Agencia de turismo – UTN / NEORIS

El objetivo del presente proyecto es diseñar e implementar un sistema de registro de datos, orientado a una agencia de turismo, que permita el ingreso, manipulación y uso de la información ingresada para brindar un servicio de calidad a los clientes.

La aplicación fue desarrollada con la finalidad de agilizar la alta demanda de los clientes que solicitan presupuestos, reservas y facturaciones de paquetes de viajes con destinos tanto nacionales como internacionales, ofreciendo la creación de clientes de tipo particular o corporativo, permitiendo reducir el tiempo de procesamiento para la gestión.

A su vez, cada paquete tiene su propia financiación, diferenciándose en los impuestos calculados para cada transacción, con la posibilidad de utilizar moneda extranjera, permitiendo realizar bonificaciones en función del importe acumulado por las compras de sus clientes.

No solo resolvemos la agenda y disponibilidad de la empresa, si no que le damos en forma original y cómoda de gestionar clientes, paquetes de viaje, facturas y reservas, pudiendo acceder, activar e inactivar los datos manera local, almacenando toda la información en formato JSON.

Alguno de los inconvenientes con los que nos encontramos fue, al pensar el diagrama UML, la correcta y eficaz distribución de las clases, tratando de evitar el uso excesivo de métodos o atributos que no contribuían al objetivo del proyecto. Cabe aclarar que, a lo largo del desarrollo, se fueron presentando situaciones donde fueron necesarios los cambios del diagrama propuesto. Como por ejemplo la relación entre un cliente, su paquete y su factura. Decidimos que cada clase tenga un id, utilizado en la lógica del programa para acceder con facilidad a cada dato y poder realizar las tareas sin tener que repetir código. Es decir, siempre se priorizó la eficacia y eficiencia de la aplicación, tanto en su funcionamiento como en su código, teniendo en mente la correcta escalabilidad.

Pensando a futuro, consideramos necesario implementar una sincronización con Google Calendar, permitiendo tanto a la agencia como a los clientes gestionar los viajes y reservas, sincronizándolos con sus agendas y rutinas diarias. Además, el acceso al calendario de Google, permite el uso de recordatorios y mensajes personalizados por la aplicación WhatsApp, logrando una comunicación directa y efectiva entre los consumidores del sistema.

Diagrama de clases UML class Class Model Paquete Nacional Cantidad De Cuotas: int Cliente ModoDePago: String CantidadDeDias: int PorcentaiePorImpuestos: float Estado: boolean Apellido: String FehaDeViaje: Date Direction: String calcularImporte(): void DNI: String IdPaquete: int getModoDePago(): String ImporteTotal: float Habilitado: boolean getPorcentajePorImpuestos(): float Nombre: String IdCliente: int setModoDePago(): void Precio: Float Nacionalidad: String setPorcentajePorImpuestos(): void Nombre: String getCantidadDeCuotas(): int Factura Provincia: String getCantidadDeDias(): int Telefono: String Fecha: Date getEstado(): boolean getFechaDeViaje(): Date IdFactura: String getApellido(): String ImporteTotal: float getIdPaquete(): int getDireccion(): String get(mporteTotal(): float Asociada-a getDNI(): String calcularImporte(): float getNombre(): String getHabilitado(): boolean getFecha(): Date getPrecio(): float getIdCliente(): int getIdFactura(): String setCantidadDeCuotas(): void getNacionalidad(): String getImporteTotal(): float setCantidadDeDias(): void getNombre(): String Paquete Internacional setFecha(): void setEstado(): void getProvincia(): String setIdFactura(): void setFechaDeViaje(): void getTelefono(): String CotizacionDolar: float setImporteTotal(): void setIdPaquete(): void setApellido(): void ImportePorImpuestos: float setImporteTotal(): void setDireccion(): void RequiereVisa: boolean setNombre(): void setDNI(): void calcular(mporte(): void setPrecio(): void setHabilitado(): void getCotizacionDolar(): float setIdCliente(): void getImportePorImpuestos(): float Contiene setNacionalidad(): void getRequiereVisa(): boolean setNombre(): void setCotizacionDolar(): void setProvincia(): void setImportePorImpuestos(): void setTelefono(): void setRequiereVisa(): void Tiene-asociado Linea De Factura 1..10 Cantidad: int IdFactura: int Lugar IdLineaFactura: int Cliente Corporativo Subtotal: float Habilitado: boolean CUIT: String Unitario: float IdLugar: int RazonSocial: String Nombre: String calcularSubtotal(): float getCUIT(): String getCantidad(): int getHabilitado(): boolean getRazonSocial(): String getIdFactura(): int getIdLugar(): int setCUIT(): void getIdLineaFactura(): int getNombre(): String setRazonSocial(): void getSubtotal(): float setHabilitado(): void getUnitario(): float setIdLugar(): void setCantidad(): void setNombre(): void setIdFactura(): void

setIdLineaFactura(): void setSubtotal(): void setUnitario(): void El diagrama refleja la consigna propuesta en el enunciado del proyecto.

Decidimos comenzar por crear una clase Paquete de la que derivan dos tipos: un Paquete Nacional y un Paquete Internacional, cada uno con los atributos correspondientes, diferenciando lo antes mencionado con respecto al impuesto y al uso de la moneda extranjera.

A la clase Paquete se le relaciona una clase Lugar, donde fueron pensados los distintos destinos que ofrece la agencia, ofrecidos a los clientes dentro de cada paquete.

Cada paquete contiene una Línea de Factura que funciona como el detalle de la venta, relacionado con la factura que contiene los datos propuestos en el enunciado.

Y por ultimo la clase Cliente que, como se puede observar, solo deriva de esta un Cliente Corporativo. Sin embargo, se pensó el Cliente Particular dentro de la Clase padre, ya que no había diferencia entre los atributos de estos dos últimos. Algo que no sucede en el Cliente Corporativo que, a diferencia del Particular, cuenta con dos atributos más: RazonSocial y Cuit.

<u>Integrantes</u>

- Ghiano David
- Malsenido Mauricio
- Novo José Manuel
- Talarico Emilio