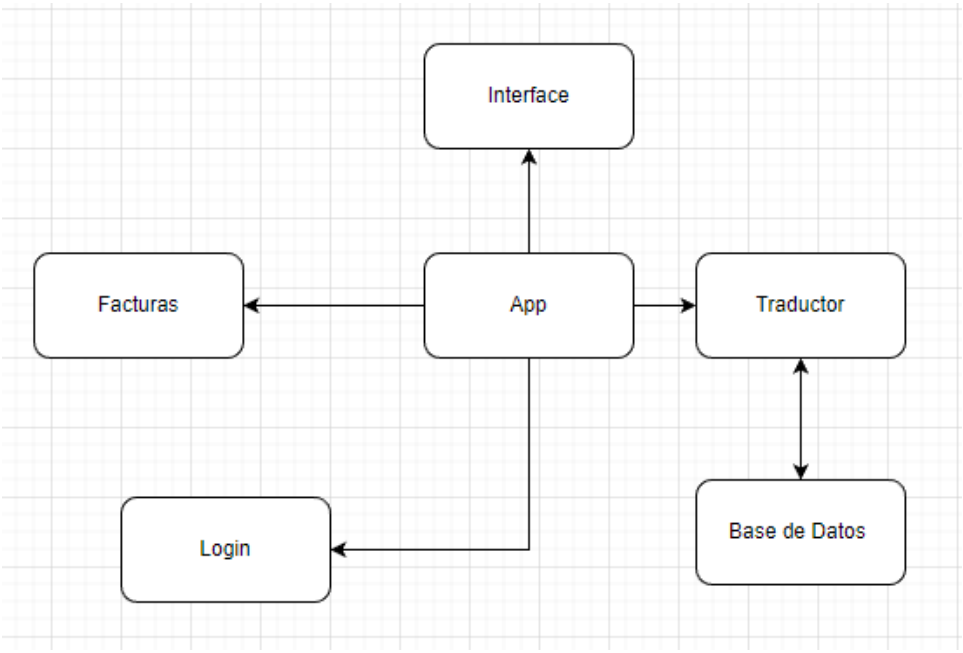
Documento diseño

Primera iteración

En la primera iteración creamos unos módulos encargados de aspectos específicos del programa. Decidimos crear estos módulos para tener un mayor entendimiento de la aplicación y para empezar a tener un mayor encapsulamiento en el programa.

Los módulos que creamos fueron:

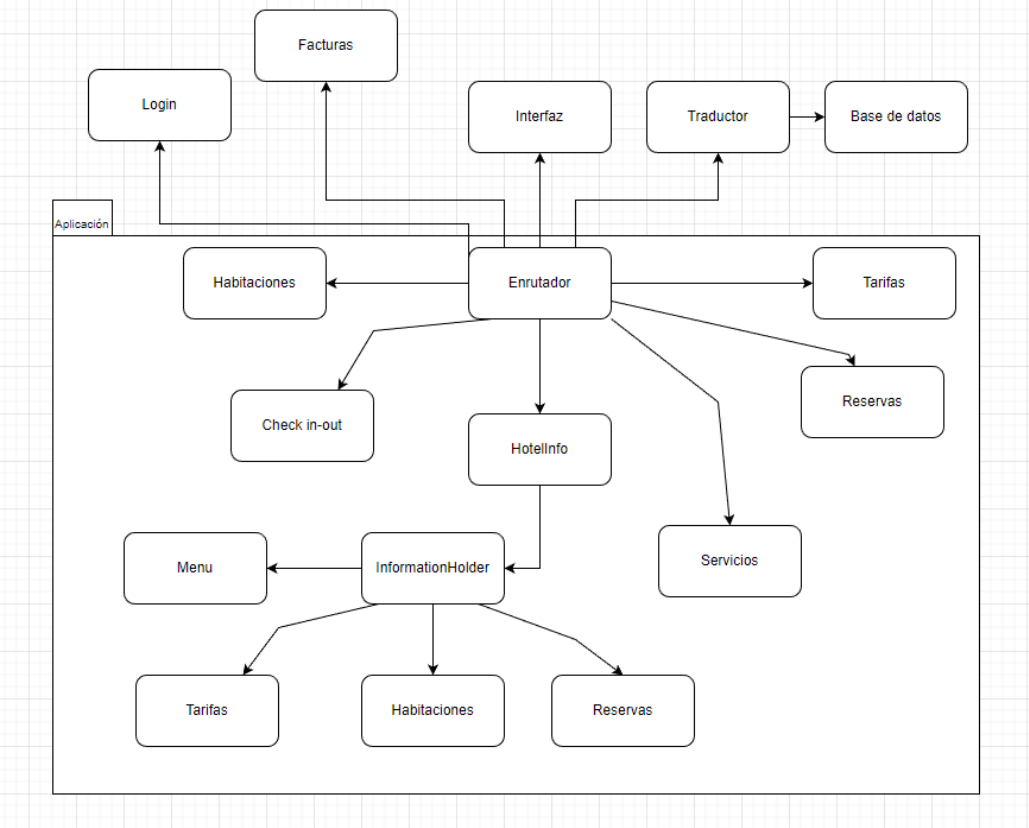
- Interfaz: Módulo con la función de mediar lo que el usuario recibe e ingresa en el programa.
- Facturas: Módulo con la función de crear y guardar las facturas de los huéspedes del hotel.
- Traductor: Módulo con la función de cambiar objetos a archivos y archivos a objetos, de forma que elementos en la base de datos puedan pasar de la base de datos al programa y del programa a la base de datos.
- Login: Módulo con la función de verificar los datos del usuario cuando vaya a entrar al programa
- App: Centro del programa. Encargado de lidiar con la lógica del programa (los requerimientos funcionales del programa). Debe almacenar y cambiar los datos del programa en forma de objetos.



Segunda Iteración

Para la segunda iteración nos enfocamos en el módulo de aplicación, ya que este módulo era el más amplio, el más importante y el más abstracto en la primera iteración. En esta segunda iteración decidimos crear un primer módulo que estuviera encargado de la conexión e interacción entre los módulos fuera de

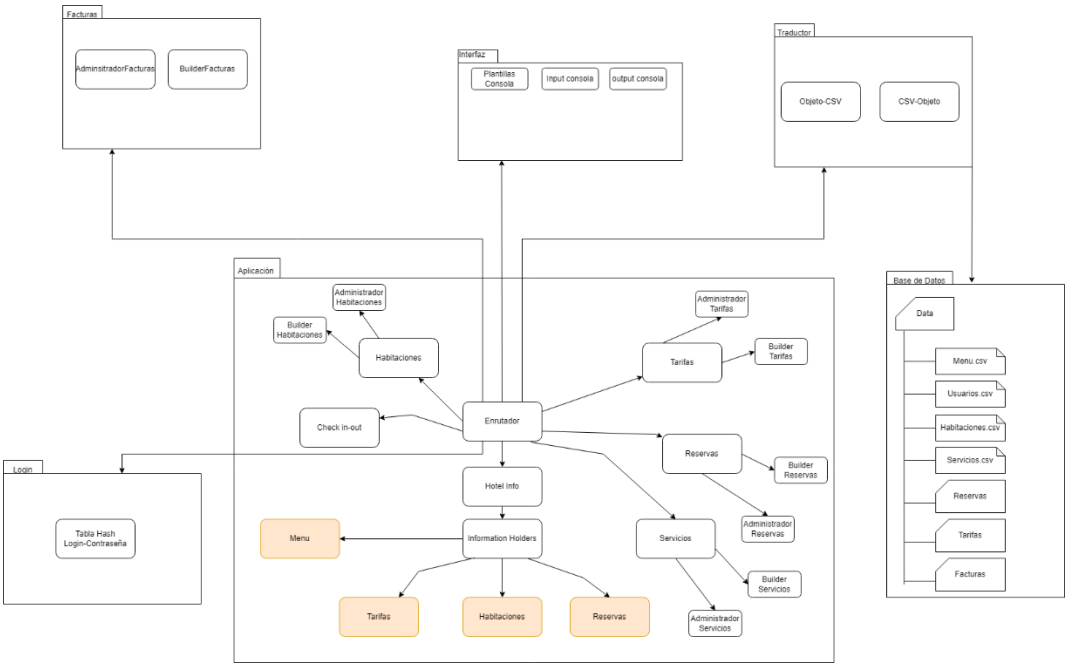
aplicación y los submódulos por dentro. Para esto creamos un módulo con el rol de enrutador. A partir de este, empezamos a crear módulos con el propósito de administrar y crear cierto tipo de datos, como las reservas, las habitaciones, las tarifas y los huéspedes.



Tercera Iteración

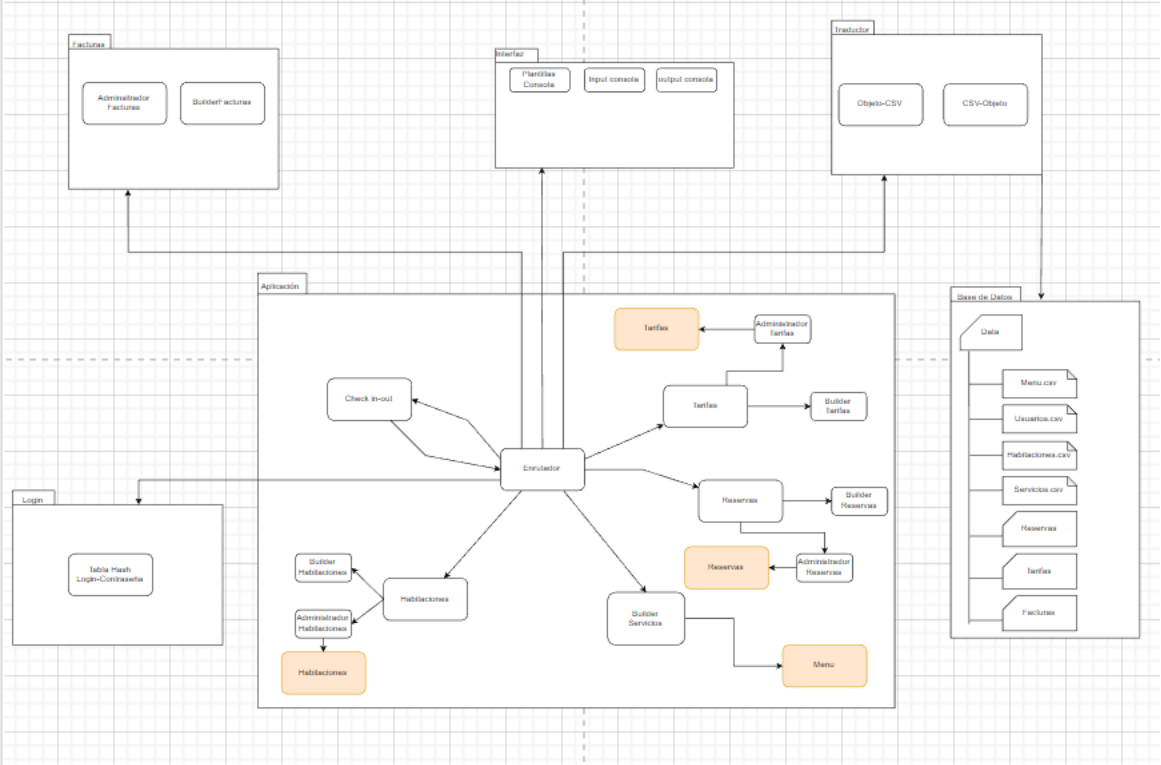
Para la tercera iteración decidimos empezar a centrarnos más en los módulos que principales del programa y los submódulos dentro de aplicación. Vimos que los submódulos dentro de aplicación tenían un componente de administrar y otro de crear. Para la parte de crear pretendemos usar el patrón builder, ya que los objetos dentro del programa tienen atributos que en un inicio no están definidos y que se van desarrollando a medida que crece el programa. Por otro lado, tenemos la parte que administra los objetos y la que se encarga de hacer los cambios en los objetos.

A parte, estructuramos la base de datos: Decidimos tener un archivo con el menú del restaurante, un archivo con la información del login de los usuarios y un archivo para las habitaciones y sus características. A su vez, tenemos 3 carpetas que contendrán los archivos de reservas, tarifas y facturas.



Cuarta Iteración

En la cuarta iteración del diseño hicimos cambios grandes en la forma en que guardábamos la información dentro del programa. Antes teníamos un único módulo que se encargaba de guardar todos los datos, pero esto hacía complicado la guardar y cambiar información en el sistema si utilizábamos los módulos que tenían esta labor. Por ello, decidimos que cada módulo que administraba y creaba un tipo de datos también guardara la información. Esto hace más clara y simple la vida de los objetos dentro del programa.



Quinta iteración:

En nuestra quinta iteración ya empezamos a definir las clases y los métodos dentro de estas. Gracias a esto nos dimos cuenta de muchos módulos que no eran necesarios, otros que debían ser añadidos y otros que tenían que cambiar su relación y posicionamiento dentro del programa. Uno de estos principales fue que la interfaz ahora contiene al enrutador principal y no en la forma inversa. También añadimos un controlador para la base de datos, de forma que este se encargue de la base de datos como tal y de operar el traductor. También para cada una de las funciones importantes hay un administrador. Este diagrama se encuentra en la entrega dos del proyecto 1 (diagramacompletoV2.png)

Sexta iteración: Esta es la creación de la interfaz gráfica (Proyecto 2)

En esta se actualizo todo lo relacionado con la interfaz, para pasar de consola a ventanas. Hay un JFrame principal que se relaciona con el enrutador principal y recoge las acciones que se realizan en los paneles. Estos paneles incluyen todos los requerimientos funcionales (cambiar tarifas, crear reservas, añadir servicios, etc.). Cambiar tarifas y ver ocupación del hotel son exclusivas de un login con admin, por lo tanto, esas opciones solo serán visibles si se entra como administrador. El diagrama con las clases de la interfaz actualizada se encuentra en esta misma carpeta. En la parte superior se encuentra todo lo relacionado a la interfaz. Se actualizaron los diagramas de secuencia incluyendo los paneles y el frame de Interfaz Principal

Séptima iteración (Proyecto 3):

- Cambios en características de cuartos y hotel:

La aplicación se expandió de tal forma que pudiera soportar nuevas características en los cuartos y del hotel. Se creó un nuevo archivo con las características del hotel y una que guardara estas especificaciones. Además, en los archivos de habitaciones base se les agregó las características descritas en el documento. Esto llevó a que se expandieran los atributos dentro de la clase de habitaciones base y que cambiara la forma en que se cargaban los datos desde los archivos de la aplicación. Por último, se resolvió mostrar estas nuevas características al usuario con una nueva ventana donde pudiera consultar las características del hotel y la habitación que quisiese.

- Pagos con tarjeta de crédito:

Se crean los botones dependiendo de la cantidad de opciones que haya dentro del archivo de texto al iniciar el programa; de modo que permite crear un botón específico para los distintos métodos de pago.

Una vez seleccionado el método se dirige a una pantalla donde puede ingresar la información, en la cual se recibirán datos de algún huésped y una tarjeta. Se procede a hacer una validación de dicha tarjeta para poder recibir el pago, modificar los servicios a pagos y el estado de reserva a finalizado (en caso de ser invalida no se recibirá el pago).

- Aplicación para huéspedes:

La aplicación para huéspedes le permite al usuario de la aplicación crear e iniciar sesión con una cuenta que el programa guarda. Cuando el usuario inicia sesión se le permite hacer una reserva en la aplicación buscando por fechas y las especificaciones que quiere que tenga las habitaciones. Además, el usuario puede consultar las especificaciones de la habitación que desee con el id de la habitación, a la vez que mira las especificaciones de lo que el hotel ofrece. Por último, cuando el usuario confirma su reserva se le presenta el precio total por el servicio de estadía en las habitaciones.

- Reportes y gráficas:

Se decidió agregar una nueva carpeta llamada “Reportes” donde se crearán las gráficas con información del hotel. Son 5 graficas entre diagramas de barras y pastel. Se generan al presionar el botón “Generar reportes” que solo está disponible en la consola de administrador.

Se utilizo la librería afrechar para crear estas graficas. La clase encargada de esto es GeneradorReportes que se encuentra en el paquete Aplicación.

Lo que hace es tomar los datos que nos interesa graficar, los agrupa en un set de datos y crea los gráficos. Por ultimo los exporta a la carpeta Reportes como png.

- Pruebas:

Para la prueba de creación de reserva, se usó el método crearReserva(). Las verificaciones que hace son las siguientes.

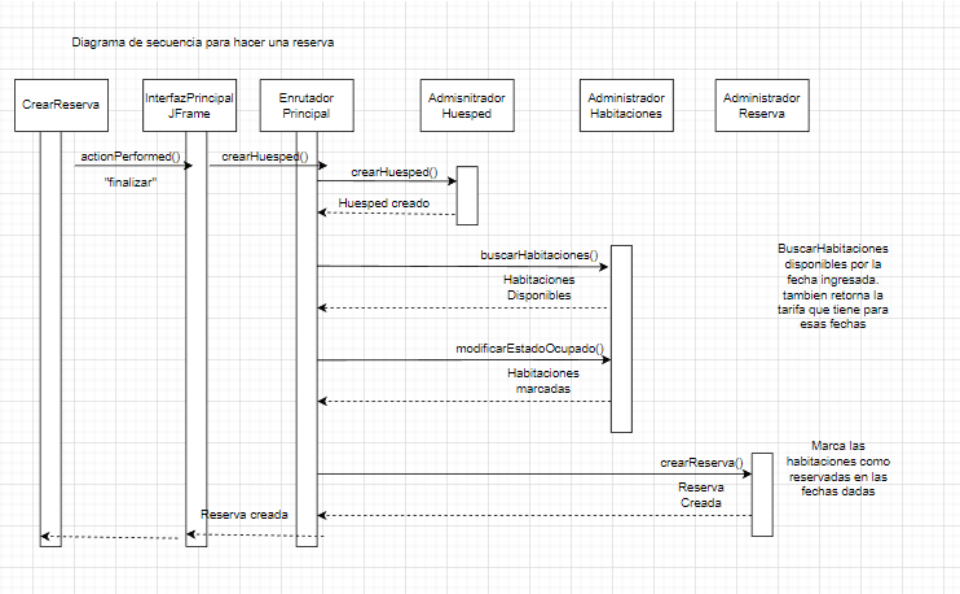
. Que el precio sea mayor que cero

. Que las habitaciones estén marcadas como ocupadas los días de la reserva, luego de haberla hecho

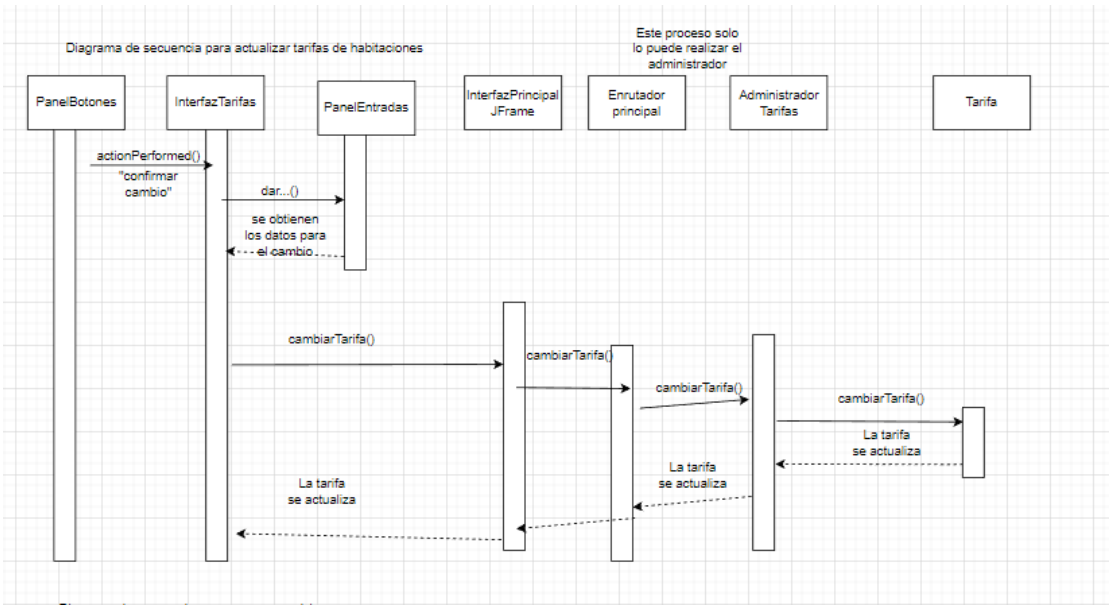
. Que se encuentre la reserva en el mapa con llave el documento de la persona que hizo la reserva.

Para las pruebas de la carga de datos, se crea un controladorBaseDatos. Hay una prueba por cada información que carga (menú, habitaciones, servicios, etc.). Lo que hace es buscar que exista información específica.

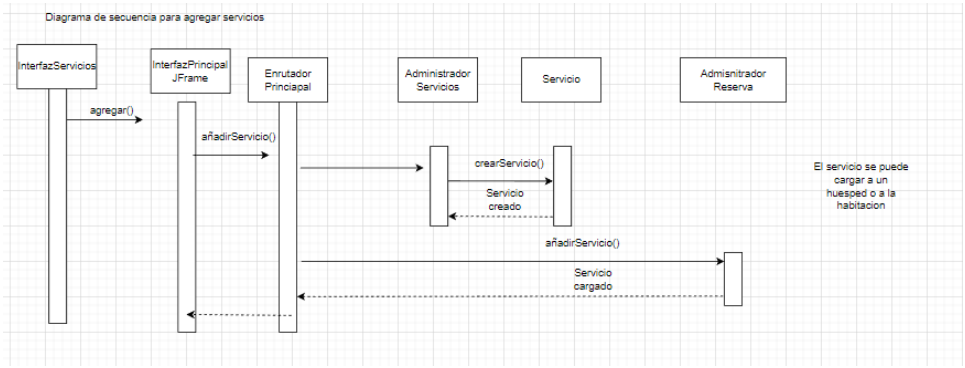
Diagramas de secuencia:



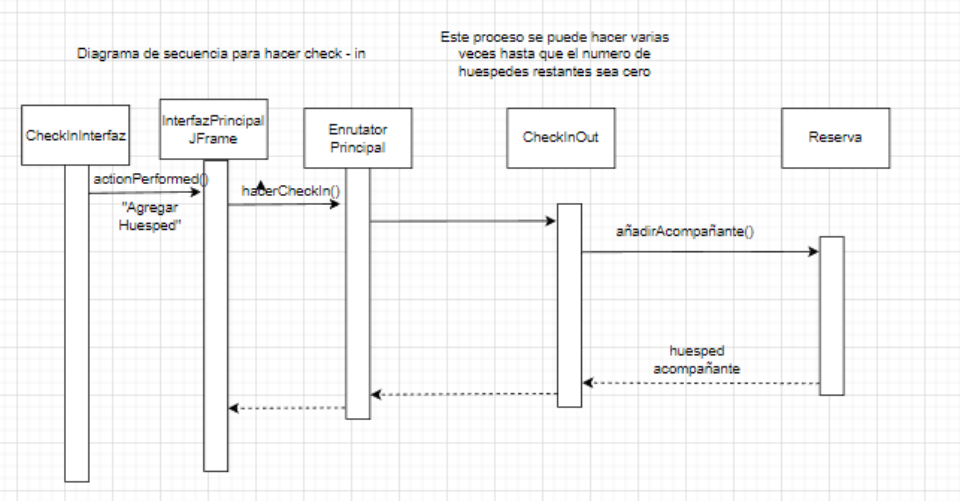
Lo primero que hace es crear el huésped principal (los acompañantes se crean cuando llegan y hacen el ingreso). Luego hace una búsqueda de habitaciones en las fechas que da el cliente, ahí informa de los precios. En cuanto eligen las habitaciones, se marcan en las fechas de reserva como ocupadas. Luego se crea la reserva.



Esta opción solo la tiene el administrador. Entra al administrador de tarifas e ingresa los datos para cambiarlas. Esta se puede realizar varias veces. Importante, si se toman fechas que ya tienen tarifa, se escoge la más barata.



Primero se crea el servicio en el administrador, y luego se añade al huésped o a la habitación, dependiendo del tipo de servicio.



En el check-in se agregan los datos de los acompañantes de la reserva

Restricciones y consideraciones:

La información se almacenará en archivos csv

La consola se encarga de que a los empleados no les aparezca la opción de modificar tarifas, que es una función solo para el administrados

Los acompañantes solo se registrarán a la entrada, en la reserva solo se dará el dato de cuantos serán

Las fechas de reserva y tarifas se guardarán en listas de 365 doubles y booleanos.

Se guardarán tres listas de tarifas, una para cada tipo de habitación. El precio aumentara de manera fija por cada característica adicional que tenga, por ejemplo, cocina integrada. Se inician en negativo para indicar que no se ha definido una tarifa.

Instrucciones:

La función main se encuentra en la clase InterfazPrincipalJFrame.

Al iniciar, se pedirá un usuario y contraseña. Aquí hay dos, el primero para admin y el segundo para empleado:

np, 1234 ;es admin

TipleA, Vegana1234; es empleado

Luego se abrirá una consola con las diferentes opciones. Cada una dirigirá a un panel que se tiene el diseño que se puede ver en la entrega uno.

En los espacios donde se piden fechas es necesario usar ese formato para que se convierta correctamente en un objeto LocalDate.