Documento de diseño– Proyecto 1

Ernesto Pérez – 202112530

Germán Moreno – 202116701

Sofia Velásquez – 202113334

Iteración #1:

Roles:

1. Proyecto
2. Usuario

Responsabilidades:

- Crear proyecto [1]
- Añadir participantes [2]

Iteración #2:

Roles:

1. Proyecto
2. Usuario
3. Actividad

Responsabilidades

- Crear proyecto [1]
- Añadir participantes [2]
- Iniciar actividad [3]
- Modificar actividad [3]
- Finalizar actividad [3

Iteración #3:

Roles:

1. Proyecto
2. Usuario
3. Actividad

Responsabilidades:

- Crear proyecto [1]
- Añadir participantes [2]
- Iniciar actividad [3]

- Modificar actividad [3]
- Finalizar actividad [3]
- Mostrar reporte de actividades [3]
- Cronometrar tiempo de trabajo [3]

Iteración #4:

Roles:

1. Proyecto
2. Dueño
3. Participante
4. Actividad
5. Reporte

Responsabilidades:

- Crear proyecto (iniciarlo) [1]
- Finalizar proyecto [1]
- Añadir participantes [2]
- Iniciar actividad [2-3]
- Modificar actividad [2-3]
- Finalizar actividad [2-3]
- Mostrar descripción de actividad [4]
- Mostrar la cantidad de actividades realizadas por usuario [5]
- Mostrar reporte del tiempo trabajado de cada usuario [5]
- Mostrar reporte de tiempo trabajado por día de cada usuario [5]
- Mostrar reporte de tiempo trabajado por tipo de actividad de cada usuario [5]
- Cronometrar tiempo de trabajo [4]

Iteración #5:

Roles:

1. Proyecto
2. Dueño
3. Participante
4. Actividad
5. Reporte

Responsabilidades:

<table>
<tr><th>Responsabilidad</th><th>Rol</th></tr>
<tr><td>Crear proyecto</td><td rowspan="2">Proyecto</td></tr>
<tr><td>Finalizar proyecto</td></tr>
<tr><td>Añadir participantes</td><td>Dueño</td></tr>
<tr><td>Iniciar actividad</td><td rowspan="3">Dueño o participante (USUARIO)</td></tr>
<tr><td>Modificar actividad</td></tr>
<tr><td>Finalizar actividad</td></tr>
</table>

Mostrar descripción de actividad	Actividad
Cronometrar tiempo de trabajo	
Mostrar la cantidad de actividades realizadas por usuario	Reporte
Mostrar reporte del tiempo trabajado de cada usuario	
Mostrar reporte de tiempo trabajado por tipo de actividad de cada usuario	

Colaboraciones:

- Proyecto colabora con actividad y con usuarios para obtener su información y poder crearse.
- Dueño colabora con proyecto para añadir participantes.
- Reporte colabora con actividad y con usuario para poder generarse.

Justificación de decisiones:

- Decidimos remover la clase cronómetro, debido a que los métodos que se planeaban implementar en ella resultaron estar mejor ubicados en la clase actividad.
- Para optimizar el futuro desarrollo de la aplicación, decidimos mantener las dos horas tanto de inicio como de fin en un solo ArrayList conteniendo a ambos.
- Para calcular los tiempos el usuario tiene una lista con todas las actividades realizadas y la actividad misma es un diccionario con llave= nombre de la actividad, valor = atributos de la actividad.
- Decidimos cambiar el estereotipo de la clase usuario a information holder, debido a que si fuera una interface solo podríamos definir el comportamiento de Dueño y Participante, pero no especificarlo. En cambio, decidimos que Dueño y Participante heredaran los métodos de Usuario, debido a que ambos tienen los mismos requerimientos (dueño tiene uno adicional).

Reflexión

Ventajas y desventajas (tradeoffs)

El diseño iterativo es muy bueno en el sentido en que es muy sistemático y permite elaborar un proyecto grande con un alto nivel de detalle. Una desventaja es el tiempo que consume esta iteración tan minuciosa de todos los elementos pues nos dimos cuenta de que podíamos hacer la primera iteración con un alto nivel de detalle y en las siguientes iteraciones más que desglosar nuestros elementos en elementos más pequeños nos hubiéramos concentrado en 'pulir' responsabilidades, roles, colaboraciones y demás, haciendo del proceso mucho más rápido sin mayores complicaciones.

Elementos problemáticos.

Por un lado, tuvimos problemas con el reporte y la manera en que este calcularía el tiempo total, por actividad y por día y como conocería esta información del usuario y también el tipo de estructura datos más apto que nos permitiera guardar los tiempos y hacer la suma total dependiendo del método llamado.

Ubicar el método de modificadorActividad dentro de la clase fue importante porque era posible tener este método dentro de la misma clase actividad, pero decidimos que enfocarlo en una clase dedicada era la mejor opción para mantener todo más simple.

Glosario de Métodos:

Clase Proyecto:

- CrearProyecto(); Crea el proyecto
- darProyecto(); Da nombre, descripcion, fecha inicial y final del proyecto y el dueño.
    - Falta dar participantes
- darinfoActividades(); Da lista de actividades por su nombre y un registro de todas las actividades con el mismo nombre para saber si se hizo durante varios días.
- darParticipantes(); da los participantes del proyecto, los participantes son
- darReporte(String correo); Pide el reporte de un usuario dado su correo

Clase Usuario:

- recibirReporte(ArrayList <Actividad>); pide a la clase Reporte dar un reporte para el correo dado como parámetro.
- iniciarActividad(String correo, String nombreActividad, tipoActividad tipo, String descripción); inicia una actividad en el proyecto.
- finalizarActividad(String correo, String nombreActividad); finaliza una actividad en el proyecto.

Clase Dueño:

- añadirParticipante (Usuario participante); Añade un participante a el proyecto con el correo y su nombre.

Clase Participante:

Nota: hereda los métodos de la clase usuario

Clase Reporte:

- cantActividades (ArrayList <Actividad>); cantidad de actividades realizadas por usuario dado su correo.
- tiempoTrabajo (ArrayList <Actividad>); reporte del tiempo trabajado del usuario
- tiempoTrabajoDia (ArrayList <Actividad>); reporte de tiempo trabajado por día del usuario
- tiempoTrabajoTipo (ArrayList <Actividad>); tiempo trabajado por tipo de actividad del usuario

Clase Actividad:

- consultarInformacion (); Imprime toda la información de la actividad (titulo, descripción, tipo, fecha inicio, hora inicio, hora final y participante que la realizo. Adicionalmente, exporta esta información en un archivo de texto txt.
- initCronometro(); inicia el cronometro para tomar el tiempo de realización de una actividad.
- stopCronometro(); termina el cronometro y recibe el tiempo en que se demoró la actividad.

Clase modificadorActividad:

- modificarActividad(); Modifica la fecha, hora de inicio y fin para las actividades, se pueden dar varias modificaciones para una sola actividad. No se puede modificar el título, descripción y tipo de una actividad.

Clase Consola:

- mostrarMenu(): void Muestra el menú de opciones
- main(String[] args):void
- ejecutarOpcion(int opcionSeleccionada):void Ejecuta una de las opciones del menú.

Modificaciones durante la implementacion

1. Se creo la clase llamada PrManager. Como su nombre lo dice simplemente se encarga de manejar el proyecto actual en el que se encuentra dado que el programa puede guardar múltiples proyectos.

Con los atributos:

- ArrayList<Proyecto> proyectos -> La lista de todos los proyectos creados

Y con los métodos:

- crearProyecto (String nombre, String descripcion, Usuario duenio, Date fechaFin)
  - Se encarga de crear el proyecto y añadir el proyecto a la lista de todos los proyectos
- getProyecto(int id)
  - Conseguir un proyecto dado el id
- getId()
  - Encargada de conocer el número de proyectos actuales para saber cuál es el siguiente id.

2. Se usa la librería serializable para guardar los objetos directamente por lo que no se trabaja con archivos de texto ni csv's.
3. Se añadieron métodos de consola y métodos para retornar informaciones que no estaban descritos en el diseño pero que su funcionalidad es muy simple y concreta.