ejecutar la función de consultar habitaciones disponibles según el tipo de habitación y para unas fechas determinadas ingresadas por medio de 2 JTextField correspondientes. Además, en esta ventana se podrán encontrar algunos JButton y JLabel, que le permitirán a un huéspede seleccionar el tipo de habitación y mostrar la información correspondiente a las habitaciones disponibles. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**CLASS USUARIOPESTANIACAMBIARINFOTARJETA**


**RESPONSABILIDAD Y JUSTIFICACIÓN DE CREACIÓN DE LA CLASE**

Esta clase se establece con el objetivo de crear el JPanel con el que interactúan todos los huéspedes del hotel y por medio de la cual podrán ejecutar la función de actualizar la información de si tarjeta bancaria. Estos cambios, son principalmente del saldo de la cuenta (que no se actualiza automáticamente) y del número de la tarjeta (que puedes ser inválido por haber, posiblemente, sido ingresado erróneamente). En esta ventana se podrán encontrar algunos JButton, JLabel y JTextField, que le permitirán a un huésped ingresar la información necesaria para actualizar la información de la tarjeta correspondiente. De esta forma, este JPanel también actuará como un ActionListener al reaccionar a los diferentes componentes del Panel, permitiendo conectar la información recibida con la implementación lógica implementada para cada funcionalidad.

**PRUEBAS REALIZADAS:**

Se realizaron dos grupos de pruebas: una enfocada en la carga de datos que se encuentra en la clase tipo JUnit Test Case llamada “test_carga” y la otra enfocada en la creación de reservas que se encuentra en la clase tipo JUnit Test Case llamada “test_reservas”.

En primer lugar, con respecto a las pruebas de carga de datos, se hicieron 25 tests individuales. Se hicieron 6 pruebas de funcionalidades para verificar que, de manera transversal, las funcionalidades de carga funcionan de manera conjunta, estas son las 6 primeras pruebas hechas a los controladores. Las siguientes 6 pruebas son para verificar que se manejan bien las “`FileNotFoundException` ” en los casos en los que se escriba mal una dirección de memoria en las funciones de carga. Posteriormente, se hicieron 3 pruebas para probar la carga de Habitaciones Ocupadas, donde cada prueba determina si se cargó una característica especifica de las habitaciones ocupadas en relación con el código implementado, pruebas de caja blanca. De manera similar, se hicieron 3 pruebas para probar la carga de Habitaciones Disponibles, donde cada prueba determina si se cargó una característica especifica de las habitaciones disponibles en relación con el código implementado, pruebas de caja blanca. De mismo modo, se hicieron 3 pruebas para verificar la carga del menú, teniendo en cuenta su implementación de código, pruebas de caja blanca. Finalmente, se hicieron 4 pruebas a la carga de servicios verificando la funcionalidad de los métodos implementados para estas.

Ahora bien, con respecto a las pruebas realizadas a la creación de reservas se realizaron 8 pruebas. Las primeras 4 pruebas son transversales a las funcionalidades propias de la carga. Las siguientes 4 pruebas referentes a métodos y características de métodos específicos de la implementación dada, pruebas de caja blanca.

**INSTRUCCIONES ADICIONALES PARA LA ADECUADA EJECUCIÓN DE LA APLICACIÓN**

- Para salvaguardar toda la información agregada y modificada de la base de datos del hotel, durante la ejecución de al aplicación, se sugiere que se seleccione el botón LogOut presente en cada menú de los usuarios, para así hacer efectiva la persistencia de la aplicación y que no haya ninguna alteración de los cambios realizados.

## REPORTES Y GRÁFICAS

Para la visualización de los reportes y gráficas dispuestas dentro de la sistema del hotel, estas van a estar dispuestas dentro de la interfaz de la aplicación del usuario Administrador. De esta forma, por medio de JButton podrán obtener información respecto a donde podrá encontrar la gráfica correspondiente a una estadística específica del hotel.

Las estadísticas consideradas y para las cuales habrá un JButton correspondiente son:

- Las ventas del restaurante.
- La suma total de las facturas generadas por el sistema a lo largo del tiempo.
- La relación tarifaHabitacion/consumoRestaurante.
- Los servicios ofrecidos (relación servicio/ventas del mismo)
- La relación tarifaHabitacion/consumoServicios.