Análisis de reto técnico Softka – Aplicativo para parque de diversiones

Después de leer las condiciones técnicas se tomaron las siguientes decisiones de diseño:

1. Sobre el modelo MVC que hacía parte de los requerimientos
   1.1. Modelo: Se crearon clases POJO para el manejo de las entidades requeridas (clientes, trabajadores, atracciones y taquillas de compra)
   1.2. Vista: Se creó un esquema de menús por consola que basado en la consulta de un número de documento en las bases de datos de empleados muestra el menú con las capacidades o tareas habilitadas para ese empleado. Para aumentar la seguridad se podría agregar una contraseña para cada empleado en un desarrollo futuro.
   1.3. Controlador: Se agregó aquí toda la lógica del negocio compuesta esencialmente por los métodos CRUD pertinentes para cada entidad del modelo y algunos métodos para las capacidades específicas de algunas entidades como los empleados, según los requerimientos hechos para cada tipo. Todos estos métodos interactúan con la información almacenada en la base de datos.
2. Sobre la aplicación de POO al proyecto
   2.1. Herencia: Dado que la mayoría de entidades representaban personas se partió de una clase padre Persona con los atributos comunes a cualquier individuo (nombre, documento, etc.) desde esta partieron dos líneas hereditarias, una para clientes y otra para empleados, cada una con nuevos herederos para atender las particularidades de cada entidad.
   2.2. Pertinencia del nivel de abstracción empleado: Pese a que para el nivel de desarrollo actual del aplicativo puede parece una estructura de clases demasiado compleja se partió de la idea de que el proyecto pudiera robustecerse en el futuro con mayor cantidad de datos y funcionalidades asociadas a cada entidad.
3. Sobre la elección de una base de datos
   3.1. Base de datos no relacional: Se eligió una base de datos no relacional puesto que la dinámica del negocio genera la posibilidad de tener entidades con diversos atributos y cuyas relaciones podrían no ser completamente claras, en este sentido la flexibilidad y facilidad de manejo de una base de datos no relacional surgió como la mejor opción.
   3.2. MongoDB: La elección MongoDB como sistema de bases de datos se basó en la facilidad de integración con Java que ofrece y sobre todo de la amplia documentación oficial disponible.
4. Sobre el grado de desarrollo alcanzado hasta el momento
   4.1. Funciones disponibles para el usuario: En este punto el usuario tiene acceso a la práctica totalidad de funcionalidades solicitades en el reto
   4.2. Funciones codificadas: Si bien todas las funciones CRUD están codificadas no todas ellas son accesibles al usuario a través de interfaz/vista pues varias de estas funciones no se aclaran a qué tipo de usuario podrían corresponder. Para tal propósito sería ideal que un desarrollo futuro incluyera un menú de administrador para acceder a estas otras funciones (como crear nuevos empleados)