Tecnologías de Desarrollo de Software

Curso 2025/2026 29 de septiembre de 2025

Caso Práctico: *Gestión de Gastos*

El presente enunciado tiene como objetivo la descripción del desarrollo de una aplicación de escritorio para la gestión y el control de gastos personales. Toda la información introducida en la aplicación debe ser persistida. La aplicación permitirá al usuario registrar de manera sencilla sus gastos personales, asociando a cada registro la cantidad y la fecha correspondiente y organizándolos en diferentes categorías predefinidas (como alimentación, transporte o entretenimiento), con la posibilidad de crear nuevas categorías según sus necesidades. Los gastos pueden ser editados o borrados en cualquier momento.
Además de la interfaz gráfica antes comentada, el sistema debe ofrecer adicionalmente una línea de comandos básica para poder realizar el registro, modificación y borrado de un gasto. Por tanto, la gestión de gastos se podrá realizar desde una interfaz gráfica y desde la línea de comandos.

El sistema ofrecerá diversas formas de presentar la información registrada. Entre ellas, se incluirá la consulta en formato de tabla/lista, así como la representación gráfica mediante diagramas de barras y/o circulares, que facilitarán la comprensión de la distribución de los gastos por categorías. Considerar el uso de visualizaciones avanzadas, como por ejemplo mostrar los gastos en un calendario (CalendarFX - https://dlsc-software-consulting-gmbh.github.io/CalendarFX/ usando una visualización Full Day, por ejemplo).

Los datos presentados pueden ser filtrados. El sistema permitirá filtrar los gastos por una lista de meses, por intervalos de fechas personalizados, por una lista de categorías o por una combinación de cualquiera de estos filtros (por ejemplo: filtrar por gastos de 'transporte' en los meses de 'julio' y 'agosto').

Asimismo, la aplicación incluirá un sistema de alertas configurable, en el que el usuario podrá establecer límites de gasto. Estas alertas podrán definirse de forma semanal, mensual y opcionalmente pueden estar además vinculadas a una categoría específica. Por ejemplo, "configurar una alerta si superamos los 500 euros de gasto a la semana" o "avisar si superamos los 100 euros al mes en videojuegos". Una vez superado el tope de gasto configurado en la alerta, el sistema generará una notificación que puede ser leída por el usuario. El sistema debe permitir que las notificaciones pasadas puedan ser también revisadas en cualquier momento (es decir, debe haber un historial de notificaciones).

Por otro lado, la aplicación permitirá el uso cuentas de gasto compartidas con otras personas. Una cuenta de gasto agrupa a personas (representadas en el sistema con un simple nombre) que son las que deben asumir una proporción del importe de gasto introducido en la cuenta. El gasto introducido en una cuenta está siempre asociado a una persona de la cuenta, representando que dicha persona ha pagado ese gasto. El resto de las personas de la cuenta le deberían una parte proporcional del gasto. Eso se refleja con el concepto de saldo de dinero pendiente para cada persona en la cuenta (inicialmente todas las personas de la cuenta tienen este valor a 0).
Por ejemplo, "sea una cuenta de gasto formada por Raquel, Luis y María. Si se introduce un gasto de 30 euros a nombre de Raquel, entonces el saldo de Raquel es +20 euros (le deben en el grupo 20 euros), mientras que el de Luis y María es de -10 euros (cada uno debe 10 euros en el grupo). Si después se introduce un gasto de 9 euros por parte de Luis, entonces su saldo pasa a ser –4 euros (sigue debiendo todavía 4 euros en el grupo), el de María pasa a –13 euros (debe ahora 13 euros al grupo) y el de Raquel pasa a +17 euros (todavía le deben 17 euros)".

Las cuentas de gasto compartidas por defecto tienen una distribución equitativa entre las personas que la conforman. No obstante, el sistema debe soportar además una cuenta de gasto compartida que permita definir el porcentaje de gasto asumido por cada persona (evidentemente la suma de los porcentajes debe ser 100%). En el ejemplo anterior podríamos haber indicado que Raquel asumiría el 40% de los gastos, mientras que Luis y María solo el 30% de los gastos cada uno.

Una vez creada una cuenta compartida con una lista de personas, dicha lista no podrá ser modificada.

Por último, la aplicación debe permitir importar datos de gasto de fuentes externas (por ejemplo, un listado de gastos generados desde una plataforma bancaria). Es decir, dado un fichero (de texto plano) con datos de gasto, la aplicación puede leerlo para incorporar los datos de los gastos en la aplicación. Se proporciona en AulaVirtual un fichero de ejemplo de una plataforma bancaria. No obstante, el sistema debe estar preparado para la importación de diferentes formatos.

Arquitectura de la aplicación y Gestión del proyecto

La aplicación seguirá el enfoque arquitectónico presentado en clase de prácticas. Se utilizará la tecnología JavaFX para la implementación de la interfaz de usuario y la librería Jackson para el almacenamiento de los datos de la aplicación en formato JSON. Se utilizará el patrón Repositorio para desacoplar la capa de almacenamiento del resto de la aplicación.

Todas las dependencias de librerías externas deben manejarse con Maven y el proyecto debe ser gestionado mediante Git. El grupo debe mandar una invitación a su profesor de prácticas para que pueda acceder al proyecto de GitHub.

Grupos de prácticas

Los alumnos deberán formar **grupos de tres componentes** que deberán ser del mismo grupo de teoría salvo casos excepcionales. Se recomienda organizar el trabajo de forma que los miembros de un grupo puedan trabajar en paralelo. Por ejemplo, un alumno puede implementar las clases de la interfaz de usuario mientras otro se encarga de las clases relativas a la persistencia de los datos. Todos los alumnos del grupo deben tener conocimiento de todos los aspectos de la práctica y deberían haber participado en la programación de cada capa de la arquitectura.

Código y patrones

El código debería ser escrito utilizando **expresiones lambda** y **streams** en aquellos puntos en los que se considere apropiado. El alumno no debe cometer errores relacionados con los patrones **GRASP** ni de los principios estudiados en clase.

Los conocimientos sobre patrones de diseño se valoran principalmente en el examen de teoría de la asignatura. No obstante, la realización de esta práctica permite el uso e **implementación** de **patrones de diseño**. Como **mínimo**, el alumno deberá aplicar correctamente el patrón *Estrategia* en las alertas, y el patrón *Adaptador* y *Método Factoría* en el importador de datos. Además, es el obligatorio el uso del patrón *Singleton* en aquellas clases cuyo acceso a la única instancia en el sistema deba ser global.

ENTREGA

Fecha de entrega: 7 de enero
Se creará una **tarea** en el Aula Virtual para que uno de los miembros del grupo de trabajo indique en el texto de la tarea la ruta del **repositorio de GitHub** donde se encuentra la entrega.

Además, en dicho texto se deben incluir "**Observaciones Finales**" donde los alumnos podrán introducir una valoración objetiva de la realización de la práctica, que incluya comentarios constructivos, una estimación del total de horas invertido en la realización del proyecto (la suma de horas de los alumnos que conforman el equipo) y cualquier otra información que consideren relevante.

Habrá que subir un fichero ZIP con el repositorio comprimido a la tarea de AulaVirtual. Esto se hace desde el botón verde "Code" del repositorio de GitHub.

Documentación

El repositorio debe contener un README.MD con el nombre de los integrantes del grupo (mail y subgrupo de pertenencia), descripción breve de qué hace el proyecto y cómo ejecutar el proyecto. Incluir además los enlaces a los documentos relevantes de la documentación.

Para el resto de documentación, crear una carpeta **/docs** en el repositorio. Incluir en esta carpeta el resto de la documentación del proyecto, en uno o varios ficheros .md (markdown). Para incluir imágenes en la documentación en .md, crear una carpeta **/imágenes** (dentro de **/docs**) donde depositar las imágenes y referenciarlas en markdown.
NOTA: Aquí un enlace con ayuda sobre la sintaxis de markdown: https://docs.github.com/es/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax

Los contenidos de la documentación son los siguientes:

1. Diagrama de clases del dominio del proyecto.
2. Especificación de las historias de usuario del proyecto.
3. Un diagrama de interacción para la una de las historias de usuario (a elección del grupo)
4. Breve explicación de la arquitectura de la aplicación y decisiones de diseño que se consideren de interés para la comprensión del trabajo.
5. Explicación de los patrones de diseño usados.
6. Breve manual de usuario (debe incluir capturas de las ventanas para apoyar las explicaciones)

Evaluación

Es requisito obligatorio para aprobar la práctica que el proyecto funcione de manera completa de acuerdo con la especificación del enunciado.

Cada parte se valorará con el siguiente porcentaje:

- Modelo del dominio de clases / interacciones e historias de usuario (15%)
- Diseño de la interfaz (25%)
- Código (legibilidad, uso de principios básicos de programación OO, buen uso de streams y expresiones lambda, uso del diseño por contrato) (40%),
- Aplicación de patrones (10%),
- Documentación entregada (10%).
- **Es obligatorio el uso de Git y Maven (penaliza su no uso/uso incorrecto con hasta –2 puntos).**