dev Senior Java

RETO 2

Sistema de Gestión de Emergencias Urbanas

RETO 2 - Sistema de Gestión de Emergencias Urbanas

Introducción

En este reto, titulado "Sistema de Gestión de Emergencias Urbanas", los estudiantes desarrollarán una aplicación basada en consola que facilite la coordinación de servicios de emergencia en una ciudad, incluyendo bomberos, ambulancias y policía. Este ejercicio busca replicar situaciones reales en las que los sistemas de software juegan un papel crucial para la toma de decisiones rápidas y efectivas. El objetivo es crear un programa que permita gestionar diferentes tipos de emergencias, asignar recursos según la prioridad y evaluar el rendimiento del sistema. A través de esta actividad, los estudiantes practicarán habilidades avanzadas de diseño y programación, mientras comprenden la resolución de problemas en sistemas críticos.

Objetivos

1. Modelar el Sistema de Emergencias
    - Crear clases para los servicios de emergencia (Bomberos, Ambulancia, Policía), cada una con atributos y métodos específicos.
    - Definir una clase principal *Emergencia* que almacene información como tipo, ubicación, nivel de gravedad y tiempo de respuesta.
2. Aplicar Encapsulamiento y Herencia
    - Diseñar una jerarquía de clases para diferentes tipos de emergencias (Incendio, Accidente Vehicular, Robo).
    - Proteger datos sensibles como ubicaciones críticas o identidades mediante modificadores de acceso.
3. Usar Polimorfismo e Interfaces
    - Crear una interfaz *Responder* con métodos como *atenderEmergencia* y *evaluarEstado*.
    - Implementar la interfaz en los servicios de emergencia y personalizar su comportamiento según el tipo de emergencia.
4. Incorporar Patrones de Diseño Básicos
    - Utilizar el patrón *Factory* para crear instancias de emergencias según su tipo.
    - Aplicar el patrón *Singleton* para centralizar la gestión de recursos disponibles en un único controlador del sistema.
5. Implementar Estrategias Avanzadas
    - Usar el patrón *Strategy* para definir cómo se priorizan las emergencias (por ejemplo, gravedad alta vs. cercanía a la base de operaciones).

- Implementar el patrón *Observer* para notificar automáticamente a los servicios relevantes cuando ocurre una nueva emergencia.

6. Integrar Clases Internas y Lambda
   - Implementar clases internas para modelar mapas urbanos, permitiendo verificar la proximidad de los recursos a las emergencias.
   - Usar expresiones lambda para filtrar listas de recursos disponibles según criterios específicos (distancia, tiempo de respuesta).
7. Evaluar y Optimizar el Sistema
   - Calcular métricas de desempeño, como el tiempo promedio de respuesta, emergencias atendidas y recursos disponibles al final de cada jornada.
   - Evaluar la eficacia de las decisiones tomadas, optimizando la asignación de recursos según los resultados.

Beneficios del Reto

1. AplicaciónRealdeProgramaciónOrientadaaObjetos
   - Los estudiantes trabajarán en un sistema relacionado con situaciones reales y críticas.
2. Análisis y Resolución de Problemas Complejos
   - Diseñarán soluciones a problemas multifacéticos, abordando priorización, optimización y trabajo con datos en tiempo real.
3. Uso Avanzado de Patrones de Diseño y Estructuras de Datos
   - Practicarán cómo aplicar patrones para diseñar sistemas robustos y escalables, preparando el terreno para proyectos profesionales.
4. Comprensión de la Integración de Sistemas en Infraestructuras Urbanas
   - Los estudiantes entenderán cómo los sistemas de software se integran en la gestión de infraestructuras urbanas y el impacto de las decisiones en situaciones críticas.

Ejemplo de Flujo del Programa

1. Inicio del Programa
   - Configuración inicial del sistema, incluyendo la asignación de recursos disponibles.
2. Menú de Opciones.

- Registrar una nueva emergencia, especificando tipo, gravedad y ubicación.
- Ver el estado actual de los recursos (vehículos, personal, combustible).
- Atender una emergencia (asignar recursos y resolverla según la prioridad).
- Mostrar estadísticas del día (emergencias atendidas, tiempo promedio de respuesta).

3. Finalizar el Programa
    - Guardar un registro del día, incluyendo métricas clave, y preparar el sistema para el siguiente ciclo.

Requerimientos Funcionales para el Sistema de Gestión de Emergencias Urbanas

1. RegistrodeEmergencias
    - Presentar un menú con diferentes tipos de emergencias (Incendio, Accidente Vehicular, Robo, etc.).
    - Solicitar detalles específicos sobre la emergencia, como su ubicación, nivel de gravedad (bajo, medio, alto) y tiempo estimado de atención inicial.
    - Mostrar un resumen del incidente y confirmar el registro.
2. Asignación de Recursos
    - Consultar la disponibilidad de recursos de emergencia (bomberos, ambulancias, unidades policiales) basados en la ubicación y tipo de emergencia.
    - Calcular los recursos necesarios según la gravedad del evento (por ejemplo, un incendio grande podría necesitar dos camiones de bomberos y tres paramédicos).
    - Permitir al usuario confirmar o ajustar la asignación de recursos antes de enviarlos.
3. Gestión de Recursos
    - Mostrar el estado actual de los recursos, como vehículos disponibles, personal activo, y combustible restante.
    - Permitir al usuario reasignar recursos entre emergencias si se agotan o si se presentan situaciones inesperadas.

4. Monitoreo del Estado de las Emergencias

    - Mostrar el progreso de atención de cada emergencia (porcentaje de resolución) en tiempo real.
    - Indicar si alguna emergencia no fue atendida a tiempo o si se resolvió con éxito.
    - Proporcionar un resumen al final de la jornada, incluyendo emergencias atendidas, recursos gastados, y tiempo promedio de respuesta.

5. Interacción con el Usuario
    - El sistema debe ofrecer un menú interactivo con opciones claras: registrar una nueva emergencia, ver el estado actual de los recursos, atender una emergencia, y terminar la jornada.
    - Validar la entrada del usuario para evitar errores (por ejemplo, asegurarse de que seleccionen un tipo de emergencia válido o asignen recursos disponibles).
6. Estadísticas y Evaluación de Desempeño
    - Calcular y mostrar métricas al final de la jornada, como:
        - Cantidad de emergencias atendidas.
        - Tiempo promedio de respuesta por tipo de emergencia.
        - Recursos gastados vs. recursos disponibles.
    - Permitir al usuario analizar los resultados para planificar mejor futuras jornadas.

Descripción General del Flujo del Sistema

El flujo del sistema de gestión de emergencias urbanas está diseñado para permitir una gestión eficiente y clara de las emergencias en una ciudad. El sistema sigue un ciclo continuo que permite registrar emergencias, asignar recursos, monitorizar el progreso de las atenciones y evaluar el rendimiento del sistema. A continuación, se describe el flujo general del sistema:

1. IniciodelSistema:
    - o Elsistemacomienzaconunaconfiguracióninicialenlaqueseasignan los recursos disponibles, como vehículos, personal y otros elementos

necesarios para la atención de emergencias. Esta fase garantiza que el sistema tenga toda la información requerida para operar.

2. Menú de Opciones:
    - Una vez iniciado el sistema, el usuario se enfrenta a un menú interactivo que le permite seleccionar entre diversas opciones:
        - Registrar una nueva emergencia: El usuario puede ingresar información sobre la emergencia, como el tipo (incendio, accidente vehicular, robo, etc.), la ubicación y el nivel de gravedad.
        - Ver el estado de los recursos disponibles: El sistema muestra el estado actual de los recursos, tales como vehículos, personal y combustible disponible.
        - Atender una emergencia: El usuario puede asignar recursos a una emergencia específica basándose en la prioridad determinada por la gravedad del evento y su proximidad.
        - Mostrar estadísticas: Al finalizar la jornada o cuando se desee evaluar el desempeño del sistema, el usuario puede ver métricas clave, como el número de emergencias atendidas, tiempo promedio de respuesta, y recursos utilizados.
3. Registro y Asignación de Emergencias
    - Cuando se registra una nueva emergencia, el sistema solicita al usuario detalles sobre el incidente y calcula automáticamente los recursos necesarios para atenderlo, según la gravedad y el tipo de emergencia.
    - El usuario puede ajustar la asignación de recursos antes de confirmar el e nvío.
4. Gestión de Recursos:
    - El sistema monitorea constantemente el estado de los recursos. Si los recursos se agotan o se requieren más unidades, el sistema ofrece la opción de reasignar recursos entre emergencias.
    - Los recursos son consumidos a medida que se utilizan, por lo que el sistema también simula la progresiva disminución de la disponibilidad a lo largo del día.
5. Monitoreo de Emergencias:
    - A lo largo del día, el sistema muestra el progreso de las emergencias en tiempo real, indicándole al usuario el porcentaje de resolución de cada incidente.

- El sistema también avisa cuando una emergencia no se atiende dentro del tiempo esperado, o si se resuelve exitosamente.

6. Evaluación del Desempeño del Sistema:
   - Al final del ciclo de trabajo o jornada, el sistema ofrece un resumen de las emergencias atendidas, los recursos gastados, y el tiempo promedio de respuesta.
   - Las métricas son generadas y mostradas al usuario para su análisis y reflexión. El sistema permite comparar los recursos gastados versus los disponibles, ofreciendo una visión clara de la eficiencia operativa del sistema.
7. Cierre del Sistema:
   - Una vez finalizada la jornada, el sistema guarda un registro de las operaciones realizadas, que incluye las estadísticas y cualquier detalle relevante sobre las emergencias atendidas.
   - El sistema se prepara para el siguiente ciclo, garantizando que los datos y recursos estén listos para ser utilizados nuevamente.

Metodología para el Reto 2: Sistema de Emergencias Urbanas

1. Trabajo en Parejas
   - o FormacióndeParejas: Los estudiantes se organizarán en parejas para realizar el reto. La colaboración será esencial para repartir responsabilidades y abordar problemas de manera conjunta.
   - División de Tareas: Se recomienda que las parejas distribuyan las tareas de manera equitativa. Un miembro podría encargarse de la lógica del sistema (gestión de emergencias, asignación de recursos), mientras que el otro se enfoque en la interfaz de usuario y validación de entradas.
2. Uso de Git y GitHub
   - o CreacióndelRepositorioenGitHub: Cada pareja creará un repositorio en GitHub para almacenar el código fuente. El repositorio debe ser público para revisión.
   - Uso de Ramas y Control de Versiones:
     - Se recomienda que cada estudiante trabaje en una rama separada.
     - Una vez completada las funcionalidades, deben fusionarlas ramas
   - Commits Regulares: Los estudiantes deberán realizar commits frecuentes con mensajes claros.

3. Video de Presentación

    - GrabacióndelVideo: Una vez completado el proyecto, los estudiantes grabarán un video donde presentarán el sistema y los desafíos que enfrentaron.
    - Contenido del Video:
        - Introducción al proyecto, presentación de los desafíos y soluciones.
        - Demostración de la ejecución del sistema.
        - Participación equilibrada de ambos estudiantes en la presentación.

4. Revisión y Feedback

    - EntregadelReto: Los estudiantes compartirán el enlace al repositorio de GitHub y el video de presentación.
    - Feedback: Los docentes revisarán el código y el video, proporcionando retroalimentación sobre la implementación, el uso de Git, y la presentación del proyecto.

Rúbrica de Evaluación para el Sistema de Gestión de Emergencias

Esta rúbrica detalla los criterios de evaluación para el desarrollo del Sistema de Gestión de Emergencias. Está diseñada para asegurar que se aborden todos los aspectos del proyecto, desde la implementación técnica hasta la presentación y documentación del trabajo. La calificación total será de 100 puntos.

Criterio	Descripción	Puntos
1. Implementación técnica.		**50**
Registro y Asignación de Emergencias	El sistema permite registrar emergencias de manera eficiente, con opciones claras para ingresar tipo, ubicación y gravedad de las emergencias.	10

Criterio	Descripción	Puntos
Gestión de Recursos	El sistema gestiona correctamente los recursos (vehículos, personal, equipos) asignandolos adecuadamente según la emergencia.	10
Simulación de Progreso	El sistema simula y muestra el progreso de la atención a las emergencias, con actualizaciones sobre el estado de cada incidente.	10
Interacción con el Usuario	El menú del sistema permite una navegación fluida y clara permitiendo seleccionar opciones, registrar emergencias y consultar el estado de los recursos.	10
Uso de funciones y modularidad	El menú del sistema permite una navegación fluida y clara permitiendo seleccionar opciones, registrar emergencias y consultar el estado de los recursos.	10
2. Uso de Git y GitHub		**20**
Uso de ramas y commits regulares	Se utilizaron ramas para separar funcionalidades y se realizaron commits frecuentes, permitiendo un desarrollo organizado y sin conflictos.	10
Mensajes de commits descriptivos	Los mensajes de los commits son claros y explican de manera precisa los cambios realizados en cada fase del desarrollo.	10
3. Presentación del video.		**20**
Introducción y descripción del reto.	Se presenta de manera clara el reto, describiendo los objetivos y la funcionalidad del sistema de gestión de emergencias.	5
demostración de la funcionalidad	El video muestra el funcionamiento completo del sistema incluyendo el registro de emergencias, la asignación de recursos y el monitoreo del progreso.	10
Claridad y equidad en la explicación	La presentación es clara, con explicaciones comprensibles de los diferentes aspectos del proyecto, y la participación es equitativa entre los presentadores.	5
4. Documentación		**10**

Criterio	Descripción	Puntos
Comentarios en el código	El código está bien comentado, con explicaciones claras sobre las funciones y bloques clave del sistema.	5
README en Gi tHub	El repositorio contiene un archivo README bien redactado, que describe el proyecto, cómo ejecutarlo, sus características y el uso del sistema.	5

Desglose de Puntos:

- ImplementaciónTécnica : 50 puntos
- UsodeGityGitHub : 20 puntos
- PresentacióndelVideo : 20 puntos
- Documentación : 10 puntos

Total Máximo: 100 puntos

Puntuación Final:

- Excelente(90-100puntos) : El sistema es completamente funcional, con una implementación técnica robusta y bien estructurada. El uso de Git/GitHub es adecuado y organizado, y la presentación en video es clara y detallada.
- Bueno (80-89 puntos): El sistema es funcional, pero presenta algunos detalles mejorables, como la optimización de la gestión de emergencias o la documentación. La interacción en el video es adecuada, pero podría ser más clara en algunos puntos.
- Satisfactorio (70-79 puntos): El sistema cumple con los requisitos básicos, pero presenta errores importantes en la funcionalidad o en la organización del proyecto. La presentación y la documentación necesitan mejoras sustanciales.
- Insuficiente (menos de 70 puntos): El sistema tiene deficiencias graves en la implementación, en el uso de Git/GitHub o en la presentación en video. La documentación es mínima o inexistente.