Reservation Management System for Temporary Public Spaces

Sistema de Gestión de Reservas para Espacios Públicos Temporales

**Nombre de autores**

Kevin Andres Henao Ospina

Jorge Augusto Borbon Romero

Luis Antonio Niño Mellizo

Politecnico Grancolombiano

# Resumen

Este trabajo presenta el desarrollo de un sistema CRUD (Crear, Leer, Actualizar, Eliminar) para la gestión de reservas de espacios públicos, implementado en Java utilizando JavaFX y el patrón de diseño MVC. El sistema permite realizar todas las operaciones básicas sobre los datos de reservas, garantizando una gestión eficiente de los espacios.

Además, se utiliza **Git** como sistema de control de versiones distribuido para gestionar el código del proyecto. Git facilita el trabajo colaborativo, manteniendo un historial de cambios claro y ordenado, permitiendo a los desarrolladores realizar modificaciones de manera simultánea sin perder la integridad del proyecto. Durante el desarrollo, Git asegura la coordinación y el seguimiento efectivo del avance del sistema.

# Palabras clave

crud, patrón mvc, javafx, control de versiones, git, gestión de bases de datos

# INTRODUCCIÓN

# El desarrollo de sistemas de gestión de información es esencial para la optimización de procesos en diversas áreas, como la administración de reservas de espacios públicos. En este contexto, los sistemas CRUD (Crear, Leer, Actualizar, Eliminar) permiten gestionar de manera eficiente los datos, facilitando la interacción de los usuarios con la información. Para estructurar y organizar el desarrollo, el patrón de diseño MVC (Modelo-Vista-Controlador) es ampliamente utilizado, ya que separa las responsabilidades de la lógica del negocio, la interfaz de usuario y el control de las interacciones, mejorando la escalabilidad y el mantenimiento del sistema.

# El lenguaje de programación Java, junto con JavaFX para la interfaz gráfica, provee una plataforma robusta para desarrollar este tipo de aplicaciones. A su vez, el uso de Git como sistema de control de versiones distribuido resulta indispensable para gestionar los cambios en el código, permitiendo la colaboración entre múltiples desarrolladores y asegurando un historial claro y detallado de cada modificación realizada durante el ciclo de vida del proyecto.

# Este trabajo se centra en la implementación de un sistema CRUD para reservas de espacios públicos, haciendo uso de tecnologías modernas y buenas prácticas de desarrollo, con un enfoque en la eficiencia y la facilidad de uso del sistema

# MÉTODO

1. Representar de forma gráfica y auténtica sobre los conceptos a usar el desarrollo de la  
aplicación como: los pilares de la programación orientada a objetos, la arquitectura Modelo-  
Vista-Controlador, controlador de versiones, Java Collection Framework, Javadoc y Junit.

Diagrama

Descripción generada automáticamente Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

2. Usando Chatgpt se determina el enunciado de la aplicación a construir. Para ello utilice el  
siguiente prompt:  
a. Enunciado de un problema poco común en una ciudad que  
requiera un único CRUD en una aplicación sobre JavaFx.

3. Copie el resultado del prompt anterior en tal cual como aparece en el documento.

Un problema poco común en una ciudad podría ser la gestión de espacios públicos temporales para eventos en plazas, parques o calles. La administración de una ciudad puede permitir que organizaciones o personas reserven estos espacios para actividades como mercados al aire libre, ferias, o eventos culturales. Sin embargo, muchas veces no existe un sistema adecuado para gestionar las solicitudes, aprobaciones y calendario de ocupación de estos espacios.

Enunciado del problema:

En la ciudad de "Villa Central", el gobierno local enfrenta dificultades para gestionar la reserva y el uso de espacios públicos temporales. Actualmente, el proceso de reserva es manual, lo que genera confusión y errores en la asignación de los espacios. Se necesita desarrollar una aplicación que permita a los administradores de la ciudad gestionar las solicitudes de reserva, asegurando que no haya conflictos de horarios, aprobaciones pendientes o eventos simultáneos en el mismo espacio.

Requerimiento:

Desarrollar un sistema de gestión de reservas de espacios públicos temporales con un único CRUD que permita:

1. **Crear** una nueva solicitud de reserva de espacio público, donde se especifiquen:
   * Nombre del evento.
   * Nombre del organizador.
   * Fecha y hora de inicio y finalización del evento.
   * Descripción del evento.
   * Ubicación específica solicitada.
2. **Leer** todas las solicitudes registradas, mostrando una lista con:
   * Nombre del evento.
   * Ubicación.
   * Fecha y hora.
3. **Actualizar** una solicitud existente, para modificar la información en caso de cambios en el evento.
4. **Eliminar una solicitud de reserva si el evento se cancela o no se realiza.**

Este CRUD se desarrollará utilizando JavaFX para la interfaz gráfica, con el fin de facilitar la interacción con los usuarios de la administración pública de Villa Central.

**4. Diseñar un prototipo gráfico para la aplicación.**  
  
  




**Video**  
<https://studio.youtube.com/video/gkjt0itFHEs/edit>  
 **GitHub**  
https://github.com/LuiisR3p/Paradigmas\_proyecto.git  
  
  
  
**5. Propuesta de cronograma y actividades para el desarrollo de la aplicación durante el semestre.**

Semana 1: Requerimientos y Diseño del Proyecto

Actividad 1: Definición de Requerimientos

Reunión para definir los requerimientos y las funcionalidades del CRUD (crear, leer, actualizar, eliminar reservas de espacios públicos).

Resultado esperado: Documento de especificación de requerimientos.

Actividad 2: Diseño de la Arquitectura del Proyecto

Diseño del sistema usando el patrón MVC (Modelo-Vista-Controlador).

Diseño de la base de datos (tablas y relaciones).

Resultado esperado: Diagrama UML de clases y diseño de la base de datos.

Semana 2: Configuración del Entorno de Desarrollo y Base de Datos

Actividad 3: Configuración del Entorno de Desarrollo

Configurar JDK, JavaFX, IDE (IntelliJ/Eclipse), y conexión a la base de datos (SQLite o MySQL).

Resultado esperado: Entorno de desarrollo listo y verificado.

Actividad 4: Creación de la Base de Datos

Implementar la base de datos con las tablas necesarias (reservas, eventos, espacios).

Resultado esperado: Base de datos funcional.

Semana 3 - 4: Desarrollo del Modelo (MVC)

Actividad 5: Implementación del Modelo

Crear clases Java para manejar las reservas (Reserva, Espacio, Evento).

Conectar las clases del modelo con la base de datos.

Resultado esperado: Clases del modelo listas y funcionales.

Actividad 6: Validación del Modelo

Implementar las validaciones de datos (fechas, espacios disponibles, etc.).

Resultado esperado: Validaciones implementadas y probadas.

Semana 5 - 6: Desarrollo de la Interfaz Gráfica (Vista en MVC)

Actividad 7: Diseño de la Interfaz Gráfica

Crear las pantallas de creación, consulta, edición y eliminación de reservas en JavaFX.

Resultado esperado: Interfaz gráfica del CRUD funcionando.

Actividad 8: Vinculación con el Modelo

Conectar la interfaz gráfica con las clases del modelo.

Resultado esperado: Interfaz gráfica y lógica del modelo integradas.

Semana 7: Implementación del Controlador (MVC)

Actividad 9: Programación del Controlador

Implementar la lógica del controlador para coordinar las acciones de la vista y el modelo (crear, leer, actualizar, eliminar).

Resultado esperado: CRUD funcional e integrado.

Semana 8: Pruebas y Corrección de Errores

Actividad 10: Pruebas Funcionales

Realizar pruebas exhaustivas en todas las funcionalidades del CRUD (alta, baja, modificación y consulta).

Resultado esperado: Reporte de errores y corrección de los mismos.

Semana 9: Mejora de la Interfaz de Usuario y Documentación

Actividad 11: Mejoras en la Interfaz de Usuario

Ajustes de la interfaz gráfica para hacerla más intuitiva y agradable visualmente.

Resultado esperado: Interfaz mejorada.

Actividad 12: Documentación

Elaboración de la documentación técnica y del manual de usuario.

Resultado esperado: Documentación completa.

Semana 10: Revisión Final y Entrega

Actividad 13: Revisión Final y Ajustes

Revisión del sistema en su totalidad, con correcciones finales.

Resultado esperado: Proyecto final listo.

Actividad 14: Preparación de la Presentación

Preparar la presentación y la demostración del sistema.

Resultado esperado: Presentación lista para la entrega final.

Resumen de Entregables:

Documento de especificación de requerimientos.

Diagrama UML y diseño de base de datos.

Base de datos configurada.

Aplicación CRUD funcional en JavaFX.

Documentación técnica y manual de usuario.

Presentación del sistema.

Este cronograma comprimido en 10 semanas asegura que cada actividad principal se aborde de manera efectiva, con un enfoque equilibrado entre desarrollo, pruebas y revisión final.  
  
  
  
**Resultados**

1. **Desarrollo funcional completo del sistema CRUD:**
   * El sistema permite la gestión eficiente de reservas de espacios públicos temporales en Bogotá, con operaciones de creación, lectura, actualización y eliminación completamente funcionales.
   * Funciones específicas para el administrador, como aprobación y rechazo de reservas, se implementaron exitosamente.
2. **Integración del patrón MVC:**
   * La arquitectura asegura una separación clara entre la lógica del negocio (modelo), la interfaz de usuario (vista) y el control de eventos (controlador).
   * Se utilizaron prácticas estandarizadas que facilitan la escalabilidad y el mantenimiento del sistema.
3. **Uso de JavaFX para la interfaz gráfica:**
   * Se diseñó una interfaz gráfica intuitiva que permite a los usuarios interactuar con el sistema de forma amigable.
   * Se implementaron validaciones para prevenir conflictos de horario y evitar entradas inválidas.
4. **Control de versiones con Git:**
   * Git permitió un desarrollo colaborativo fluido, manteniendo un historial claro de cambios y facilitando la integración de nuevas funcionalidades.

# Discusión

1. **Impacto del Sistema:**
   * Este sistema automatiza un proceso que anteriormente era manual y propenso a errores, garantizando la correcta asignación de espacios públicos y reduciendo conflictos de programación.
   * La interfaz gráfica de usuario mejora significativamente la experiencia del usuario final, tanto para administradores como para solicitantes.
2. **Desafíos enfrentados:**
   * **Validación de horarios:** Fue un reto implementar una lógica robusta que detecte y evite conflictos entre reservas en tiempo real.
   * **Gestión de serialización y deserialización:** La persistencia de datos utilizando archivos planteó problemas iniciales de compatibilidad, que fueron solucionados mediante pruebas y ajustes del formato de serialización.
3. **Mejoras potenciales:**
   * Migrar la persistencia de datos a una base de datos como MySQL o SQLite para mayor escalabilidad.
   * Implementar autenticación avanzada para usuarios y administradores para mayor seguridad.
   * Añadir notificaciones automáticas para informar a los solicitantes sobre el estado de sus reservas.

# Conclusión

El sistema de gestión de reservas de espacios públicos se desarrolló con éxito, cumpliendo con los objetivos planteados. Las operaciones CRUD permiten un control total sobre las reservas, asegurando que no haya conflictos de horarios ni asignaciones incorrectas.

La implementación del patrón MVC asegura una arquitectura robusta, escalable y fácil de mantener, mientras que el uso de JavaFX proporciona una interfaz gráfica eficiente y amigable. Además, el uso de Git como sistema de control de versiones contribuyó significativamente al desarrollo organizado y colaborativo.

Este sistema no solo automatiza un proceso crítico, sino que también establece una base sólida para futuras expansiones, como la integración de una base de datos más robusta, soporte para múltiples idiomas y características avanzadas de análisis de datos.

# REFERENCIAS BIBLIOGRÁFICAS

 Chacon, S., & Straub, B. (2014). *Pro Git* (2nd ed.). Apress. https://doi.org/10.1007/978-1-4842-0076-6

 Loeliger, J., & McCullough, M. (2012). *Version control with Git: Powerful tools and techniques for collaborative software development* (2nd ed.). O'Reilly Media.

 Swicegood, T. (2008). *Pragmatic Version Control Using Git*. Pragmatic Bookshelf.

 Silverman, M. (2020). *Learn Version Control with Git: A step-by-step course for the complete beginner*. Independently Published.