Manual del Técnico TicketMaster

1. **Introducción:**

Esta destinado a desarrolladores y técnicos encargados del mantenimiento, implementación y despliegue del Sistema **TicketMaster**. Aquí se documentan las tecnologías empleadas, arquitectura del proyecto, estructura del código fuente, procedimientos de compilación, asi como soluciones a problemas comunes.

1. **Arquitectura:**

El sistema esta dividido en capas siguiendo el patrón MVC(Modelo-Vista-Controlador):

* **Modelos**(ticketmaster.model): Representan entidades como Ticket, Usuario, Rol, Permiso, etc.
* **Controladores**(ticketmaster.controller): Lógica que enlaza la interfaz gráfica con los datos.
* **DAO**(ticketmaster.DAO): Acceso a datos persistentes (archivos de texto).
* **Vista**(resources/view/): Interfaces graficas en formato. fxml.
* **Script**(scripts). Archivo de la creación de tablas PostgreSQL.
* **Base de Datos**(ticketmaster.database): Conexión el sistema con la base de datos postgreSQL.

1. **Diagramas:**
   1. **Diagrama de Casos de Uso**

Incluye los siguientes casos principales:

* Registrar tickets
* Cambiar estado de tickets
* Agregar notas a tickets
* Registrar usuarios
* Asignar roles y permisos
* Configurar flujos de trabajo
* Registrar departamentos
* Consultar tickets
  1. **Diagrama de Clases**

|  |  |
| --- | --- |
| **Ticket** | |
| * **Id** * **Titulo** * **Descripción** * **Estado** * **Prioridad** * **Fecha** | |
| **Usuario** |
| * **Nombre** * **Rol** * **Departamento** |

|  |
| --- |
| **Rol** |
| * **Nombre** * **Permisos** |

1. **Algoritmos y Flujos**

**Crear Ticket**

if (!titulo.isEmpty() && !descripcion.isEmpty()) {

Ticket nuevo = new Ticket(...);

TicketDAO.guardar(nuevo);

}

**Cambiar Estado**

ticket.setEstado("Resuelto");

TicketDAO.actualizar(ticket);

**Agregar Nota**

ticket.agregarNota("Comentario del técnico");

TicketDAO.actualizar(ticket);

1. **Tecnologías Utilizadas**

* Java 23+
* JavaFX (FXML)
* Scene Builder
* PostgreSQL
* Archivos de texto .txt como sistema de almacenamiento
* GitHub (control de versiones)

1. **Configuración y Ejecución**

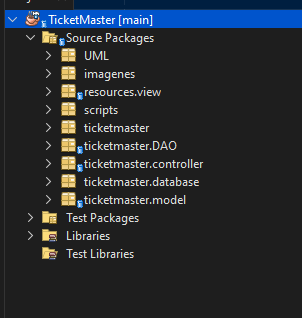
**Requisitos**

* JDK 17+
* NetBeans con soporte JavaFX
* Scene Builder para diseño de interfaces
* Librerías javaFX vinculadas al proyecto

**Compilación y Ejecución**

* Clona el repositorio desde GitHub
* Abre el proyecto en Netbeans o tu IDE
* Asegúrate de tener configuradas las rutas de JavaFX en las opciones del proyecto.
* Ejecuta la clase Ticket.java

1. **Estructura de Archivo**



1. **Solución de Problemas**

|  |  |
| --- | --- |
| **Problema** | **Solución** |
| * Error al cargar FXML * No guarda información * Botón no responde * Error al leer archivos * FXML con campos nulos (null) | * Verifica que los fx:id coincidan con el controlador. * Asegúrate de que la ruta de archivo .txt exista y tenga permisos. * Verifica el método on Action y su existencia en el controlador. * Usa UTF-8 y revisa que las líneas tengan el formato esperado (Split(“ * Asegúrate de cargar el archivo correcto y que los IDs esten bien escritos. |

1. **FAQ**

* **¿Puedo usar una base de datos en lugar de archivos?**

Si basta con reemplazar las clases DAO por una implementación que use JDBC y consultas SQL.

* **¿Qué sucede si borro un archivo txt?**

Se perderán los datos correspondientes. Se recomienda respaldos periódicos.

* **¿El sistema tiene seguridad?**

La interfaz de roles y permisos permite limitar funcionalidades según rol.

1. **Control de Versiones**

git add .

git commit -m “Implementación completa del módulo de gestión de tickets”

git push origin main