Documento de Diseño del Sistema Parque de Diversiones

1. Contexto y Alcance

El proyecto busca desarrollar un sistema en Java que gestione las operaciones administrativas de un parque de diversiones, incluyendo:

- Catálogo de atracciones (mecánicas y culturales) y espectáculos.
- Gestión de empleados y asignación de labores.
- Venta y validación de tiquetes.

Alcance:

- Persistencia local en archivos (no base de datos externa).
- Usuarios con login y contraseña.
- Implementación de toda la lógica, sin necesidad de interfaz gráfica (por ahora).
- Programas de prueba que permitan validar el funcionamiento desde consola.

Restricciones:

- Toda la persistencia debe realizarse fuera de la carpeta del código fuente.
- La aplicación debe estar escrita completamente en Java.
- No se espera interfaz gráfica ni validación de entradas del usuario.

2. Objetivos y No-Objetivos

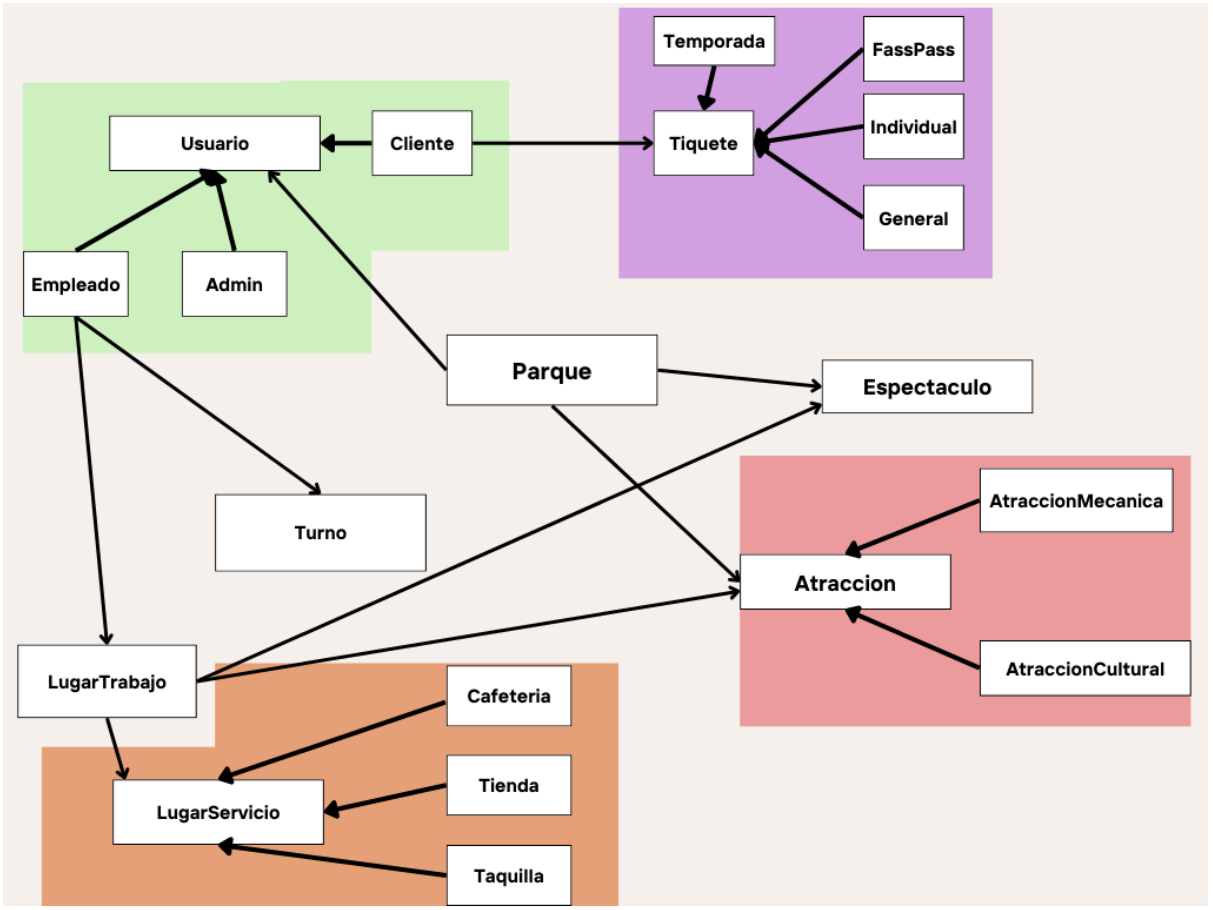
Objetivos:

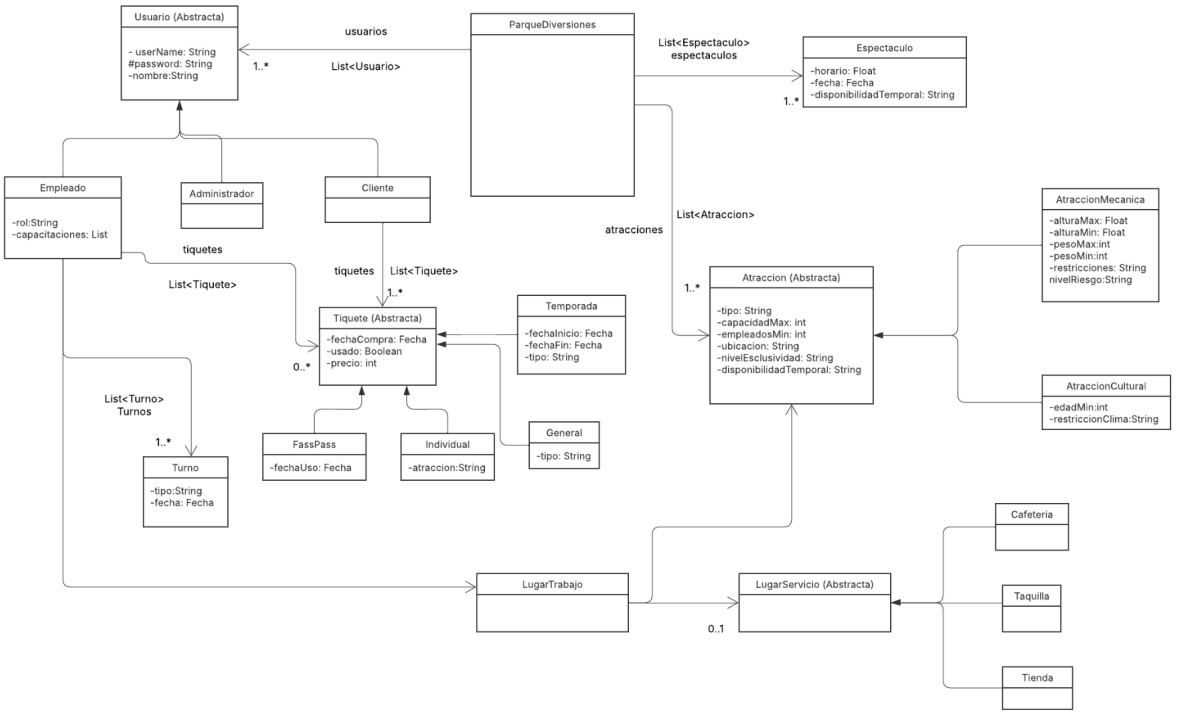
- Modelar las entidades principales del parque: atracciones (mecánicas y culturales), espectáculos, empleados, usuarios y tiquetes utilizando principios de orientación a objetos.
- Implementar las funcionalidades centrales para la gestión de estas entidades (creación, consulta, modificación, eliminación cuando sea aplicable).
- Desarrollar la lógica para la venta y validación de tiquetes considerando los diferentes tipos y restricciones.
- Implementar un mecanismo de persistencia de datos para asegurar que la información del parque se conserve entre ejecuciones.

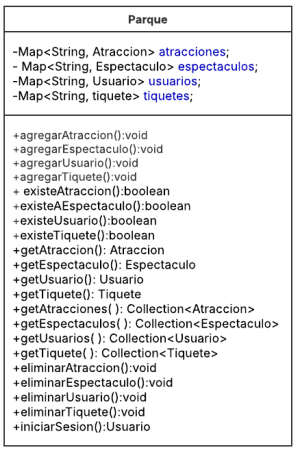
No-Objetivos:

- Implementación de una interfaz de usuario gráfica .
- Implementación de un sistema de gestión de filas para atracciones.
- Integración con sistemas de pago externos.
- Gestión de inventario detallada para tiendas o cafeterías (más allá de lo necesario para la asignación de empleados).
- Manejo avanzado de eventos climáticos en tiempo real con fuentes de datos externas.
- Un sistema de seguridad completo con roles y permisos granulares más allá de la identificación del tipo de usuario básico .
- Implementación de transacciones atómicas complejas que requieran mecanismos avanzados de bases de datos.

3. Diagrama de Contexto del Sistema:




Usuario (Abstracta)
- userName: String
#password: String
-nombre:String
usuarios
1..*
List<Usuario>
ParqueDiversiones
List<Espectaculo>
espectaculos
Espectaculo
-horario: Float
-fecha: Fecha
-disponibilidadTemporal: String
1..*
Empleado
-rol:String
-capacitaciones: List
Administrador
Cliente
tiquetes
List<Tiquete>
tiquetes
List<Tiquete>
1..*
atracciones
List<Atraccion>
1..*
AtraccionMecanica
-alturaMax: Float
-alturaMin: Float
-pesoMax:int
-pesoMin:int
-restricciones: String
nivelRiesgo:String
Atraccion (Abstracta)
-tipo: String
-capacidadMax: int
-empleadosMin: int
-ubicacion: String
-nivelEsclusividad: String
-disponibilidadTemporal: String
Tiquete (Abstracta)
-fechaCompra: Fecha
-usado: Boolean
-precio: int
0..*
Temporada
-fechaInicio: Fecha
-fechaFin: Fecha
-tipo: String
AtraccionCultural
-edadMin:int
-restriccionClima:String
List<Turno>
Turnos
1..*
Turno
-tipo:String
-fecha: Fecha
FassPass
-fechaUso: Fecha
Individual
-atraccion:String
General
-tipo: String
Cafeteria
LugarTrabajo
LugarServicio (Abstracta)
0..1
Taquilla
Tienda


Parque
-Map<String, Atraccion> atracciones;
- Map<String, Espectaculo> espectaculos;
-Map<String, Usuario> usuarios;
-Map<String, tiquete> tiquetes;
+agregarAtraccion():void
+agregarEspectaculo():void
+agregarUsuario():void
+agregarTiquete():void
+ existeAtraccion():boolean
+existeAEspectaculo():boolean
+existeUsuario():boolean
+existeTiquete():boolean
+getAtraccion(): Atraccion
+getEspectaculo(): Espectaculo
+getUsuario(): Usuario
+getTiquete(): Tiquete
+getAtracciones( ): Collection<Atraccion>
+getEspectaculos( ): Collection<Espectaculo>
+getUsuarios( ): Collection<Usuario>
+getTiquete( ): Collection<Tiquete>
+eliminarAtraccion():void
+eliminarEspectaculo():void
+eliminarUsuario():void
+eliminarTiquete():void
+iniciarSesion():Usuario


Espectaculo
#nombre: String
#descripcion: String
horarios: List<LocalDateTime>
#esDeTemporada: boolean
#fechaInicioTemporada:LocalDate
fechaFinTemporada:LocalDate
CondicionClimatica:String
#empleadosMin: int
#empleadosAsignados:List<Empleado>


Atraccion
#nombre: String
#capacidadMax: int
ubicacion: String
#nivelEsclusividad: String
#esDeTemporada: boolean
#fechaInicioTemporada:LocalDate
fechaFinTemporada:LocalDate
CondicionClimatica:String
#empleadosMin: int
#empleadosAsignados:List<Empleado>
+getNombre():String
+setNombre()String
+getCapacidadMax():int
+setCapacidadMax()int
+get