Guía de la Cuarta Entrega del proyecto

Competencias a desarrollar:

- Toma en cuenta la opinión de los futuros usuarios al mostrarles la aplicación.
- Mejora el diseño actual de su solución incluyendo o aplicando los conceptos de herencia y polimorfismo donde sea adecuado.
- Amplía el o los diagramas de clases UML para incluir todas las funcionalidades.
- Provee una experiencia robusta al usuario.

Se completan los requisitos funcionales:

Esta entrega requerirá una nueva priorización de los pasos restantes a seguir (pendientes y/o por enfrentar) para completar la solución propuesta. Este prototipo debería ser un *beta*: incluye todos los requisitos funcionales listados por el grupo, su diseño se apoya en principios y buenas prácticas, herencia y polimorfismo; y presenta una interfaz robusta contra todo tipo de errores. Con ello se pretende expresar que la experiencia del usuario debe ser fluida, y debe guiarlo o reorientarlo cuando sea necesario para cumplir con lo que el usuario desea hacer. Como un *beta* de su proyecto, esta entrega debe estar completa pero sujeta a cambios según la retroalimentación recibida al hacer pruebas con usuarios.

Identificación y descripción de cambios a las clases o nuevas clases necesarias:

Identifique las clases que necesiten modificación para completar la solución al problema y/o para aprovechar la herencia y el polimorfismo, así como posibles clases nuevas para el mismo fin. Aplique los cambios correspondientes a su diagrama UML antiguo.

Para cada uno de los cambios introducidos y las clases nuevas desarrolladas, elabore una descripción donde especifique qué cambio se ha hecho, dónde, por qué y cuál será su impacto en el resto del sistema existente.

Retroalimentación de usuarios:

Como se trata de un prototipo *beta*, los usuarios deberían poder usar la aplicación por sus cuentas. Procure convencer a algunos de sus usuarios de que muestren o compartan (virtual o remotamente) su prototipo con otras personas. Provéales material (código o videos, por ejemplo) si es necesario. Asegúrese de incluir, en sus tomas de retroalimentación, una mezcla de pruebas presenciales y pruebas “remotas”, en las que los usuarios prueban su solución sin su asistencia y le envían una descripción de su experiencia y de sus sugerencias, preguntas, recomendaciones y/o críticas.

Planificación y gestión:

Elabore una lista con las tareas propuestas para completar los requisitos funcionales especificados por el grupo, junto con el miembro del equipo que será responsable de cumplir cada tarea y su fecha probable de terminación. Siga los siguientes pasos:

- Liste los requisitos que se implementarán en esta entrega con su respectiva prioridad con respecto al desarrollo de la entrega.
- Divida los requisitos funcionales que desarrollará en tareas más pequeñas.
- Especifique cada tarea:
    - Nombre de la tarea
    - Descripción de la tarea
    - Horas estimadas de desarrollo
    - Responsable de desarrollarla
    - Fecha probable de terminación de la tarea
- Haga un calendario de planificación donde incorpore todas las tareas que redactó con su respectiva fecha de terminación.

Cada integrante del grupo debe llenar el formulario 1 con la información de las tareas que le fueron asignadas y el tiempo que le tomó elaborarlas. El formulario está lleno con información hipotética para que sirva como ejemplo del contenido que debe llevar. En la tabla 2 puede encontrar las instrucciones y el propósito del Formulario.

Implementación:

- Implemente como mínimo 1 requisito funcional de los encontrados **por miembro del equipo** de desarrollo. La implementación debe estar completa, de manera que quede listo para mostrar a usuarios.

Control de versiones:

- Utilice Github (https://github.com/), o el repositorio Git en línea de su preferencia, como controlador de versiones. Debe haber más de 2 *commits* semanales por miembro del grupo de desarrollo.

Formulario

Nombre: Pepito Pérez **Carné:** 16789

Fecha	Inicio	Fin	Tiempo Interrupción (min)	Tiempo trabajado (min)	Tarea	Comentarios
05/08/2020	08:00	10:00	5	115	Jerarquía de clases del usuario	Vi un par de TikTok's.
06/08/20	14:00	15:00	30	30	Programación defensiva	Grabé un par de TikTok's.

Formulario 1. Gestión del tiempo en el cumplimiento de las tareas planificadas

Objetivos	• Registrar el tiempo invertido en cada fase del proyecto para referencia • Concientizar al grupo y a cada miembro del uso de su tiempo en relación con el proyecto • Usar estos datos para completar el Resumen Plan Proyecto
Notas	• Sea lo más exact@ posible • Si requiere espacio adicional usar otra copia del formulario
Fecha	Fecha en que se registra el tiempo
Inicio	Hora en que comienza a trabajar en la tarea
Fin	Hora en que deja de trabajar en la tarea
Tiempo Interrupción	Cualquier interrupción que ocurre durante la tarea y la razón de esta. Ej. Teléfono, baño, etc. La razón debe ser suficientemente específica para entender la naturaleza de la interrupción, pero no debe ser excesivamente específica para evitar información innecesaria o personal/privada.
Tiempo trabajado	Tiempo en minutos real gastado en la tarea (Fin – Inicio) – Interrupción
Tarea	Nombre o descripción de la fase, paso o tarea en la que esté trabajando
Comentarios	Cualquier comentario interesante o aclaratorio

Tabla 2. Objetivos e instructivo para llenar el formulario 1

Material a entregar en Canvas:

- Archivo .pdf que incluya:
  - Los requisitos con los que se completará esta entrega.
  - La identificación y descripción de las modificaciones a clases y/o clases nuevas, con especial atención a las aplicaciones de herencia y polimorfismo.
  - La planificación de las tareas.
  - Registro de retroalimentación de usuarios potenciales (al menos dos usuarios por miembro del grupo). Asegúrense de organizar la retroalimentación para que se distinga claramente la fuente (prueba compartida por usuario, prueba presencial, prueba remota, etc.).
  - Los formularios de registro de trabajo de cada un@ de l@s integrantes del grupo.
- Archivo de imagen (.jpg o .png) con el diagrama de clases incluyendo cambios, clases nuevas y comentarios al diseño.
- Archivos de código (extensión .java).
- Link del repositorio del proyecto en Github o el repositorio Git en línea de su preferencia.

Evaluación Grupal:

- **Requisitos funcionales (15 puntos):** se revisó la lista de requisitos funcionales ya desarrollada para identificar, añadir y/o eliminar los requisitos para esta entrega, y se presentó el listado de estos requerimientos, ordenados por una nueva fase de priorización.
  Todo requisito funcional (en otras palabras, requisitos para que el sistema cumpla con sus objetivos principales) debe completarse para esta entrega, incluyendo pendientes o cambios a los de la entrega anterior. Este prototipo estará sujeto a cambios según la retroalimentación recibida, pero no debe tener cambios pendientes. La gracia es que esta sea una prueba tipo *beta* de su solución.
- **Descripción de cambios y clases nuevas (20 puntos):** el grupo de desarrollo describe a detalle cada una de las clases a añadir y/o cambios a realizar al sistema existente. El objetivo es que se entienda dónde se hará cada cambio, por qué y cuál será su efecto sobre el resto del sistema. Es importante que se reestructure el diseño original como sea necesario para incluir y aprovechar los conceptos de herencia y polimorfismo en cualquiera de sus presentaciones. Esto debe estar claramente descrito e identificado.
- **Retroalimentación de usuarios (20 puntos)**: evidencia de la retroalimentación obtenida al llevar (virtualmente) el sistema a personas que estén relacionadas con la problemática elegida por el grupo, y pedirles que lo usen. Esta toma de retroalimentación, considerando que el proyecto debe estar esencialmente completo para esta entrega, debería poder hacerse sin apoyo presencial de l@s desarrollador@s. Por tanto, se requiere una mezcla de pruebas con retroalimentación en tiempo real (como llamadas o videollamadas) y pruebas “independientes”, en las que el usuario lleva a cabo la prueba por su cuenta y provee retroalimentación al cabo de la misma.

También se recomienda, si es posible, los usuarios de prueba compartan la solución con otr@s para obtener retroalimentación de segundas y terceras "capas" de usuarios.

- **Código y programa funcionando (30 puntos):** el código entregado cumple con los requisitos funcionales, sigue los principios de programación OO, está debidamente documentado y aplica los conceptos y herramientas vistos en clase hasta ahora. Esto incluye, donde aplique:
    - Relaciones de herencia entre clases para reutilización de código y centralización de cambios.
    - Uso de clases abstractas y polimorfismo vía herencia.
    - Encapsulación para manejar correctamente el acceso a propiedades de cada objeto.
    - Desarrollo e implementación de interfaces para uniformizar las formas de acceso e intercambio de información con diferentes clases.
    - Aplicación del patrón MVC y de los principios de desarrollo orientado a objetos vistos en clase (principio de responsabilidad única + alta cohesión/bajo acomplamento; ley de Demeter; principio Abierto-Cerrado).
    - Uso de sobrecarga de métodos y atributos/métodos estáticos para flexibilizar o hacer más rígido el código.
- **Planificación (15 puntos):** se elaboró una lista de tareas en la que se reparte de forma equitativa el trabajo entre tod@s los integrantes del grupo de desarrollo. Tod@s tienen tareas en las que tienen que programar y elaborar parte del análisis y el diseño. Se incluyen las tareas de muestra del sistema a usuarios finales en la planificación de la segunda fase del proyecto. Se evidencia en el registro de contribuciones al repositorio en GitHub el trabajo equitativo que ha contribuido cada miembro del grupo.

Evaluación individual:

Cada estudiante tendrá una nota individual (**Ni**) del proyecto, que se calcula de la siguiente manera:

1. Se califica el proyecto en general, de acuerdo a la distribución de puntos descrita en esta guía.
2. Se realiza un promedio del porcentaje de autoevaluación (**A**) y coevaluación (**C**) de sus compañeros de grupo. La autoevaluación y coevaluación son actividades que cada miembro del grupo debe realizar para calificar su propio desempeño y el de sus compañer@s en esta fase del proyecto.
3. El porcentaje promedio individual multiplica la nota del proyecto (**Np**) en general.

$$Ni = [(A+C1+C2+C3) /4]*Np$$

Importante: se ignorarán, del cálculo de nota individual, las auto y coevaluaciones que no sean realizadas. Si ningún miembro del grupo realiza su auto y coevaluaciones, la nota de la entrega será 0 para todo el grupo.