

Proyecto 1 - Documento de diseño Property Management System

Marco Alejandro Ramírez - Daniela Torres

June 4, 2023

1 Contexto del problema

Antes de iniciar con el proceso de diseño se definen las funcionalidades de alto nivel que el juego debe estar en capacidad de satisfacer a la hora de interactuar con la interfaz. Lo anterior se realiza con el propósito de que, independiente de la implementación de la interfaz, sea factible desplegar visualmente el juego correctamente. La figura 1 ilustra las interacciones de forma genérica, así como los archivos de texto que describen los registros (Check-in y Check-out), tarifas, reservas, menú y la base de datos con los datos de login de los empleados y sus respectivos roles.

Luego de añadir más requerimientos al proyecto aumentamos el gráfico con 4 bases de datos adicionales (Las resaltadas en color lila), una para almacenar las facturas, otra para la información de los huespedes en la HuespedApp, BankingData para almacenar de forma organizada la información de cada uno de los bancos que soporta la aplicación y los indicadores, los cuales son 3 gráficas dinámicas y un archivo de texto que tiene reportes acerca de los huespedes del hotel.

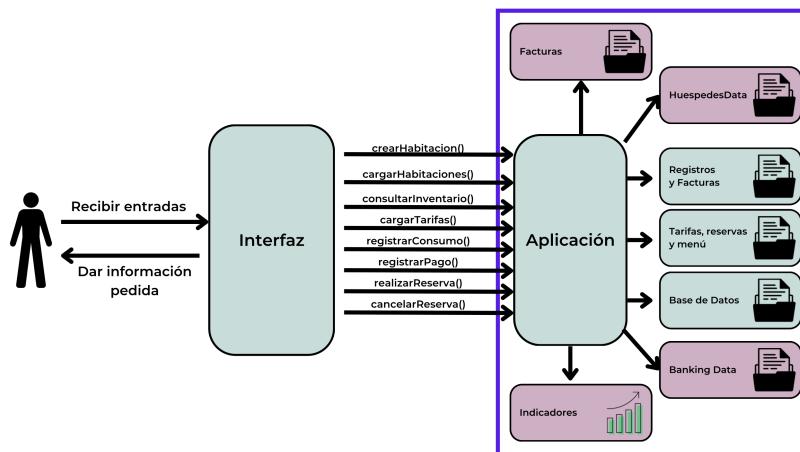


Figure 1: Definición del contexto del problema

En cuanto a la interacción con la interfaz, el usuario (empleado) solo puede ver las opciones acordes a su empleo, debido a que, un administrador no va a tener las mismas opciones que un recepcionista.

Por otro lado, la interfaz desde la interfaz del administrador debe poder crear habitaciones, cargar habitaciones desde un archivo externo, consultar y cargar inventario, registrar y cargar tarifas de habitaciones. A su vez, todos los empleados pueden registrar consumos y pagos, pero solo los recepcionistas pueden crear y cancelar reservas.

Los cambios en tarifas, reservas, registros, facturas o menú se verán registrados en archivos externos para poder ser persistentes con esta información, debido a que, en dado caso de que se cambie algún registro, el siguiente empleado en acceder a la aplicación tendrá la información actualizada.

2 Nivel 1

2.1 Componentes candidatos y estereotipos

1. El Login de la app tiene el estereotipo de Service provider, en vista de que, dependiendo del estatus que tenga el empleado se van a mostrar ciertas funciones o no
2. El Inventario es un Information holder porque en esta parte solo se mantiene, actualiza y sube información acerca del inventario, habitaciones, tarifas o cargar menus.
3. La sección de servicios sólo almacena la información sobre los consumos registrados a nombre de los clientes y sus respectivos pagos, las el alojamiento en el hotel por persona y el servicio de restaurante. Por estas razones es un Information holder.
4. La sección de Controller almacena llama a las otras funciones para que ejecuten las instrucciones que el pide, por eso es un Controller. Esta sección está hecha de esta forma para seguir el estilo de control de **Control Delegado**.

En color lila podemos encontrar los nuevos candidatos que añadimos al proyecto con sus respectivos estereotípos.

5. Huéspeds App tiene el estereotipo de información holder, pues va a interactuar con el resto de componentes, pidiéndoles información, brindándoles información y almacenandola para mostrarla en la interfaz gráfica.
6. Bancos es el componente encargado de realizar los pagos de las reservas y servicios que consumen los clientes. Tiene el estereotipo de Service Provider debido a que, le brinda el servicio de dejar pagar a los usuarios a las dos aplicaciones principales, también tiene el estereotipo de Information Holder gracias a que, contiene todos los datos bancarios de los huéspedes que pagaron con un banco determinado.

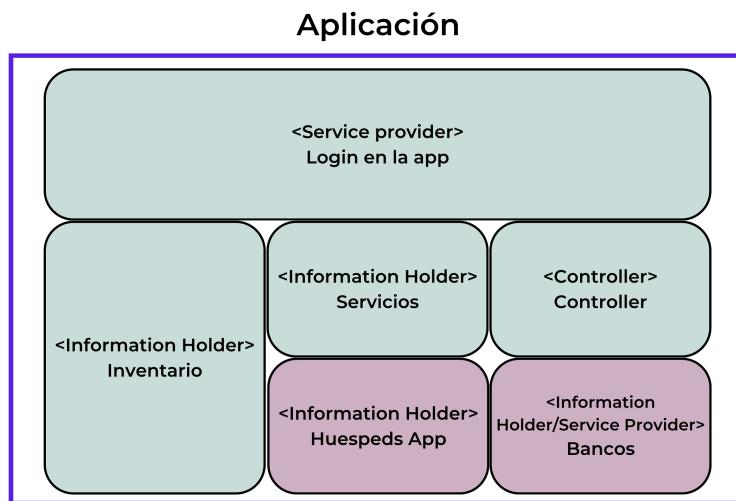


Figure 2: Componentes candidatos y sus respectivos estereotipos

2.2 Responsabilidades

Num	Responsabilidades	Componente
1	Permitir el Login al Usuario	Login en la App
2	Asignar rol al Usuario dependiendo de que tipo de empleado sea	Login en la App
3	Informar si el usuario y/o contraseña está mal escrito	Login en la App
4	Permitir cambiar contraseña	Login en la App
5	Cargar Inventario Habitaciones	Inventario
6	Cargar Habitación individual	Inventario
7	Editar Habitación	Inventario
8	Consultar inventario	Inventario
9	Cargar Tarifa	Inventario
10	Editar Tarifa	Inventario
11	Avisar tarifa faltante	Inventario
12	Cargar menú	Inventario
13	Editar menú	Inventario
14	Registrar servicios consumidos	Servicios
15	Registrar pagos	Servicios
16	Crear factura	Servicios
17	Realizar reserva	Servicios
18	Cancelar reserva	Servicios
19	Consultar tarifas por noches	Servicios
20	Registrar salida	Servicios
21	Recibir y devolver información a la interfaz (con ayuda de las otras secciones)	Controller
22	Crear Usuario para clientes	HuéspedApp
23	Autenticar Usuario para ingreso de cuenta de clientes	HuéspedApp
24	Consultar habitaciones disponibles	HuéspedApp
25	Reservar habitación específica	HuéspedApp
26	Permitir pagar con descuento apenas se hace la reserva	Bancos
27	Permitir pagar luego de hacer la reserva	Bancos
28	Permitir pagar todos los insumos consumidos en el check-out	Bancos

Tabla 1: Asignación de responsabilidades

Después de la asignación cabe resaltar que, la mayoría de responsabilidades se "desbloquean" después de que Login en la App dé el visto bueno y muestre las opciones que se pueden hacer, pues no todos los usuarios pueden acceder a las mismas funcionalidades (Un recepcionista no puede cargar las habitaciones).

2.3 Colaboraciones

Ahora bien, entre las responsabilidades definidas se ha considerado también la colaboración entre los distintos componentes a fin de satisfacerlas. Una característica reside en el estilo de control, al ser de Control Delegado todas las operaciones son pedidas **necesariamente** primero por el Controller, y vuelve al Controller para que sea dado a la interfaz. Lo anterior se caracteriza a continuación.

- **Registrar Salida (Check-out):** implica que el Controller al recibir las instrucciones por parte de la interfaz y permiso del Login:

1. Le pregunta a Servicios si el ID del reservante existe y si todos los servicios consumidos están pagos.
2. Recibe la información por parte de Servicios.
Si todo está pago, se registra la salida exitosamente.
Si algo **no** está pago, se le indica al cliente que debe pagar, por ende se manda a Servicios para que registren el pago.

- **Realizar reserva:** implica que el Controller al recibir las instrucciones por parte de la interfaz y permiso del Login:

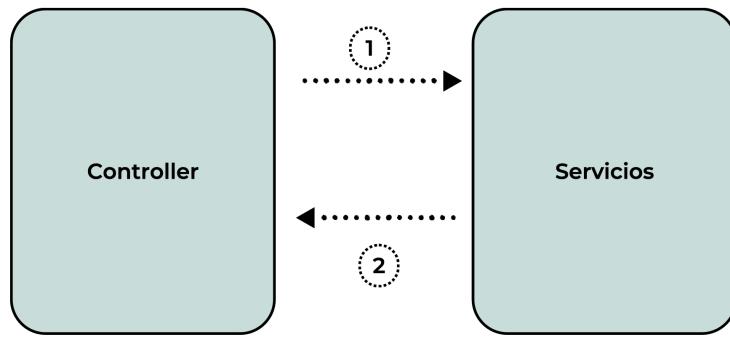


Figure 3: Colaboraciones entre componentes a la hora de registrar la salida de un huésped

1. Le pregunta a Inventario las tarifas del día a reservar
2. Recibe la información por parte del Inventario.
3. Le pregunta a Inventario si hay alguna habitación del tipo deseado libre en los días a reservar.
4. Recibe la información por parte del Inventario.
5. Si hay habitaciones disponibles, habla con Servicios para registrar el servicio de hospedaje.
Si no, el recepcionista debe comunicar al cliente que no hay habitaciones de ese tipo disponibles en ese rango de fechas.

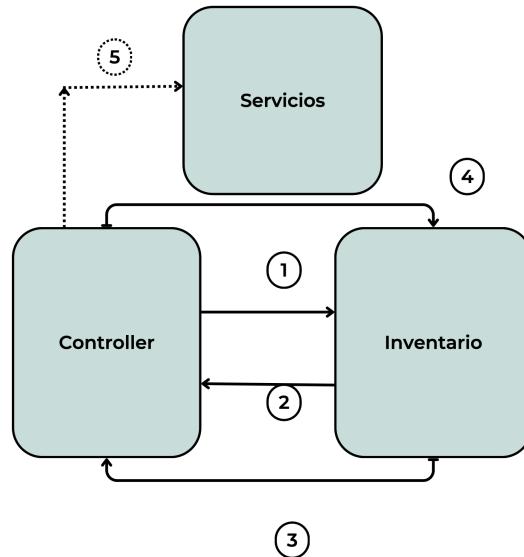


Figure 4: Colaboraciones entre componentes a la hora de realizar reserva

- **Cancelar reserva:** implica que el Controller al recibir las instrucciones por parte de la interfaz y permiso del Login:

1. Le pregunta a Servicios la fecha para la que se reservó la habitación
2. Recibe la información por parte de Servicios
3. Si la fecha de reserva respecto a la fecha de la solicitud de la cancelación es mayor a 48 horas, habla con Servicios para que borre el registro del servicio hospedaje de ese cliente.
// De lo contrario, el recepcionista notifica que no se puede cancelar la reserva.

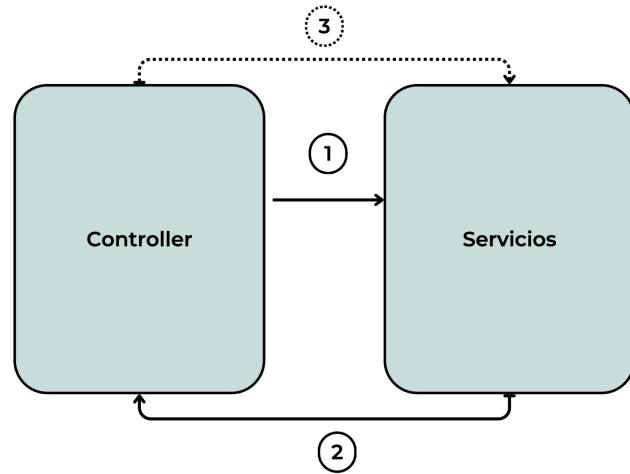


Figure 5: Colaboraciones entre componentes a la hora de cancelar reserva

- **Crear factura:** implica el Controller al recibir las instrucciones por parte de la interfaz y permiso del Login:

1. Le pregunta a Servicios todos los servicios registrados a nombre del cliente
2. Recibe la información por parte de Servicios
3. Crea e imprime una factura con los servicios que ha consumido este cliente, especificando en cada uno si ya estan pagos o no.

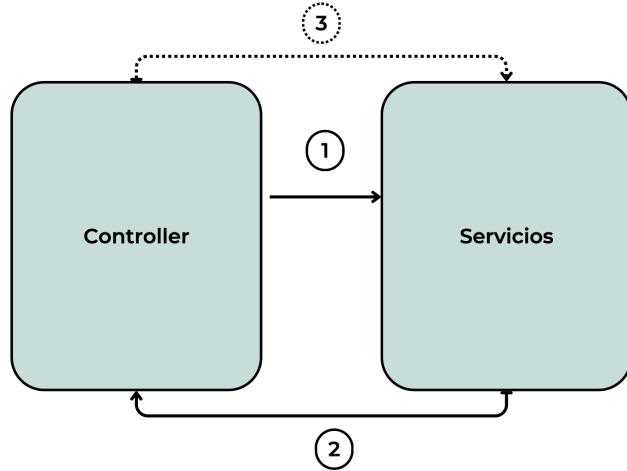


Figure 6: Colaboraciones entre componentes a la hora de Crear Factura

2.4 Colaboraciones Nuevas

Hay muchas nuevas colaboraciones con los nuevos componentes, pero las más importantes son:

- **Pagar Reserva:**

1. La App de los huespedes solicita procesar un pago a la clase Pagos
2. Este lo recibe y lo re-direcciona al banco solicitado
3. El banco solicita la información a servicios el costo del servicio de hacer la reserva
4. Servicios le pregunta al inventario cuanto cuestan las tarifas en las fechas solicitadas

Acá ya nos devolvemos, dando respuestas a todas las solicitudes hechas

5. Inventario informa a Servicios cual sería el precio de la reserva en función de lo que esté estipulado en la tabla de precios.
6. Servicios con esta información ya puede decir cuento cuesta el servicio, y al haber pagado a la hora de la reserva, el precio a pagar tiene un 10% de descuento.
7. Esta información la recibe el banco deseado para hacer el pago y hace 2 cosas:
 - (a) Realiza el cobro en la tarjeta al titular correspondiente
 - (b) Escribe esta transacción en la base de datos designada al banco, poniendo evidencia de que se pagó la reserva
 - (c) Informa a Pagos que el pagó se realizó exitosamente o fue rechazado
8. Finalmente, La aplicación de los huespedes recibe la información para mostrarle al usuario que su pago fue aceptado o rechazado.

Con este ejemplo podemos ver lo complejo que se volvieron los procesos colaborativos en la aplicación, pues una actividad común y muy recurrente necesita de muchos componentes pequeños. En esta parte del proyecto es muy importante haber seguido los planes de diseño, debido a que, el orden ayuda a que se tenga una buena lógica y no hayan acciones o métodos repetidos entre las clases.

Para este análisis no tuvimos en cuenta las bases de datos que se deberían consultar para dar una buena respuesta. Las bases de datos necesarias son:

1. HuespedesData
2. Banking Data

3. Tarifas

Podemos ver el diagrama que ilustra este proceso colaborativo en la figura 7.

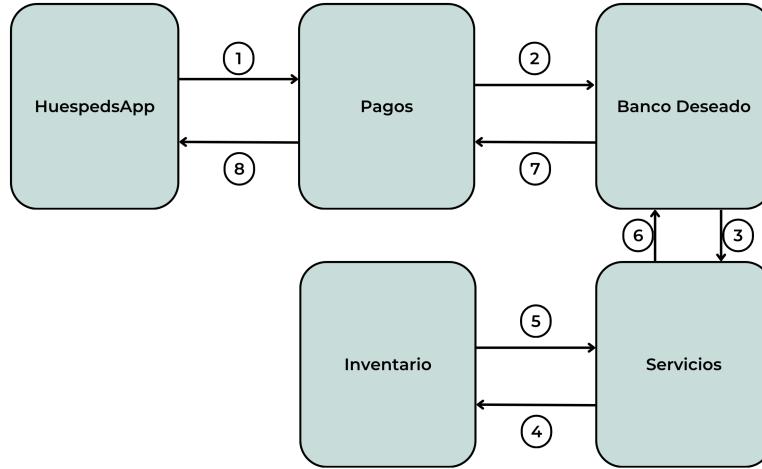


Figure 7: Colaboraciones entre componentes a la hora de pagar una reserva

3 Nivel 2

Siguiendo con el proceso de diseño por niveles, se procede a descomponer cada uno de los componentes perfilados en el nivel anterior

3.1 Login en la App

Login en la App solo necesita verificar si el usuario existe o no, y asignarle el rol que va a desempeñar en la app. El rol se asigna según el correo que use el empleado para iniciar sesión. Dependiendo del rol se le permitirá/restringirá usar ciertas funciones en la aplicación.

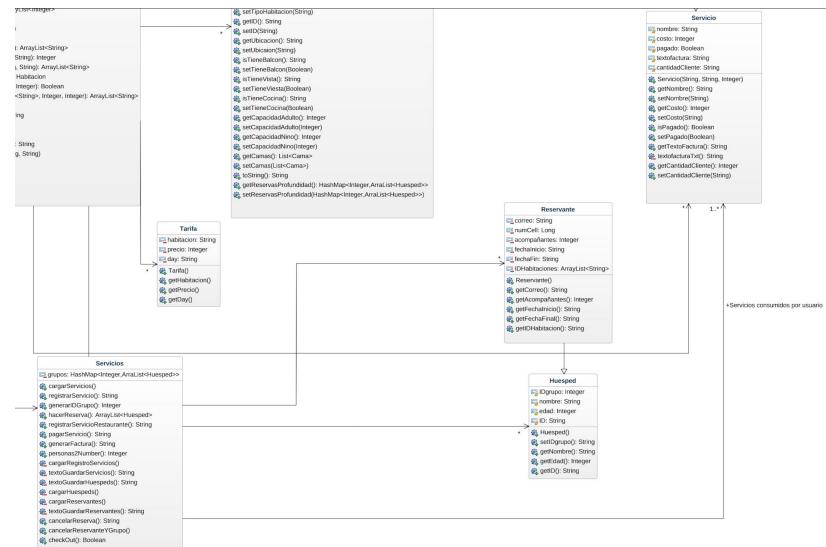


Figure 8: Descomposición del componente “Login en la App”.

3.2 Inventario

El inventario al encargarse de forma exclusiva de contener, las habitaciones, las tarifas, servicios del hotel y servicios del restaurante, estos servicios se pueden consultar por cualquier empleado y editar/cargar/crear por el administrador. El componente de Servicios de Restaurante se interpreta como una clase que hereda a la clase Servicio, debido a que tienen muchos atributos y métodos en común.

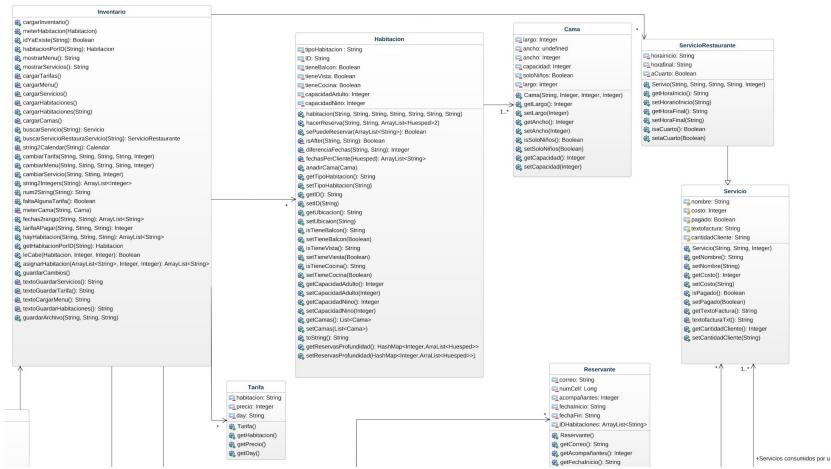


Figure 9: Descomposición del componente “Inventario”.

3.3 Servicios

La clase Servicios se encarga de forma exclusiva de contener los servicios consumidos por cada cliente. A su vez, almacena la información de los huéspedes, reservantes y grupos (conformación familiar/grupal del reservante y sus acompañantes), esto con el fin de poder acceder a la información de todo el grupo a la hora de generar facturas y permitir el Check-out.

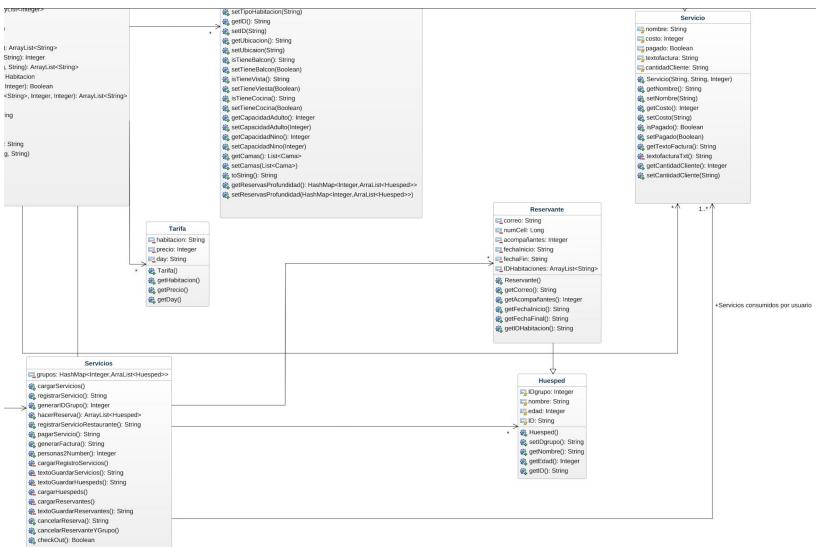


Figure 10: Descomposición del componente “Servicios”.

3.4 Controller

El Controller solo tiene un atributo, el cual es "usuario" de la clase Usuario, este atributo es donde se almacena el usuario actual que está usando el sistema (Se inicializa en null). Por otra parte, esta sección solo va a tener métodos públicos y estáticos, los cuales serán ejecutados en función de lo que pida el usuario (Empleado del hotel) en cada momento. Cada método llama a otro método de las otras secciones (Inventario o Servicios).

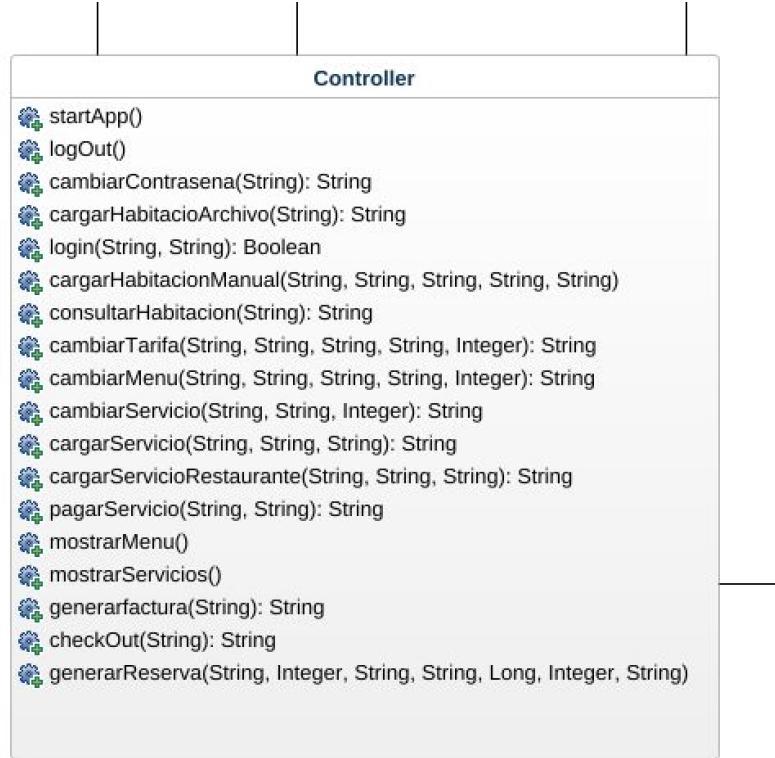


Figure 11: Descomposición del componente “Controller”.

3.5 HuespedesApp

La App contiene varias clases para funcionar, las cuales serían los JFrames que tiene asociados para que el usuario pueda ver que está pasando, cada uno tiene sus respectivos botones y funcionalidades para que la App sirva correctamente. Por otra parte, esta sección depende fuertemente de otras secciones de código, como del Inventario, Servicios, Pagos y LogIn.

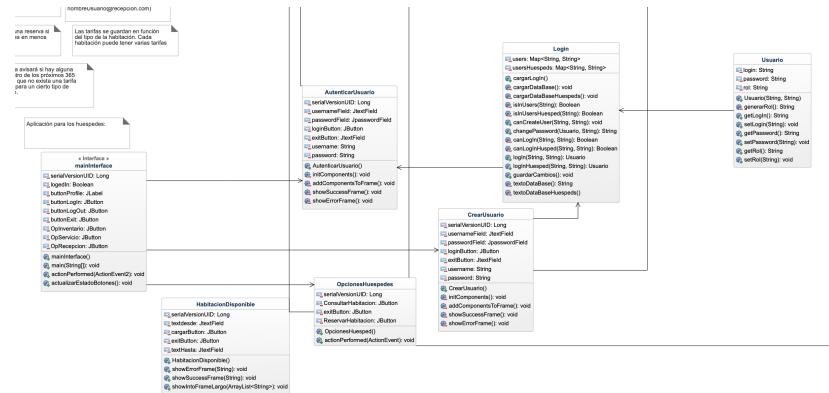


Figure 12: Descomposición del componente “Huespedes App”.

3.6 Pagos

La sección de Pagos contiene varias clases para funcionar, las cuales en su mayoría heredan de una grande que se llama Pagos, la cual unifica los comportamientos generales para cada pasarela de pago aceptada por la app. Por otra parte, no depende tanto de otros módulos, pues al ser un Service Provider es considerablemente más independiente del resto.



Figure 13: Descomposición del componente “Huespedes App”.

4 Diseño final

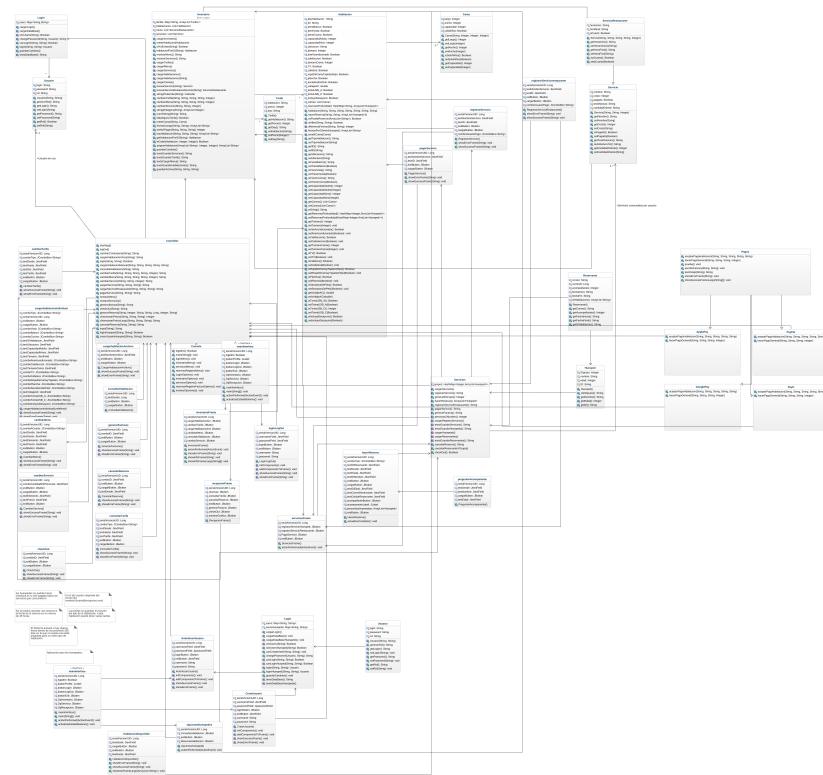


Figure 14: Diseño final obtenido.