Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Cliente	**El Gobierno Municipal de Palmira**
Usuario	Autoridades de la Ciudad de Palmira (Policía/Seguridad, Bomberos/Servicios de emergencia y gestores de tránsito) Conductores de transporte Público y Privado Ciudadanos
Requerimientos funcionales	*1. Gestión de Rutas de Transporte* *RF1 - El sistema debe permitir registrar nuevas rutas de transporte con su respectivo ID, distancia, tiempo estimado, punto de inicio y punto final.* *RF2 - El sistema debe permitir consultar las rutas registradas.* *RF3 - El sistema debe permitir ordenar las rutas por distancia (de menor a mayor) o por tiempo estimado (de menor a mayor).* *RF4 - El sistema debe permitir buscar la mejor ruta disponible con base en un criterio definido por los desarrolladores (ejemplo: ruta más corta, menor tiempo, etc.).* *2. Gestión de Incidentes de Seguridad* *RF5 - El sistema debe permitir registrar incidentes con su respectivo ID, tipo (robo, accidente, incendio, etc.), ubicación, fecha/hora del reporte, descripción y estado (pendiente, en proceso, resuelto).* *RF6 - El sistema debe permitir consultar los incidentes registrados.* *RF7 - El sistema debe permitir ordenar los incidentes por fecha/hora (del más reciente al más antiguo).* *RF8 - El sistema debe permitir buscar un incidente por su ID.* *3. Gestión de Conductores y Pasajeros* *RF9 - El sistema debe permitir registrar conductores con su respectivo ID, nombre, vehículo asignado y estado (disponible, en ruta).*

	RF10 - El sistema debe permitir registrar pasajeros con su respectivo ID, nombre, ruta asignada y contacto. *RF11 - El sistema debe permitir buscar un conductor por su nombre.* *4. Manejo de Archivos (Entrada/Salida de Datos)* *RF12 - El sistema debe permitir leer y escribir datos en archivos JSON, incluyendo rutas, incidentes, pasajeros y conductores.* *5. Reportes* *RF13 - El sistema debe generar un reporte en consola con las rutas ordenadas por distancia o tiempo estimado.* *RF14 - El sistema debe generar un reporte en consola con los incidentes ordenados por fecha/hora.* *RF15 - El sistema debe mostrar los resultados de las búsquedas específicas de incidentes por ID y conductores por nombre.* *6. Manejo de Excepciones y Errores* *RF16 - El sistema debe manejar errores comunes en la ejecución, como archivos JSON corruptos, datos mal formateados o búsquedas sin resultados.* *RF17 - El sistema debe implementar excepciones personalizadas para situaciones específicas, como intentos de registrar datos duplicados o búsqueda de elementos inexistentes.*
Contexto del problema	La ciudad de **Palmira, en el Valle del Cauca (Colombia)**, ha experimentado un **rápido crecimiento económico y poblacional**, lo que ha traído consigo desafíos significativos en términos de **seguridad y movilidad**. **Problemas actuales:** 1. **Seguridad:** Altos índices de delincuencia, robos y emergencias sin un sistema centralizado de monitoreo. 2. **Movilidad:** Congestión vehicular debido a la falta de rutas optimizadas y ausencia de información en tiempo real. 3. **Acceso a la información:** Ciudadanos y autoridades carecen de datos actualizados para la toma de decisiones estratégicas. Para abordar estos problemas, el **Gobierno Municipal de Palmira** ha decidido desarrollar el **Sistema de Gestión y Monitoreo de Movilidad y Seguridad (SGMMS)**. Este sistema permitirá **monitorear incidentes de**

	seguridad, optimizar rutas de transporte y proporcionar información en tiempo real a ciudadanos y autoridades. **Objetivos principales del SGMMS:** - **Monitoreo en tiempo real** de vehículos de emergencia y patrullas de seguridad. - **Respuesta rápida y eficiente** a incidentes de seguridad y emergencias. - **Información en tiempo real** sobre tráfico y transporte público. - **Optimización de rutas** con base en datos históricos y actuales. - **Generación de reportes y estadísticas** para la toma de decisiones. - **Notificaciones a la comunidad** sobre incidentes, rutas alternativas y recomendaciones de seguridad. El desarrollo del **prototipo inicial** del SGMMS estará enfocado en la gestión de **rutas de transporte, incidentes de seguridad, pasajeros y conductores**, garantizando el manejo adecuado de datos mediante estructuras dinámicas y algoritmos de búsqueda y ordenamiento.
Requerimientos no funcionales	**RNF1** - El sistema debe manejar errores comunes en la ejecución y proporcionar mensajes claros al usuario sobre situaciones inesperadas. **RNF2** - El sistema debe implementar **excepciones personalizadas** para el manejo de errores específicos. **RNF3** - El sistema debe utilizar **listas enlazadas simples, dobles o circulares** para almacenar y manipular información sobre rutas e incidentes. **No se permite el uso de ArrayList**. **RNF4** - El sistema debe permitir el almacenamiento y recuperación de datos en **archivos JSON**. **RNF5** - Se debe reportar **indicadores de calidad del software** en al menos **10 commits**, midiendo: • **Densidad de errores-fallos** = Total de fallos / Total de pruebas. • **Confiabilidad** = 1 - Densidad de fallos. • **Completitud** = Casos de prueba / Total de funcionalidades. **RNF6** - El sistema debe generar **reportes en consola**, mostrando: • Rutas ordenadas por distancia o tiempo. • Incidentes ordenados por fecha/hora. • Resultados de búsquedas específicas.

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	***RF1*** *- Registrar nuevas rutas de transporte*		
Resumen	*El sistema debe permitir registrar nuevas rutas de transporte con su respectivo ID, distancia, tiempo estimado, punto de inicio y punto final.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	idRuta	String	*No debe estar vacío, no debe repetirse en el sistema.*
	distancia	Double	*Debe ser mayor que 0.*
	tiempoEstimado	Int	*Debe ser mayor que 0 (minutos).*
	puntoInicio	String	*No debe estar vacío.*
	puntoFin	String	No debe estar vacío.
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	mensajeConfirmacion	String	*Ej. "Ruta registrada con éxito: [idRuta]"*
	mensajeErrorRegistro	String	Ej. "Error: datos inválidos o ruta duplicada"

Identificador y nombre	*RF2 – Consultar rutas registradas*		
Resumen	*El sistema debe permitir consultar las rutas que se han registrado.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
			Podría no requerir parámetros, o un filtro opcional (ej. ID).
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	listaRutas	Lista/Array de Ruta	*Cada Ruta con sus atributos (ID, distancia, etc.).*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF3 - Ordenar rutas*		
Resumen	*[El resumen del RF debe ser claro, preciso, conciso, factible, entendible, verificable, cuantificable, completo, buena ortografía, puntuación y gramática. Debe incluir las entradas, actividades y condiciones necesarias para transformar las entradas en salidas, las salidas y la postcondición.]*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	criterioOrden	String	*Debe ser "distancia" o "tiempo".*
	listaRutas	Lista/Array de Ruta	Debe contener al menos 1 elemento para poder ordenar.
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	listaRutasOrdenadas	Lista/Array de Ruta	*Rutas ordenadas según el criterio (distancia o tiempo).*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF4 - Buscar la mejor ruta*		
Resumen	*El sistema debe permitir buscar la mejor ruta disponible con base en un criterio definido (ej. menor distancia, menor tiempo, etc.).*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	listaRutas	Lista/Array de Ruta	*Debe tener rutas disponibles.*
	criterioBusqueda	String	Ej. "menorDistancia", "menorTiempo", etc.
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	rutaOptima	Objeto Ruta	*Retorna la ruta que cumpla el criterio óptimo.*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF5 - Registrar incidentes*		
Resumen	*El sistema debe permitir registrar incidentes con ID, tipo, ubicación, fecha/hora, descripción y estado.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	idIncidente	String	*No debe estar vacío, no debe repetirse.*
	tipo	String	*Ej. "robo", "accidente", "incendio", etc.*
	ubicación	String	*No debe estar vacío.*
	fechaHora	String	*Formato válido (ej. YYYY-MM-DD HH:MM).*
	descripcion	String	*(Opcional) Máximo 255 caracteres.*
	estado	String	"pendiente", "en proceso", "resuelto".
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	mensajeConfirmacion	String	*Ej. "Incidente registrado con éxito: [idIncidente]"*
	errorRegistro	String	Ej. "Error: datos inválidos o incidente duplicado"

Identificador y nombre	*RF6 - Consultar incidentes registrados*		
Resumen	*El sistema debe permitir consultar la lista de incidentes registrados.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	(N/A)	(N/A)	*Podría tener filtros opcionales (ej. estado).*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	listaIncidentes	Lista/Array de Incidente	*Cada objeto con atributos (ID, tipo, fecha/hora, etc.).*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF7 - Ordenar incidentes por fecha/hora*		
Resumen	*El sistema debe permitir ordenar los incidentes por fecha/hora (del más reciente al más antiguo).*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	listaIncidentes	Lista/Array de Incidente	*Debe contener al menos 1 incidente para ordenar.*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	listaIncidentesOrdenados	Lista/Array de Incidente	*Incidentes ordenados por fecha/hora descendente (más reciente).]*

Identificador y nombre	*RF8 - Buscar un incidente por ID*		
Resumen	*El sistema debe permitir buscar un incidente específico ingresando su ID.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	idIncidente	String	*No debe estar vacío, debe existir en el sistema.*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	incidenteEncontrado	Objeto Incidente	*Retorna el incidente si se encuentra, o null/error si no existe.*

Identificador y nombre	*RF9 - Registrar conductores*		
Resumen	*El sistema debe permitir registrar conductores con su ID, nombre, vehículo asignado y estado (disponible, en ruta).*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	idConductor	String	*No debe estar vacío, no debe repetirse.*
	nombreConductor	String	*No debe estar vacío.*
	vehiculoAsignado	String	*No debe estar vacío.*
	estadoConductor	String	"disponible" o "en ruta".
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	mensajeConfirmacion	String	*Ej. "Conductor registrado con éxito: [idConductor]"*
	errorRegistro	String	Ej. "Error: datos inválidos o conductor duplicado"

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF10 - Registrar pasajeros*		
Resumen	*El sistema debe permitir registrar pasajeros con su ID, nombre, ruta asignada y contacto.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	idPasajero	String	*No debe estar vacío, no debe repetirse.*
	nombrePasajero	String	*No debe estar vacío.*
	rutaAsignada	String	*Puede ser el ID de la ruta, si existe.*
	contacto	String	No debe estar vacío (teléfono o email).
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	mensajeConfirmacion	String	*Ej. "Pasajero registrado con éxito: [idPasajero]"*
	errorRegistro	String	Ej. "Error: datos inválidos o pasajero duplicado"

Identificador y nombre	*RF11 - Buscar un conductor por nombre*		
Resumen	*El sistema debe permitir buscar un conductor específico ingresando su nombre.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	nombreConductor	String	*No debe estar vacío. Debe existir un conductor con ese nombre.*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	conductorEncontrado	Objeto Conductor	*Retorna el conductor con ese nombre o null/mensaje de error si no existe.*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF12 - Leer y escribir datos en JSON*		
Resumen	*El sistema debe permitir leer y escribir datos en archivos JSON, incluyendo rutas, incidentes, pasajeros y conductores.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	rutaArchivoJSON	String	*Debe ser una ruta válida del sistema de archivos.*
	datosAGuardar	Estructura/Lista de objetos (Rutas, etc.)	Debe contener información en el formato esperado para JSON.
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	resultadoLectura	Boolean/String	*Ej. "Lectura de archivo JSON exitosa"*
	resultadoEscritura	Boolean/String	Ej. "Escritura de archivo JSON exitosa"
	excepcionArchivos	Excepción/ String	Ej. "Error: archivo JSON corrupto"

Identificador y nombre	*RF13 - Reporte de rutas ordenadas*		
Resumen	*El sistema debe generar un reporte en consola con las rutas ordenadas por distancia o tiempo estimado.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	listaRutas	Lista/Array de Ruta	*Debe contener al menos 1 ruta para reportar.*
	criterioReporte	String	"distancia" o "tiempo".
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	reporteRutas	String	*Texto en consola mostrando las rutas ordenadas (ID, distancia, etc.).*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF14 - Reporte de incidentes ordenados*		
Resumen	*El sistema debe generar un reporte en consola con los incidentes ordenados por fecha/hora (más reciente al más antiguo).*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	listaIncidentes	Lista/Array de Incidente	*Debe contener al menos 1 incidente.*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	reporteIncidentes	String	*Texto en consola mostrando los incidentes ordenados (ID, fecha/hora).*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF15 - Resultados de búsquedas (incidentes y conductores)*		
Resumen	*El sistema debe mostrar los resultados de las búsquedas específicas de incidentes por ID y conductores por nombre.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	idIncidente *(opt.)*	String	*No debe estar vacío para buscar incidente.]*
	nombreConductor *(opt.)*	String	No debe estar vacío para buscar conductor.
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	resultadoBusqueda	String	*Texto en consola mostrando el incidente/conductor encontrado o error si no existe.*

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF16 - Manejo de errores comunes*		
Resumen	*El sistema debe manejar errores comunes (archivos JSON corruptos, datos mal formateados, búsquedas sin resultados, etc.) con mensajes claros.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	(depende de la operación)	(varía)	*Pueden ser rutas de archivo, datos de registro, etc.*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	mensajeError	String	*Ej. "Error: datos inválidos", "Archivo corrupto", etc.*
	excepcionPersonalizada	Clase Excepción propia	Ej. DatosDuplicadosException("Ruta ya existe")

Universidad Icesi
Departamento de Computación y Sistemas Inteligentes
Identificación del problema y análisis de requerimientos
Algoritmos y Programación II

Identificador y nombre	*RF17 - Excepciones personalizadas*		
Resumen	*El sistema debe implementar excepciones propias (datos duplicados, elemento inexistente, etc.) para casos específicos.*		
Entradas	**Nombre entrada**	**Tipo de dato**	**Condición valores válidos**
	(N/A)	(N/A)	*Se lanzan cuando ocurren condiciones de error específicas.*
Resultado o Postcondición			
Salidas	**Nombre salida**	**Tipo de dato**	**Formato**
	excepcionPersonalizada	Clase de Excepción propia	*Mensaje descriptivo, ej. "Datos duplicados: [ID]".*