Práctica 4

**ÍNDICE**

1. Definición del problema

2. Extracción de requisitos

2.1. Requisitos Funcionales

2.2. Requisitos No Funcionales

3. Historias de usuario

4. Casos de uso

5. Diagrama de clases

6. Diagramas de secuencia

7. Metodología SCRUM

7.1. Product Backlog

7.2. Sprint Backlog

7.3. Burndown Chart

8. Matrices de validación

1. Definición del Problema

Un cliente nos ha pedido un programa para almacenar alumnos. Dicho programa permite al usuario poder almacenar alumnos, modificarlos, borrarlos, entre otras funcionalidades.

Además, dicho cliente nos ha pedido que su programa funcione en el sistema operativo Linux, y que el fichero en el que se almacenen los alumnos sea de tipo binario.

El cliente nos ha indicado que el programa no sólo lo va a manejar él, sino que puede ser que lo utilicen varios profesores, los cuales serán sus ayudantes, por lo que el programa debe ser capaz de diferenciar entre un profesor u otro, y limitar la funcionalidad dependiendo del profesor que sea.

2. Extracción de Requisitos

El cliente nos ha pedido un programa que almacene alumnos, y este alumno tenga varios campos para almacenar datos. Los campos son los siguientes:

- DNI del alumno

- Nombre del alumno

- Apellidos del alumno

- Líder de un grupo o no

- Nº de grupo

- Email

- Edad

Además, nos ha pedido una serie de requisitos, tanto funcionales como no funcionales.

2.1 Requisitos Funcionales

El cliente nos ha pedido que su programa debe contener los siguientes requisitos funcionales:

- Buscar alumno por DNI

- Modificar alumno

- Mostrar todos los alumnos

- Modificar líder de grupo

- Borrar alumno

- Insertar alumno

- Borrar todos los alumnos

- Hacer copia de seguridad

- Cargar copia de seguridad

- Dar de alta a un profesor

- Dar de baja a un profesor

- Entrar al sistema mediante Usuario y Credencial

- Salir del sistema

2.2. Requisitos no funcionales

El cliente nos ha pedido que el programa contenga los siguientes requisitos no funcionales:

- Funcionar en sistema Linux

- El sistema pueda cargar un archivo binario

- Para cargar alumno se necesita mínimo DNI y apellido

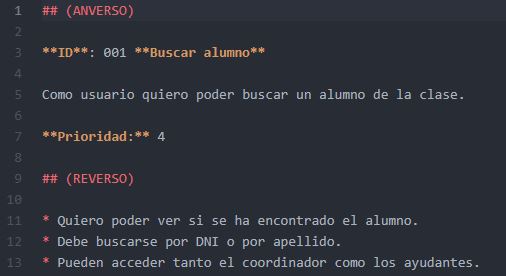
- El nº máximo de alumnos es de 150

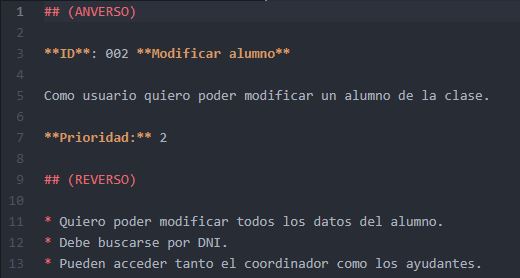
- Los datos se almacenan en un fichero binario

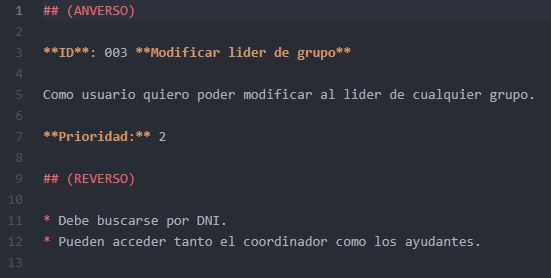
- Se considera la posibilidad de que haya varios profesores, cada uno con una credencial

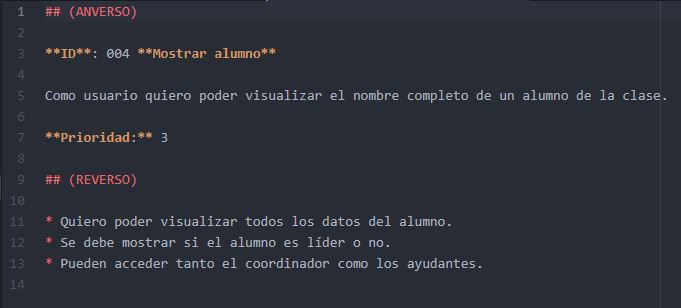
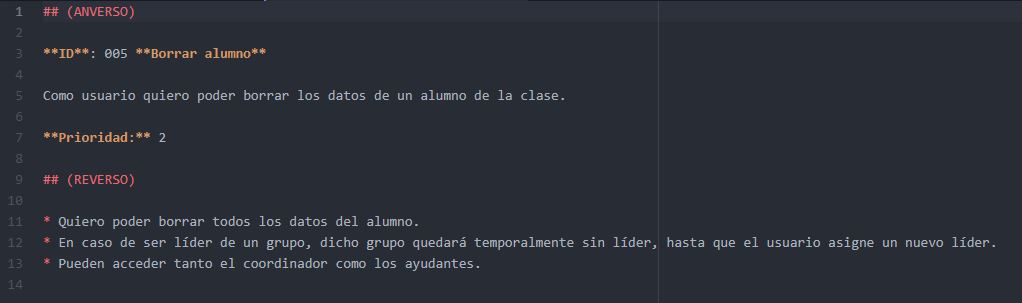
- Los profesores estarán divididos en Coordinador, el cual sólo habrá uno, y Ayudantes

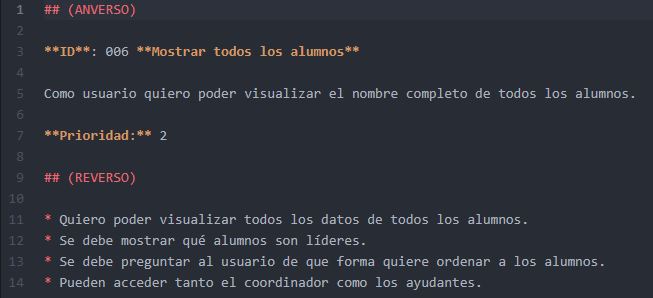
3. Historias de Usuario

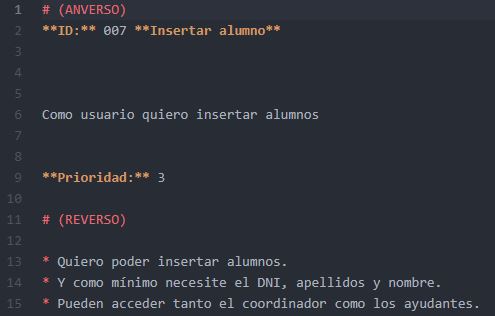
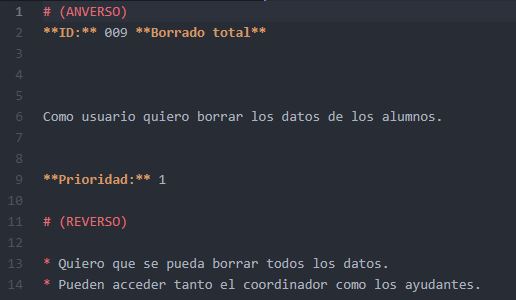
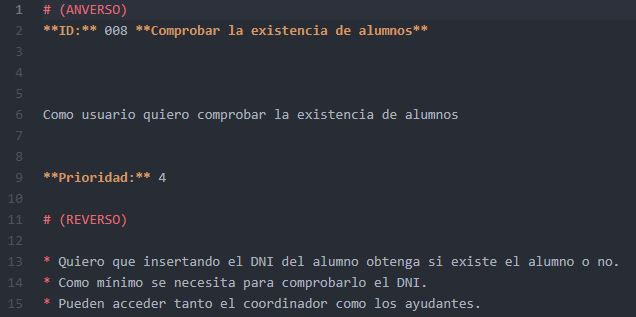
****Conforme a los requisitos funcionales que nos ha solicitado el cliente, se han obtenido las siguientes Historias de Usuario:

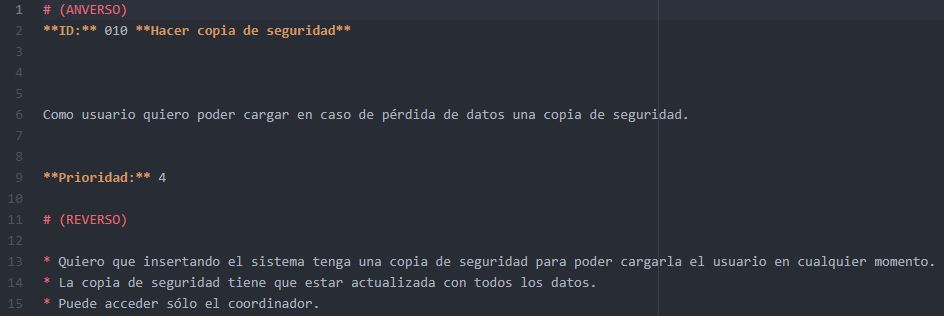
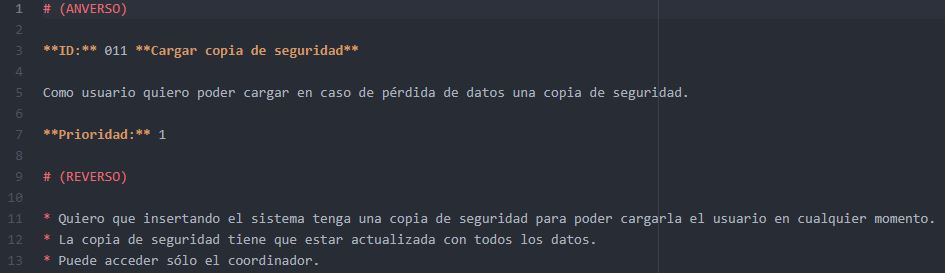
****

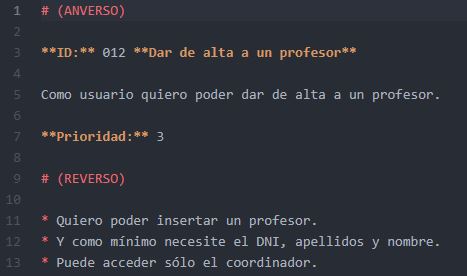
****

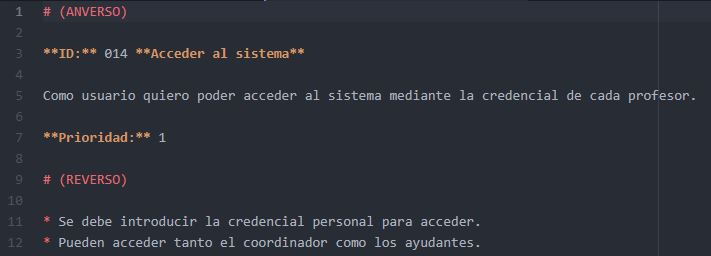
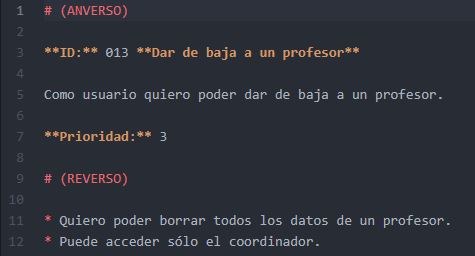
****

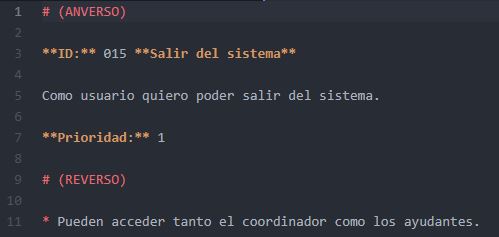
****

****

****

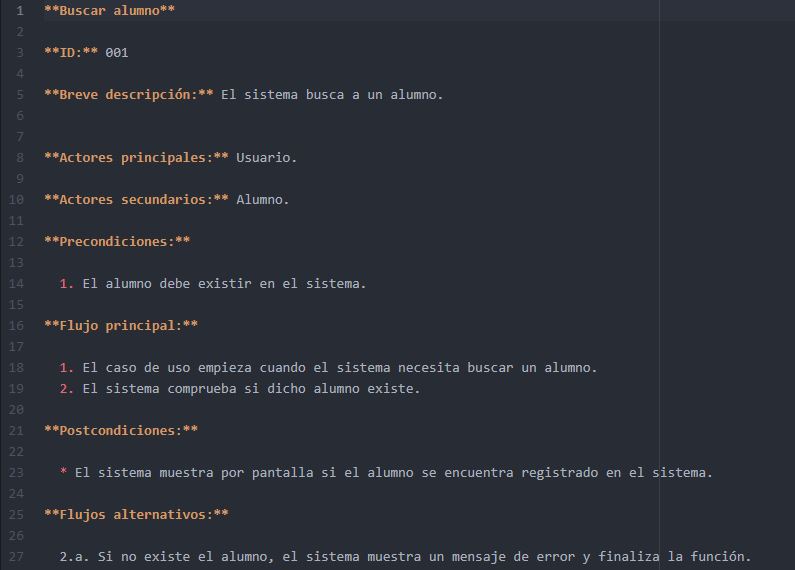
****

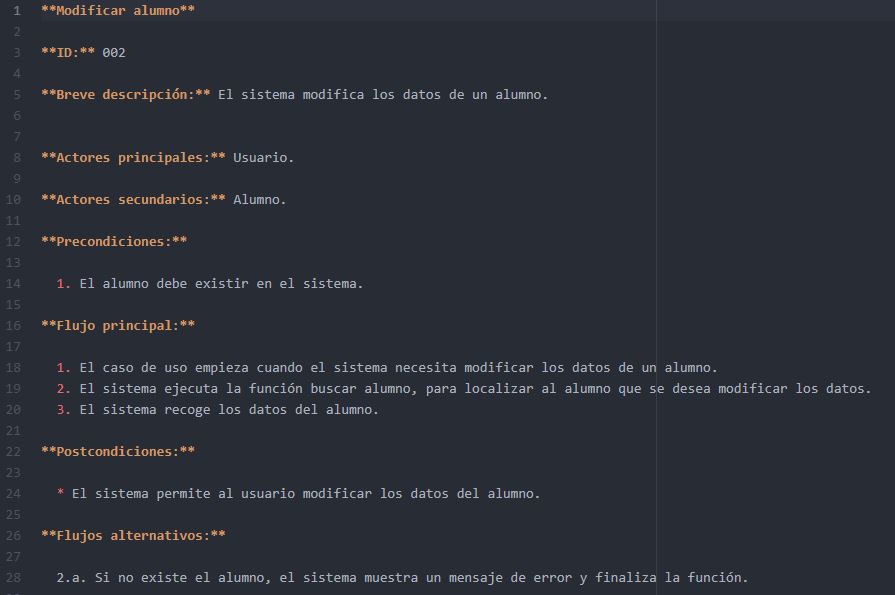
****

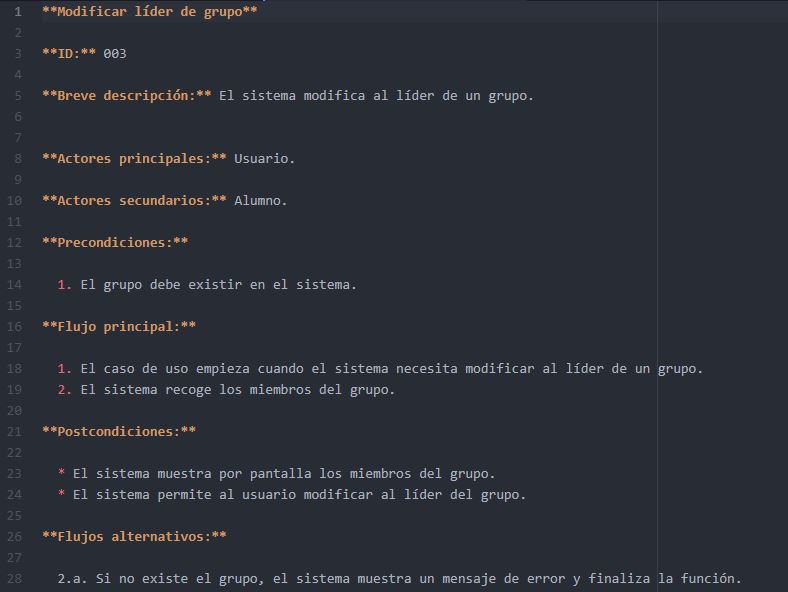
****

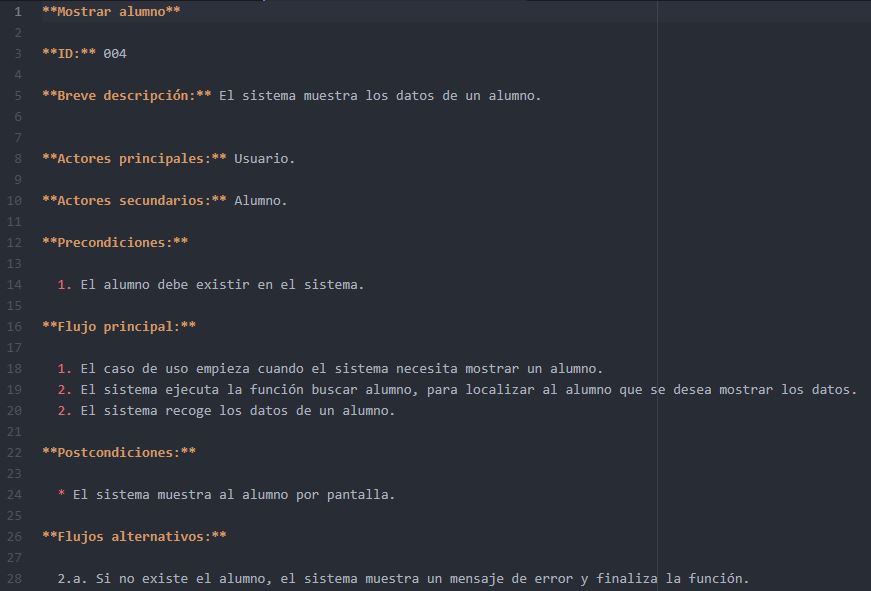
4. Casos de Uso

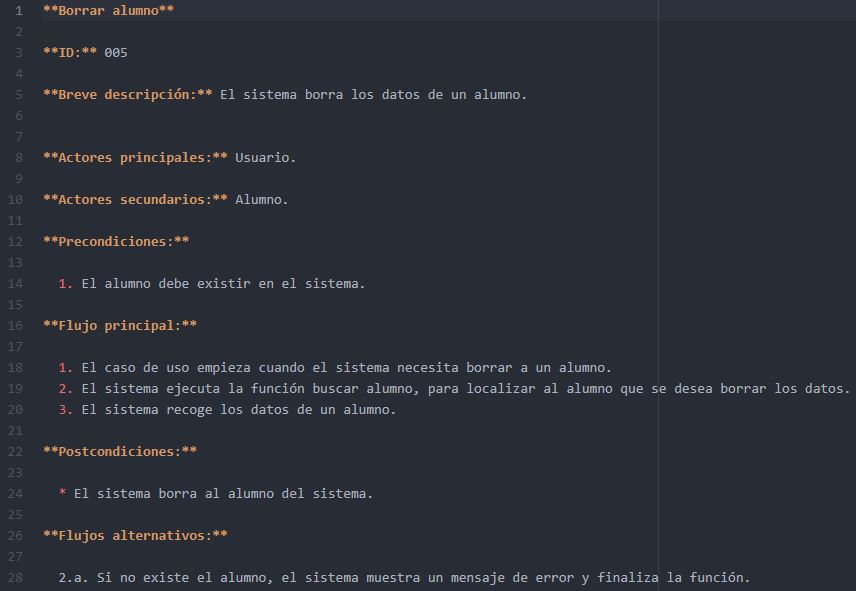
Tras crear las diferentes historias de usuario en base a los requisitos funcionales que nos solicitó el cliente, se crean los siguientes casos de uso:

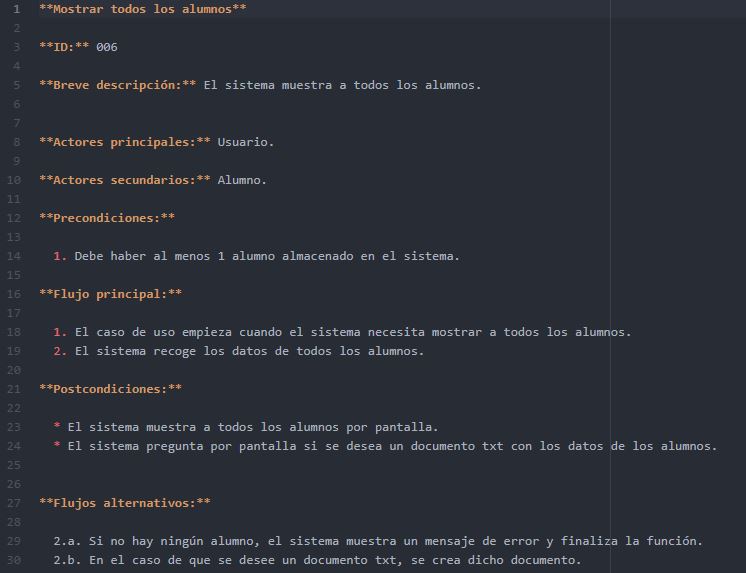


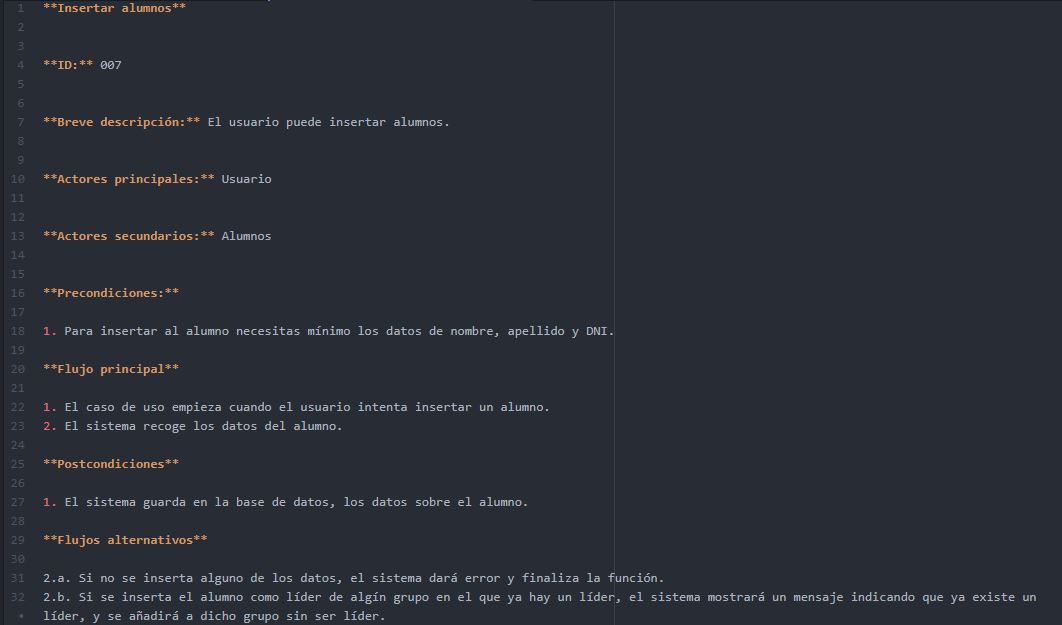


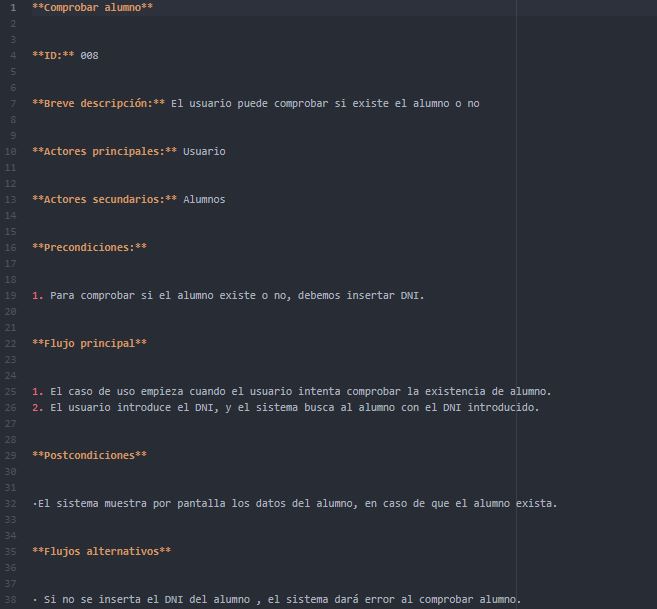


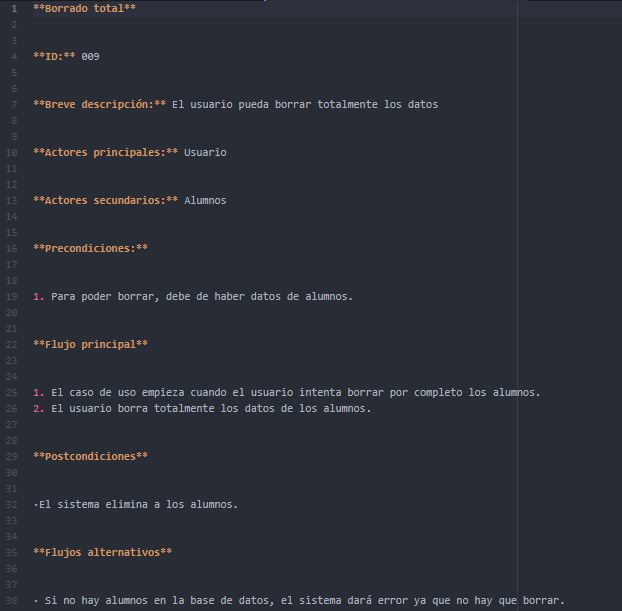


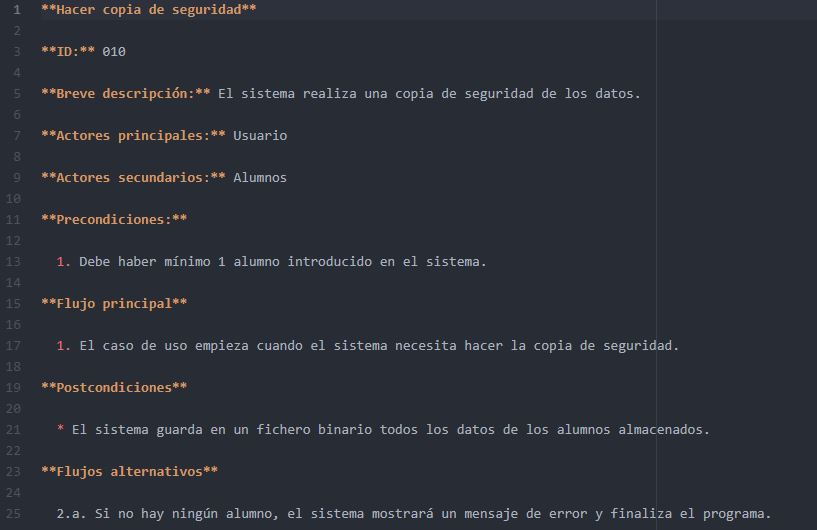


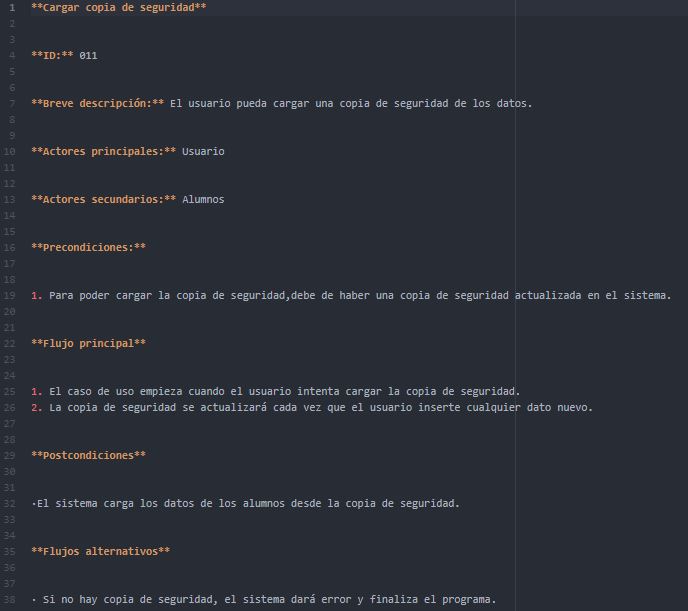
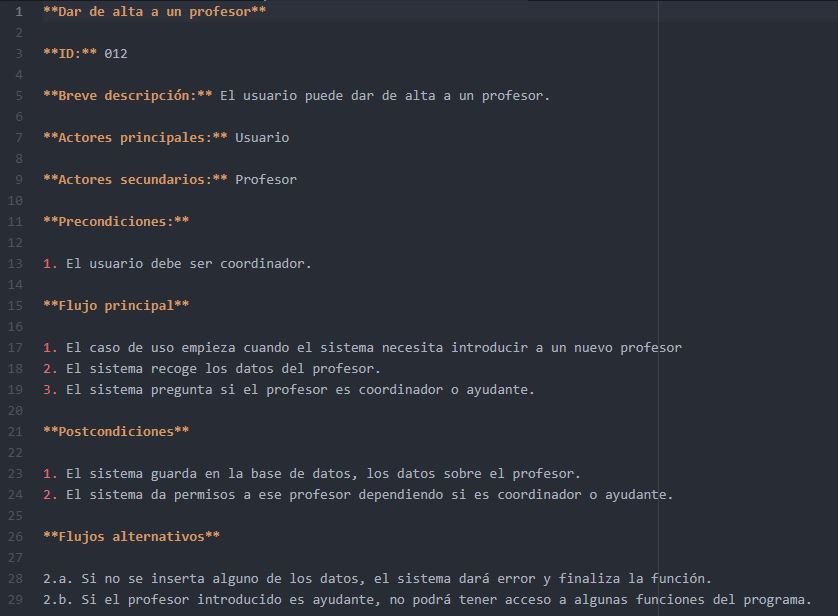


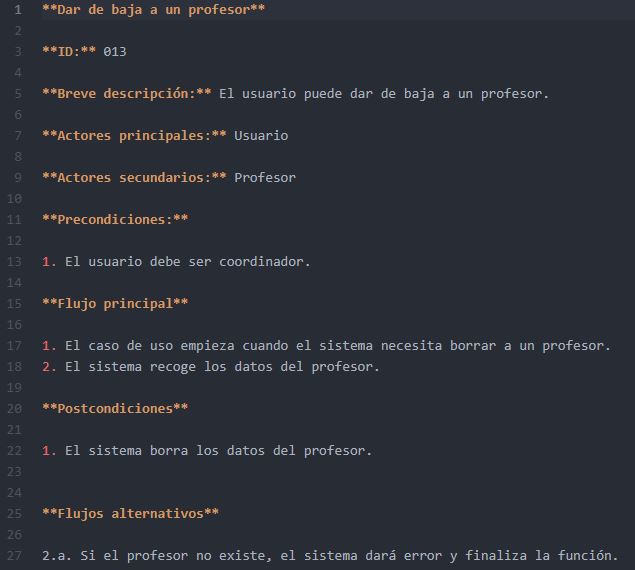
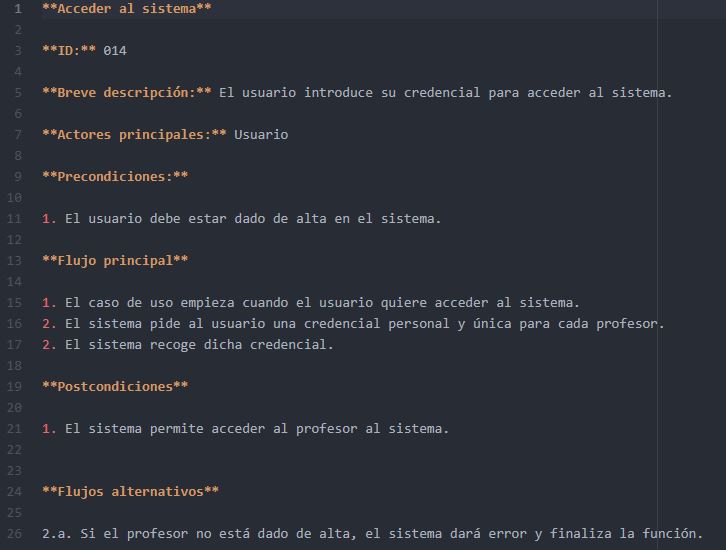


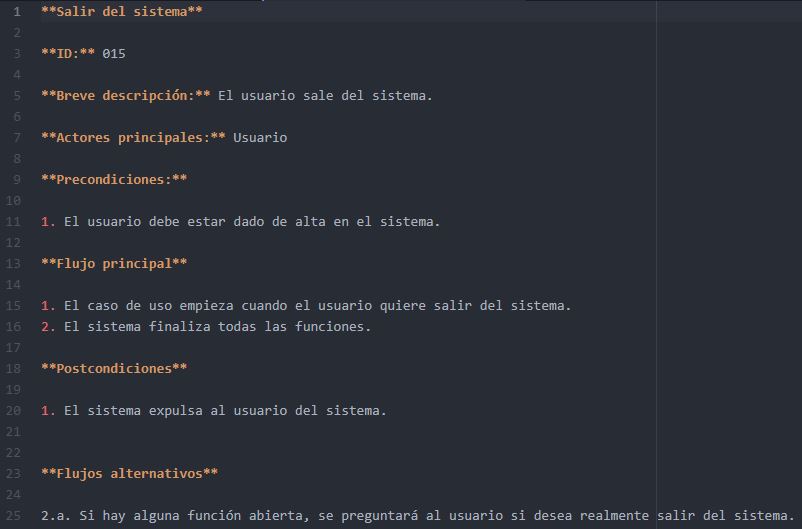












7.-Metodología SCRUM

**La metodología SCRUM** permite abordar proyectos complejos desarrollados en entornos dinámicos y cambiantes de un modo flexible. Esta metodología está basada en entregas parciales y regulares del producto final en base al valor que ofrecen a los clientes. En nuestro caso, utilizamos para nuestra entrega 3 elementos de la metodología: Product Backlog, Sprint Backlogs y Burndown charts.

7.1.- Product Backlog

Contiene todas las funcionalidades ordenadas por prioridades. Es la agrupación de todas las historias de usuario del producto.

En nuestro caso, el product backlog contiene 15 funcionalidades ordenadas por prioridad y en cada funcionalidad su historia de usuario. Además, el tiempo estimado por función.

Las funcionalidades se clasifican en 5 prioridades, que van desde prioridad 0 a prioridad 4. Siendo prioridad 0 lo más esencial del producto.

Las funcionalidades (ordenadas por ID) son las siguientes:

·· Buscar Alumno

·· Modificar Alumno

·· Modificar líder de Grupo

·· Mostrar Alumno

·· Borrar alumno

·· Mostrar todos los alumnos

·· Insertar Alumno

·· Comprobar la existencia de alumnos

·· Borrado total

·· Hacer copia de seguridad

·· Cargar copia de seguridad

·· Dar de alta a un profesor

·· Dar de baja a un profesor

·· Acceder al sistema

·· Salir del sistema

7.2.- Sprint backlogs

Funcionalidad a desarrollar en un sprint determinado.

En nuestro caso, presentamos 3 sprint backlogs. En el primero determinamos a cada integrante del grupo sus funcionalidades a crear. Como en nuestro grupo somos solo 2 integrantes, determinamos a un miembro las funciones de prioridad 2 y 3, mientras que al otro integrante se le determinó las demás funciones y la declaración de clases get y set.

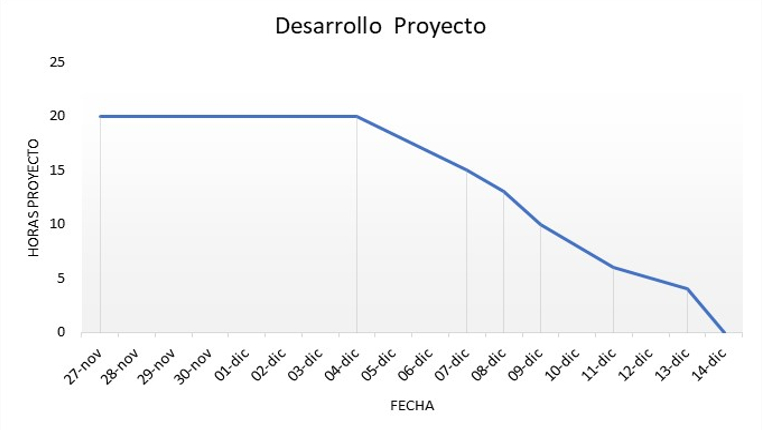
En el segundo determinamos a cada integrante del grupo la tarea de seguir creando sus funcionalidades respectivamente. Pero ya no era necesario la declaración de clases get y set, ya que ya habían sido creadas.

En el tercero y último determinamos a un miembro a acabar las funcionalidades de modificar alumno, modificar líder, dar de alta a un profesor, dar de baja a un profesor y mostrar alumno, además de comprobar las funciones y corregir respectivos fallos. Mientras que al otro miembro solo se le determinó comprobar fallos en sus funciones y constructores.

7.3.- Burndown charts

Gráfico que muestra la cantidad de trabajo hecho.

En nuestro caso, este gráfico muestra 20 horas de estimación para hacer el producto.



Los integrantes del grupo empiezan a desarrollar sus funcionalidades el 4 de Diciembre y para el 7 de Diciembre han acortado 5 horas de trabajo. Mientras que desde esa fecha hasta el 13 de Diciembre se acortan 10 horas de trabajo. Y para acabar el producto se dedican 5 horas desde el 13 de Diciembre hasta el 14 de Diciembre, en la que se corrigen errores en las funcionalidades y constructores.

8.-Matriz de validación

FICHA TÉCNICA DEL CLIENTE

··Nombre:

··Profesión:

··Ocupación:

··Teléfono:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Aspectos** | Muy  Adecuado  5 | Adecuado    4 | Medianamente  Adecuado  3 | Poco  Adecuado  2 | Nada  Adecuado  1 |
| Trabajo |  |  |  |  |  |
| Funciones |  |  |  |  |  |
| Casos de uso |  |  |  |  |  |
| Historias de usuario |  |  |  |  |  |
| Diagrama de clases |  |  |  |  |  |
| Diagrama de secuencia |  |  |  |  |  |
| Product Backlog |  |  |  |  |  |
| Sprint backlog |  |  |  |  |  |
| Burndown charts |  |  |  |  |  |
| Ajustarse al tiempo de entrega |  |  |  |  |  |
| Calidad del código o producto |  |  |  |  |  |

9.-Bibliografía

-Material de Moodle: <https://moodle.uco.es/m1819/course/view.php?id=2230>

-Material para código y funciones:

<http://www.cplusplus.com/>

-Material para diagramas:

# <https://www.youtube.com/watch?v=Q1kH7XKxK5I>

-Material de GitHub:

<https://moodle.uco.es/m1819/pluginfile.php/110517/mod_resource/content/2/P1_IngenieriaSoftware_Presentacion.pdf>