

Universidad del Rosario

Facultad de Economía

Programación de Computadores

Sistema de Gestión de Transporte Urbano

Integrantes:

Ivanna Castro Mendoza
Daniela Silva Vargas
Tomas Rangel Palacio
Nicolle Garaviz Sanchez

Profesor: Roy Alejandro Gómez Ávila

Fecha de Entrega: Noviembre 2025

Período: 2025-II

Índice

1. Planteamiento del Problema	3
1.1. Contexto y Justificación	3
2. Objetivos	3
2.1. Objetivo General	3
2.2. Objetivos Específicos	3
3. Alcance del Sistema	4
3.1. Funcionalidades para el Conductor	4
3.2. Funcionalidades para el Pasajero	4
3.3. Limitaciones del Sistema	4
4. Reglas del Negocio	5
5. Diagrama de Flujo	6
6. Clases Principales	7
6.1. Clase Ruta	7
6.2. Clase Bus	7
6.3. Clase Conductor	8
6.4. Clase Pasajero	8
6.5. Clase Viaje	8
7. Plan de Trabajo y Organización	9
7.1. Roles del Equipo	9
7.2. Cronograma Semanal	9
8. Viabilidad Técnica y Herramientas	9
8.1. Tecnologías a Utilizar	9

8.2. Justificación Técnica	10
--------------------------------------	----

1 Planteamiento del Problema

Desarrollar un programa informático que permita gestionar de manera eficiente las rutas, los conductores y los pasajeros de un sistema de transporte. El sistema deberá registrar y organizar la información de cada ruta disponible, asignar conductores a los recorridos y controlar el número de pasajeros transportados. Además, el programa incluirá una funcionalidad para calcular las ganancias generadas en cada viaje o ruta, teniendo en cuenta los costos y los ingresos obtenidos, con el fin de facilitar el análisis y la toma de decisiones dentro del sistema de transporte.

1.1 Contexto y Justificación

En ciudades como Bogotá, los sistemas de transporte urbano enfrentan desafíos constantes en la gestión eficiente de rutas, control de pasajeros y recaudo de tarifas. La falta de sistemas integrados que proporcionen información en tiempo real afecta tanto a los usuarios como a los operadores del servicio.

Este sistema de gestión de transporte busca optimizar la operación mediante digitalización, permitiendo:

- Control en tiempo real de la ocupación de buses
- Gestión automática de pagos con tarjeta
- Cálculo preciso de ganancias por ruta
- Mejora en la experiencia del usuario mediante información disponible

2 Objetivos

2.1 Objetivo General

Desarrollar un programa informático que permita organizar de manera eficiente rutas, conductores y pasajeros de un sistema de transporte de buses, distribución de recorridos, registro de esta información y estimaciones de las ganancias, con el fin de optimizar la toma de decisiones y perfeccionar el funcionamiento del servicio.

2.2 Objetivos Específicos

1. Estructurar un módulo que organice las rutas, registrando y actualizando la información relacionada con la disponibilidad de los buses y cupos en cada uno (100 % de las rutas registradas).

2. Desarrollar un sistema que permita vincular conductores con las rutas correspondientes según su jornada laboral (asignación automática 100 % funcional).
3. Elaborar un sistema que muestre el registro de pasajeros transportados en tiempo real y que al mismo tiempo actualice la disponibilidad de cupos (actualización en tiempo real con máximo 2 segundos de delay).
4. Estructurar un módulo que determine las ganancias generadas por cada viaje o ruta, considerando el número de pasajeros transportados (cálculo preciso al 100 %).

3 Alcance del Sistema

3.1 Funcionalidades para el Conductor

- Registro de pasajeros ingresados en tiempo real
- Visualización de total recaudado por ruta
- Control de inicio/fin de viaje
- Consulta de ruta asignada
- Al ingresar al programa, el conductor tendrá un registro de cuántas personas ingresaron al bus y el total de dinero recaudado al realizar esa ruta.

3.2 Funcionalidades para el Pasajero

- Consulta de ubicación de buses en tiempo real
- Verificación de cupos disponibles
- Recarga de tarjeta virtual
- Historial de viajes realizados
- Al ingresar al programa, el pasajero tendrá acceso a la ubicación de los buses, el tiempo estimado de llegada y el número de pasajeros a bordo, para conocer la disponibilidad en tiempo real.

3.3 Limitaciones del Sistema

- Sistema opera con máximo 7 rutas predefinidas
- Capacidad máxima por bus: 35 pasajeros
- Tarifa fija de \$5.000 por trayecto
- Entre 3-6 paradas por ruta

4 Reglas del Negocio

- Toda ruta tendrá conductor asignado
- Cada pasajero deberá pasar la tarjeta para pagar el pasaje y registrar su entrada
- Cada pasajero deberá registrar su salida del bus
- Cada bus tendrá una capacidad máxima de 35 personas
- La tarifa será estándar para todas las rutas: 5.000 pesos colombianos
- Control de puestos de pasajeros mediante tarjeta de entrada/salida
- Cada bus tiene un solo destino y entre 3-6 paradas
- Total de 7 puntos donde los buses inician y finalizan

5 Diagrama de Flujo

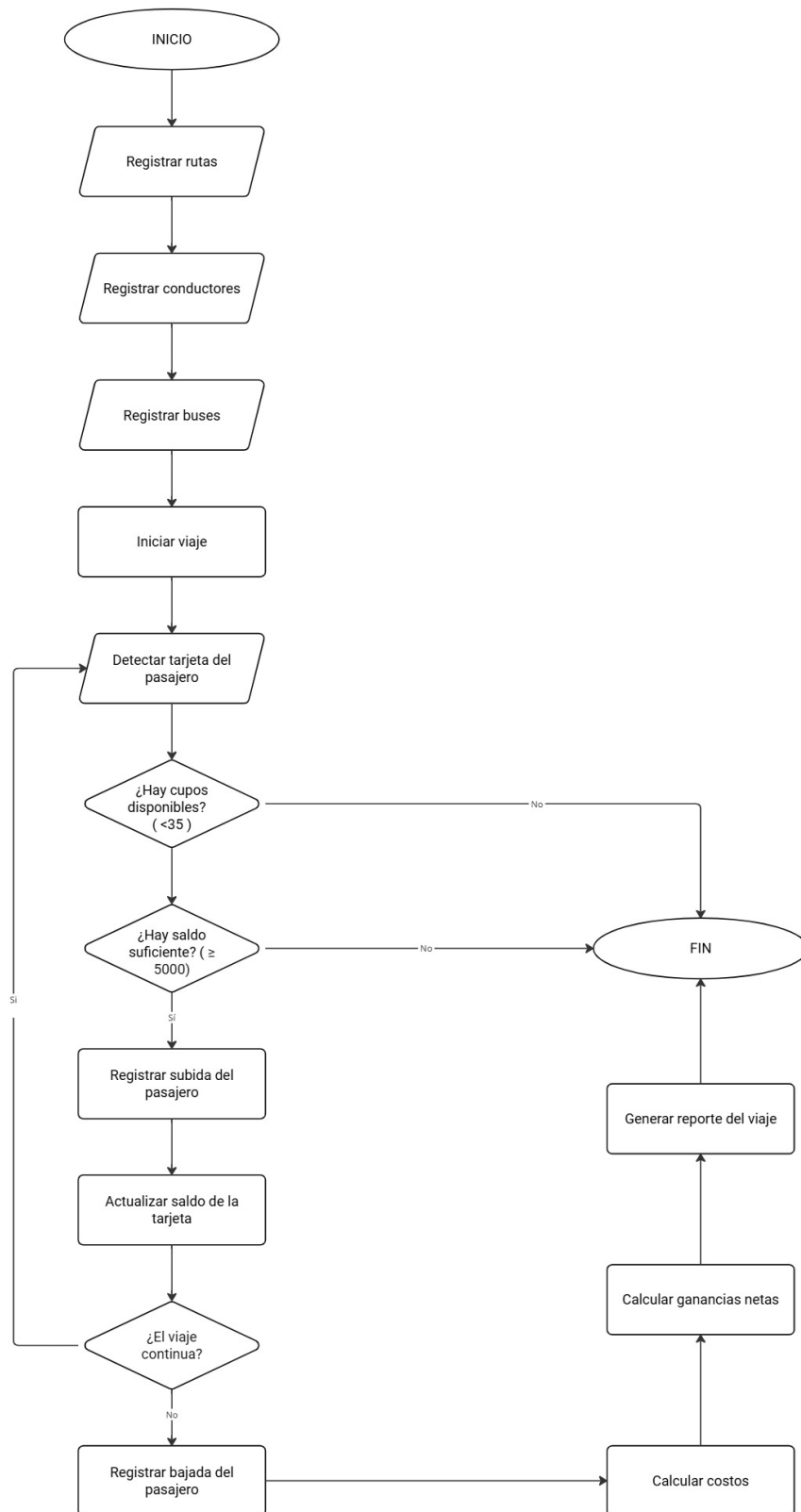


Figura 1: Diagrama de flujo del sistema de transporte

6 Clases Principales

6.1 Clase Ruta

Atributos	Descripción
– idRuta : int	Identificador único de la ruta
– destino : str	Punto final de la ruta
– puntoOrigen : str	Punto de inicio de la ruta
– paradas : list<str>	Lista de paradas intermedias
– horaSalida : time	Hora de salida programada
– horaLlegadaEstimada : time	Hora estimada de llegada
– conductorAsignado : str	Conductor asignado a la ruta
– busAsignado : str	Bus asignado a la ruta
Métodos	Descripción
+ asignarConductor(conductor : Conductor)	Asigna un conductor a la ruta
+ asignarBus(bus : Bus)	Asigna un bus a la ruta
+ actualizarCupo(ingreso : boolean)	Actualiza la disponibilidad de cupos
+ obtenerDisponibilidad() : int	Retorna cupos disponibles
+ actualizarUbicacion(nuevaUbicacion : String)	Actualiza ubicación actual
+ calcularGanancias() : double	Calcula ganancias de la ruta

6.2 Clase Bus

Atributos	Descripción
– idBus : int	Identificador único del bus
– placa : str	Placa del vehículo
– capacidad : int = 35	Capacidad máxima de pasajeros
– ubicacionActual : str	Ubicación actual del bus
– rutaAsignada : int	Ruta asignada al bus
– pasajerosActuales : int	Número actual de pasajeros
Métodos	Descripción
+ asignarConductor(conductor : Conductor)	Asigna conductor al bus
+ actualizarUbicacion(ubicacion : String)	Actualiza ubicación del bus
+ asignarRuta(ruta : Ruta)	Asigna una ruta al bus
+ obtenerCupoDisponible() : int	Retorna cupos disponibles

6.3 Clase Conductor

Atributos	Descripción
– idConductor : int	Identificador único del conductor
– nombre : str	Nombre completo del conductor
– cedula : str	Número de cédula
– telefono : str	Número de teléfono
– rutaActual : int	Ruta actual asignada
– totalGanancias : float	Ganancias acumuladas
Métodos	Descripción
+ asignarRuta(ruta : Ruta)	Asigna una ruta al conductor
+ registrarGanancia(monto : float)	Registra ganancias obtenidas
+ obtenerGanancias() : float	Retorna total de ganancias

6.4 Clase Pasajero

Atributos	Descripción
– idPasajero : int	Identificador único del pasajero
– saldoTarjeta : float	Saldo disponible en la tarjeta
– ubicacionActual : str	Ubicación actual del pasajero
– enBus : boolean	Indica si está en un bus
Métodos	Descripción
+ pagarPasaje(tarifa : float)	Realiza pago del pasaje
+ registrarIngreso(bus : Bus)	Registra ingreso a un bus
+ registrarSalida(bus : Bus)	Registra salida de un bus
+ recargarTarjeta(monto : float)	Recarga saldo de la tarjeta

6.5 Clase Viaje

Atributos	Descripción
– idViaje : int	Identificador único del viaje
– ruta : int	Ruta del viaje
– bus : int	Bus utilizado
– conductor : int	Conductor asignado
– pasajerosTransportados : int	Total de pasajeros transportados
– gananciaGenerada : float	Ganancias del viaje
– fecha : date	Fecha del viaje
Métodos	Descripción
+ registrarPasajero()	Registra un pasajero en el viaje
+ registrarGanancias(total : double)	Registra ganancias obtenidas
+ cerrarViaje()	Finaliza el viaje y genera reporte

7 Plan de Trabajo y Organización

7.1 Roles del Equipo

Rol	Responsable
Líder Técnico	X
Desarrollo Principal	X
Documentación	X
Pruebas y Calidad	X

Cuadro 1: Distribución de roles del equipo

7.2 Cronograma Semanal

Semana	Tareas Principales	Responsables
1	Definición de clases y estructura POO	X
2	Implementación módulo Rutas y Conductores	X
3	Desarrollo módulo Pasajeros y Tarjetas	X
4	Integración y pruebas del sistema	X
5	Depuración y documentación final	X

Cuadro 2: Cronograma de desarrollo del proyecto

8 Viabilidad Técnica y Herramientas

8.1 Tecnologías a Utilizar

- **Lenguaje:** Python 3.x
- **Paradigma:** Programación Orientada a Objetos (POO)
- **Almacenamiento:** Archivos CSV
- **Control de versiones:** GitHub
- **Interfaz:** Consola

8.2 Justificación Técnica

Python es ideal para este proyecto debido a:

- Sintaxis clara para implementar las clases definidas
- Capacidades nativas para manejo de archivos
- Amplia documentación y comunidad
- Compatibilidad con los temas vistos en clase (POO, estructuras de datos, archivos)
- Facilidad para implementar el diagrama de flujo propuesto