UNIMINUTO
Corporación Universitaria Minuto de Dios
Educación de calidad al alcance de todos
Vigilada MinEducación

ALTA CALIDAD

FACULTAD DE INGENIERIA
INGENIERIA DE SISTEMAS

DESARROLLO BASADO EN PLATAFORMAS

ACTIVIDAD 1

AUTORES:
CAMILA MARTINEZ
CRISTIAN NOVOA

DOCENTE:
JUAN CARLOS MARTINEZ

BOGOTÁ D.C

Profesor

1

a. Resuma el ciclo de vida de construcción de un programa

El ciclo de vida de construcción de un programa generalmente sigue estas pasos

1. Análisis del problema: Se define el problema que resolverá con el programa
2. Diseño: Se planifica la estructura del programa, algoritmos y diagramas de flujo
3. Codificación: Se escribe el código en un lenguaje de programación
4. Pruebas y depuración: Se verifica que el programa funcione correctamente y se corrigen errores
5. Implementación: Se despliega el programa en un entorno real
6. Mantenimiento y actualización: Se realizan mejoras y correcciones a lo largo del tiempo

b. Aspectos del análisis de un problema

El analisis de un problema incluye

1. Identificación del problema: Se comprende la necesidad o el desafío a resolver
2. Recolección de requisitos: Se determinan las necesidades del usuario y las restricciones del sistema
3. Definición de entradas y salidas: Se identifican los datos de entrada y los resultados esperados
4. Limitaciones y restricciones: Se consideran aspectos como tiempo de ejecución, recursos y seguridad
5. Alternativas de solución: Se evalúan diferentes enfoques para resolver el problema

c. Etapas del proceso de solución de problemas

Consta de las siguientes etapas

1. Comprensión del problema: Analizar en detalle el problema y sus requerimientos
2. Planificación de la solución: Diseñar el algoritmo o estrategia para resolver el problema
3. Implementación de la solución: Escribir el código o desarrollar la solución planteada
4. Pruebas y validación: Verificar que la solución funcione correctamente en distintos escenarios
5. Documentación y mantenimiento: Registrar el funcionamiento del sistema y mejorar su rendimiento

d. Elementos a entregar a un cliente

los elementos que se deben de entregar a un cliente en un proyecto de software incluye

1. Código fuente: Si el cliente requiere acceso al código del programa
2. Manual de usuario: Documento con instrucciones para usar el sistema
3. Manual técnico: Información sobre la arquitectura y el mantenimiento del software
4. Documentación de requisitos y diseño: Especificaciones técnicas y funcionales del sistema
5. Producto final: El software instalado y listo para su uso
6. Capacitación: Formación para los usuarios finales si es necesario
7. Soporte técnico y garantía: Asistencia en caso de errores o mejoras

e. Tarea No. 1

Cliente	El banco que desea implementar el programa para gestionar sus cajeros automáticos
Usuario	los clientes del banco que utilizarán los cajeros automaticos para realizar transacciones
Requerimiento funcional	R_1: El programa debe permitir a los usuarios retirar dinero de sus cuentas R_2: El programa debe permitir a los usuarios consultar el saldo de sus cuentas
Mundo del problema	• Es necesario conocer la información de las cuentas de los usuarios, como el número de cuenta, el saldo disponible y la identificación del titular • El programa debe tener acceso a la base de datos del banco para verificar la autenticidad de las transacciones y actualizar los saldos despues de cada retiro • Debe haber un sistema de seguridad para autenticar a los usuarios, como el uso de tarjetas y PINs • El programa debe manejar posibles errores, como fondos insuficientes o fallas en la conexión con la base de datos
Requerimiento no funcional	• Seguridad: El programa debe garantizar la seguridad de las transacciones y la privacidad de la información de los usuarios • Disponibilidad: los cajeros automáticos deben estar operativos las 24 horas del día, los 7 días de la semana • Usabilidad: La interfaz del cajero debe ser intuitiva y fácil de usar para los clientes • Rendimiento: El programa debe procesar las transacciones de manera rapida y eficiente para evitar largas esperas

F. Tarea 2

1.

Nombre	R1: Realizar un depósito en una cuenta
Resumen	Permite a un usuario depositar una cantidad de dinero en su cuenta bancaria
Entradas	Número de cuenta Cantidad a depositar
Resultado	El saldo de la cuenta ha sido incrementado con la cantidad depositada

2

Nombre	R2: Realizar un retiro de una cuenta
Resumen	Permite a un usuario retirar una cantidad de dinero de su cuenta bancaria
Entradas	Número de cuenta Cantidad a retirar
Resultado	El saldo de la cuenta ha sido disminuido con la cantidad retirada siempre que haya fondos suficientes

3.

Nombre	R3: Consultar el saldo de una cuenta
Resumen	Permite a un usuario consultar el saldo actual de su cuenta bancaria
Entradas	Número de cuenta
Resultado	Se muestra el saldo actual de la cuenta al usuario

Observaciones

6. Tarea 3

1. Requerimientos

Nombre	R1: Determinar el tipo de triángulo
Resumen	Permite determinar si un triángulo es equilátero, isósceles o escaleno basado en las longitudes de sus lados
Entradas	longitudes de los tres lados del triángulo
Resultado	Se muestra el tipo de triángulo (equilátero, isósceles o escaleno)

2. Requerimiento

Nombre	R2: Calcular el perimetro del triángulo
Resumen	Permite calcular el perímetro de un triángulo sumando las longitudes de sus tres lados
Entradas	longitudes de los 3 lados del triángulo
Resultado	Se muestra el perimetro del triángulo

3. Requerimiento

Nombre	R3: Calcular el area del triangulo
Resumen	Permite calcular el area de un triangulo utilizando la formula de Heron
Entradas	longitudes de los 3 lados del triangulo
Resultado	Se muestra el area del triángulo

H. Tarea 4

	Nombre	
Entidad	Triángulo	La entidad principal que representa un triángulo con tres lados y tres ángulos
Entidad	lado	Representa cada uno de los tres lados del triángulo, con una longitud especifica
Entidad	Ángulo	Representa cada uno de los tres angulos del triangulo con una medida en grados

Punto de reflexión ¿Qué pasa si no identificamos bien las entidades del mundo?
Podemos enfrentar serios desafíos en el diseño y desarrollo del programa. Un mal diseño basado en entidades incorrectas puede resultar una estructura confusa y difícil de mantener, lo que complica la implementación de funcionalidades. Además, podríamos implementar características que no se alinean con las necesidades reales del problema.

Punto de reflexión ¿Cómo decidir si se trata efectivamente de una entidad y no sólo de una característica de una entidad ya identificada?
Es importante considerar varios factores, primero evaluar si el concepto puede existir de manera independiente y si tiene sus propias características y comportamientos. Si es así, es probable que sea una entidad. Segundo, analizar si el concepto tiene relaciones significativas con otras entidades, lo que también sugiere que es una entidad en si mismo

I. Tarea 5

Clase: CuentaBancaria

Atributo	Valores Posibles
numeroCuenta	String (Identificador de la cuenta)
titular	String (Nombre del titular de la cuenta)
Saldo	Double (saldo actual de la cuenta
FechaApertura	String (Fecha de apertura de la cuenta)
Estado	String (Estado de la cuenta: Activa, Inactiva, bloqueada

- Diagrama UML de CuentaBancaria

CuentaBancaria
- numeroCuenta : String - Titular : String - Saldo : Double - FechaApertura : String - estado : String

- Diagrama UML CuentaCorriente

CuentaCorriente
- limiteSobregiro : Double - tasaInteres : Double

- Diagrama UML de CuentaAhorros

CuentaAhorros
- tasaInteresAnual : Double - saldoMinimo : double

- Diagrama UML de CDT

CDT
- numeroCDT : String - montoInvertido : double - Plazo : int - tasaInteres : double

- Diagrama UML de Mes

Mes
- numeroMes : int - nombreMes : String

- Clase : CuentaCorriente

Atributo	ValoresPosibles
limiteSobregiro	double
tasaInteres	double

- Clase : CuentaAhorros

Atributo	Valores Posibles
tasaInteresAnual	double (Tasa de interés anual)
saldoMinimo	double (Saldo mínimo requerido para evitar cargos

- Clase : CDT

Atributo	ValoresPosibles
numeroCDT	String (Identificador único del CDT
montoInvertido	double (Monto invertido en el CDT)
Plazo	int (Plazo en meses del CDT)
TasaInteres	double (Tasa de interés del CDT)

- Clase : Mes

Atributo	ValoresPosibles
numeroMes	int (Número del mes en la simulación)
nombreMes	String (Nombre del mes)

J. Tarea 6

Conclusiones:

El algoritmo presentado para viajar en el metro de París, aunque proporciona una guía general, carece de suficiente precisión y detalle para evitar ambigüedades. Por ejemplo, no especifica cómo identificar correctamente las estaciones en el mapa, como distinguir entre multiples líneas que podrían llevar al destino, o qué hacer en caso de transferencias entre líneas. Además, supone que el usuario tiene un conocimiento previo del sistema de metro y utiliza su "sentido común" para interpretar las instrucciones, lo que puede llevar diferentes interpretaciones y errores. Para que un algoritmo sea efectivo, debe ser lo suficientemente detallado y claro para cualquier persona

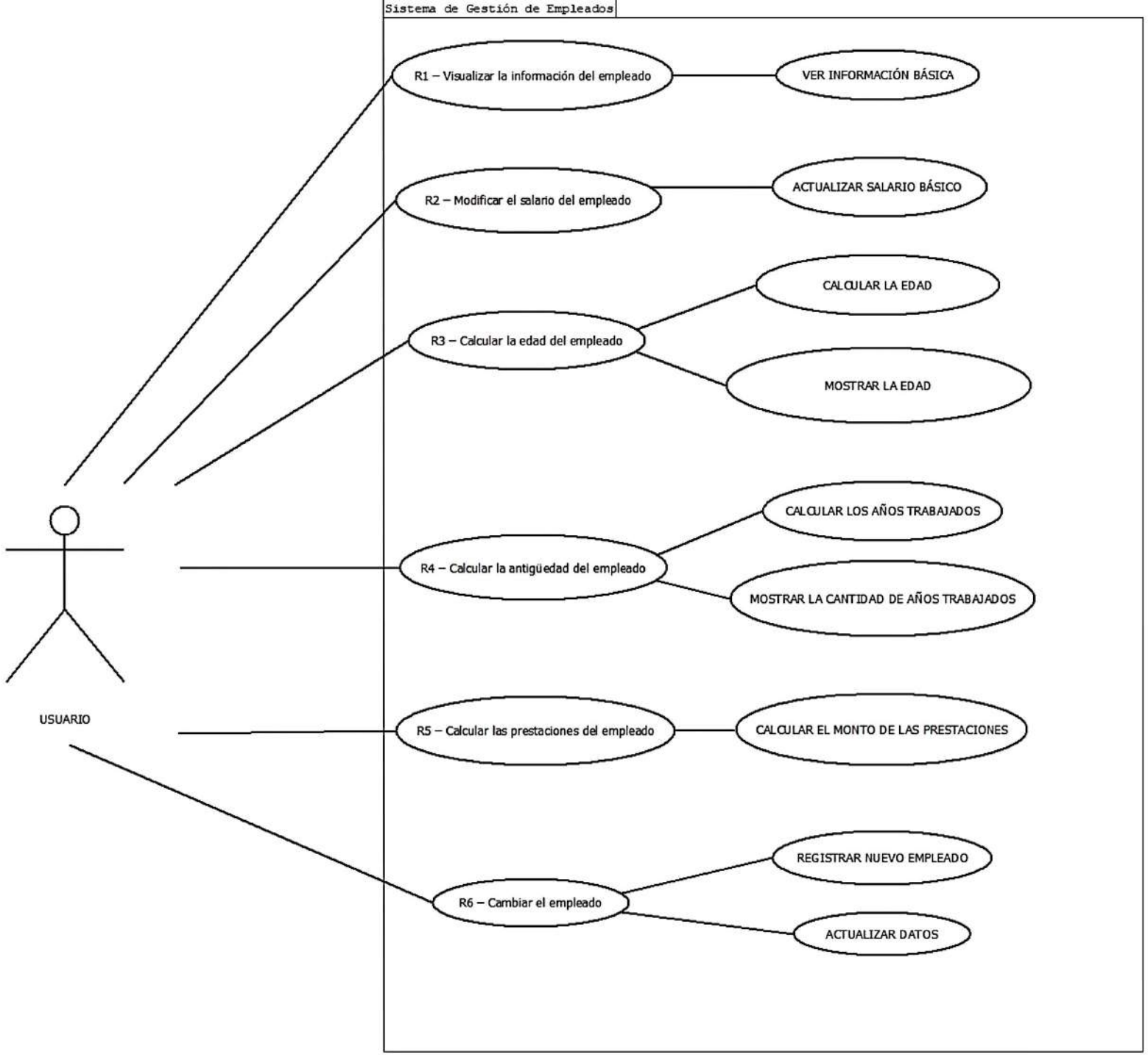
K. Ejemplo 1: El empleado

- Enunciado del problema:

Se requiere una aplicación que permita gestionar la información de un empleado, proporcionando funcionalidades para visualizar sus datos, modificar su salario, calcular la edad, antigüedad y prestaciones, así como cambiar el empleado registrado en el sistema

- Requerimientos funcionales

1. R1 - Visualizar la información del empleado
2. R2 - Modificar el salario del empleado
3. R3 - Calcular la edad del empleado
4. R4 - Calcular la antigüedad del empleado
5. R5 - Calcular las prestaciones del empleado
6. R6 - Cambiar el empleado



M. Punto m

Clase: Empleado

Atributos (variables)

- nombre (String): Nombre del empleado
- apellido (String): Apellido del empleado
- género (String): Genero del empleado
- fecha_nacimiento (Date): Fecha de nacimiento del empleado
- fecha_ingreso (Date): Fecha de ingreso a la empresa
- salario (Double): Salario basico del empleado
- imagen (String): Ruta o URL de la imagen del empleado

N. Punto N

Idea de proyecto 1: Sistema de gestión de Tareas y prioridades

Requerimiento funcional 1	Nombre: RegistrarTarea
	Resumen: Permite al usuario registrar una nueva tarea con su descripción, prioridad y tiempo estimado
	Entradas: Descripción (String), Prioridad (Int), Tiempo (Double)
	Resultado: La tarea se añade a la lista de tareas pendientes
Requerimiento funcional 2	Nombre: AsignarPrioridad
	Resumen: Permite al usuario asignar o modificar la prioridad de una tarea existente
	Entradas: ID de la tarea (INT), nueva prioridad (int)
	Resultado: La prioridad de la tarea se actualiza en la lista
Requerimiento funcional 3	Nombre: CalcularTiempoTotal
	Resumen: Calcula el tiempo total estimado para completar todas las tareas pendientes
	Entradas: Ninguna
	Resultado: Devuelve el tiempo total estimado (double)

Requerimiento funcional 4	Nombre:	Optimizar Tareas
	Resumen:	Utiliza un algoritmo para reorganizar las tareas basandose en su prioridad y tiempos estimados
	Entradas:	Ninguna
	Resultado:	La lista de tareas se reordena para optimizar la productividad

Idea de Proyecto: 2 : Sistema de recomendación de Rutas de viaje

Requerimiento funcional 1	Nombre	RegistrarDestino
	Resumen	Permite al usuario registrar un nuevo destino con su nombre, coordenadas y costo asociado
	Entradas	Nombre(String), Coordenadas(double,double), Costo(double)
	Resultado	El destino se añade a la lista de destinos disponibles
Requerimiento funcional 2	Nombre	CalculaRutaOptima
	Resumen	Utiliza un algoritmo (como Dijkstra) para calcular la ruta más eficiente entre dos destinos
	Entradas	Destino Inicial(String), Destino Final(String)
	Resultado	Devuelve la ruta óptima y el costo total
Requerimiento funcional 3	Nombre	MostrarRuta
	Resumen	Muestra la ruta calculada en un formato legible para el usuario
	Entradas	Ruta(List<String>)
	Resultado	la ruta se muestra en la interfaz del usuario
Requerimiento funcional 4	Nombre	ActualizarCosto
	Resumen	Permite al usuario actualizar el costo asociado a un destino
	Entradas	Nombre del Destino(String), Nuevo Costo (Double)
	Resultado	El costo del destino se actualiza en la lista

1.1 Describa y justifique el problema, indicando porque se trata de un problema soluble (referencia pág. 5):

Problema	Manejo de reservas de un avión con capacidad limitada y diferentes clases de sillas
Cliente	Empresa de aerolíneas que necesita un sistema eficiente para gestionar reservas de pasajeros.
Usuario	Personal de la aerolínea encargado de la gestión de reservas y pasajeros
Requerimiento funcional	-Asignar sillas a pasajeros según preferencias de clase y posición. -Consultar, buscar y eliminar reservas. -Calcular porcentaje de ocupación y valores de ventas.
Mundo del problema	Gestión de reservas en un avión con 50 sillas (8 ejecutivas y 42 económicas), considerando preferencias de los pasajeros y optimización del espacio
Requerimiento no funcional	-Sistema debe ser fácil de usar y eficiente. -Debe manejar datos de pasajeros de manera segura. -Debe proporcionar cálculos precisos y rápidos.

Justificación del problema soluble:

1. **Cliente y Usuario:** El problema es relevante para una empresa de aerolíneas que necesita gestionar reservas de manera eficiente. El sistema beneficiará al personal de la aerolínea al simplificar y agilizar el proceso de reservas.
2. **Requerimientos Funcionales:** Los requisitos funcionales están claramente definidos, lo que permite desarrollar un sistema que cumpla con las necesidades específicas de la aerolínea. Las operaciones de asignación, consulta, eliminación y búsqueda son tareas manejables con un programa bien diseñado.
3. **Mundo del Problema:** El contexto del problema es limitado y bien definido (50 sillas con clases específicas), lo que facilita la creación de un sistema que pueda manejar estas restricciones de manera efectiva.
4. **Requerimientos No Funcionales:** Los requisitos no funcionales, como la usabilidad, seguridad y precisión, son alcanzables con las tecnologías y metodologías de desarrollo adecuadas.

1.2 Señale mínimo 4 requerimientos funcionales indicando el tipo de dato que existen en el lenguaje de implementación para las variables que serán utilizadas en las entradas y salidas del requerimiento

	Nombre	**Asignar una silla a un pasajero**
REQUERIMIENTO FUNCIONAL 1	Resumen	Asignar una silla a un pasajero según sus preferencias de clase y posición.
	Entradas	- Nombre del pasajero (String) - Cédula del pasajero (String) - Preferencia de clase (String: "ejecutiva" o "económica") - Preferencia de posición (String: "ventana", "pasillo", "centro")
	Salidas	- Confirmación de asignación (String: "Asignación exitosa" o "No hay sillas disponibles") - Detalles de la silla asignada (String: "Clase: ejecutiva, Posición: ventana, Silla: 1A")
	Nombre	**Consultar una reserva**
REQUERIMIETO FUNCIONAL 2	Resumen	Consultar los detalles de una reserva utilizando la cédula del pasajero.
	Entradas	- Cédula del pasajero (String)
	Salidas	- Detalles de la reserva (String: "Nombre: Juan Pérez, Cédula: 123456789, Clase: económica, Posición: pasillo, Silla: 10B") - Mensaje de error si no se encuentra la reserva (String: "Reserva no encontrada")
	Nombre	**Eliminar reserva**
REQUERIMIENTO FUNCIONAL 3	Resumen	Eliminar una reserva existente utilizando la cédula del pasajero.
	Entradas	- Cédula del pasajero (String)
	Salidas	- Confirmación de eliminación (String: "Reserva eliminada exitosamente") - Mensaje de error si no se encuentra la reserva (String: "Reserva no encontrada")
	Nombre	**Calcular el porcentaje de ocupación del avión**
REQUERIMIENTO FUNCIONAL 4	Resumen	Calcular el porcentaje de sillas ocupadas en el avión.

	Nombre	Calcular el porcentaje de ocupación del avión
	Entradas	- Número de sillas ocupadas (Int) - Número total de sillas (Int: 50)
	Salidas	- Porcentaje de ocupación (Float: 75.0%)

2.1 Identifique las entidades del mundo problema, mínimo 3 (consulta como referencia la pág. 16, J. Villalobos) RUBBY CASALLAS GUTIERREZ, JORGE ALBERTO VILLALOBOS SALCEDO, "Fundamentos de Programación: Aprendizaje Activo Basado en Casos" En: México 2006. Ed:Pearson Education ISBN: 970-26-0846-5

ENTIDADES (Las que necesitas)	NOMBRE	DESCRIPCIÓN
1	Pasajero	Representa a un pasajero que realiza una reserva. Contiene información como nombre y cédula.
2	Silla	Representa una silla en el avión. Contiene información sobre su clase (ejecutiva o económica) y posición (ventana, pasillo, centro).
3	Reserva	Representa la reserva de una silla por un pasajero. Contiene información sobre el pasajero, la silla asignada y detalles de la reserva.

2.2 Señale las características de las entidades descritas:

Entidad #1: Pasajero

NOMBRE DE LA ENTIDAD #1	VALORES POSIBLES (Rango aceptable)	TIPO DE DATO (C++) Y EXPLICACIÓN
Nombre	Cualquier cadena de caracteres (ej: "Juan Pérez")	std::string - Para almacenar nombres completos con espacios y caracteres especiales.
Cédula	Números enteros positivos (ej: 1234567890)	int o long long - Dependiendo del tamaño de la cédula, se usa int para valores pequeños o long long para valores más grandes.

Entidad #2: Silla

NOMBRE DE LA ENTIDAD #2	VALORES POSIBLES (Rango aceptable)	TIPO DE DATO (C++) Y EXPLICACIÓN
Clase	"Ejecutiva" o "Económica"	std::string - Para almacenar la clase de la silla como una cadena de texto.
Posición	"Ventana", "Pasillo", "Centro"	std::string - Para almacenar la posición de la silla como una cadena de texto.

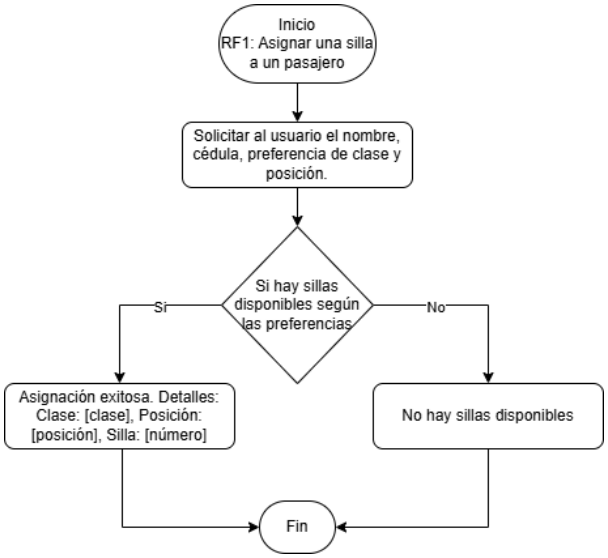
Entidad #3: Reserva

NOMBRE DE LA ENTIDAD #3	VALORES POSIBLES (Rango aceptable)	TIPO DE DATO (C++) Y EXPLICACIÓN
Pasajero	Objeto de tipo Pasajero	Pasajero - Se usa un objeto de la clase Pasajero para representar al pasajero que realiza la reserva.
Silla	Objeto de tipo Silla	Silla - Se usa un objeto de la clase Silla para representar la silla asignada.
Detalles de la reserva	Cualquier cadena de caracteres (ej: "Reserva confirmada")	std::string - Para almacenar información adicional sobre la reserva.

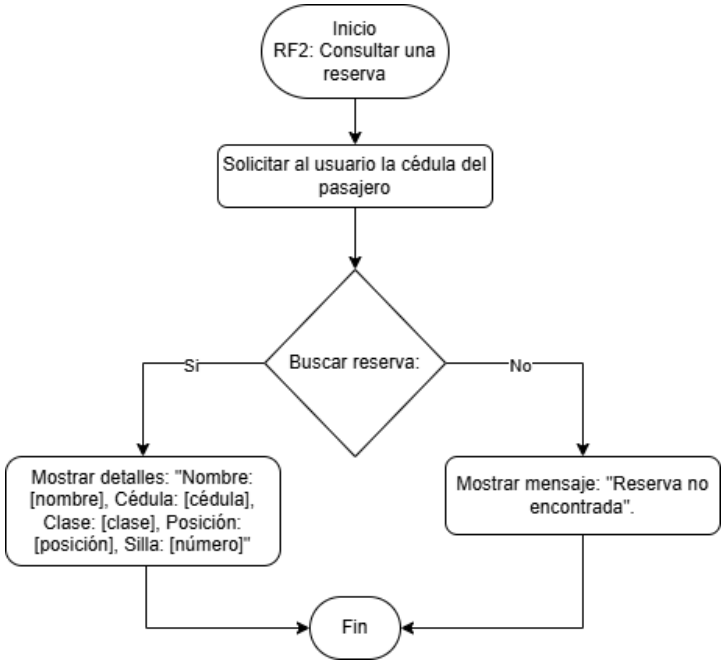
2.3 Establezca las relaciones entre las entidades de forma lógica, en un esquema gráfico, estableciendo las entidades y las relaciones (investiga: Diagrama E/R) (Pág. 21-22):

2.4 Por cada requerimiento funcional desarrollado en el numeral 1.2, crea un diagrama de flujo:

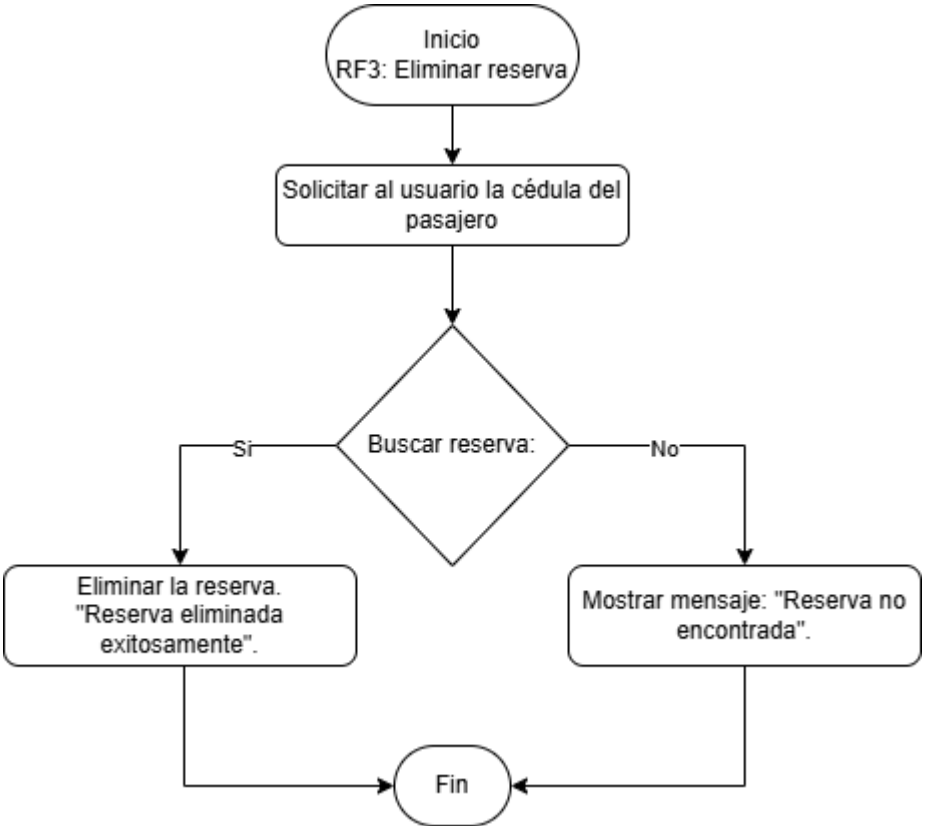
Requerimiento Funcional 1



Requerimiento Funcional 2



Requerimiento Funcional 3



Requerimiento Funcional 4