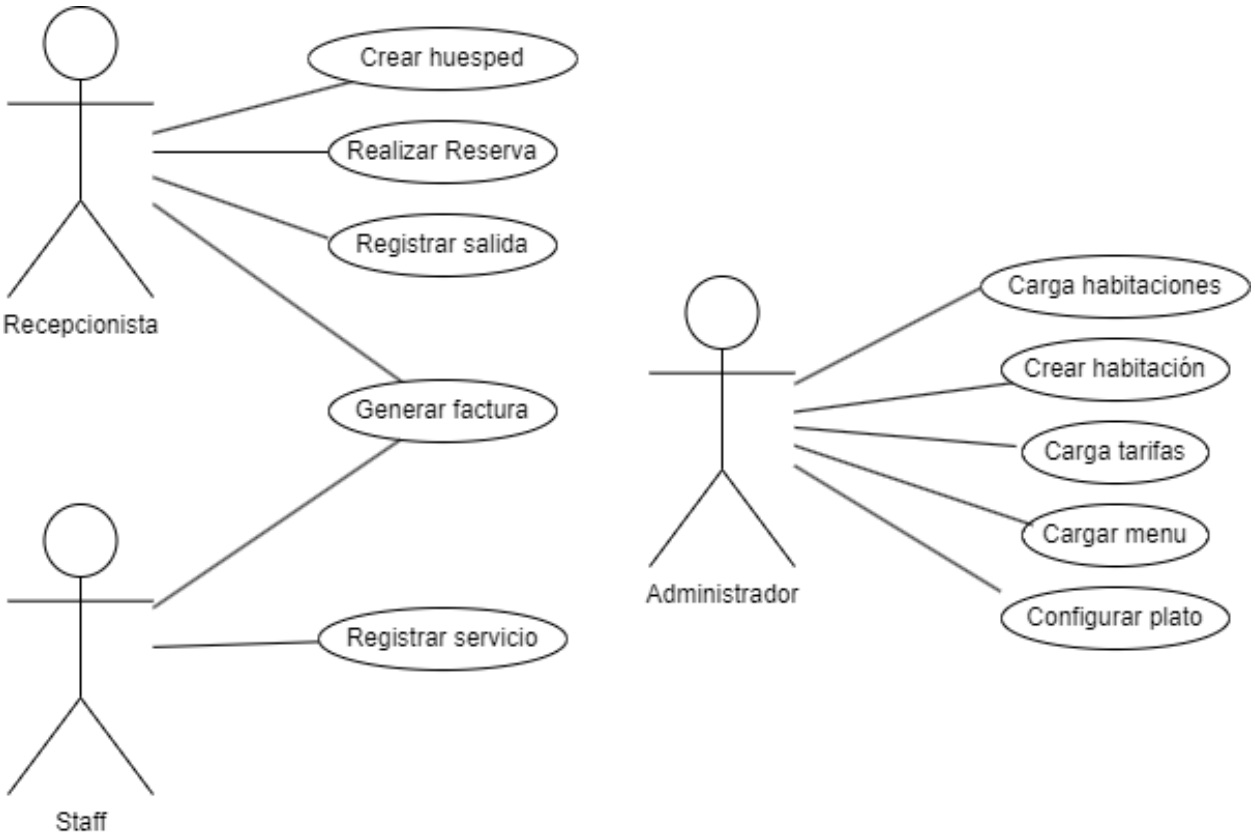
Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477

Proyecto 1 – Entrega 2: Diseño

El objetivo de este documento es presentar un análisis detallado y completo de los requisitos, características y funcionalidades necesarias para desarrollar una aplicación de alta calidad y eficiente para un hotel.

Para empezar con el análisis del diseño de nuestra aplicación vamos a hablar sobre el estilo de control que adoptamos en el diseño, nuestro estilo de control es delegado ya que nosotros distribuimos nuestras responsabilidades en un gran rol que es Empleado y de ahí se delega a más pequeños como lo son Administrador, Staff, Recepcionista.

Diagrama de casos de uso el cual ejemplifica nuestro estilo de control:



A continuación, definiremos los objetos/roles de nuestra aplicación, esto nos ayudara a identificar quien o quienes son los responsables de cierta cosa dentro de la aplicación

Objetos y roles

1. Information Holder: mantiene y entrega información
    a. CheckOut
    b. reserva
    c. Habitación

2. Structurer: mantiene las relaciones entre objetos y provee información sobre esas relaciones
    a. Huésped

   b. Grupo

3. Service provider: hace algún trabajo para los demás (ofrece un servicio complejo)
   a. Recepcionista
   b. Staff
   c. BufferReader
   d. BufferWriter

4. Coordinador: su principal responsabilidad es delegar tareas a otros componentes
   a. Huésped

5. Controller: toma decisiones importantes y controla las acciones de otros componentes
   a. Administrador

6. Interfacer: transforma información y peticiones entre diferentes partes de un sistema
   a. Servicios

Ahora gracias a que definimos anteriormente los roles dentro de nuestra aplicación, debemos definir las responsabilidades de igual manera. En nuestras aplicaciones contamos con que las responsabilidades son los métodos de ciertos componentes, dichos componentes son objetos en nuestra aplicación. Además, contamos con componentes externos a nuestra aplicación que también debe cumplir con ciertas responsabilidades dentro de nuestra aplicación.

Responsabilidades

#N	Responsabilidad	Componente
1	Registrar Huésped	Recepcionista
2	Generar Factura	Staff, Recepcionista
3	Entrar al Sistema	Empleado
4	Salir del Sistema	Empleado
5	Hacer Reserva	Recepcionista
6	Dar Cotización	Recepcionista
7	Cancelar Reserva	Recepcionista
8	Registrar Salida	Recepcionista
9	Registrar Pago	Staff
10	Registrar Servicio	Staff
11	Generar Factura Final	Sistema
12	Registrar Huéspedes	Recepcionista
13	Cargar Habitaciones	Administrador
14	Crear Habitación	Administrador
15	Guardar el archivo de la factura	Sistema
16	Mantener las relaciones entre clases	Empleado
17	Asignar habitación	Recepcionista
18	Cobrar servicio extra	Staff
19	Acabar Reserva	Recepcionista

Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477

20	Cargar Información	Administrador
21	Ver disponibilidad de habitación	Recepcionista
22	Leer líneas de texto	BufferReader
23	Escribir archivos de texto	BufferWriter

Justificación Decisiones tomadas en el diseño:

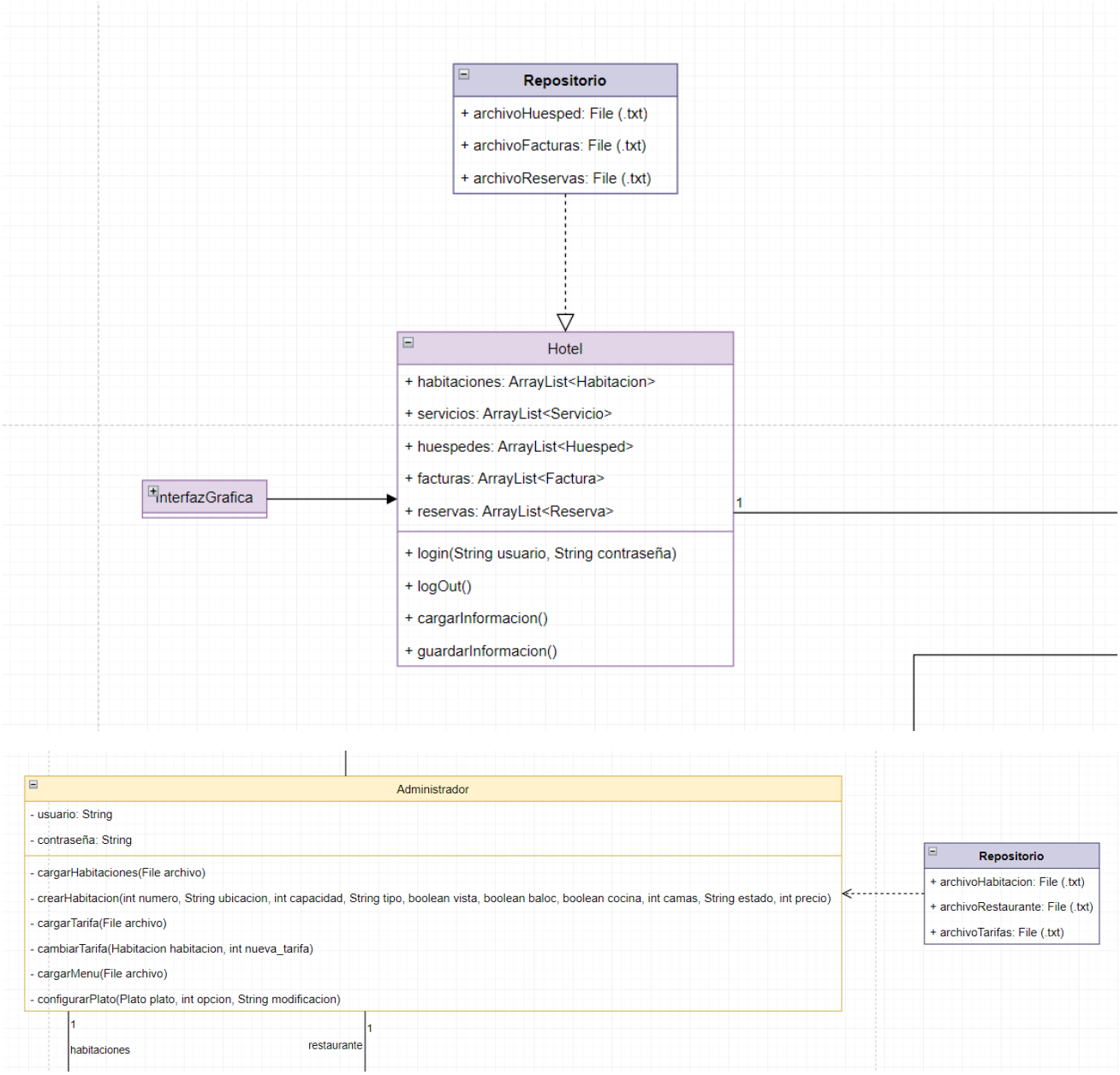
En el diagrama, tenemos tres roles principales que interactúan con el sistema y a través de ellos el usuario realiza reservas, utiliza servicios y realiza pagos. Estos tres roles, se comportan de la misma manera, por eso existe una clase Empleado de la cual se hereda Staff, Recepcionista y Administrador. La clase Staff la relacionamos a Huésped y a Servicios, ya que el staff se encarga de ofrecer y registrar un servicio que toma uno o varios huéspedes. La clase Recepcionista está relacionada a Huésped, Factura, Reserva y Habitación, ya que el recepcionista se encarga de crear un huésped en el sistema, inicializar su reserva, agregar las habitaciones y guardar su factura. La clase Administrador se relaciona con Restaurante y Habitación ya que es el encargado de cargar los datos sobre la información de estos.

Por otra parte, el Huésped está relacionado con Grupo que es uno o más huéspedes que ingresan al hotel. Este grupo, además de tener una lista de Huéspedes también guarda una lista de habitaciones, así se relaciona con la clase Habitación, que guarda toda la información acerca de sus camas, ubicación, capacidad, estado, precio y de estas derivamos los tres tipos diferentes de habitaciones. Se ofrecen servicios diferentes, por eso creamos una clase abstracta Servicios y tenemos tres servicios los cuales son Spa, guía turística y Restaurante donde guardamos la información acerca de su precio, horario y ubicación. A su vez, el Restaurante está relacionado a la clase Plato, que guarda el nombre del plato, de la bebida, su precio, el rango de hora y su lugar, donde en un restaurante guardamos información de uno o muchos platos. Por último, la clase factura guarda la información del id de factura, el grupo, la fecha, los servicios extra, los impuestos y la tarifa total y se relaciona con Consumo que guarda la información sobre los servicios consumidos por el huésped.

Justificación de persistencia

A continuación, se muestra el diagrama de persistencia con su respectiva justificación de porque decidimos realizar esas conexiones

Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477



Decidimos elaborar este diagrama de persistencia de la siguiente manera:

- Nuestra parte externa de la aplicación llamada repositorio (almacenamiento) esta enlazada con el hotel ya que de esta forma mantendremos la información de las Habitaciones, Servicios, Huéspedes, Facturas y de las reservas.
- Por otro lado, nuestro repositorio se conecta con el administrador ya que él es el encargado de cargar la información de la habitación en particular, del restaurante y de las tarifas.

Requerimientos de la aplicación

Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477
Vale la pena recalcar en nuestros requerimientos funcionales de la aplicación, así como los no funcionales, ya que fueron de ayuda al momento de diseñar los diagramas y responsabilidades.

Requerimientos funcionales:

- Administrador
  - -CargarHabitaciones
  - -CrearHabitacion
  - -CargarTarifa
  - -CambiarTarifa
  - -CargarMenu

- Recepcionista
  - -iniciarReserva
  - -darCotización
  - -cancelarReserva
  - -registrarHuespedes
  - -getHabitaciones

- Staff
  - -RegistrarServicio
  - -RegistrarPago
  - -GenerarFactura

- Sistema
  - -login
  - -generarFacturaFinal
  - -logout

- Diferente a los objetos y clases de la App
  - Consultar información almacenada en los archivos de texto
  - Crear archivos de texto plano separados por “;” con la información necesaria para continuar con la persistencia de la aplicación
  - Modificar los archivos de texto plano siguiendo la persistencia antes mencionada.

Requerimientos no funcionales:

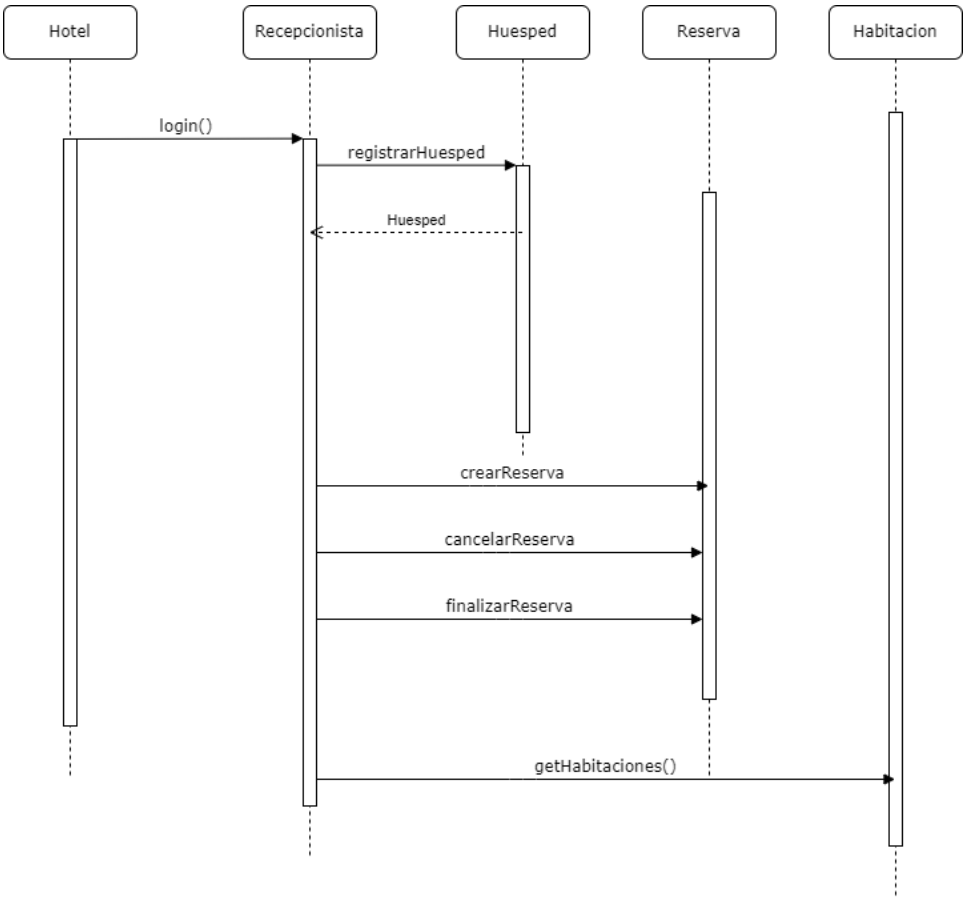
- Mantener la información de la aplicación en memoria de disco y no en memoria RAM.

Contamos 3 diagramas de secuencia que ejemplifican 3 requerimientos específicos por parte de un actor en nuestra aplicación.

Diagramas de Secuencia

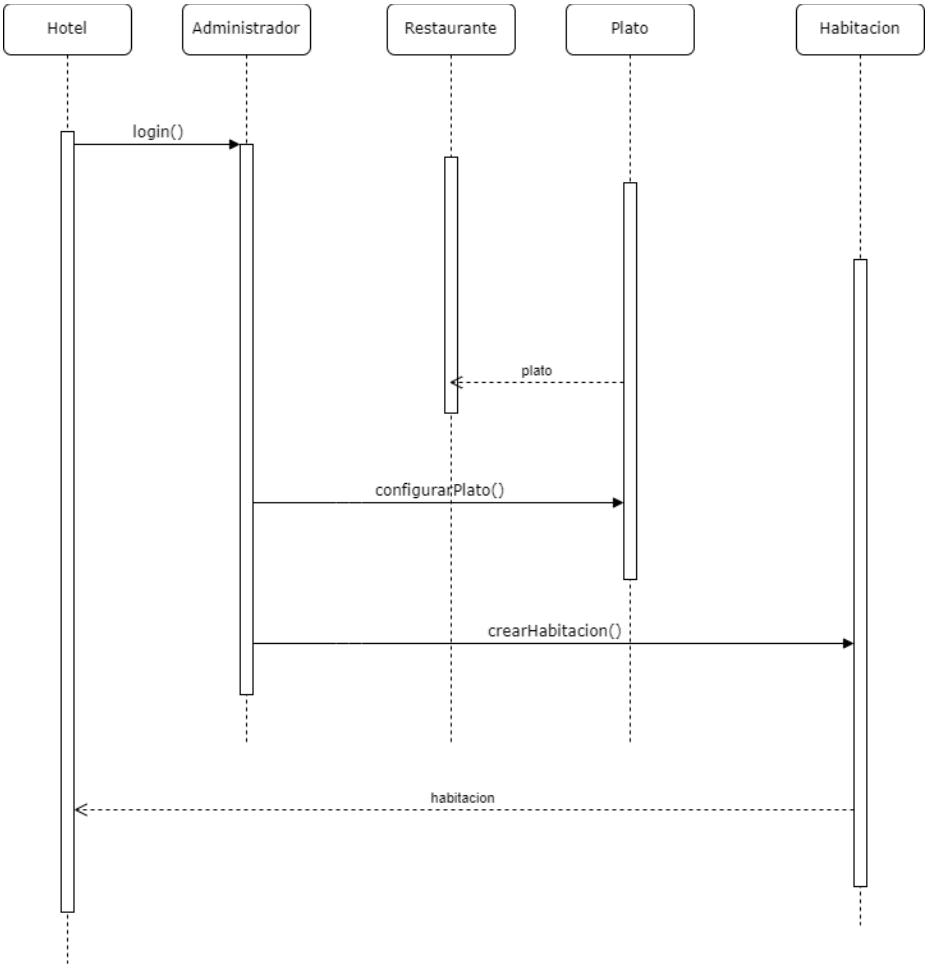
Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477

- Diagrama de secuencia para el requerimiento critico de registrar un huésped por parte del Recepcionista



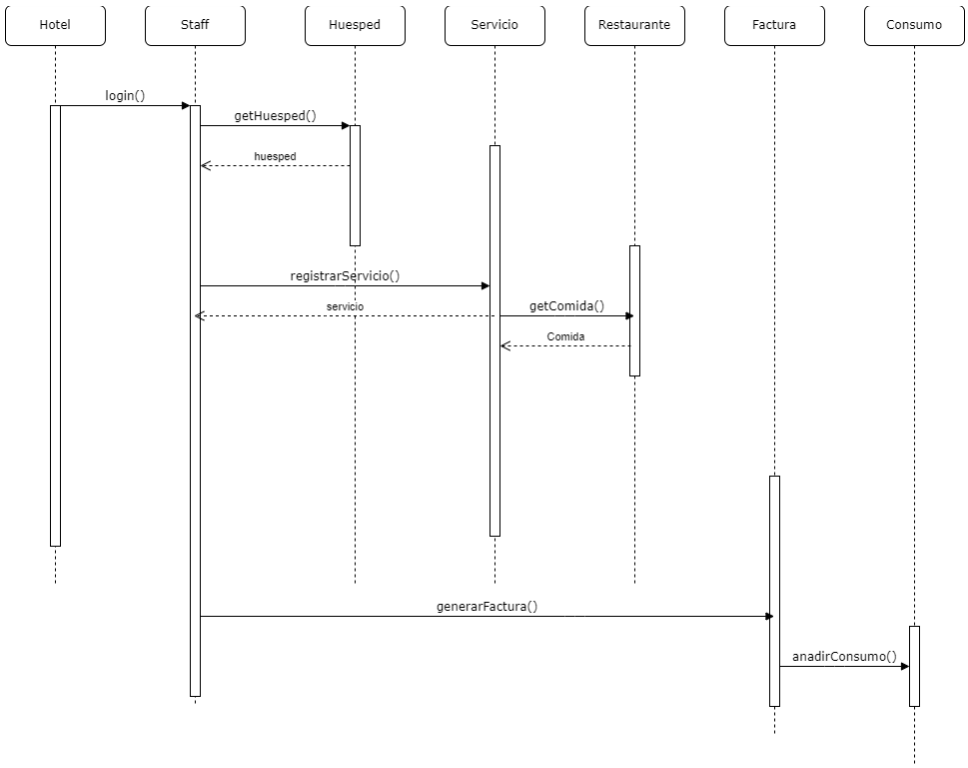
- Diagrama de secuencia para el requerimiento critico de configurar un plato y/o crear habitación por parte del Administrador

Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477



- Diagrama de secuencia para el requerimiento critico de registrar un servicio/generar Factura/ ver un huésped por parte del Staff

Luis Fernando Ruiz 202211513
Santiago Navarrete Varela 202211202
Andrea Galindo Cera 202122477



En este documento se encuentran los diagramas que fueron creados a partir del proceso de diseño antes descrito, algunos de ellos no están en este documento, pero están disponibles en formato .jpg en el GitHub junto con los que se encuentran en este documento para su mejor visualización.