Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Diseño, implementación e integración de un plugin para la realización de encuestas y exámenes en la red social Pesedia.

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autora: Silvia Carpena Tobarra

Tutor: Agustín Rafael Espinosa Minguet

2016-2017

Resumen

Este proyecto de final de grado consiste desarrollar un *plugin* que permita crear y responder encuestas y exámenes para la red social educativa Pesedia, creada por el grupo de investigación Tecnología Informática - Inteligencia Artificial(GTI-IA) de la Universitat Politècnica de València y basada en la plataforma de código abierto Elgg. Para ello ha sido necesario llevar a cabo diferentes fases: la especificación de requisitos del proyecto, el análisis, el diseño, la implementación, la integración y, por último, las pruebas para verificar la usabilidad y funcionalidad del *plugin.* Para ello se han utilizado en su totalidad herramientas de código abierto, siguiendo además las buenas prácticas recomendadas por Elgg para crear un nuevo *plugin* que pueda ser añadido cuando se desee a la plataforma.

Palabras clave: plugin, encuesta, examen, Elgg, PHP

Resum

Aquest projecte de fi de grau consisteix en el desenvolupament d'un plugin que permeta crear i respondre enquestes i exàmens per a la xarxa social educativa Pesedia, creada pel grup d'investigació Tecnologia Informàtica - Intel·ligència Artificial (GTI-IA) de la Universitat Politècnica de València, i basat en la plataforma de codi obert Elgg. Amb aquesta finalitat ha estat necessari dur a terme diferents tasques: l'especificació de requisits del projecte, l'anàlisi, el disseny, la implementació, la integració, i, finalment, les proves per a verificar la usabilitat i funcionalitat del plugin. Tot el programari utilitzat per assolir aquest objectiu és de codi obert, i a més a més, segueix les bones pràctiques recomanades per Elgg per a crear un nou plugin que puga ser afegit a la plataforma quan es desitge.

Paralules clau: plugin, enquestes, exàmens, Elgg, PHP.

Abstract

This final project consist on developping a plugin with the functionality of create and answer polls and quizs for the educative social network Pesedia, created by the Grupo de Tecnología Informática e Inteligencia Artificial(GTI-IA) of the Universitat Politècnica de València based on the open-source plaform Elgg. Throughout the project it has been necessary to carry out the following phases: the specification of the plugin requirements, analysis, desing, implementation, integration and finally, tests, in order to guarantee the usability and functionality of the plugin. The whole project has been developed using open-source tools, following the recommended good practices of the Elgg documentation, in order to create a plugin that can be added to the platform.

Keywords : plugin, poll, quiz, Elgg, PHP

Índice General

Índice de tablas

Índice de figuras

1. Introducción

Actualmente las redes sociales son una de las herramientas más utilizadas en Internet. Estas plataformas tienen como principal objetivo la comunicación entre sus usuarios, y ofrecen multitud de formas para llevarla a cabo: emisión de contenido multimedia en vivo o bajo demanda, mensajes directos o diferentes tipos de publicaciones e interacciones con el contenido publicado. Las redes sociales no son una novedad, y existen multitud de ellas y con diferentes propósitos. En ellas se agrupan individuos que comparten aficiones, parentesco, trabajo, pasatiempos, etc.

El inicio de las redes sociales se encuentra aproximadamente en el año 2000, cuando múltiples plataformas surgieron con el fin de posibilitar la comunicación entre las personas a través de Internet. Durante los años posteriores fueron apareciendo algunos de los sitios más populares de esta índole: MySpace (2003), Twitter (2006), Facebook (2007), Google+ (2011).

Son muchas las redes sociales que cuentan con el respaldo de empresas multinacionales y han acabado convirtiéndose en un auténtico negocio; no obstante, no es el caso de todas ellas.

Existen redes sociales de código abierto que nos permiten su desarrollo a nuestro gusto, como lo son BuddyPress, Elgg, Dolphin o NewsCloud. Las ventajas de una red social de código abierto frente a una que no lo sea es que podemos contribuir en su desarrollo, implementando nosotros mismos parte de su funcionalidad: extendiéndola, depurándola e incluso mejorándola.

En este proyecto se ha utilizado Pesedia como red social a la que se añadirá la funcionalidad (en forma de *plugin*) de crear y responder encuestas y exámenes. Pesedia no es en sí una plataforma de desarrollo, sino que es una red social creada por el grupo de investigación Tecnología Informática - Inteligencia Artificial de la Universitat Politècnica de València basada en el sistema de gestión de contenidos (CMS) Elgg.

Algunas de las redes sociales más conocidas como lo son Facebook o Twitter ya presentan la funcionalidad de crear encuestas de preguntas cortas y respuesta única, pero una red social con fines educativos como lo es Pesedia requiere de la incorporación de un *plugin* que, además de crear encuestas personalizadas, también permita crear exámenes y la evaluación de los mismos.

1.1 Objetivos

El objetivo principal de este proyecto es la incorporación de un *plugin* a la red social Pesedia que cumpla con los siguientes objetivos secundarios:

- Preparar dicho *plugin* para su incorporación en Pesedia, red social basada en Elgg creada por el grupo de investigación Tecnología Informática - Inteligencia Artificial de la Universitat Politècnica de València. Trabajar, por lo tanto, en un proyecto de gran envergadura en el que participan muchas otras personas.

- Permitir el acceso al *plugin* desde una de las opciones del menú principal de Pesedia (siendo esta la única alteración del diseño original de Pesedia que queda fuera del *plugin* en sí).

- Permitir al usuario de Pesedia crear encuestas y exámenes desde cero añadiendo diferentes tipos de pregunta. La encuesta o examen podrá ser eliminada y editada en cualquier momento.

- Proporcionar al creador de la encuesta o examen diferentes opciones de configuración como la hora de activación y entrega de la encuesta o examen, tiempo límite para enviar la respuesta o número de veces que se podrá enviar la respuesta.

- Permitir que el usuario pueda asignar permisos a sus propias encuestas y exámenes para decidir quién puede acceder a ellos.

- Crear una arquitectura que tenga en cuenta la internacionalización y localización; es decir, programar el *plugin* de manera que pueda adaptarse a diferentes idiomas y regiones sin la necesidad de modificar su código fuente. Este proyecto soportará los idiomas castellano e inglés en el momento de su lanzamiento.

- Demostrar la capacidad de la autora para desenvolverse con nuevas tecnologías no conocidas hasta el inicio del proyecto como la plataforma Elgg, el lenguaje de programación PHP o llevar a cabo una especificación de requisitos, análisis y diseño de un proyecto completo.

1.2 Estructura de la memoria

En primer lugar, en esta memoria se puede ver una puesta en contexto del *plugin* que se va a desarrollar: por qué añadir la funcionalidad de crear encuestas y exámenes a través de un *plugin*, qué es Pesedia, qué es Elgg y cuáles han sido las otras aplicaciones y software que se han utilizado como referencia para crear el *plugin*.

En segundo lugar se ha procedido a realizar una especificación formal de requisitos siguiendo para ello el estándar IEEE 29148-2011, donde se detallan los requisitos y funcionalidades que ha de cumplir el *plugin* en base al ámbito del proyecto.

En tercer lugar, se habla de la fase de análisis, donde se especifica cómo se van a satisfacer los requisitos y funcionalidades de la especificación de requisitos, comenzando a dar una primera estructura a la forma de organizar los datos con los que se trabajará el *plugin*.

En cuarto lugar, en la fase de diseño se explica cuál será el modelo de datos del *plugin*, respaldado por el modelo de datos ya presente en Elgg, así como también se lleva a cabo unos primeros bocetos de las vistas del *plugin*.

En quinto lugar, la fase de implementación consta de las tecnologías empleadas para el desarrollo del *plugin*, así como de una explicación detallada de los directorios y archivos que se han utilizado para la programación del mismo.

En sexto lugar se encuentra el apartado de despliegue, donde se pueden observar los pasos seguidos para poder instalar y trabajar con Elgg, y por lo tanto, para poder utilizar un *plugin* en esta plataforma.

En séptimo lugar se exponen en la memoria las pruebas que se han realizado tanto durante el desarrollo del *plugin* como tras su finalización, haciendo mención de las mejoras y cambios que se han llevado a cabo.

Por último, esta memoria acaba con una conclusión donde se realiza una reflexión acerca del desarrollo del *plugin* en sí, mencionando las dificultades que han surgido y las ventajas que este proyecto ha aportado.

2. Contexto

Un *plugin* en un contexto como lo es el de la informática podría definirse como una aplicación que permite incluir una nueva funcionalidad o característica al programa o herramienta original.

La aparición del primer *plugin* se sitúa en la década de los 70, cuando un editor de texto llamado EDT permitió que otro programa externo incluyese en él una nueva funcionalidad.

Lo más habitual es que el *plugin* sea ejecutado por el programa principal, donde el usuario puede elegir si activarlo o no a través de una interfaz. Los *plugins*, una vez activados, pueden haber sido configurados para ejecutarse antes o después de un evento concreto, ante una acción del usuario o añadir una interfaz propia.

Podemos encontrar *plugins* en sistemas operativos, plataformas de *blogging*, navegadores web, sistemas de gestión de contenidos, reproductores de audio y video y, en definitiva, en gran multitud de *software* de todo tipo.

Este proyecto pretende añadir un *plugin* que a través de la incorporación de una interfaz gráfica permita la creación de encuestas y exámenes, así como su respuesta y visualización de resultados en Pesedia, red social basada en Elgg.

Resulta también conveniente hablar en detalle sobre qué es Elgg exactamente, y el porqué de la elección de incorporar esta funcionalidad nueva a través de un *plugin*.

2.1 ¿Qué es Elgg?

Elgg es una plataforma de servicios de red social de código abierto que proporciona un marco de trabajo el cual permite construir todo tipo de entornos sociales. Fue creada en el año 2004 por Ben Werdmuller y Dave Tosh y actualmente posee dos licencias: MIT (que se aplica al código del núcleo únicamente) y GPLv2 (que se aplica tanto al código del núcleo de Elgg como a los *plugins* que vienen instalados en él por defecto).

Algunos ejemplos de uso de Elgg serían usarla para crear la red social de un campus universitario, la creación de intranets privadas o una plataforma colaborativa de una empresa que sirva como herramienta de comunicación entre ella y sus clientes.

En Elgg los *plugins* permiten extender nuestro sitio añadiendo nuevas funcionalidades, idiomas y temas. La mayoría de *plugins* de Elgg han sido desarrollados por miembros de la comunidad Elgg. Cualquier desarrollador puede descargar los *plugins* disponibles de forma gratuita en la tienda de Elgg o Github, subir el suyo propio y comentar o puntuar los *plugins* ya disponibles. Actualmente Elgg cuenta con más de 2214 *plugins* y más de 6366337 descargas de estos.

2.2 ¿Qué es Pesedia?

Pesedia es una red social con fines educativos creada por el grupo de investigación Tecnología Informática - Inteligencia Artificial de la Universitat Politècnica de València basada en herramientas de código abierto, entre ellas Elgg.

Desde su implementación, Pesedia ha sido utilizada en la Escola de Estiu de la UPV tanto por alumnos como por profesores, estando bajo un dominio de acceso público y seguro (HTTPS). A pesar de esto, el acceso a Pesedia ha estado habilitado únicamente a aquellos usuarios que dispusiesen de un nombre de usuario y contraseña, limitando el libre registro.

Actualmente se ha solicitado a la Secretaría Autonómica de Educación la implantación de Pesedia en diferentes institutos, con el fin de que a ella tengan acceso tanto alumnos, padres y profesores.

La evolución de Pesedia es constante y la inclusión de la funcionalidad de crear encuestas y exámenes es algo de gran utilidad para un sitio con fines educativos, por lo que en este proyecto se diseñará, implementará e integrará el *plugin* de manera que este pueda ser

utilizado por el grupo de investigación Tecnología Informática-Inteligencia Artificial en cualquiera de los sitios Pesedia a los que se desee añadir esta funcionalidad.

2.3 Aplicaciones y *plugins* relacionados

Antes de comenzar con la especificación de requisitos se ha llevado a cabo un estudio de algunas de las aplicaciones o plataformas actuales que tienen una funcionalidad igual o similar al *plugin* que se implementará en este proyecto. Esto ha permitido no solo identificar el contexto existente en la web en lo respectivo a la creación y respuesta de encuestas y exámenes, sino también descubrir funcionalidades muy útiles, e incluso necesarias, que han sido incluidas en el *plugin*.

A continuación se muestran unos ejemplos representativos.

PoliformaT

Es una plataforma online de la Universitat Politècnica de València (UPV) donde los alumnos y profesores pueden encontrar y publicar todo el material disponible para las diferentes asignaturas y cursos que se ofrecen. PoliformaT permite a los profesores la creación de exámenes *online* para que todos los alumnos matriculados en la asignatura puedan realizarlos de manera remota y posteriormente ver su calificación.

PoliformaT ha servido como referente para este proyecto en muchos aspectos, como, por ejemplo, la amplia gama de opciones de configuración que ofrece a la hora de crear un examen.

Figura 1: Captura de pantalla de la creación de un examen en PoliformaT

Quizworks

Quizworks es un *software* online que permite la creación de "Cuestionarios en forma de juego", "Exámenes" y "Evaluaciones". La opción "Examen" es la que más se acercaría al propósito de este proyecto (dentro de la categoría exámenes), aunque difiere en que Quizworks ofrece la posibilidad de crear preguntas con una serie de respuestas correctas introducidas en forma de texto y nuestro *plugin*, aparte de esta opción, también permitirá la creación de preguntas con otro tipo de formato: preguntas de respuesta múltiple, preguntas de respuesta única y preguntas de texto largo.

Figura 2: Captura de pantalla de la creación de un examen con Quizworks

Google Forms

Esta herramienta gratuita de Google es quizás una de las más utilizadas actualmente para la creación de encuestas online. Aunque sí que requiere tener una cuenta de Google para poder crear la encuesta, enviar respuestas no requiere de ningún tipo de registro (existe no obstante una opción de configuración que permite limitar las respuestas a una por persona, lo cual sí que hace necesario acceder a la encuesta con una cuenta de Google). Google Forms ha sido un gran referente a la hora de comenzar a dar forma a este proyecto, ya que es una plataforma muy completa, intuitiva y usable.

Google Forms, no obstante, no permite crear exámenes.

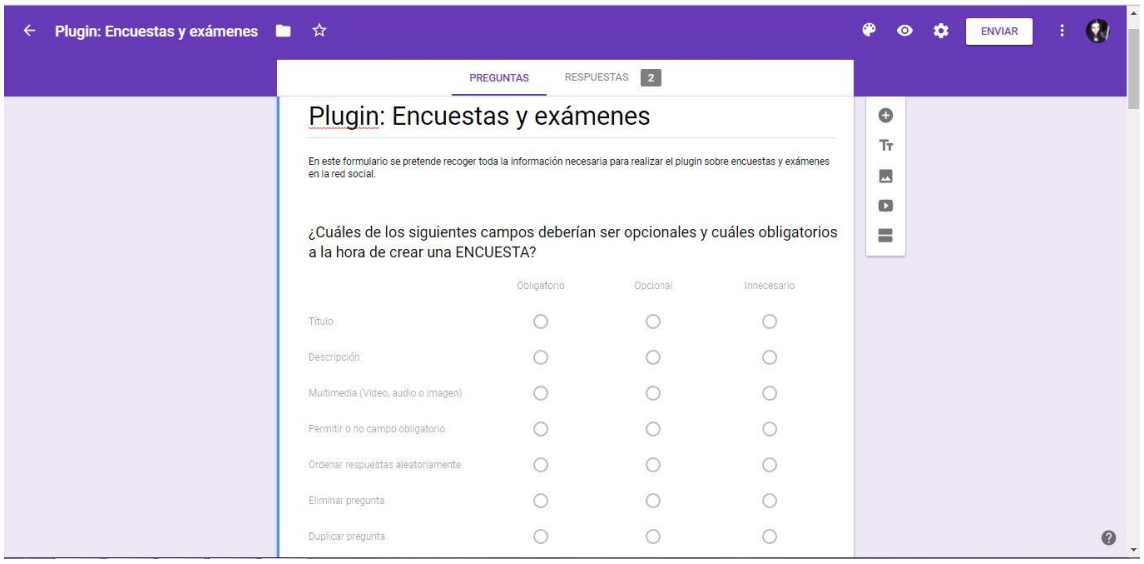

Figura 3: Captura de pantalla de la creación de una encuesta con Google Forms

Facebook

Facebook, una de las redes sociales más utilizadas a día de hoy, también permite crear encuestas. Estas, no obstante, ofrecen una funcionalidad que si bien puede resultar muy útil, es demasiado básica para el ámbito de este proyecto.

Las encuestas de Facebook permiten añadir una única pregunta con tantas respuestas como el creador de la encuesta desee. Los usuarios pueden votar aquella opción que les parezca, y cambiar su resultado a voluntad. Se puede configurar la encuesta para que los usuarios voten una única opción o varias, y también para que los propios usuarios añadan opciones a la encuesta.

Figura 4: Captura de pantalla de la creación de una encuesta en Facebook

Daypo

Daypo es una herramienta online que permite la realización de tests únicamente (lo que sería el equivalente a los exámenes del *plugin* que se va a desarrollar en este proyecto), actualmente cuanta con más de 231652 tests indexados que cualquier persona puede responder. Los tests permiten añadir preguntas de respuesta única, respuesta múltiple y texto corto (que se valida como correcta si el resultado de quien responda el test coincide exactamente con la respuesta introducida por el creador del test), entre otras.

Entre las diversas configuraciones que permite esta herramienta está incluir un tiempo límite de respuesta a las preguntas, elegir si el usuario que responde el test puede ver su respuesta correcta o elegir si las respuestas han de mostrarse de forma aleatoria. Estas funcionalidades han sido muy útiles para poder tener otro caso real como referencia a la hora de decidir qué opciones de configuración se podrían incluir en el *plugin* y de qué forma.

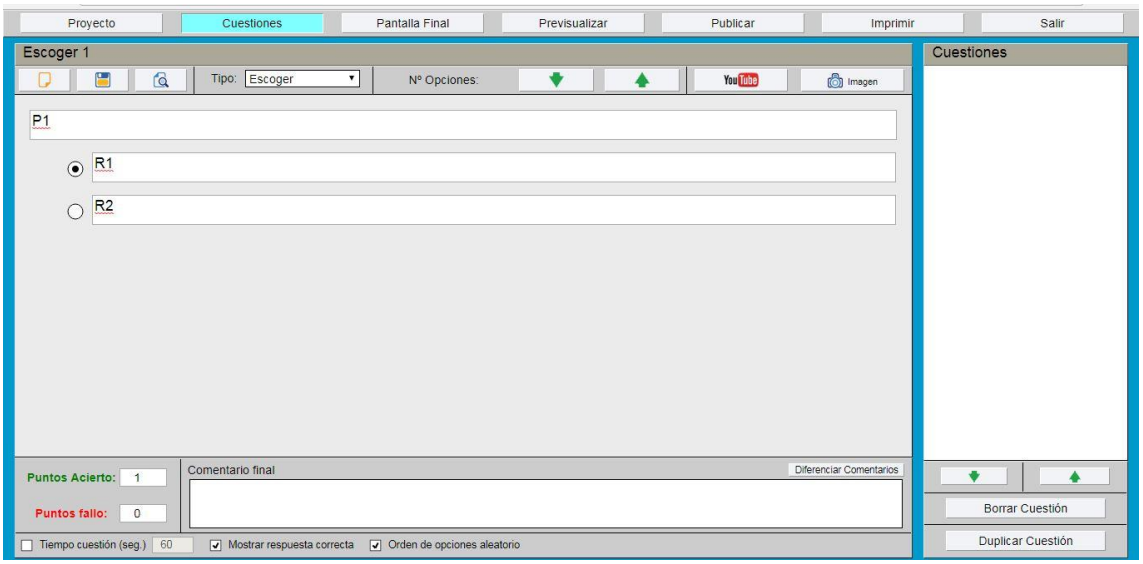

Figura 5: Captura de pantalla de la creacion de un test con Daypo

ProProfs

Esta herramienta requiere de registro previo para poder utilizarla. Permite la creación de encuestas y exámenes con los tipos de pregunta que ya se han ido mencionando anteriormente: respuesta múltiple y respuesta simple, texto y texto largo. ProProfs permite mostrar a aquellos que realicen exámenes las respuestas correctas al mismo tiempo que se realiza el examen, así como datos acerca del mismo al final: Número de respuestas acertadas, nota, lista de preguntas con sus respectivas respuestas y comentarios aclaratorios de cada una de las preguntas por el creador del examen.

Tanto en encuestas como exámenes permite la visualización de diferentes tipos de gráficos y estadísticas de los resultados.

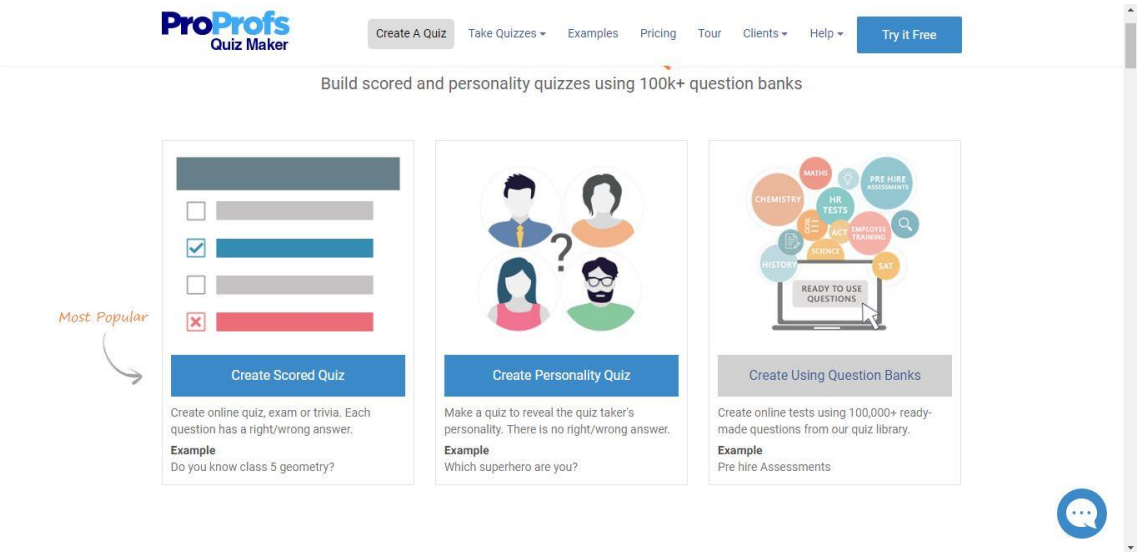

Figura 6: Captura de pantalla de la página principal de ProProfs

Edmodo

Esta es una de las plataformas que más similitud presentará con nuestro *plugin*. Edmodo requiere registrarse de manera gratuita para poder utilizar la plataforma, y a partir de ahí, permite crear tres tipos de contenido principales: Entregables, exámenes y encuestas. Los entregables son archivos o enlaces que se envían directamente a los alumnos (en esta plataforma se hace la distinción de dos roles de usuario, alumnos y profesores) con las directrices del propio entregable. Los exámenes, al igual que en *plugin* que se desarrollará en este proyecto, cuentan con preguntas de diferente tipo que el profesor puede puntuar y marcar sus resultados correctos. El examen puede ser configurado de formas diferentes: el profesor puede elegir si el alumno puede ver o no la corrección del examen tras su entrega, también puede elegir el tiempo que tiene el alumno para realizar el examen y si se desea que las preguntas se muestren de forma aleatoria.

En lo que a encuestas se refiere, únicamente podemos crear encuestas de una pregunta y de respuesta única, por lo que en este aspecto nuestro *plugin* sí que contempla una funcionalidad más extendida.

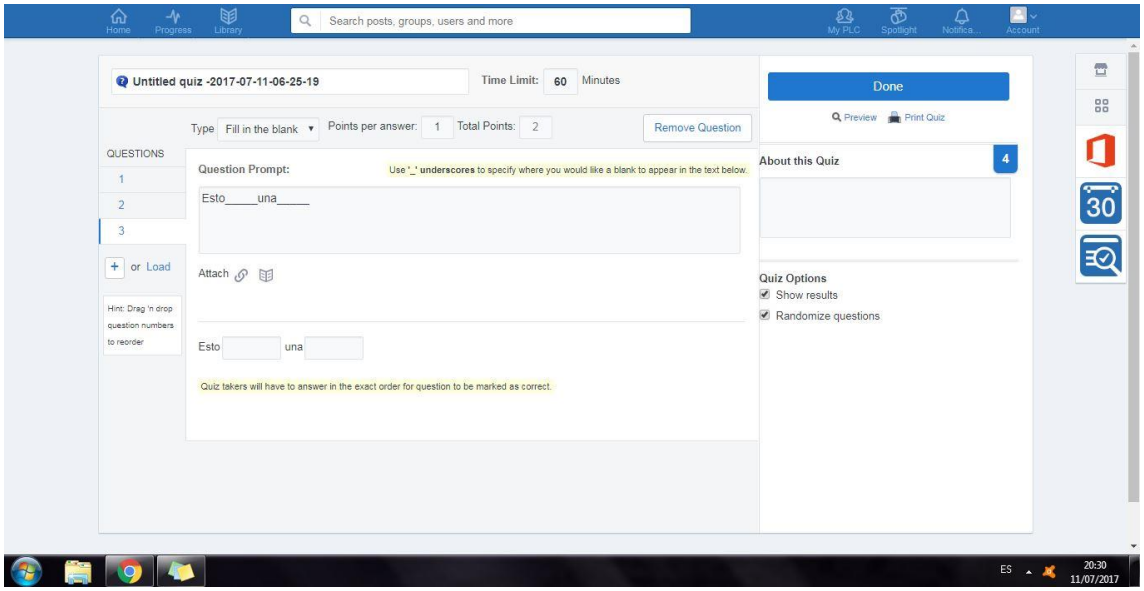

Figura 7: Captura de pantalla de un examen de Edmodo

Plugin Poll en Elgg

El propio Elgg incluye un *plugin* que permite la realización de encuestas, llamado Poll. Este *plugin* no obstante permite el lanzamiento de una sola pregunta de respuesta única con tantas opciones de respuestas como el usuario desee. El *plugin* desarrollado en este proyecto permite (en lo que a encuestas se refiere) no solo la creación de varias preguntas, sino diferentes tipos de respuesta: respuesta múltiple, respuesta única, texto y texto largo, aparte de añadir también la funcionalidad de crear exámenes.

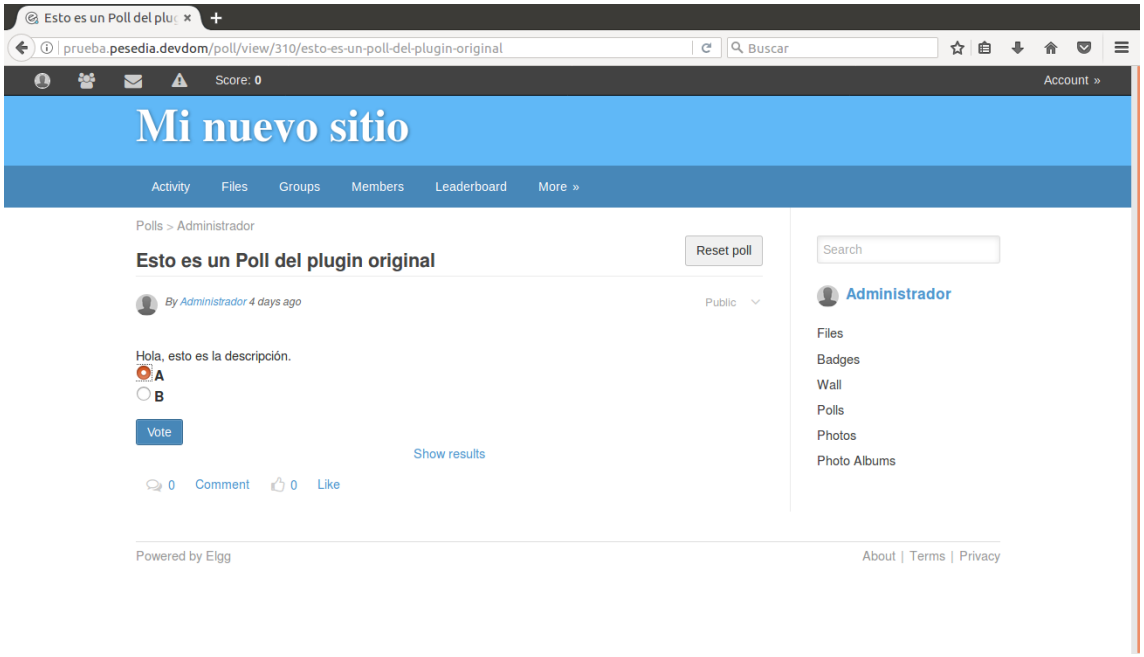

Figura 8: Captura de pantalla de un poll en Elgg

3. Especificación de requisitos

En este apartado de la memoria se va a proceder a realizar la especificación de requisitos teniendo como base el estándar IEEE 29148-2011, lo cual permitirá una visión detallada de qué ofrecerá exactamente este proyecto tras su desarrollo.

Una parte importante para poder llevar a cabo este apartado de la memoria ha conllevado reunirse con parte del equipo de desarrollo de Pesedia (en el cual se encuentra el profesor tutor de este proyecto), que han servido para determinar los límites y contenido del *plugin* de encuestas y exámenes, desde temas relacionados con las opciones de configuración que ofrecerán las encuestas y los exámenes hasta los tipos de pregunta disponible.

3.1 Introducción a la especificación de requisitos

3.1.1 Propósito

Hacer posible la integración en un sitio Pesedia de un *plugin* que permita la creación, respuesta y visualización de resultados tanto de encuestas como de exámenes por parte del usuario siguiendo los propios patrones de diseño y vistas de Elgg, la plataforma original en la que está basada Pesedia.

3.1.2 Ámbito

Aparte de la funcionalidad básica mencionada en el propósito de la especificación de requisitos el producto también contemplará los siguientes aspectos:

- Internacionalización y localización: El producto estará disponible en inglés, y castellano. También presentará el formato de fecha adecuado correspondiente al idioma que esté en uso, sin necesidad de modificar el código del *plugin*.

- Despliegue: La aplicación (el *plugin*, en este caso) se encontrará disponible en la red social Pesedia bajo el nombre de "Encuestas y exámenes" y con el nombre en el sistema de encuestas_examenes.

- Seguridad: El *plugin* contará con todas las medidas de seguridad contempladas en Pesedia además de con un sistema de permisos configurable por el creador de la encuesta o examen.

- Integración: El *plugin* podrá integrarse en cualquier sitio Pesedia siguiendo todos los pasos y requisitos que cualquier *plugin* creado para la plataforma Elgg.

- Escalabilidad: La escalabilidad del *plugin* de encuestas y exámenes recaerá en la escalabilidad del sitio Pesedia en el que esté instalado. Esto dependerá del servidor en el que esté alojado el sitio, y a pesar de que queda fuera del alcance de este proyecto, la comunidad de Elgg afirma que actualmente Elgg es capaz de manejar 50.000 usuarios registrados. En cualquier caso, de haber limitaciones de *hardware* y una mejora del mismo, el código del *plugin* no se verá afectado.

3.1.3 Definiciones, acrónimos y abreviaturas:

- **CMS**: Content Management System (o en español, Sistema Gestor de Contenido), sistema que permite a los usuarios crear contenido y administrarlo y que por norma general cuenta con una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio web.

- ***Core***: Código principal del programa. En el caso de este proyecto, al hablar del *core* de Elgg se hará referencia a su código base, el cual ha de mantenerse inmutable durante el desarrollo del *plugin*.

- **CSV**: Comma-separated values. El formato .csv es un tipo de documento que contiene datos en forma de tabla, donde las columnas están separadas por comas. Los archivos .csv también pueden contener una primera línea usada como cabecera, la cual contiene los nombres de las columnas.

- ***Framework***: En castellano marco o entorno de trabajo. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio (Riehle, 2000).

- **GMT**: *Greenwich mean time* (en español, el tiempo medio de Greenwich), es el estándar de tiempo utilizado (junto con UTC) actualmente.

- **MVC**: Modelo vista controlador. Propuesta de diseño de software donde se establece una separación en tres capas: datos, lógica de negocio e interfaz, cada una de ellas con una funcionalidad diferente.

- ***Plugin***: Complemento encargado de añadir funcionalidades extra al *software* original.

- **SGBD**: Sistema de gestión de bases de datos. Software que permite la manipulación de la información de una base de datos: añadir información, modificarla o eliminarla.

- **URL**: *Uniform Resource Locator*. Referencia a un recurso web.

- **UTF-8:** *8-bit Unicode Transformation Format*. Formato de codificación de caracteres. Actualmente es una de las tres posibilidades de codificación reconocidas por Unicode y lenguajes web, o cuatro en ISO 10646.

- **WYSIWYG**: *What You See Is What You Get*. Aplicable a los procesadores de texto que permiten visualizar el que será el resultado final del documento mientras se está escribiendo.

- **W3C**: *World Wide Web Consortium*. Consorcio internacional que se encarga de desarrollar y promover los estándares para la *World Wide Web*.

- **XML***: Extensible Markup Language*. Meta-lenguaje que permite definir etiquetas personalizadas para descripción y organización de datos.

3.2 Descripción general

En este apartado se pretende describir el producto de forma abstracta y genérica, sin entrar en detalle en lo referente a requisitos o especificaciones técnicas.

3.2.1 Perspectiva del producto

Se pretende que se pueda hacer uso del producto independientemente de la máquina en la que se utilice en lo referido a sistema operativo o tipo de dispositivo (ordenador, tableta, teléfono móvil, etc.). El producto podrá utilizarse siempre y cuando dicha máquina disponga de navegador web que cumpla los requisitos para acceder a Elgg y a sus correspondientes *plugins* instalados.

3.2.2 Funciones del producto

El producto deberá permitir:

- La activación o desactivación del *plugin* en cualquier momento que lo desee el administrador del sitio Pesedia.

- Al usuario que cree la encuesta o examen:

  - Elegir fecha de activación y entrega de la encuesta o examen.
  - Introducir un tiempo límite para la respuesta de la encuesta o examen.
  - Configurar si desea que el resultado de la encuesta o examen se envíe automáticamente cuando haya expirado el tiempo límite de su realización.
  - Seleccionar quién puede responder la encuesta o examen.
  - Visualizar los resultados de la encuesta o examen en cualquier momento.
  - Permitir a aquellos que respondan un examen ver su nota una vez se haya entregado, ver su nota y las respuestas correctas, o no ver ninguna de las dos cosas.

        - Permitir un único envío de la encuesta o examen o permitir múltiples envíos (en cuyo caso prevalece la última respuesta).
        - Permitir o no el envío de respuestas vacías.
        - Añadir preguntas del tipo texto, texto largo, respuesta única y respuesta múltiple.
        - Seleccionar una nota mínima con la que se aprobará el examen o no.
        - Seleccionar la nota individual de cada pregunta (tanto en caso de acierto como en caso de fallo). Posteriormente estas notas serán escaladas a 10 por el sistema.
        - Eliminar y editar cuando lo desee dicha encuesta o examen.
        - Reiniciar los resultados del examen (eliminar todas las respuestas).
        - Descargar el formato .csv los resultados obtenidos de una encuesta o examen.

    - Al usuario que responda la encuesta/examen:
        - Visualizar los resultados que él mismo ha enviado.
        - Ver su nota (en caso de que sea un examen y el creador lo haya decidido así).
        - Visualizar los resultados correctos del examen (en caso de que sea un examen y el creador lo haya decidido así).

El producto, siguiendo la filosofía de Elgg, será una herramienta de código abierto.

3.2.3 Características de los usuarios

El producto está orientado a todo usuario que acceda a Pesedia.

Se ha de tener en cuenta que el acceso real a la encuesta o examen dependerá en todo momento de que el creador de esta haya dado permisos o no a determinado tipo de usuario. En este caso, se presupone que todos los tipos de usuario tienen permisos para responder la encuesta o examen.

USUARIO AUTENTICADO (REGISTRADO EN PESEDIA)	
Acciones permitidas	Crear encuesta o examen.
	Editar su encuesta o examen.
	Eliminar su encuesta o examen.
	Reiniciar su encuesta o examen.
	Ver lista de encuestas y exámenes publicados.
	Acceder a cualquier encuesta o examen.
	Enviar el resultado de la encuesta o examen con sus propias respuestas.
	Ver los resultados obtenidos de la encuesta o examen (en caso de que sea su creador).
	Ver su nota de un examen (en caso de que el creador lo haya decidido así en la configuración).
	Ver las respuestas correctas de un examen (en caso de que el creador lo haya decidido así en la configuración).
	Descargar en formato .csv los resultados de su encuesta o examen.
	Elegir el idioma del *plugin*.

Tabla 1: Características de los usuarios autenticados

USUARIO ADMINISTRADOR	
Acciones permitidas	Activar o desactivar el *plugin*
	Crear encuesta o examen.
	Editar su encuesta o examen.
	Eliminar su encuesta o examen.
	Reiniciar su encuesta o examen.
	Ver lista de encuestas y exámenes publicados.
	Acceder a cualquier encuesta o examen.
	Enviar el resultado de la encuesta o examen con sus propias respuestas.
	Ver los resultados de la encuesta o examen (en caso de que sea su creador).
	Ver su nota de un examen (en caso de que el creador lo haya decidido así en la configuración).
	Ver las respuestas correctas de un examen (en caso de que el creador lo haya decidido así en la configuración).
	Descargar en formato .csv los resultados de su encuesta o examen.
	Elegir el idioma del *plugin*

Tabla 2: Características del usuario administrador

3.2.4 Restricciones generales

- El producto únicamente funcionará si es integrado en una plataforma Elgg.

- El producto ha sido programado usando principalmente PHP, JavaScript, CSS, HTML y mySQL, igual que la plataforma original.

- El servidor que aloje el sitio Pesedia deberá disponer de espacio de almacenamiento suficiente para guardar los datos correspondientes a las encuestas y exámenes.

3.2.5 Suposiciones y dependencias

Para poder hacer uso del producto se espera que el usuario disponga de un dispositivo con un navegador que cumpla con los estándares recomendados por el World Wide Web Consorcium (W3C), que es el caso de la mayoría de navegadores de uso común.

3.3 Requisitos específicos

Se procede a describir en este apartado el producto de forma detallada, abordando ahora sí aspectos técnicos.

3.3.1 Requisitos de interfaz externos

La interfaz de la aplicación deberá cumplir los siguientes requisitos:

- **IR-1.** La interfaz contará con una serie de vistas básicas que entran dentro de los requisitos de Elgg a la hora de crear un *plugin*. El desarrollo de este sistema de vistas se ha seguido al pie de la letra teniendo como base la documentación de la propia plataforma Elgg.

- **IR-2.** La interfaz ha de redimensionarse correctamente dependiendo de la resolución de la pantalla desde la que se está visualizando; es decir, debe ser adaptativa.

- **IR-3.** Debe estar disponible en castellano e inglés y además debe estar preparada para ser traducida a cualquier otro idioma.

- **IR-4.** No debe depender de eventos de ratón ni otro tipo de eventos que no sean multiplataforma.

-**IR-5.** El sistema siempre deberá mantener informados a los usuarios acerca de qué está ocurriendo a través de mensajes de confirmación o error (Nielsen, 1995).

- **IR-6**. El sistema debería “hablar el lenguaje de los usuarios”; es decir, se hará uso de palabras, frases y conceptos que sean familiares al usuario, en lugar de usar términos relacionados con el sistema (Nielsen, 1995).

-IR-7. Reconocimiento antes que recuerdo: se deben hacer visibles los objetos, acciones y opciones. Las instrucciones para el uso del sistema deben estar a la vista o ser fácilmente recuperables cuando sea necesario (Nielsen, 1995).

3.3.2 Requisitos funcionales

El sistema deberá permitir al usuario llevar a cabo las siguientes acciones:

- **FR-1**. Acceder al *plugin* de encuestas y exámenes a través de una pestaña en el menú principal de Pesedia.

- **FR-2**. Ver las encuestas y exámenes. El usuario podrá ver sus propias encuestas y exámenes en una lista, así como también el resto de encuestas y exámenes que estén públicos para dicho usuario.

-FR-3. Añadir encuesta o examen. El usuario podrá crear una nueva encuesta o examen cuando desee, añadiendo tantas preguntas como guste.

- **FR-4**. Seleccionar fecha de activación y envío límite de la encuesta o examen que desea publicar.

- **FR-5**. Limitar el tiempo que la encuesta o examen permanecerá activa para ser respondida. En caso de que el tiempo expire, el creador podrá elegir en las opciones de configuración si los resultados respondidos hasta el momento se enviarán al servidor o no.

- **FR-6.** Configurar si se permitirá un único envío de la encuesta o examen o se permitirán múltiples envíos. En el último caso, el envío que prevalecerá será el último.

- **FR-7.** Marcar preguntas como obligatorias o no durante la creación de la encuesta o examen.

- **FR-8**. Escalar a 10 las puntuaciones obtenidas de los exámenes, así como los pesos de las notas de las preguntas que los forman.

- **FR-9**. Añadir una puntuación mínima del 0 al 10 para que el examen se considere aprobado (si el creador del examen lo desea).

-FR-10. Añadir la puntuación de cada pregunta del examen: positiva en caso de que sea correcta, negativa en caso de que no lo sea. El *plugin* también permitirá no restar puntuación alguna al responder una pregunta de manera errónea si es lo que el creador del examen desea. En caso de la suma de las notas de todas las preguntas del examen no sea 10, se escalará a esta cifra.

- **FR-11**. Para los tipos de pregunta “pregunta de respuesta múltiple” y “pregunta de respuesta única” el creador de la encuesta o examen podrá añadir tantas respuestas como desee.

- **FR -12**. El creador del examen podrá decidir si se permitirá o no el envío de exámenes vacíos (sin ninguna respuesta).

- **FR-13**. Editar encuesta o examen. Una vez el usuario publique una encuesta o examen podrá editarse; no obstante, en caso de editar las opciones de respuesta de las preguntas los resultados obtenidos hasta el momento se eliminarán.

- **FR-14**. El usuario creador de una encuesta o examen podrá eliminarla cuando desee, haya finalizado o no la fecha límite de esta.

- **FR-15**. El usuario creador de una encuesta o examen podrá reiniciarla cuando desee, borrando todas respuestas obtenidas hasta el momento.

- **FR-16**. Ver resultados. El usuario creador de la encuesta o examen podrá ver los resultados de su encuesta o examen cuando desee, haya finalizado o no.

- **FR-17**. El usuario creador de un examen podrá configurar si desea que se muestre la nota obtenida en el examen a aquel que lo realice, así como si desea que aparte de esta nota se muestre también la corrección del examen.

- **FR-18**. El creador de la encuesta o examen podrá seleccionar a quién va dirigida la encuesta o examen.

- **FR-19.** El creador de la encuesta o examen podrá también descargar los resultados en formato .csv.

3.3.3 Requisitos de la base de datos

Al instalar Elgg se han identificado los siguientes requisitos referentes a la base de datos:

-LR-1: La base de datos debe ser mySQL y se podrá gestionar de manera fácil y rápida a través de la instalación de la interfaz gráfica PHPmyadmin.

-LR-2: Trabajar con entidades. A pesar de que Elgg permite el manejo directo de consultas a la base de datos a través de código y *scripts* personalizados recomienda encarecidamente trabajar con entidades (que es lo que se ha hecho durante todo el proyecto). Elgg cuenta con sus propios objetos, *ElggObject*, y estos son referenciados utilizando funciones específicas para ello. Seguir la estructura de datos de Elgg ha sido una parte fundamental y compleja en este proyecto, en la cual se hace hincapié en el apartado de Diseño.

3.4 Restricciones de diseño

Para poder desarrollar el *plugin* ha sido necesario disponer de una instalación de Elgg 2.3 (la versión más reciente y, por lo tanto, más estable por el momento). Siguiendo las recomendaciones de instalación de dicha versión, el proyecto se ha realizado por completo en el sistema operativo Ubuntu 16.04, en el cual se ha hecho la instalación de un servidor Apache2, una base de datos mySQL y la versión 5.6 de PHP.

Al igual que la plataforma original, el *plugin* ha sido desarrollado principalmente en PHP con ciertas partes en Javascript, HTML y CSS.

El *plugin* no funcionará para una versión anterior a Elgg 2.3.

3.5 Atributos del sistema

3.5.1 Fiabilidad

La probabilidad de que el *plugin* presente el comportamiento esperado por este será de 1 siempre y cuando Pesedia se encuentre en funcionamiento y el *plugin* de Encuestas y Exámenes se encuentre activado. En caso de desactivarlo mientras está en uso, podría haber pérdidas graves de datos que no han sido guardados en la base de datos.

3.5.2 Disponibilidad

El *plugin*, siempre que esté activo, deberá estar disponible el 100% del tiempo. En caso de que deje de haber conexión a internet este dejará de estar disponible, y el servidor en el que esté alojado el sitio de Pesedia devolverá el estado HTTP que corresponda.

3.5.3 Seguridad

Los sitios Pesedia funcionan bajo dominios que hacen uso del protocolo HTTPS, por lo que todo el tráfico de información se encuentra cifrado mediante SSL/TLS.

Actualmente el libre registro de usuarios no está permitido.

En lo referente a usuarios registrados, estos accederán a la plataforma con su usuario y contraseñas personales.

4. Análisis

El objetivo de la fase de análisis es definir los requisitos del *plugin*, sin entrar en detalle en cómo se vayan a satisfacer, lo cual se tratará en las fases de Diseño e Implementación.

4.1 Diagrama de clases

Elgg es orientado a objetos, y por lo tanto, a lo largo de este proyecto se va a trabajar con clases y objetos o instancias de estas. Concretamente estas clases son en Elgg denominadas entidades, y se habla de ellas en profundidad (junto con el resto del modelo de datos) en la fase de Diseño.

Antes de presentar el diagrama en sí, es conveniente realizar una breve descripción de las clases o entidades de las que se hace uso en el *plugin* de encuestas y exámenes. Entre paréntesis podemos ver el nombre de este objeto en el sistema.

- **Usuario (*user*):** Una parte fundamental de Pesedia son los usuarios que la forman. Este tipo de objetos no son generados por el *plugin* de encuestas y exámenes, sino que son un objeto originario de Elgg; no obstante, sí que interfieren en el *plugin*. El usuario es la representación en la red social de la persona que hay tras la máquina, por lo que podrá crear encuestas y exámenes a voluntad y, por lo tanto, configurarlas a su gusto.

- **Encuesta (*encuesta*):** Una encuesta es un objeto que contiene una serie preguntas con sus respectivas respuestas publicadas por un usuario. El resto de usuarios que tengan permisos para acceder a dicha encuesta podrán responder a las preguntas con la respuesta que deseen, y posteriormente el autor podrá ver las estadísticas obtenidas de su encuesta. Las encuestas no tienen puntuación, ni tampoco existen respuestas correctas para sus preguntas.

- **Examen (*examen*):** Muy similar a una encuesta, el objeto examen sí que dispone de una puntuación para sus preguntas en función de si las respuestas enviadas por los usuarios son o no las correctas. Las respuestas correctas las establece el creador del examen durante su elaboración.

- **Resultados de la encuesta/ Resultados del examen (*resultadosEncuesta/resultadosExamen*):** Los resultados tanto de encuestas como de exámenes se van almacenando en un objeto del tipo *resultadosEncuesta* o *resultadosExamen*, respectivamente.

- **Voto de la encuesta/ Respuesta del examen (*voto/respuesta*):** Cada vez que un usuario envía una respuesta de una encuesta o examen se crea un objeto en el

sistema del tipo *voto* para las encuestas y *respuesta* para los exámenes. Es el conjunto de estas respuestas que envían los usuarios las que conforman los resultados de la encuesta o examen. Este objeto también permite a cada usuario que ha enviado su encuesta o examen ver posteriormente su resultado.

- **Pregunta:** Tanto encuestas como exámenes están formados por preguntas (aparte de las opciones de configuración). El creador puede añadir tantas preguntas como desee. Finalmente se decidió añadir las preguntas como un mapa ordenado dentro de los objetos encuesta y examen; no obstante, puesto que cuenta con atributos propios, se ha decidido incluirla también en el diagrama de clases.

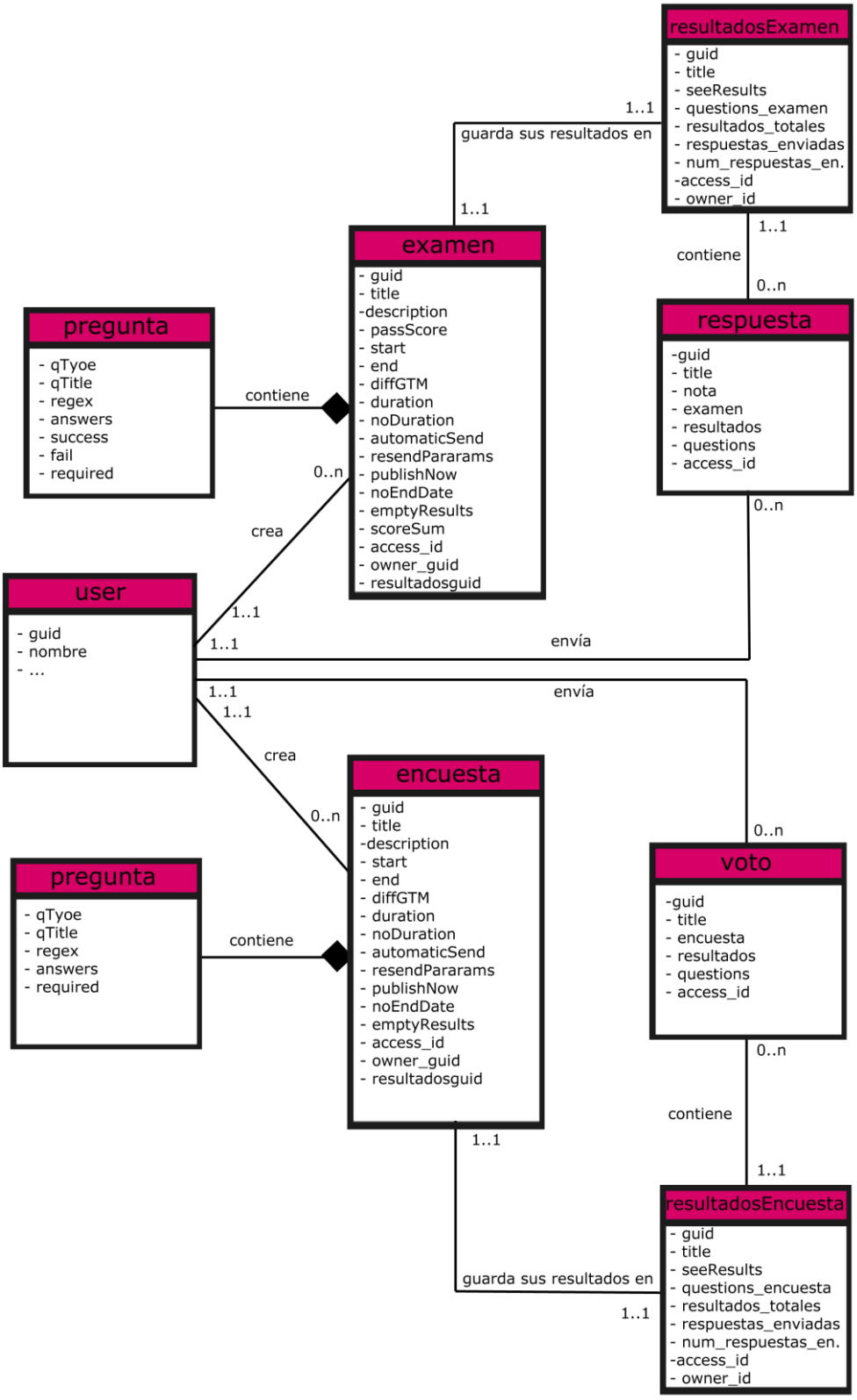

Figura 9: Diagrama de clases

4.2 Diagrama de casos de uso

Se incluyen en el diagrama de casos de uso las interacciones del usuario con el sistema representadas gráficamente. En este caso, al igual que hasta ahora, suponemos que el usuario que responde la encuesta o examen cuenta con permiso por parte del creador para realizar las acciones que se muestran.

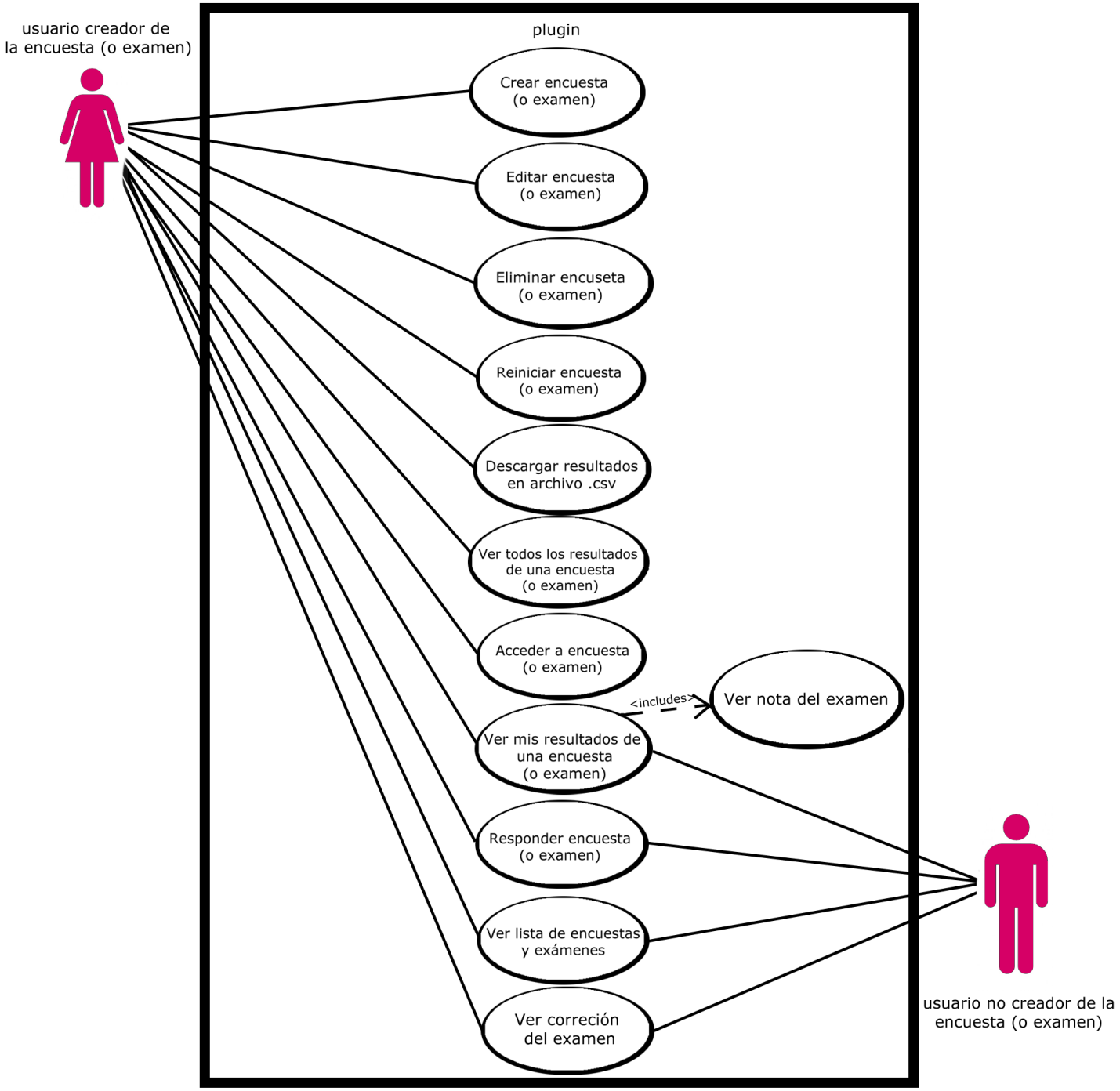

Figura 10: Diagrama de casos de uso

4.3 Casos de uso

Caso de uso	**Ver lista de encuestas y exámenes**
Actores	Usuario autenticado.
Descripción	Cualquier usuario de Pesedia podrá ver la lista de encuestas y exámenes disponibles para ser respondidos.
Secuencia principal	- El usuario accede al *plugin* a través de la opción disponible en el menú principal de Pesedia. - El usuario podrá ver la lista de encuestas si está situado en la pestaña “Encuestas”. - El usuario podrá ver la lista de exámenes si está situado en la pestaña “Exámenes”.
Flujos alternativos	- Si el usuario hace clic en el autor de cualquier encuesta o examen, podrá listar únicamente las encuestas o exámenes publicados por ese autor.
Requisitos	Ninguno.

Tabla 3: Caso de uso #1: "Ver lista de encuestas y exámenes"

Caso de uso	**Crear encuesta (o examen)**
Actores	Usuario autenticado.
Descripción	Cualquier usuario de Pesedia puede crear una encuesta o un examen accediendo al *plugin*.
Secuencia principal	- Desde la interfaz principal del *plugin* el usuario podrá hacer clic sobre el botón “Crear Encuesta” o “Crear Examen”. - El usuario deberá asignar un título y una descripción (opcional) al examen. - El usuario deberá elegir entre todas las opciones de configuración disponibles aquellas que desea para su encuesta o examen. - El usuario podrá añadir tantas preguntas (con sus respectivas respuestas en caso de que sean preguntas de respuesta única o preguntas de respuesta múltiple) como desee. En el caso de los exámenes, estas

	preguntas deberán tener una puntuación y una serie de respuestas correctas para poder calcular la nota del examen. - El examen se considerará creado cuando el usuario haca clic sobre el botón "Crear examen".
Flujos alternativos	- En caso de que haya algún campo obligatorio que el usuario no haya rellenado, el examen no podrá ser guardado en la base de datos. Los campos necesarios de ser rellenados se señalarán al usuario mediante un borde de color rojo. - En caso de que haya algún dato incorrecto, como el caso de un formato erróneo de la fecha o una puntuación negativa, tampoco se permitirá al usuario enviar el examen al servidor.
Requisitos	Ninguno.

Tabla 4: Caso de uso #2: "Crear encuesta (o examen)"

Caso de uso	**Acceder a una encuesta (o examen)**
Actores	Usuario autenticado.
Descripción	El usuario podrá acceder a cualquier examen y ver las opciones disponibles para interactuar con el contenido.
Secuencia principal	- Se podrá acceder a una encuesta o examen inmediatamente después de su creación (el creador del mismo), o a través de la página principal del *plugin*, haciendo clic sobre su título.
Flujos alternativos	Ninguno.
Requisitos	Que el creador del examen haya dado permisos de acceso al usuario que quiera acceder a la encuesta o examen.

Tabla 5: Caso de uso #3: Acceder a una encuesta (o examen)

Caso de uso	**Responder encuesta (o examen)**
Actores	Usuario autenticado.
Descripción	Una vez se ha creado una encuesta o examen cualquiera que tenga acceso a él puede enviar una repuesta.
Secuencia principal	- Acceder a la encuesta o examen haciendo clic en "Acceder a la encuesta" o "Acceder al examen". - Responder las preguntas de la encuesta o examen. - Enviar la encuesta o examen haciendo clic en "Enviar".
Flujos alternativos	- En caso de que haya alguna pregunta obligatoria que no se haya respondido se mostrará al usuario un mensaje avisándole de ello, y no se le permitirá enviar la respuesta. - En caso de que se intente enviar una encuesta o examen vacío y el creador haya especificado en las opciones de configuración que no se permiten envíos vacíos, se mostrará al usuario un mensaje avisándole de ello y no se le permitirá enviar una respuesta. - En caso de que ocurran alguna de las dos situaciones anteriores y el límite de tiempo de respuesta de la encuesta o examen haya llegado a su fin, no se podrá enviar una respuesta al servidor aunque haya sido configurado el envío automático de respuesta, y el *plugin* redireccionará al usuario a la página principal.
Requisitos	Que el creador del examen haya dado permisos de acceso al usuario que quiera acceder a la encuesta o examen.

Tabla 6: Caso de uso #4: Responder encuesta (o examen)

Caso de uso	**Editar encuesta (o examen)**
Actores	Usuario autenticado creador de la propia encuesta o examen.
Descripción	Una encuesta o un examen pueden editarse cualquier momento: Cambiar las opciones de configuración, añadir o eliminar preguntas, etc.
Secuencia principal	- Acceder a la opción "Editar encuesta" o "Editar examen" de entre las opciones disponibles que ofrece la propia encuesta o examen. - Realizar los cambios necesarios que el creador de la encuesta o examen desee.

	- Guardar los cambios haciendo clic en “Guardar encuesta” o “Guardar examen”.
Flujos alternativos	- En caso de que haya algún campo obligatorio que el usuario no haya rellenado, el examen no podrá ser guardado en la base de datos. Los campos necesarios de ser rellenados se señalarán al usuario mediante un borde de color rojo. - En caso de que haya algún dato incorrecto, como el caso de un formato erróneo de la fecha o una puntuación negativa, tampoco se permitirá al usuario enviar el examen al servidor. - En caso de que haya habido alguna modificación en las preguntas del examen, los resultados de este se reiniciarán automáticamente.
Requisitos	Solo puede editar la encuesta o examen el usuario que lo haya creado.

Tabla 7: Caso de uso #5: Editar encuesta (o examen)

Caso de uso	**Eliminar encuesta (o examen)**
Actores	Usuario autenticado creador de la propia encuesta o examen
Descripción	La encuesta o examen y sus respuestas y resultados serán eliminados de la base de datos.
Secuencia principal	- Hacer clic en la opción “Eliminar encuesta” o “Eliminar examen” de entre las opciones disponibles que ofrece la propia encuesta o examen. - Confirmar que se desea eliminar la encuesta o examen. - Se eliminarán todos los objetos voto (o respuesta) asociados a la encuesta o examen y se eliminarán los objetos resultadosEncuesta o resultadosExamen, así como la propia encuesta o examen. - El *plugin* mostrará un mensaje de confirmación en caso de que todo haya ido correctamente.
Flujos alternativos	- En caso de que no se pueda eliminar la encuesta o examen por alguna razón, se mostrará un mensaje de error al usuario.
Requisitos	Solo puede editar la encuesta o examen el usuario que lo haya creado.

Tabla 8: Caso de uso #6: Eliminar encuesta (o examen)

Caso de uso	**Reiniciar encuesta (o examen)**
Actores	Usuario autenticado creador de la propia encuesta o examen.
Descripción	Se eliminarán todas las respuestas enviadas por los usuarios hasta el momento referentes a esa encuesta o examen.
Secuencia principal	- Hacer clic en la opción “Reiniciar encuesta” o “Reiniciar examen” de entre las opciones disponibles que ofrece la propia encuesta o examen. - Se eliminarán todos los objetos voto (o respuesta) asociados a la encuesta o examen y se actualizarán los objetos resultadosEncuesta o resutladosExamen. - Confirmar que se desea reiniciar la encuesta o examen. - El *plugin* mostrará un mensaje de confirmación en caso de que todo haya ido correctamente.
Flujos alternativos	- En caso de que no se pueda reiniciar la encuesta o examen por alguna razón, se mostrará un mensaje de error al usuario.
Requisitos	Solo puede editar la encuesta o examen el usuario que lo haya creado.

Tabla 9: Caso de uso #7: Reiniciar encuesta (o examen)

Caso de uso	**Ver mi respuesta de una encuesta (o examen)**
Actores	Usuario autenticado
Descripción	Una vez un usuario envía una respuesta a una encuesta o examen, podrá ver su última respuesta enviada.
Secuencia principal	- Hacer clic en la opción “Ver mi respuesta” de entre las opciones disponibles que ofrece la propia encuesta o examen. - El *plugin* mostrará la respuesta que el usuario envío de dicha encuesta o examen.
Flujos alternativos	- En el caso de un examen, si el creador de este lo ha configurado así, a través de esta opción también se podrá ver la nota obtenida en el examen.
Requisitos	Haber respondido la encuesta o examen.

Tabla 10: Caso de uso #8: Ver mi respuesta de una encuesta (o examen)

Caso de uso	**Ver la corrección de un examen**
Actores	Usuario autenticado
Descripción	Una vez un usuario ha enviado una respuesta a un examen, si el creador lo ha configurado así, podrá acceder a la corrección de este y ver cuáles eran las respuestas correctas.
Secuencia principal	- Hacer clic en la opción "Ver corrección" de entre las opciones disponibles que ofrece el propio examen. - El *plugin* mostrará las respuestas del usuario así como cuáles eran las respuestas correctas del examen. - También aparecerá la nota obtenida.
Flujos alternativos	Ninguno.
Requisitos	Haber respondido el examen y que el creador lo haya configurado de forma que se permita ver la corrección. Esta opción no está disponible para encuestas.

Tabla 11: Caso de uso #9: Ver la corrección de un examen

Caso de uso	**Ver resultados totales de una encuesta (o examen)**
Actores	Usuario autenticado creador de la propia encuesta o examen.
Descripción	Visualización de todas las preguntas del examen y las respuestas obtenidas hasta el momento. Las respuestas a las preguntas de texto se muestran en un listado, mientras que las respuestas de las preguntas de respuesta múltiple y respuesta única se muestran mediante barras.
Secuencia principal	- Hacer clic en la opción "Ver resultados" de entre las opciones disponibles que ofrece la propia encuesta o examen. - El *plugin* mostrará las respuestas de todos los usuarios juntas.
Flujos alternativos	Ninguno.
Requisitos	Ser el creador de la encuesta o examen.

Tabla 12: Caso de uso #10: Ver resultados totales de una encuesta (o examen)

Caso de uso	**Descargar los resultados de una encuesta (o examen) en formato CSV**
Actores	Usuario autenticado creador de la propia encuesta o examen
Descripción	Se descargará a través del navegador un archivo .csv con todos los resultados de la encuesta o examen, junto con el nombre del usuario que ha emitido la respuesta.
Secuencia principal	- Hacer clic en el botón "Descargar resultados en archivo .csv" que aparece al ver todos los resultados de la encuesta o examen. - El *plugin* generará un archivo con todos los resultados y este será descargado automáticamente a través del navegador.
Flujos alternativos	Ninguno.
Requisitos	Ser el creador de la encuesta o examen.

Tabla 13: Caso de uso #11: Descargar los resultados de una encuesta (o examen) en formato CSV

5. Diseño del *plugin*

Pesedia es una red social que tiene como principal objetivo ayudar a los usuarios a mejorar el alcance de su información. Para tal fin, fueron desarrolladas medidas específicas de privacidad y control utilizando Elgg como plataforma principal, y algunos de sus *plugins* disponibles para poder extender su funcionalidad básica.

A la hora de decidir añadir la funcionalidad de incluir encuestas y exámenes que pudiesen publicar y responder los usuarios de Pesedia, se llegó a la conclusión de que un *plugin* sería la mejor forma de hacerlo, ya que este tipo de extensiones sirven precisamente para eso: añadir funcionalidades nuevas a la plataforma original.

Existen diferentes sitios Pesedia, y cada uno de ellos puede ofrecer servicios diferentes dependiendo de las necesidades del cliente; así pues, el *plugin* objetivo de este proyecto podrá ser clonado e instalado en tantos sitios Pesedia como se desee. Para su desarrollo, se ha trabajado con un sitio Pesedia instalado en un servidor local con sistema operativo Ubuntu. Esto no supone una diferencia en el comportamiento de Elgg, únicamente ha requerido el trabajo extra de instalar, aparte de la plataforma, el servidor Apache en la máquina de trabajo.

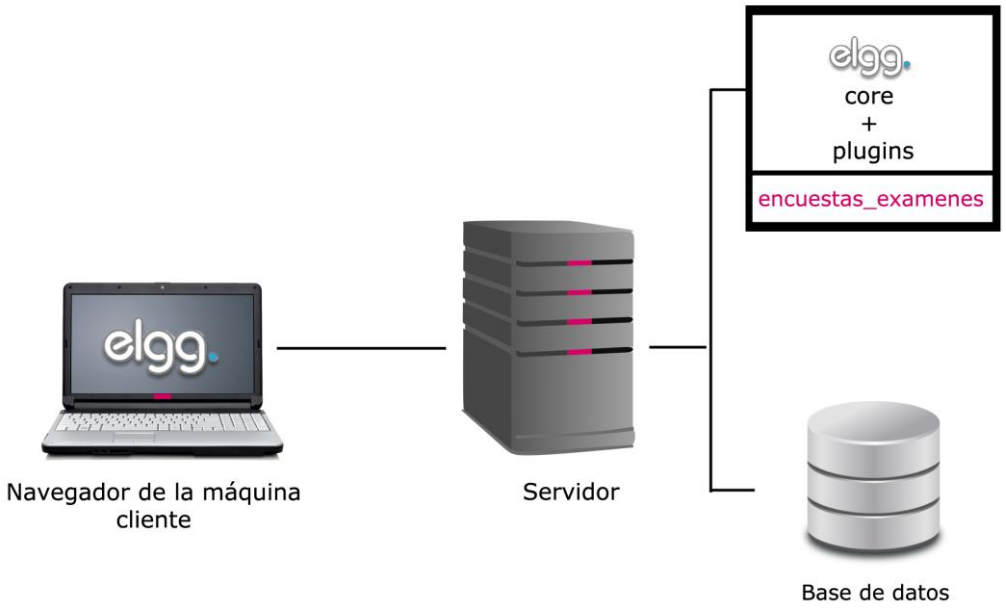

Figura 11: Esquema del entorno de trabajo

Este esquema, no obstante, es muy básico si tenemos en cuenta la envergadura de Elgg.

A rasgos generales podríamos decir que Elgg está compuesto por un núcleo de código (*core*) más sus extensiones (*plugins*). Estas extensiones o *plguins* no tienen por qué ser instalados aparte al *core* de Elgg, sino que muchos ya vienen instalados por defecto. Posteriormente, el administrador de Elgg puede desactivar estos *plugins* o instalar nuevos.

A la hora de diseñar el *plugin* realizado en este proyecto ha sido necesario en primer lugar llevar a cabo una exhaustiva investigación acerca de Elgg, su funcionamiento, estructura y el aspecto más relevante a la hora de llevar a cabo el proyecto: su modelo de datos. El modelo de datos de Elgg no es en absoluto trivial, y merece ser tratado en detalle, ya que su comprensión ha sido fundamental a la hora del diseño y la implementación del *plguin*. Antes de tratar el modelo de datos de Elgg y para poder comprender el porqué de este en su totalidad, ha sido muy importante contextualizarlo, y es que Elgg sigue el patrón de diseño modelo-vista-controlador (MVC). Los *plugins* que se añadan a Elgg también deben seguir este patrón, y el nuestro no será menos.

5.1 El patrón de diseño MVC en Elgg

El MVC propone separar el código de los programas por sus diferentes responsabilidades. Este patrón de diseño surgió hace varias décadas, antes incluso de la aparición de la Web. Ha sido no obstante durante los últimos años cuando ha ganado mucha fuerza por ser utilizado en muchos *frameworks* de desarrollo web como modelo para la arquitectura de aplicaciones web, tal es el caso de Elgg.

A la hora de comenzar a comprender cómo funciona la arquitectura de un *plugin* de Elgg ha sido crucial entender en primer lugar cómo la plataforma trabaja con este MVC. Se va a proceder por lo tanto a detallar qué son exactamente estos modelos, vistas y controladores, y qué implicación van a tener en el *plugin* que se va a desarrollar.

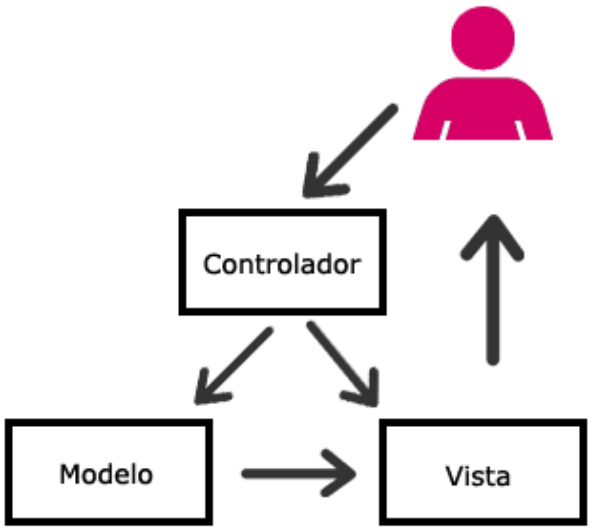

Figura 12: Modelo-vista-controlador

5.1.1 Modelos

De entre las tres capas que forman el MVC, esta capa es la que trabaja con los datos, y más concretamente, con el acceso y actualización de la información. Los datos en este caso se encuentran en una base de datos mySQL. Aunque normalmente estos accesos a las bases de datos se realizan a través de consultas directas *(selects, updates, inserts*, etc) Elgg permite trabajar con cierta abstracción a través de sus entidades.

5.1.1.1 Modelo de datos de Elgg

El modelo de datos de Elgg se basa en una estructura de datos unificada que consiste en entidades, así como relaciones de estas entidades y metadatos específicos de cada una de ellas. Una entidad no es otra cosa que un objeto; es decir, una unidad dentro del programa que consta de un estado y un comportamiento.

Elgg recomienda hacer uso de estas entidades a la hora de desarrollar un *plugin*, con el fin de no manipular directamente la base de datos, creando así sistemas más estables y con una notable mejora en la experiencia del usuario. Esto se debe a que es más sencilla la manipulación de entidades que la manipulación de accesos a la base de datos de Elgg, y además, permite crear código reutilizable, pues el contenido creado por *plugins* diferentes puede ser reutilizado de manera consistente.

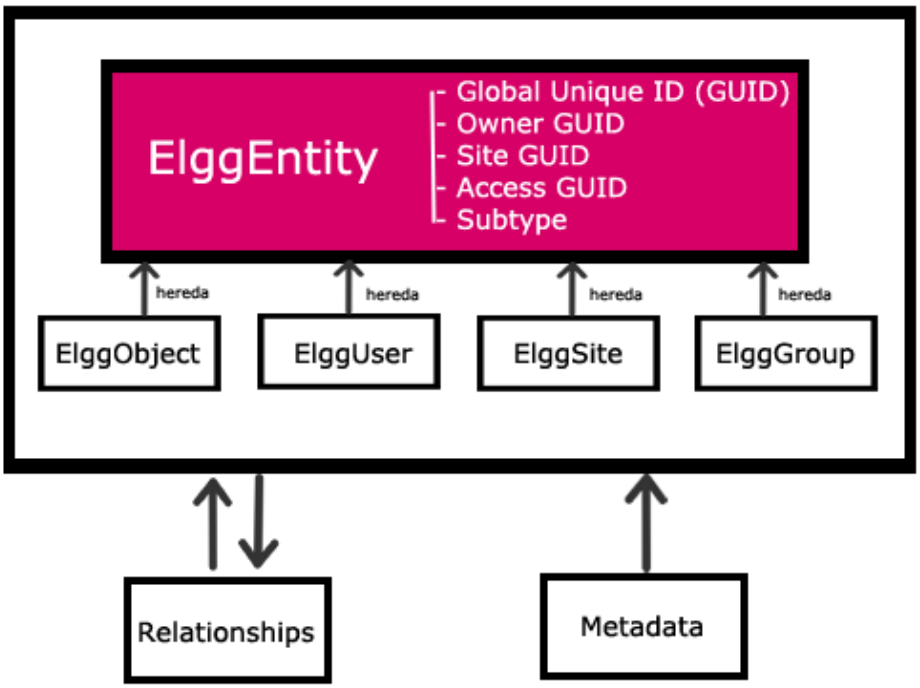

Figura 13: Modelo de datos de Elgg

Entidades

Como se ha mencionado anteriormente, una entidad es un objeto. Toda entidad en el sistema hereda la clase *ElggEntity*, que contiene las siguientes variables o propiedades:

- Identificador numérico único (*guid*). Cabe destacar del *guid* que ha sido fundamental a lo largo del desarrollo del *plguin* para poder manipular las entidades.
- Permisos de acceso a la entidad. Un usuario nunca accederá a una entidad si no tiene permisos para ello.
- Un subtipo (muy importante para nuestro *plugin*, los subtipos se detallan a continuación).
- Un propietario.
- El sitio al que pertenece la entidad, haciendo referencia al sitio creado mediante la instalación de Elgg.
- Un contenedor utilizado para asociar el contenido con un usuario o grupo. Cada objeto dispone de un *container_id* con el *guid* del grupo o usuario que lo creó, de esta manera, se facilita en gran medida la búsqueda de la entidad en la base de datos.

Existen en Elgg cuatro tipos de entidades, cada una de ellas con propiedades y métodos característicos.

Tipo	Clase PHP	Representa
object	`ElggObject`	La mayoría del contenido creado por los usuarios: publicaciones, contenido subido a la red, y en el caso de nuestro *plugin*, encuestas y exámenes, resultados y respuestas de los mismos.
group	`ElggGroup`	Un grupo de usuarios organizado con su propia página de perfil.
user	`ElggUser`	Un usuario del sistema
site	`ElggSite`	El sitio creado a través de la instalación de Elgg

Tabla 14: Tipos de entidades en Elgg

Cada entidad puede también tener un **subtipo** personalizado. En este proyecto se decidieron crear seis subtipos diferentes de entidad (en realidad podría decirse que son tres; no obstante, cada uno de ellos tiene su análogo para encuestas y para exámenes) que se pueden ver en el diagrama de clases de la fase de Análisis, y que son las siguientes:

- *examen* (*ElggObject*): Como su nombre indica, cada objeto con este subtipo es un examen que contiene tanto las opciones de configuración del mismo como todas las preguntas que lo forman.

- *resultadosExamen* (*ElggObject*): Contiene toda la información referente a los resultados del examen: respuestas correctas a las preguntas, recuento de respuestas, lista de quién ha enviado cada respuesta y cuántas veces, etc

- *respuesta* (*ElggObject*): Constituye la respuesta de un usuario a un examen: cuándo se ha enviado, cuáles son las respuestas a las preguntas del examen, puntuación obtenida en el examen, etc.

- *encuesta* (*ElggObjetct*): Al igual que el examen, un objeto con el subtipo *encuesta* constituye la propia encuesta en sí, con sus preguntas y sus opciones de configuración.

- *resultadosEncuesta* (*ElggObject*): Este objeto almacena todos los resultados obtenidos en la encuesta, al igual que *resultadosExamen*.

- *voto* (*ElggObject*): De manera análoga al subtipo respuesta, un voto es un resultado de un usuario ante una encuesta.

Como se puede ver en la descripción, cada entidad cuenta con ciertos datos específicos, y es que en Elgg existe la posibilidad de extender una entidad utilizando para ello los metadatos (*metadata*), otro de los elementos relevantes en su modelo de datos.

Metadatos

Los metadatos en Elgg permiten añadir información adicional a una entidad más allá de los campos soportados por esta por defecto, que son, aparte de los mencionados en el punto anterior, el título (*title*) y la descripción (*description*).

Los metadatos son instancias de la clase *ElggMetadata*, y aunque también cuentan con identificador propio, diferente al identificador del objeto al que pertenecen, a lo largo de este proyecto se ha trabajado siempre con el objeto y sus metadatos en conjunto (tal y como la documentación de Elgg recomienda), no tratándolos como objetos propios.

Con el fin de profundizar en las entidades con las que se ha trabajado en este proyecto y en sus metadatos, se procede a mostrar las siguientes tablas donde se puede visualizar qué metadatos exactamente están asociados a cada entidad:

examen

Metadatos	Descripción
title	Título del examen (este metadato viene por defecto en Elgg)
description	Descripción del examen (este metadato viene por defecto en Elgg)
passScore	Nota mínima con la que se aprobará el examen de 0 a 10 (en caso de haberla)
start	Fecha de activación del examen (en caso de haberla)
end	Fecha límite de entrega del examen (en caso de haberla)
diffGTM	Número entero que representa la diferencia de la hora local del dispositivo con el que se ha publicado el examen con la hora en formato GTM+0. Este metadato sirve para mostrar la misma hora de activación y entrega sin importar la franja horaria del dispositivo desde donde se publique el examen.
duration	Tiempo que tiene el usuario para responder al examen (en caso de haberlo)
noDuration	Indicador de si habrá o no límite de tiempo para responder al examen.

automaticSend	Indicador de si el examen se enviará o no automáticamente al agotarse el tiempo límite para responderlo en caso de que el usuario no lo haya enviado en ese tiempo.
resendParams	Número de veces que se podrá enviar el examen: una, número elegido por el creador del examen o ilimitadas.
publishNow	Indicador de si el examen se publicará directamente tras su creación (es decir, no hay fecha de activación).
noEndDate	Indicador de que el examen se mantendrá siempre activo (es decir, no hay fecha de cierre).
emptyResults	Indicador de si se permitirá o no el envío de un examen vacío (sin ninguna pregunta respondida).
scoreSum	Sumatorio de las puntaciones de las preguntas del examen, que serán escaladas a 10.
questions	Lista de preguntas del examen. Cada pregunta es un mapa ordenado que contiene los siguientes datos: tipo de pregunta (*qType*), título de la pregunta (*qTittle*), si se admite como resultado una expresión regular (*regex*), lista de respuestas disponibles (*answers*), nota en caso de la pregunta se responda correctamente (*success*), descuento de nota en caso de que la respuesta a la pregunta sea incorrecta (*fail*), y si la pregunta es de respuesta obligatoria (*required*).
access_id	Quién puede responder al examen.
owner_guid	*guid* del creador del examen.
resultadosguid	*guid* del objeto *resultadosExamen* asociado a este examen.

Tabla 15: Metadatos de la entidad examen

resultadosExamen

Metadatos	Descripción
title	Título del examen asociado
seeResults	Parámetro que indica si el usuario podrá ver su nota tras enviar el examen, si podrá ver su nota y el examen corregido o si no podrá ver nada de esto.
questions_examen	Mapa ordenado de los resultados del examen. Cada pregunta del examen cuenta con una lista asociada que contiene las correspondientes respuestas a las preguntas (en caso de las preguntas de texto, esta respuesta es única). Cada respuesta cuenta con un diccionario que contiene los *guids* de los usuarios que la seleccionaron (*guidVotes*) y el número de "votos" que ha obtenido en total dicha respuesta (*value*).
resultados_totales	Total de respuestas enviadas a este examen.
respuestas_enviadas	Diccionario que tiene como clave las *guids* de los usuarios que han enviado una respuesta al examen y como valor el número de respuestas que han enviado.
num_respuestas_enviadas	Número de respuestas máximas permitidas en el examen en caso de no ser ni una ni ilimitadas.
access_id	Quién puede acceder a los resultados del examen (heredado del propio examen)
owner_id	Creador del examen

Tabla 16: Metadatos de la entidad resultadosExamen

respuesta

Metadatos	Descripción
title	Título del examen
nota	Nota de esta respuesta evaluada
examen	*guid* del examen al que pertenece la respuesta
resultados	*guid* del *resultadosExamen* asociado al examen
questions	Mapa ordenado de las respuestas a cada una de las preguntas del examen
access_id	Acceso a la *respuesta*
owner_id	Propietario de la *respuesta*

Tabla 17: Metadatos de la entidad respuesta

encuesta

Metadatos	Descripción
title	Título de la encuesta (este metadato viene por defecto en Elgg)
description	Descripción de la encuesta (este metadato viene por defecto en Elgg)
start	Fecha de activación de la encuesta (en caso de haberla)
end	Fecha límite de entrega de la encuesta (en caso de haberla)
diffGTM	Número entero que representa la diferencia de la hora local del dispositivo con el que se ha publicado la encuesta con la hora en formato GTM+0. Este metadato sirve para mostrar la misma hora de activación y entrega sin importar

	la franja horaria del dispositivo desde donde se publique la encuesta.
duration	Tiempo que tiene el usuario para responder la encuesta (en caso de haberlo)
noDuration	Indicador de si habrá o no límite de tiempo para responder la encuesta.
automaticSend	Indicador de si la encuesta se enviará o no automáticamente al agotarse el tiempo límite para responderla en caso de que el usuario no la haya enviado en ese tiempo.
resendParams	Número de veces que se podrá enviar la encuesta: una, número elegido por el creador de la encuesta o ilimitadas.
publishNow	Indicador de si la encuesta se publicará directamente tras su creación (es decir, no hay fecha de activación).
noEndDate	Indicador de que la encuesta se mantendrá siempre activa (es decir, no hay fecha de cierre).
emptyResults	Indicador de si se permitirá o no el envío de una encuesta vacía (sin ninguna pregunta respondida).
questions	Lista de preguntas de la encuesta. Cada pregunta es un mapa ordenado que contiene los siguientes datos: tipo de pregunta (*qType*), título de la pregunta (*qTittle*), lista de respuestas disponibles (*answers*), y si la pregunta es de respuesta obligatoria (*required*).
access_id	Quién puede responder a la encuesta.
owner_guid	*guid* del creador de la encuesta.
resultadosguid	*guid* del objeto *resultadosEncuesta* asociado a esta encuesta.

Tabla 18: Metadatos de la entidad encuesta

resultadosEncuesta

Metadatos	Descripción
title	Título de la encuesta asociada
questions_encuesta	Mapa ordenado de los resultados de la encuesta. Cada pregunta de la encuesta cuenta con una lista asociada que contiene las correspondientes respuestas a las preguntas (en caso de las preguntas de texto, esta respuesta es única). Cada respuesta cuenta con un diccionario que contiene los *guids* de los usuarios que la seleccionaron (*guidVotes*) y el número de "votos" que ha obtenido en total dicha respuesta (*value*).
resultados_totales	Total de respuestas enviadas a esta encuesta.
respuestas_enviadas	Diccionario que tiene como clave las *guids* de los usuarios que han enviado una respuesta a la encuesta y como valor el número de respuestas que han enviado.
num_respuestas_enviadas	Número de respuestas máximas permitidas en la encuesta en caso de no ser ni una ni ilimitadas.
access_id	Quién puede acceder a los resultados de la encuesta (heredado de la propia encuesta)
owner_id	Creador de la encuesta

Tabla 19: Metadatos de la entidad resultadosEncuesta

voto

Metadatos	Descripción
title	Título de la encuesta
encuesta	*guid* de la encuesta al que pertenece el voto
resultados	*guid* del *resultadosEncuesta* asociado a la encuesta
questions	Mapa ordenado de las respuestas a cada una de las preguntas de la encuesta
access_id	Acceso al *voto*
owner_id	Propietario del *voto*

Tabla 20: Metadatos de la entidad voto

Relaciones

Las relaciones permiten realizar asociaciones entre entidades con diferente tipo y subtipo. También han decidido utilizarse en el *plugin* de encuestas y exámenes, ya que permiten acceder a unas entidades a través de otras de manera eficiente.

En Elgg, una relación pertenece a la clase *ElggRelationship*, y presenta las siguientes características:

API name	Ejemplo	Representa
`guid_one`	A	Entidad que está siendo enlazada
`relationship`	algo	Cómo está siendo enlazada
`guid_two`	B	A qué entidad está siendo enlazada

Tabla 21: Relaciones entre entidades en Elgg

El comportamiento de las relaciones es el siguiente:

"**{A}** es un **{algo}** de **{B}**."

En este proyecto se ha establecido que habrá la siguiente relación, con el fin de facilitar la recuperación de datos:

{*respuesta* (o *voto*)} es una **{ parte }** de **{ *resultadosExamen* (o *resutladosEncuestas*)}**

5.1.2 Vistas

Las vistas, tal y como su nombre indica, contienen el código que permitirá la visualización de las interfaces de usuario. En el caso de Elgg, las vistas son creadas con código HTML y PHP. Elgg contiene muchas vistas por defecto en su código fuente, pero es la unión de estas y su manipulación lo que dará lugar a las vistas de las interfaces del *plugin*.

Elgg recomienda como buena práctica seguir una estructura concreta en lo referente al uso de algunas de sus vistas. Existen en Elgg cuatro "lienzos" de trabajo principales, que son:

- Una columna
- Dos columnas
- Contenido
- Widget (que se podría traducir como complemento en este contexto)

Estas vistas son accesibles a través de una función del siguiente tipo:

```
$canvas_area = elgg_view_layout($canvas_name, array(
        'content' => $content,
        'section' => $section
));
```

Cada vista tiene diferentes secciones que la conforman. Cada una de estas secciones (diferentes para cada vista) contiene el código HTML que será visualizado en la pantalla del dispositivo en el que se ejecute el *plugin*.

Si bien en Pesedia se hace uso de todas estas vistas, el *plugin* desarrollado a lo largo de este proyecto se visualizará en el cuerpo central de la página, por lo que en todo momento se hace uso del lienzo contenido (*content*).

A su vez, este lienzo ha de ser colocado sobre la vista "*page*" de Elgg, a través de una función como la siguiente:

```
echo elgg_view_page($title, $canvas_area);
```

En la siguiente imagen se pueden distinguir con claridad estos dos tipos de vistas tan utilizados en el *plugin*:

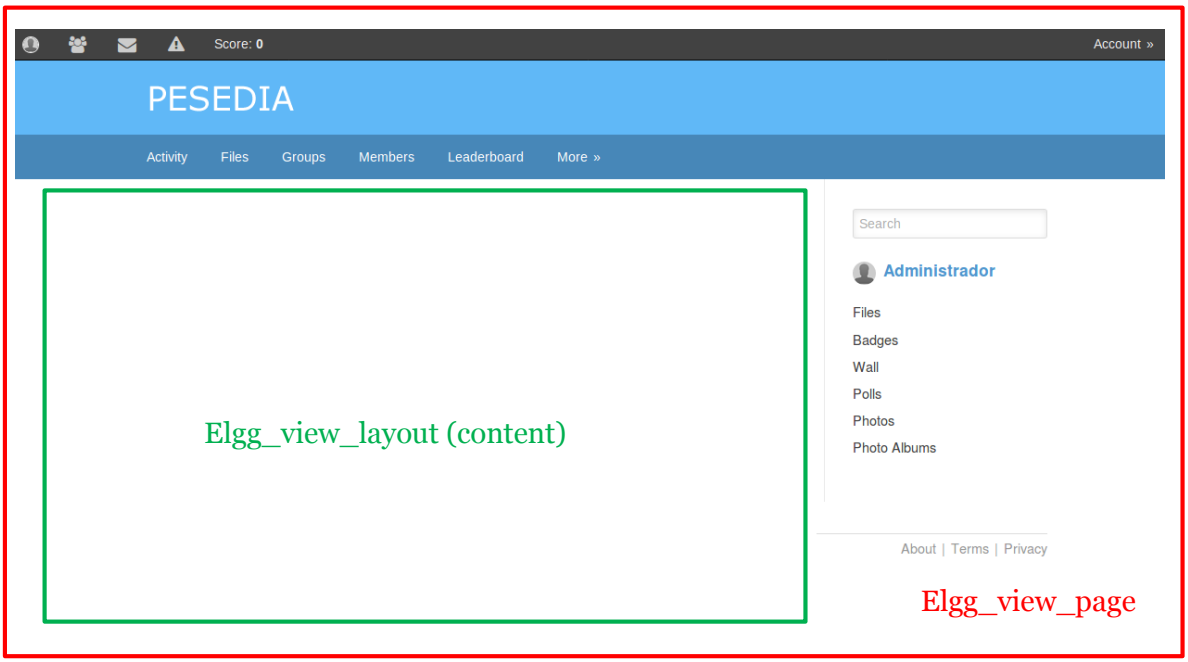

Figura 14: Vistas layout y page

A pesar de que estas vistas se usan siempre en el desarrollo del *plugin*, no son las únicas, se han ido incluyendo diferentes vistas dentro de *layout*, al igual que se ha podido ver que la vista *layout* está dentro de la vista *page*.

En el apartado de Implementación del *plugin* se detalla cómo se ha implementado el *plugin* utilizando esta potente herramienta que son las vistas, y cómo se han creado vistas específicas para cumplir con las necesidades del proyecto.

En la fase de diseño también se ha llevado a cabo un prototipo del *plugin*, que se expone en el apartado 5.2: Prototipo del *plugin*.

5.1.3 Controladores

Son el nexo de unión entre los modelos y las vistas. Los controladores recogen las acciones que se muestran en la vista y hacen llegar esta información a los modelos, que son los que manipulan realmente los datos.

Si bien las vistas son las encargadas de hacernos ver lo que queremos ver, los controladores son los que llevan a cabo las acciones que nosotros seleccionamos a través de las vistas. A lo largo de este proyecto se ha estado trabajando con los siguientes tipos de controladores de Elgg:

- *Page Handler* o "controlador de páginas": Elgg ofrece la posibilidad de facilitar el control de las páginas que componen un *plugin* a través de este tipo de controlador, que permite crear URL's personalizadas y configurar su redirección al archivo PHP que ejecutará las vistas adecuadas.
  A lo largo de la interacción con el *plguin* de encuestas y exámenes se accede a numerosas URL's diferentes, algunas de ellas contienen atributos específicos, pero no es el caso de todas ellas.
  Un controlador de este tipo debe ser registrado en el archivo *start.php* del *plugin* (se habla de este archivo en profundidad en el apartado Implementación), de la siguiente forma:

  ```
  elgg_register_page_handler('your_plugin', 'your_plugin_page_handler');
  ```

  '*your_plugin_handler'* no es ni más ni menos que la función que se ha debido implementar para redireccionar correctamente todas las URLs, la cual ha sido implementada supliendo las necesidades del *plugin* de encuestas exámenes, tal y como se puede ver en detalles en el apartado Implementación.

- *Action Handler* o "controlador de acciones": Son las acciones las que realmente provocan una interacción entre el *plugin* y el usuario, ya que son estas acciones las que alteran la base de datos, y por lo tanto son el desencadenante para crear, editar y eliminar contenido. Las acciones en Elgg han de ser registradas en el archivo *start.php* (al igual que ocurre con el controlador de páginas) de la siguiente forma:

  ```
  elgg_register_action("example", __DIR__ . "/actions/example.php");
  ```

En este caso la salida del formulario "*example"* será procesada por la acción "*example.php"*, donde estará la funcionalidad que altera la base de datos. De nuevo, la implicación de este controlador y cómo se ha trabajado con él en el *plguin* de encuestas y exámenes se detalla en la parte de Implementación.

- *Event Handler* o "controlador de eventos" y *Plugin Hooks*. Merece la pena hablar de estos dos tipos de controlador al mismo tiempo, ya que ambos son muy parecidos y, no obstante, tienen una diferencia muy importante, y es que si bien los eventos pueden ser cancelados, la ejecución de un *hook* no puede cancelarse. Estos tipos de controlador son los encargados de llevar a cabo todas las acciones y funcionalidades extras que podría realizar un *plugin* en Elgg. En el caso del *plugin* desarrollado a lo largo de este proyecto, se han implementado dos tipos de *hooks* diferentes:
    - El primero es relacionado con el manejo de URLs del *plugin*. Dependiendo del tipo de entidad del que se quiera mostrar una vista (*encuesta*, *examen*, *voto*, *respuesta*, *resultadosEncuesta* o *resultadosExamen*), este *hook* se encarga de crear la URL adecuada. Posteriormente, es el controlador de páginas el que, dependiendo de la URL, muestra la vista que corresponda.
    - Los siguientes *hooks* utilizados han sido para todos los archivos Javascript del proyecto, los cuales constituyen cada uno un *hook* diferente asociado al archivo PHP que se ejecute cada vez.

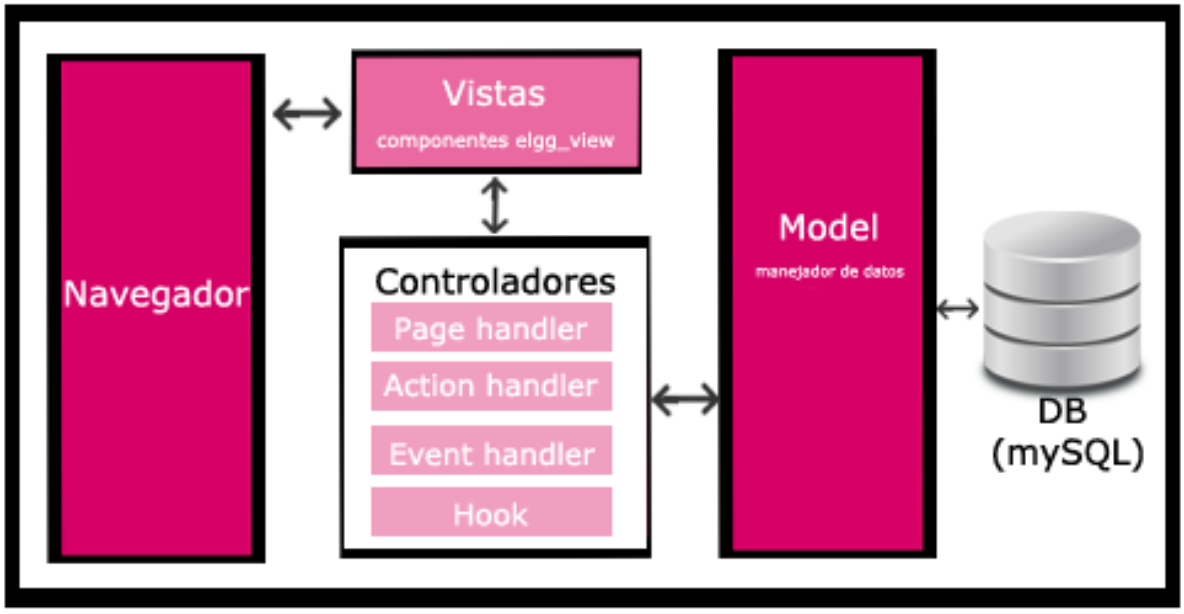

Figura 15: MVC en Elgg

5.2 Prototipo del *plugin* de encuestas y exámenes

Con el fin de poder comenzar a trabajar en una buena experiencia de usuario desde la fase del diseño del proyecto se ha preferido realizar un prototipo que ha ido evolucionando hasta ser lo suficientemente consistente como para comenzar con una primera implementación del *plugin*. La evaluación tanto del prototipo como del diseño se comenta en profundidad en el punto de pruebas realizadas.

El prototipo del *plugin* ha consistido en realizar un boceto de las vistas relevantes de este utilizando para ello la herramienta JustInMind. Se trata de un prototipo de baja fidelidad, ya que aunque las vistas pueden guardar mucha similitud con la apariencia real de las interfaces del *plugin*, la implementación tras ellas es nula.

Las vistas pues son las que se muestran a continuación. Como se puede ver, en todo momento se ha trabajado con un escenario muy similar al real, ya que el contenido del *plugin* se mostrará en la parte del cuerpo de la página, tal y como se mencionó en el apartado anterior.

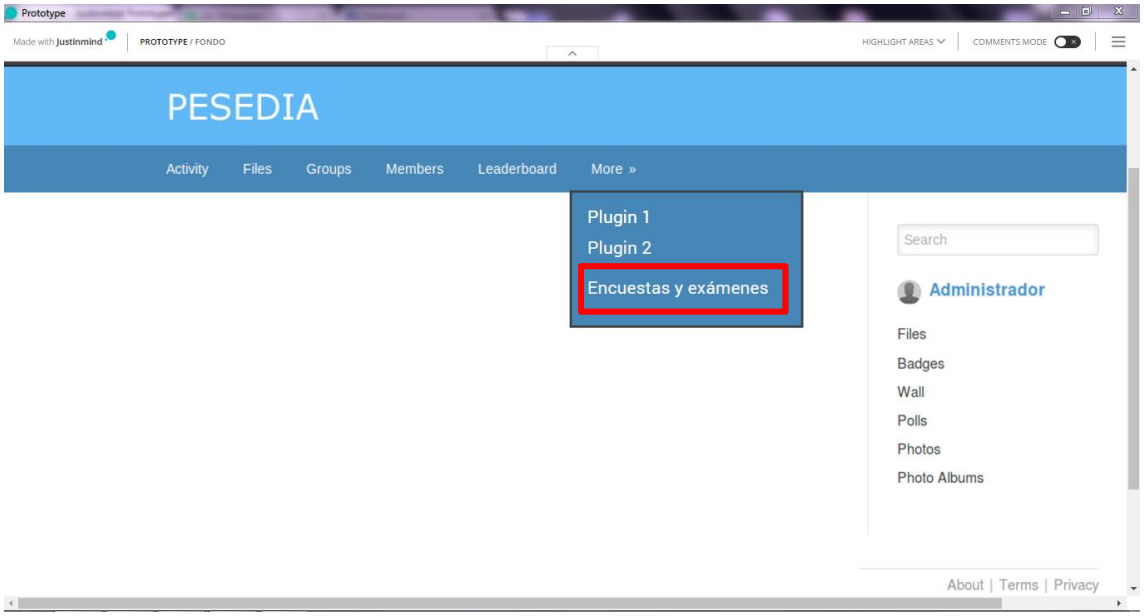

Figura 16: Prototipo #1: Acceder al plugin Encuestas y Exámenes

En la figura 16 se pretende mostrar la forma en la que se podrá acceder al *plugin* (una vez esté activado). Como se puede ver, aparece un enlace a este en el propio menú principal de Pesedia.

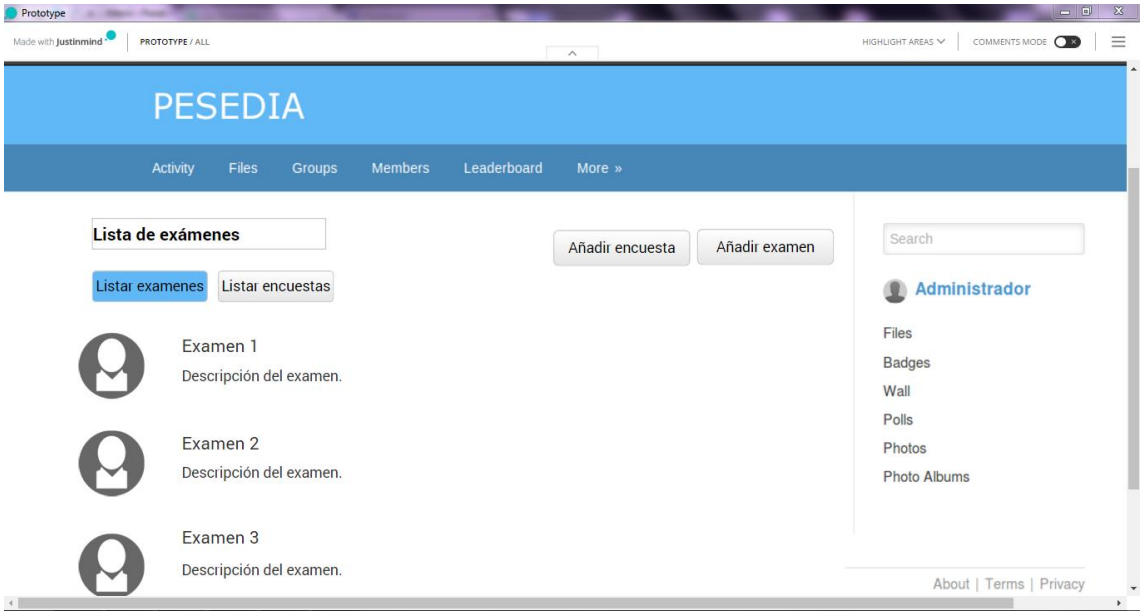

Figura 17: Prototipo #2: Página principal del plugin

Ahora sí podría decirse que "estamos dentro del *plugin*". Como se puede ver, esta vista mostrará un listado tanto de encuestas como de exámenes, así como la posibilidad de crear tanto lo uno como lo otro.

A continuación, supondremos que el usuario hace *clic* en "Añadir examen". Puesto que en muchos aspectos son similares encuestas y exámenes, este prototipo se ha realizado únicamente con el caso de trabajar con un examen. Esto se debe a que crear el examen, editarlo, eliminarlo, responderlo, ver sus resultados, etc. se realiza de forma análoga a las encuestas.

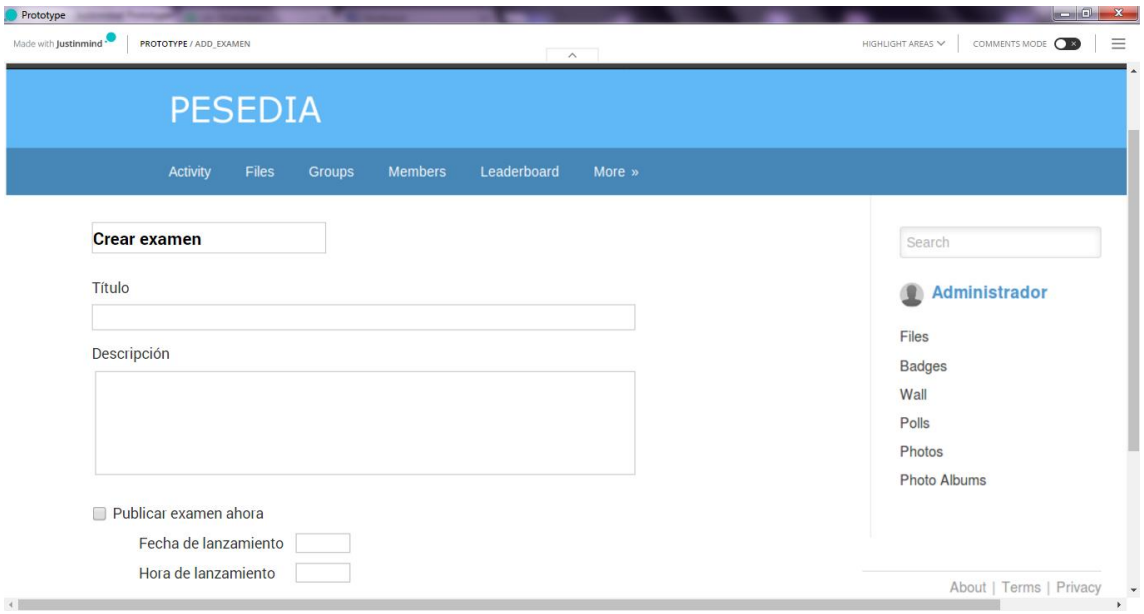

Figura 18: Prototipo #3: Crear examen 1

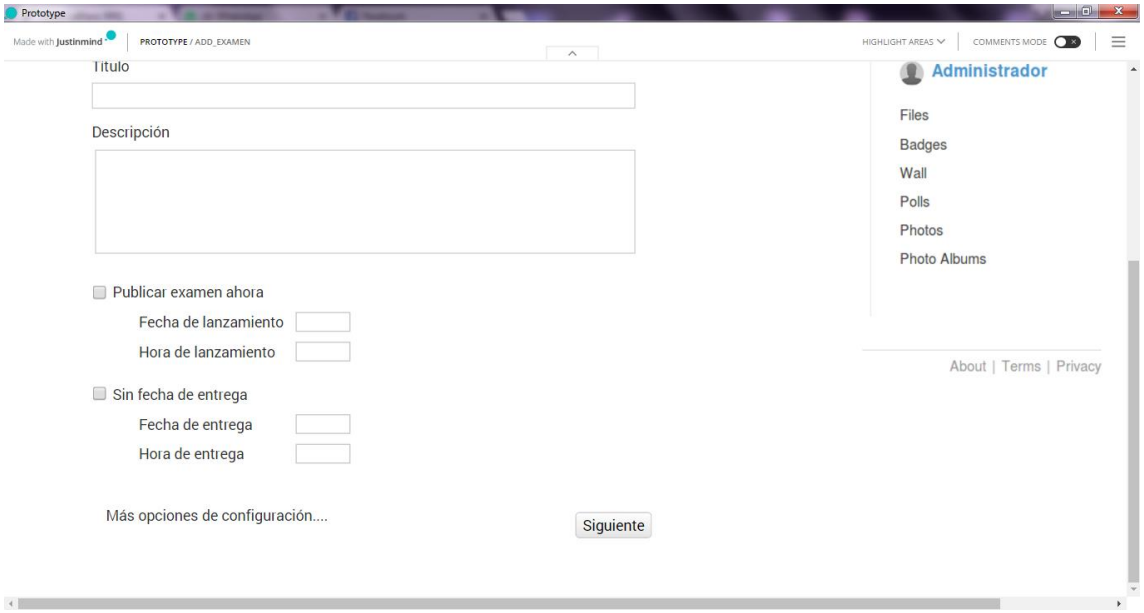

Figura 19: Prototipo #4: Crear examen 2

Al crear un nuevo examen lo primero que nos muestra la vista son el título, la descripción y la configuración del examen. Al hacer *clic* en "Siguiente" accederemos a la siguiente vista:

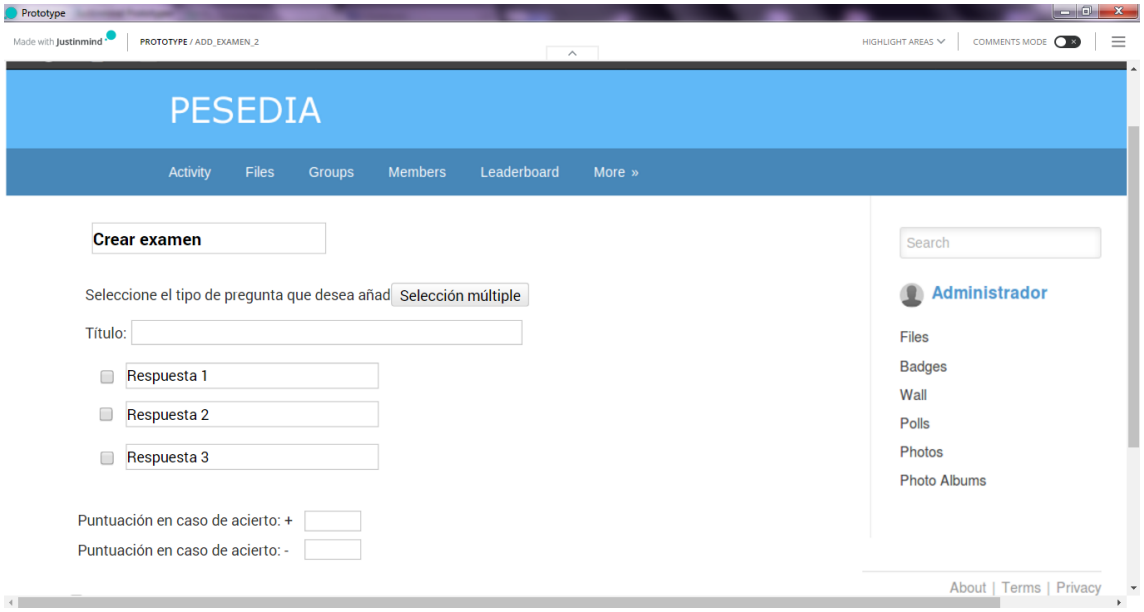

Figura 20: Prototipo #5: Crear examen 3

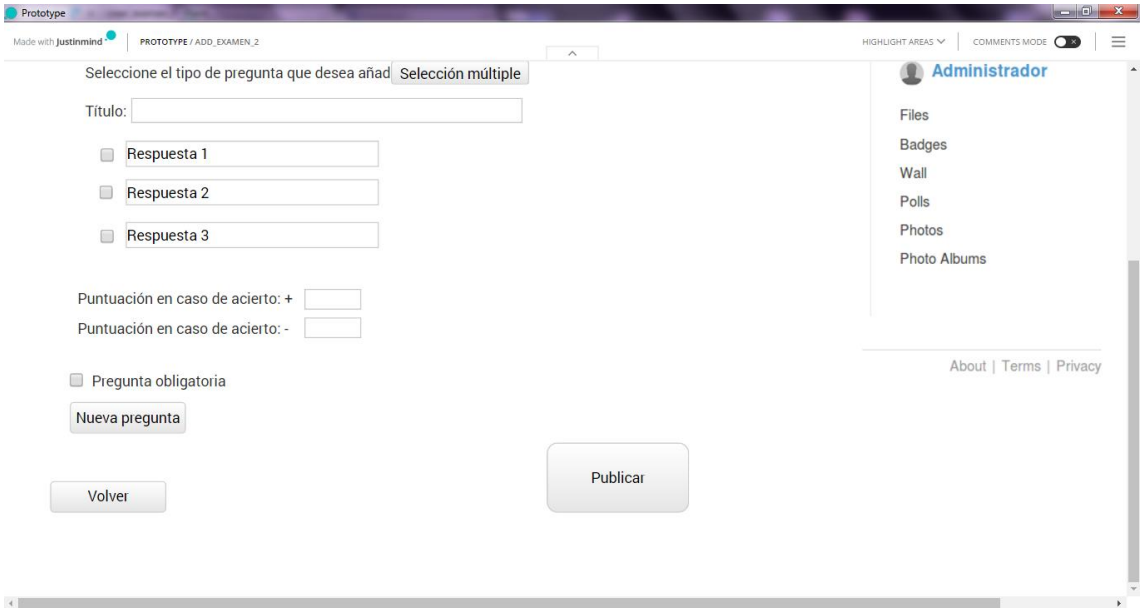

Figura 21: Prototipo #6: Crear examen 4

Se puede observar que cabe la posibilidad de que el usuario vuelva a la configuración del examen cuando desee. En esta parte del examen que muestran las figuras 20 y 21 crearemos las preguntas que lo forman, así como las opciones de configuración específicas de cada pregunta. Una vez el examen esté terminado, podrá ser publicado, y se mostrará a todos aquellos que tengan acceso a él.

Figura 22: Prototipo #7: Vista general del examen

Una vez el examen ha sido creado, es posible acceder a él haciendo clic sobre su título en la página principal del *plugin*. Esta vista ha sido incluida tras una primera evaluación del prototipo, lo cual se detalla en el apartado de pruebas.

Como podemos observar, todas las interacciones disponibles con el examen pueden controlarse en esta vista. Cabe destacar que ciertas opciones, como son ”Ver todos los

resultados", "Editar examen", "Reiniciar examen" y "Eliminar examen" únicamente estarán disponibles para el creador del examen, funcionalidad que se decidió en la fase de análisis del proyecto.

Al hacer clic en "Acceder al examen" se muestra la siguiente vista:

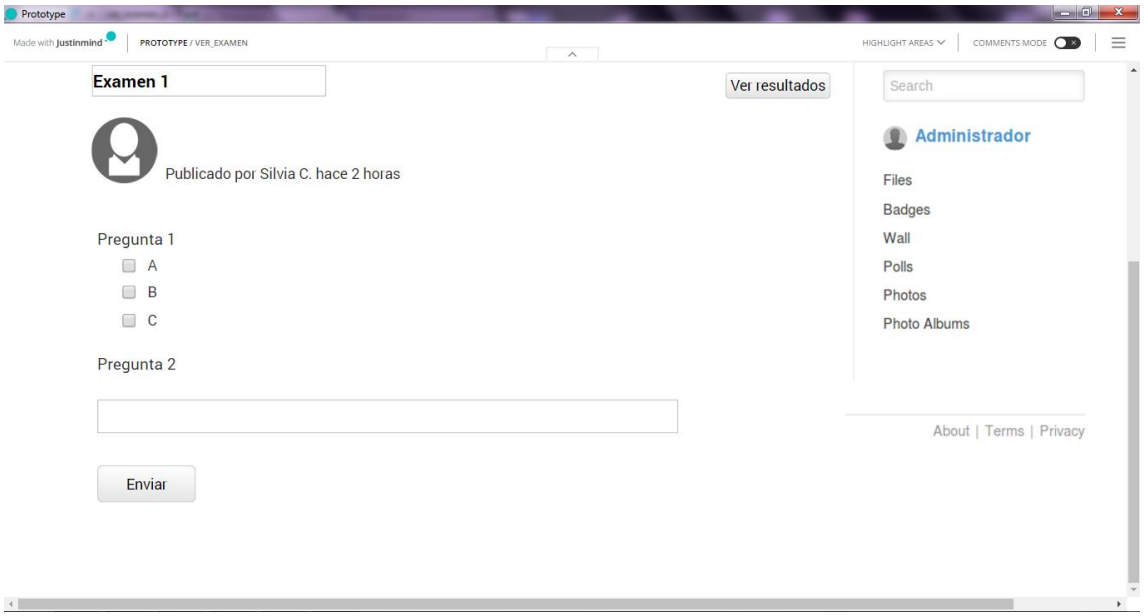

Figura 23: Prototipo #8: Responder examen

En la vista del examen mostrada en la figura 23 se muestran todas las preguntas de este, así como el tiempo límite para su realización en caso de haberlo. Finalmente también han sido incluidas en la implementación ciertas anotaciones en el propio examen, como las veces que ha sido enviado y los envíos restantes.

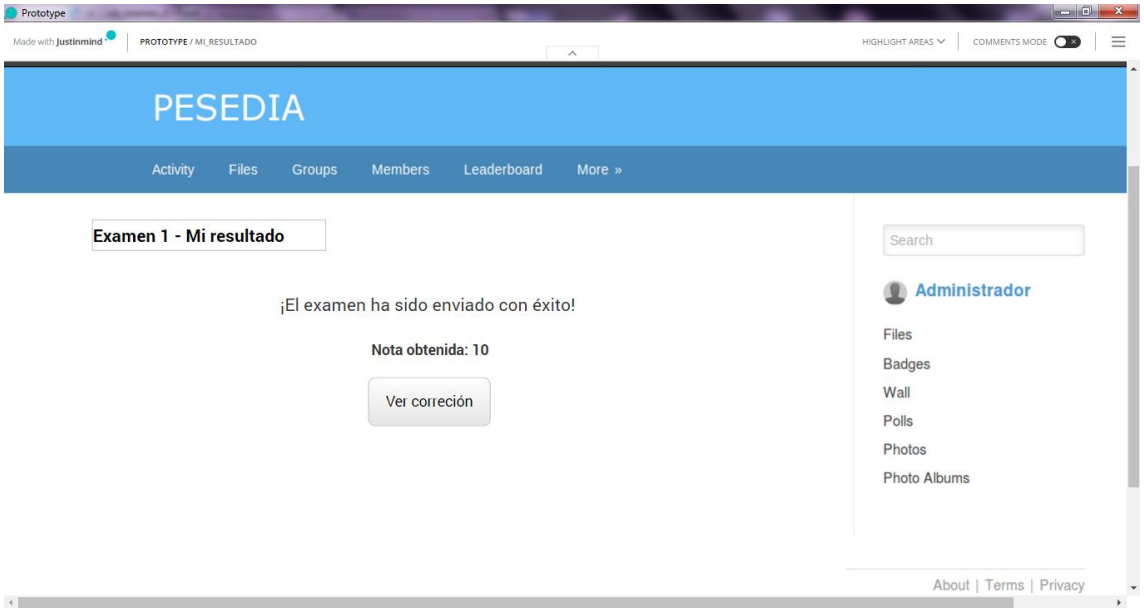

Figura 24: Prototipo #9: Mi resultado

Una vez el examen ha sido enviado aparece automáticamente la vista de la figura 24. En caso de que el creador del examen lo haya decidido así, el usuario podrá ver su nota y acceder a la corrección del mismo.

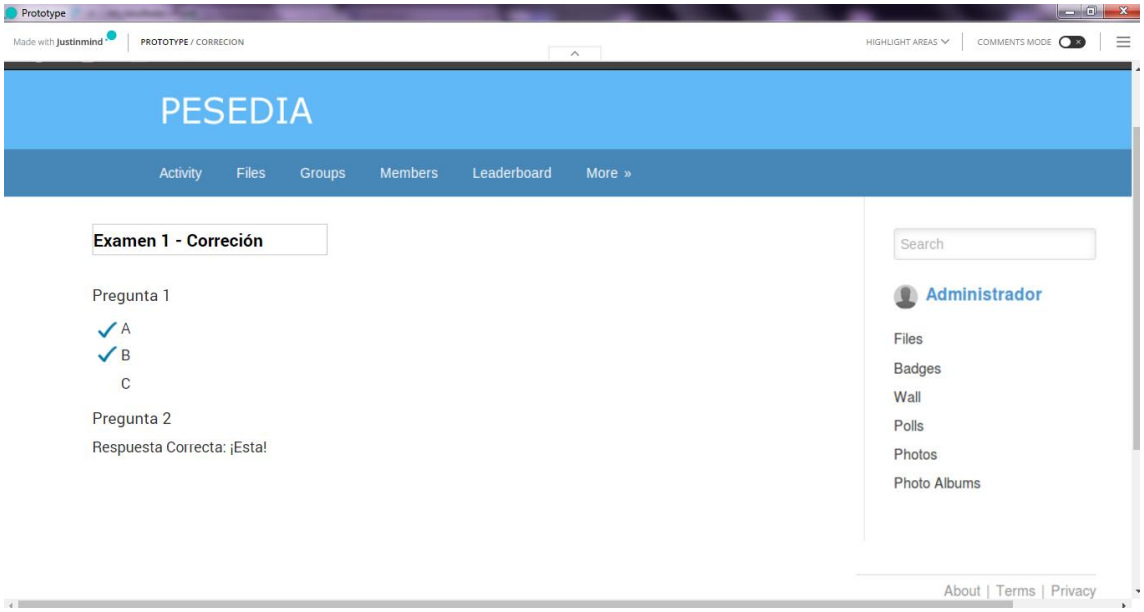

Figura 25: Prototipo #10: Corrección

En la corrección del examen se muestra cuáles eran las respuestas correctas del examen, así como las del usuario, para que este pueda ver dónde ha fallado.

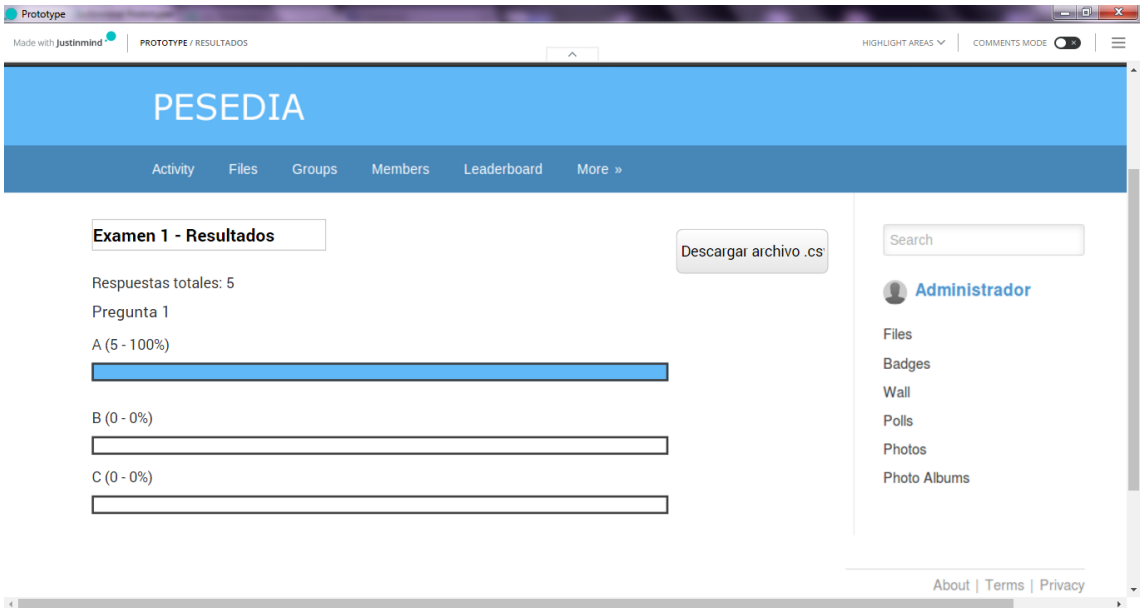

Figura 26: Prototipo #11: Todos los resultados

Por último, el creador del examen podrá acceder a una vista que muestre las estadísticas de todas las respuestas del examen, así como la opción de descargarlas en formato .csv.

6. Implementación del *plugin*

Una vez está claro qué ofrecerá exactamente el *plugin* de encuestas y exámenes, cuál deberá ser la meta de este y con una instalación de Pesedia funcional en nuestro equipo de trabajo, se ha procedido a la implementación del *plugin*. Será aquí donde debamos poner en funcionamiento todos aquellos requisitos que definimos en análisis de este proyecto siguiendo para ello todos los conceptos del diseño que se ha tratado a lo largo de todo el apartado anterior: modelo de datos y vistas. Es la implementación del *plugin* lo que permite su funcionamiento, a través de la ejecución del código que lo conforma.

6.1 Tecnologías empleadas

Aquí se pretende hacer mención de todas las tecnologías o herramientas utilizadas para poder llevar a cabo el proyecto. Se destacarán las razones por las cuales se han escogido estas herramientas, y qué funciones llevan a cabo dentro del *plugin*.

- **Elgg**: Elgg es una plataforma de código abierto que fue escogida por los creadores de Pesedia para desarrollar gran parte de la lógica y modelo de la red social, aparte de proporcionar la interfaz gráfica. Elgg es la base de nuestro proyecto, el *plugin* se va a desarrollar únicamente para Elgg, por lo que va a hacer uso de toda la tecnología de la que Elgg hace uso, además de seguir las indicaciones y requisitos que ha de cumplir un *plugin* para esta plataforma, como el patrón de diseño MVC descrito en el apartado de diseño o la estructura de directorios que se detalla en el apartado siguiente de esta memoria.

- **PHP**: Es uno de los lenguajes de código abierto más populares en el ámbito del desarrollo web. PHP es sin lugar a duda el lenguaje de programación más utilizado en este proyecto, ya que los *plugins* de Elgg han de ser necesariamente desarrollados en PHP, por lo que aunque podría ser considerado una imposición de la plataforma, realmente es la mejor opción, ya que goza de una documentación excelente, así como es muy apropiado para el desarrollo de este tipo de aplicaciones. Algo importante a tener en cuenta respecto a este lenguaje es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente.

- **HTML5**: Ha sido utilizado por el *plugin* para organizar el contenido de las vistas. Dada la opción de poder realizar incrustaciones de HTML en el código PHP y viceversa, la combinación de estas tecnologías ha sido muy utilizada a lo largo de este proyecto.

- **Javascript** y **jQuery**: A pesar de que este lenguaje de programación no ha sido altamente usado en el *plugin*, ha permitido dinamizar el contenido de varias de las vistas de manera muy útil, aunque no del todo trivial. La mayor parte del código Javascript se ha utilizado de hecho para la creación dinámica de encuestas y exámenes, aunque dada su ejecución en la parte cliente (al contrario que PHP),

también ha sido muy útil para poder obtener datos relativos a la localización de la máquina cliente.

- **CSS**: A pesar de que Elgg goza de numerosas vistas que facilitan mucho trabajar con el estilo de la visualización del *plugin*, se han dado ciertas pinceladas de CSS para poder embellecer algunas partes de la interfaz.

- **mySQL**: Elgg impone mySQL como base de datos necesaria para trabajar con la plataforma; además, mySQL es una base de datos respaldada por una gran comunidad de usuarios y buena documentación que hace su uso mucho más sencillo y cómodo.

- **Apache**: Se ha elegido Apache como servidor por razones muy similares a la elección de la base de datos, con el añadido de que en la documentación de Elgg podemos encontrar también la configuración de Apache para el correcto funcionamiento de la plataforma.

- **Bash**: Este lenguaje no se ha utilizado para el desarrollo del *plugin* en sí, pero ha sido una herramienta imprescindible para poder trabajar con la línea de comandos de Linux, tanto para la instalación y su configuración como para realizar copias de seguridad.

6.2 Estructura de directorios

A la hora de implementar un *plugin* en Elgg hay que tener en cuenta que se deben seguir ciertas buenas prácticas sugeridas en la documentación de la plataforma. Si bien algunas de estas prácticas ya se han mencionado en la fase de diseño, referentes al modelo de datos o a las vistas de un *plugin*, Elgg también sugiere seguir ciertos patrones referentes a la estructuración de los directorios de un *plguin*. El *core* de la aplicación no deberá modificarse bajo ningún concepto, por lo que nuestro *plugin* estará fuera de este, en un directorio específico para los *plugins* de Elgg llamado */mods*. Para empezar pues a programar el *plugin* de encuestas y exámenes, el primer paso ha sido crear un directorio con su nombre en el directorio *pesedia/public_html/mods*.

A partir de ahora todo el desarrollo del *plugin* se llevará a cabo dentro del directorio *pesedia/public_html/mods/encuestas_examenes*.

Para que Elgg reconozca un *plugin* como tal este deberá contar con dos ficheros en su raíz: *start.php* y *manifest.xml*.

start.php

Este fichero contiene el registro de la mayoría de controladores del *plugin* (exceptuando aquellos controladores del tipo *hook* que ejecutan los archivos Javascript). Todos estos registros se llevan a cabo en la función *encuestas_examenes_init()*, la cual se ejecuta una vez se inicializa el núcleo del sistema de Elgg si el *plugin* está activado.

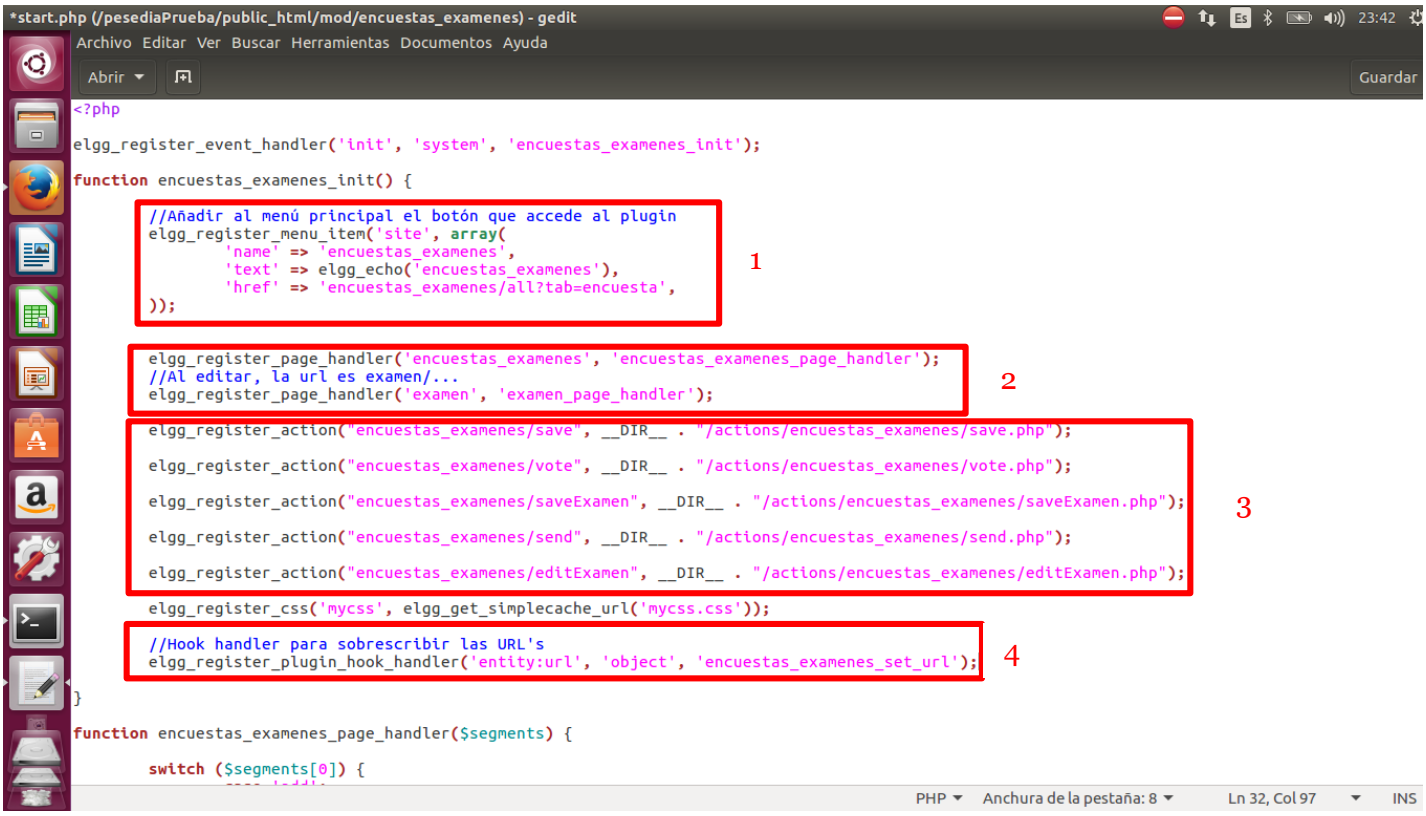

Figura 27: archivo start.php 1

Cabe destacar en esta función lo siguiente:

- **1**- A través de la llamada a *elgg_register_menu_item* se añade en el menú principal de Pesedia el acceso al *plugin* a modo de una nueva opción de menú.
- **2** – Registro de los controladores de página (*page handlers*). Este tipo de controladores fueron introducidos en la fase de diseño, y tal y como se mencionó, llama a las funciones "*encuestas_examenes_page_handler*", "*examen_page_handler* " y *"encuestas_page_handler"* con la finalidad de redireccionar las URLs de las distintas páginas a las que se accede a los archivos .php que se encargan de mostrar las vistas.
- **3** – Registro de las acciones del *plugin*. A través de las funciones *elgg_register_action* asociamos los formularios que están en la carpeta *./views/default/forms/encuestas_examenes/(primer parámetro de la función)* a sus acciones correspondientes, disponibles en la carpeta *./actions/encuestas_examenes/(tercer parámetro de la función)*, que son las

encardas de crear y modificar las entidades de la base de datos. __DIR__ hace referencia al directorio actual en el servidor.

- 4 – Este *hook* es el encargado de ejecutar la función *encuestas_examenes_set_url* (cuya captura se encuentra más adelante).

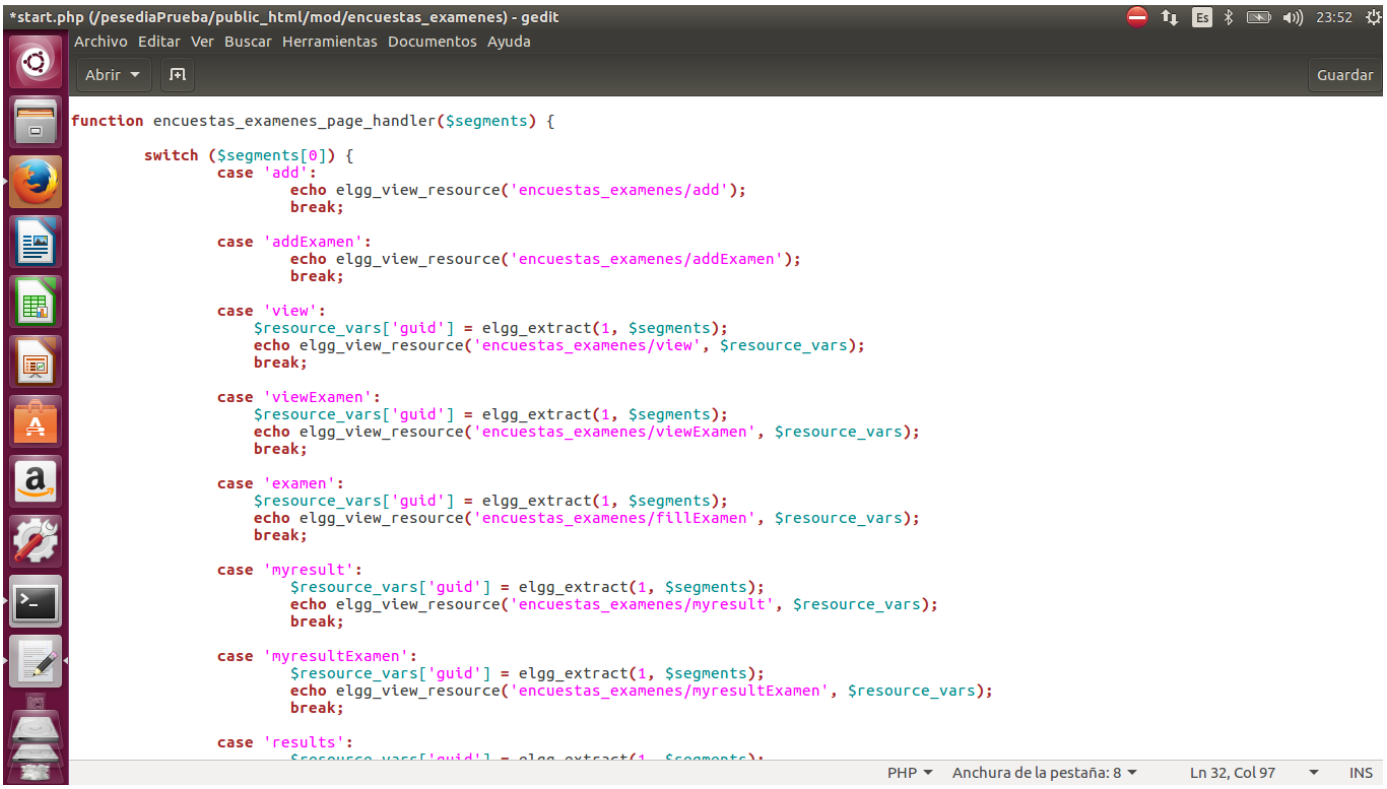

Figura 28: archivo start.php 2

He aquí parte de la función *encuestas_examenes_page_handler*, la cual funciona de la siguiente manera cuando Pesedia accede a una URL del tipo */nombre_de_la_sección/nombre_de_la_página/guid_de_la_entidad* (No siempre tiene por qué haber *guid_de_la_entidad*):

1- Elgg comprueba que hay un *page handler* asociado a *nombre_de_la_sección* y llama a la función correspondiente pasando como parámetros un *array* tal que así: *array('página', 'guid_de_la_entidad')*.

2- La función coge el primer parámetro de este *array* y a través de un *switch* extrae si procede el *guid* de la entidad y genera la vista asociada a la URL. Normalmente estas vistas se encuentran en la carpeta */views/default/resources/*, a la cual se accede a través de la función *elgg_view_resource*. A esta vista podemos pasar como parámetro el *guid* de la entidad con la que se esté trabajando en este momento, que normalmente será el *guid* de la encuesta o examen correspondiente. Esta vista en su interior contiene a su vez la generación de las vistas *elgg_view_page* y *elgg_view_layout*, encargadas de generar la interfaz de la página.

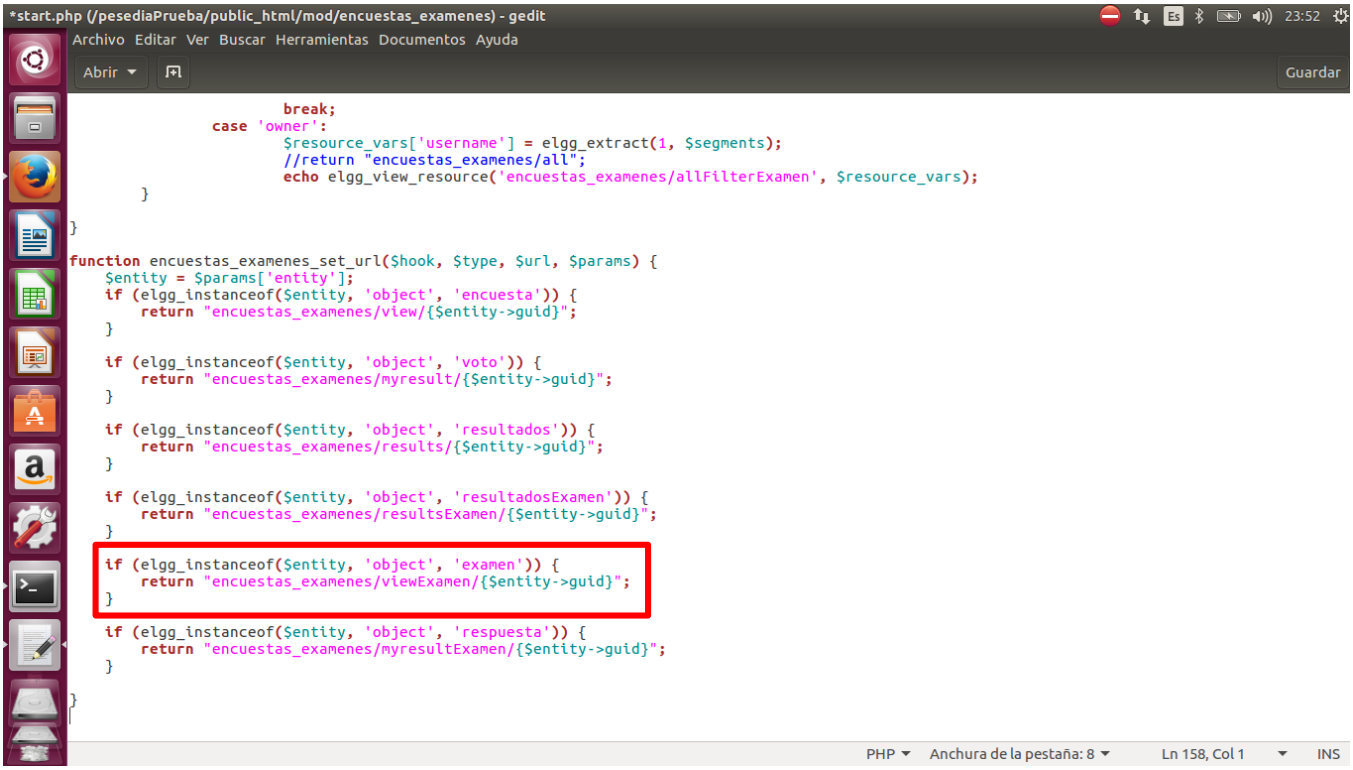

Figura 29: archivo start.php 3

En esta captura podemos ver la función *encuestas_examenes_set_url*. En ella hemos remarcado una sección del código que sirva como ejemplo para poder ver con claridad cómo, en caso de que la entidad que se quiera visualizar sea del tipo especificado (en este caso, en caso de que la entidad sea un objeto *elggObject* del subtipo *examen*), se redireccionará a la URL *encuestas_examenes/viewExamen/guid_del_examen*. Será el *page handler* el que se encargue de asociar esta URL con su vista correspondiente.

manifest.xml

Este otro fichero es un archivo estándar .xml con codificación UTF-8 que contiene información sobre el propio *plugin*: nombre, autor, descripción, dependencias, versión, etc.

Esta información es utilizada por Elgg a la hora de listar todos los *plugins* disponibles para su activación o desactivación.

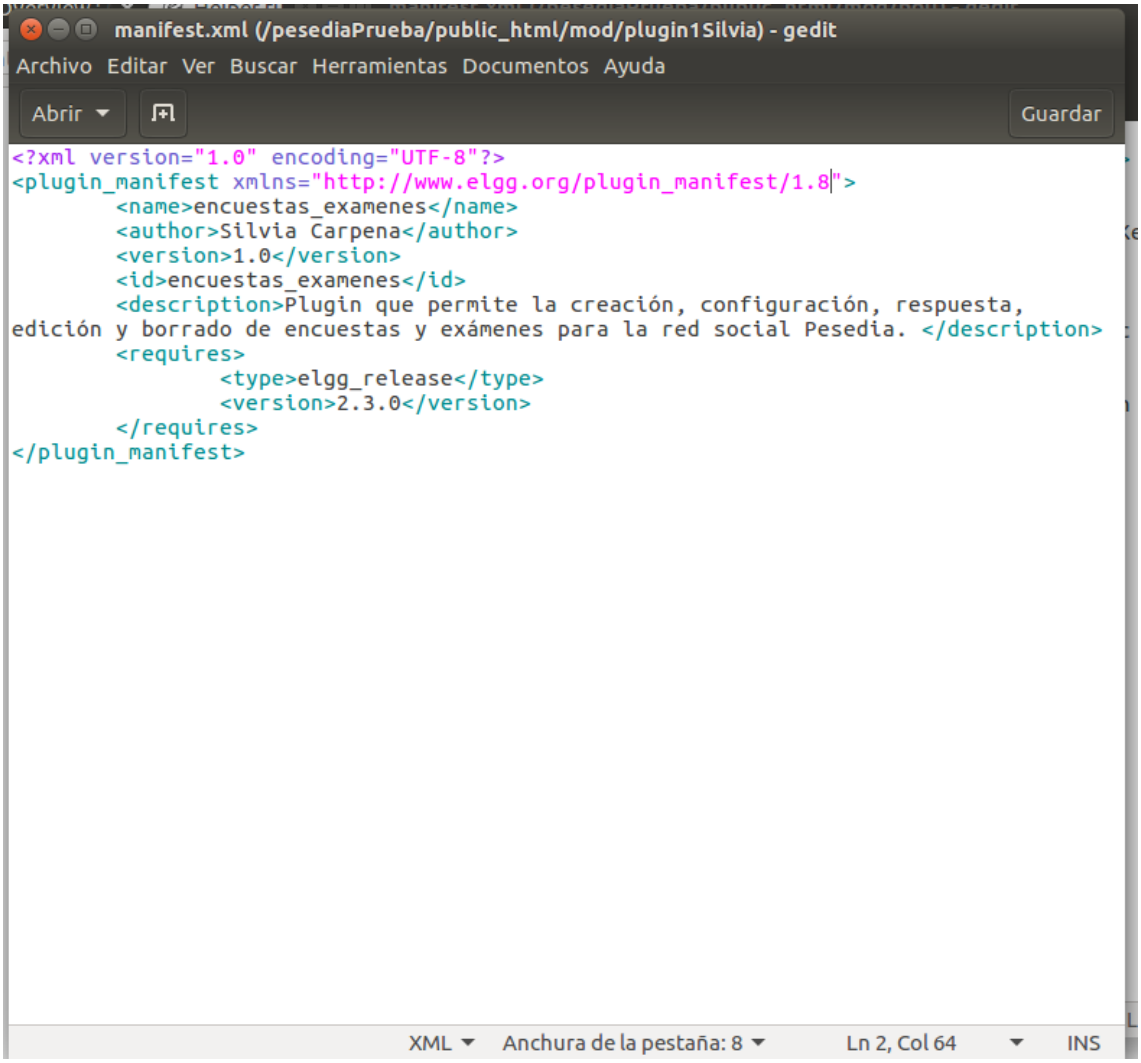

Figura 30: archivo manifest.xml

Volviendo a la raíz de un *plugin* de Elgg, existen una serie de ficheros y directorios que, a pesar de no ser obligatorios, sí que son necesarios para implementar el funcionamiento de este proyecto. En el caso del *plugin* de encuestas y exámenes, se han incluido los siguientes directorios y ficheros:

views/default/

Este directorio está formado por una serie de subdirectorios que constituyen la generación de vistas del *plugin*. Los ficheros contenidos no son las vistas en sí, no obstante, en ellos se llamarán a las funciones correspondientes para que Elgg genere las vistas, tal y como se ha explicado en detalle en la fase de diseño. Los subdirectorios más relevantes usados en el *plugin* de encuestas y exámenes son los siguientes:

- ***forms/encuestas_examenes*** – En este directorio podemos encontrar todos los formularios asociados a una acción que se usan en el *plugin*. Concretamente, estos formularios son los siguientes:
    - *saveEncuesta.php* y *saveExamen.php*– Generan las vistas para la creación de encuestas y exámenes respectivamente.

    - *vote.php* y *send.php* – Estos formularios son los que permiten al usuario rellenar la encuesta y el examen respectivamente y obtener del envío un objeto del subtipo *voto* o *respuesta*.

    - *editEncuesta.php y editExamen.php* – La edición de los dos tipos de contenido del *plugin* no es algo sencillo, para ello, hay que recuperar los datos de la encuesta o examen y mostrarlos en una vista que ha de ser idéntica a la generada por *saveEncuesta.php* y *saveExamen.php*, pero cuya implementación es completamente diferente no solo por la recuperación de datos, sino también por la posterior dinamización del formulario de creación de la encuesta o examen, así como el envío de cambios y la propagación de estos en la acción asociada al formulario.

Con el fin de profundizar un poco más en cómo se han implementado estos archivos, se ha decidido hacer un análisis de ciertos fragmentos de *saveExamen.php*, ya que si bien cada formulario es diferente, en esencia presentan muchas similitudes.

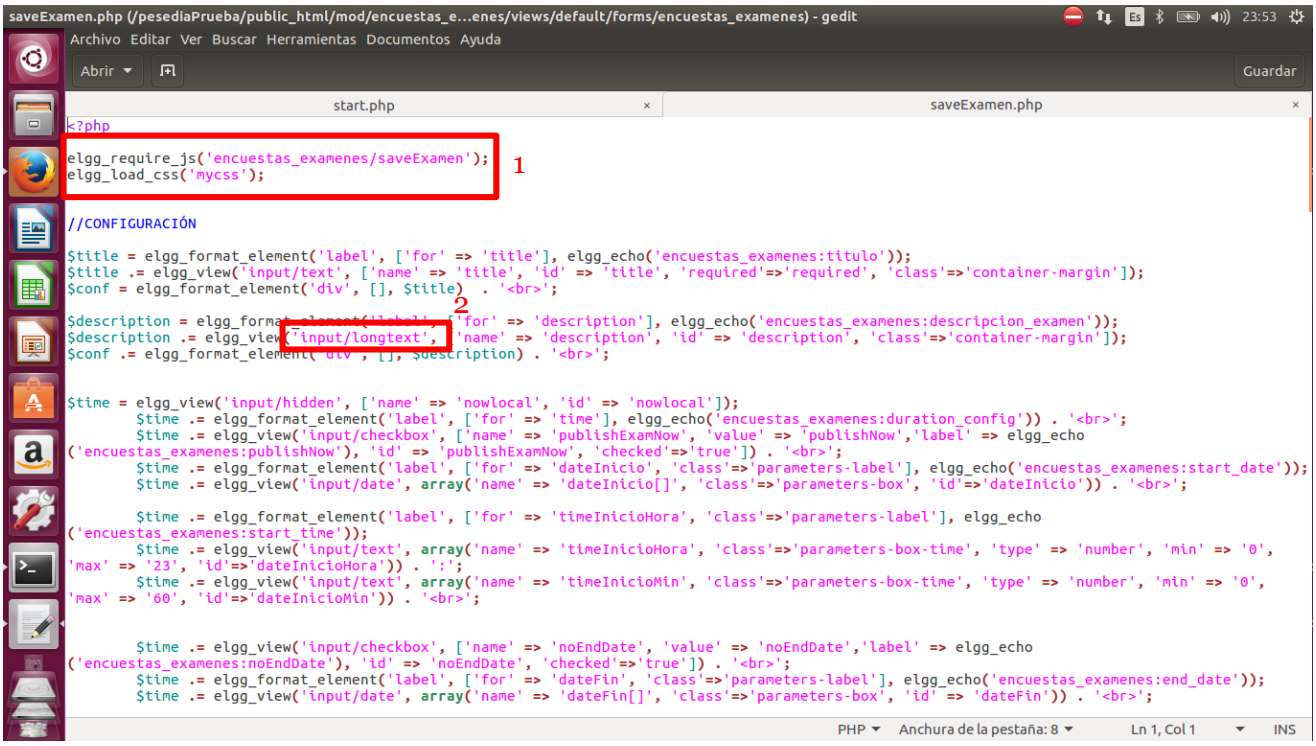

Figura 31: archivo /views/default/forms/saveExamen.php 1

El archivo está dividido en dos partes, por un lado, está el código referente a la configuración del examen, y por otra, el código referente a las preguntas del examen. La figura 31 nos muestra parte del código de la configuración del examen.

1- Se han resaltado en color rojo las llamadas a las funciones *elgg_requiere_js* y *elgg_load_css*, que se encargan de vincular *saveExamen.php* a los archivos Javascript y CSS correspondientes. En este caso, *saveExamen.php* dispone de un archivo Javascript propio que se encarga de que el formulario de creación del examen sea dinámico. El archivo CSS no obstante es común para todas las vistas del *plugin*, ya que gran parte de su código es reutilizado por vistas contenidas en directorios y archivos diferentes.

2- Cabe destacar también de esta captura el uso de los *inputs* para formularios disponibles en Elgg. El código de estos *inputs* se encuentra en el núcleo de Elgg, y a lo largo del desarrollo del *plugin* se han usado los siguientes:

- ***input/text*** - Devuelve una entrada de texto `<input type="text">`.
- ***input/longtext*** – Devuelve una entrada de texto del tipo WYSISWYG.
- ***input/checkbox*** – Devuelve una única *checkbox* `<input type="checkbox">`.
- ***input/checkboxes*** – Devuelve un grupo de *checkboxes* bajo el mismo nombre ("*name*").
- ***input/radio*** – Devuelve uno o más botones de tipo radio `<input type="radio">`.
- ***input/submit*** – Devuelve un botón del tipo *submit* `<input type="submit">`.
- ***input/button*** - Devuelve un botón `<button></button>`.
- ***input/select*** – Devuelve una entrada del tipo `<select></select>`.
- ***input/hidden*** - Devuelve una entrada oculta del tipo `<input type="hidden">`.
- ***input/date*** – Devuelve un selector de fechas jQuery (*datepicker*).

También hubiese sido posible crear el formulario con HTML, no obstante, se ha optado por usar estos *inputs* en medida de lo posible, ya que de esta forma se fomenta la reutilización de código y el uso de herramientas proporcionadas por la plataforma.

Figura 32: archivo /views/default/forms/saveExamen.php 2

Tras la generación de todas las opciones de configuración a través de estos inputs, en este fragmento de código podemos ver cómo, siguiendo el mismo método, se crean las preguntas del examen.

3- También se ha hecho uso de unas vistas de iconos que Elgg contiene también por defecto, concretamente, de tres de ellos:

elgg_view_icon('delete')

elgg_view_icon('arrow-left')

elgg_view_icon('checkmark')

4- Existen cuatro tipos pregunta tanto para una encuesta como para un examen: texto, texto largo, pregunta de respuesta única o pregunta de respuesta múltiple, que se corresponden con los inputs *input/text*, *input/longtext*, *input/radio* e *input/checkboxes*, respectivamente.
En este caso, el título de la pregunta se generará de igual manera para todos los tipos de pregunta, así como un menú desplegable (*drop-down*) del tipo *input/select* que permitirá al creador del examen elegir el tipo de pregunta que desee añadir.

A continuación se muestran diferentes capturas de los tipos de pregunta mencionados

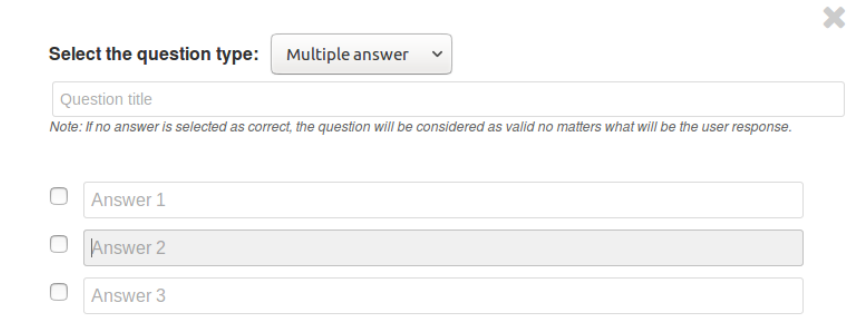

Figura 33: Pregunta de respuesta múltiple en un examen

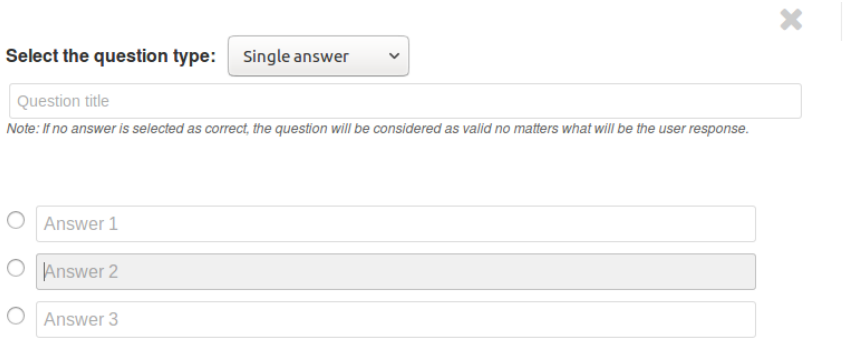

Figura 34: Pregunta de respuesta única en un examen

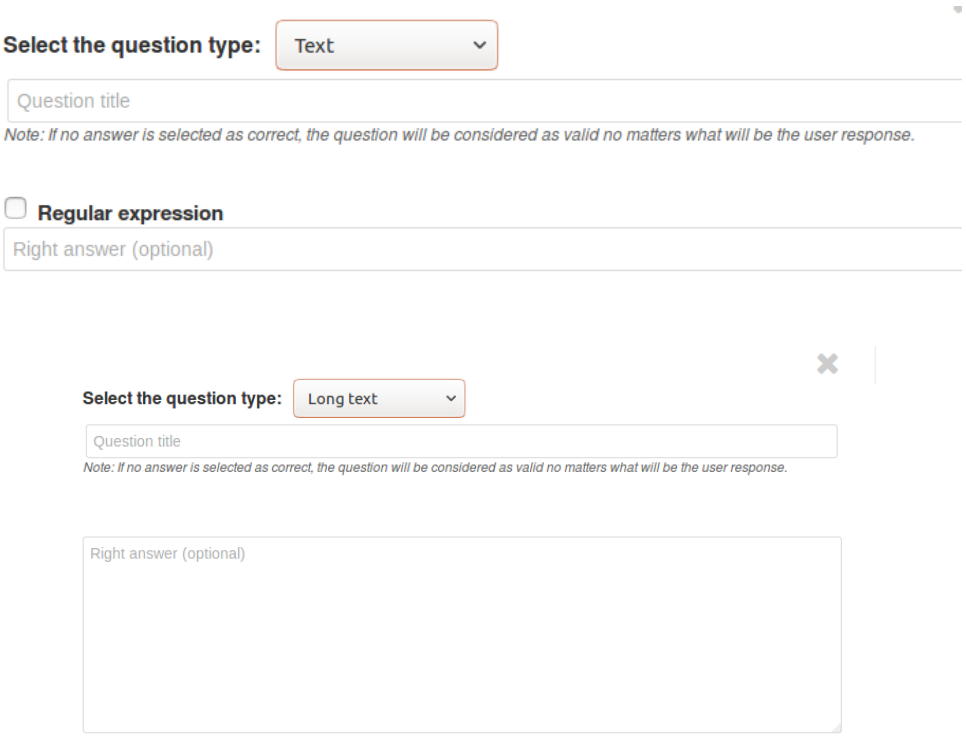

Figura 35: Preguntas de respuesta corta y respuesta larga en un examen

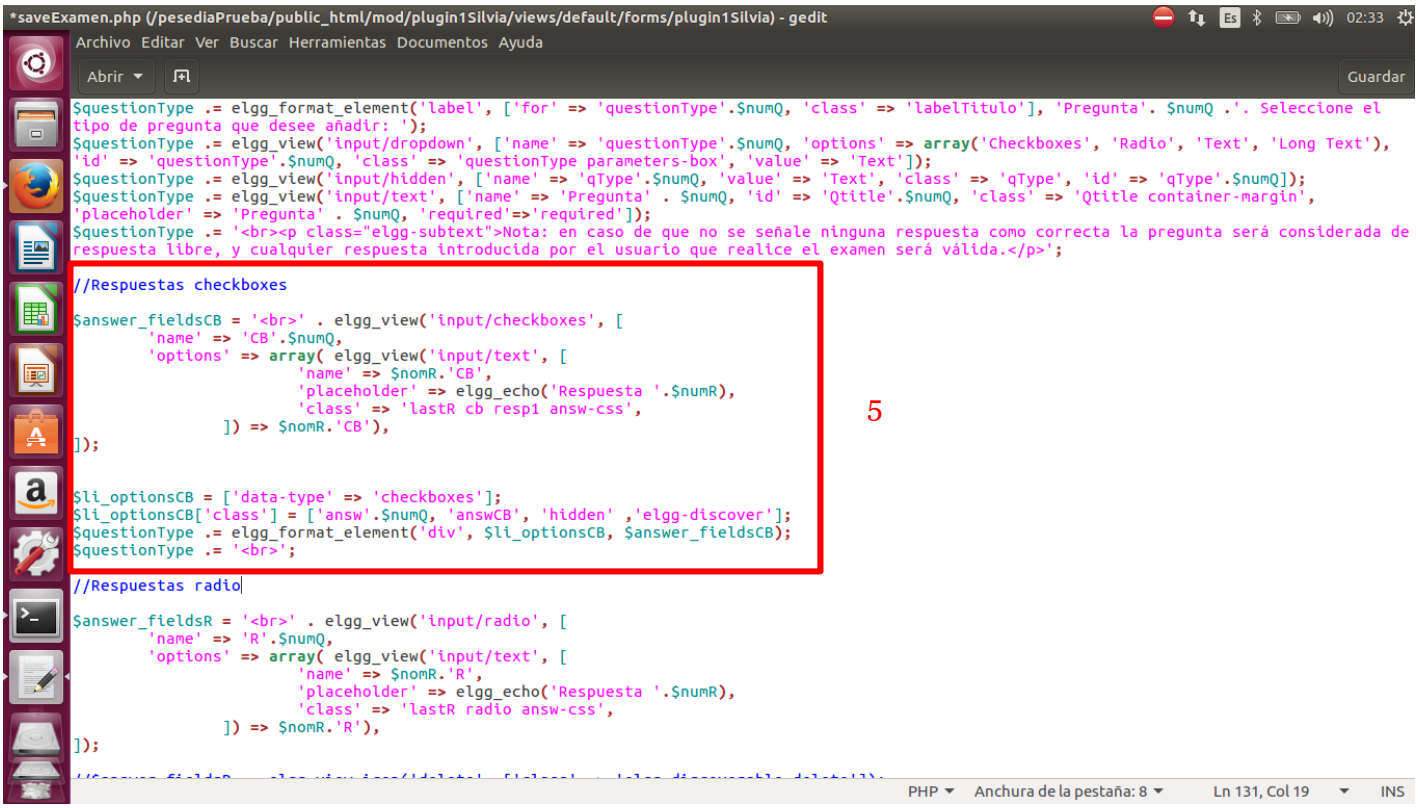

Figura 36: archivo /views/default/forms/saveExamen.php 3

5- En este fragmento de código aparece destacado como ejemplo cómo se implementa una pregunta del tipo respuesta múltiple, y es que el examen (al contrario que la encuesta), no solo debe permitir la introducción de las múltiples respuestas entre las que el usuario podrá elegir, sino que también deberá permitir marcar cuáles de ellas son las correctas para poder así evaluar el examen. Es por ello que en este caso, tanto preguntas de respuesta múltiple como en preguntas de respuesta única cuentan con un *input/text* para que el creador del examen pueda escribir directamente las opciones de la respuesta y marcar cuáles de estas son correctas.

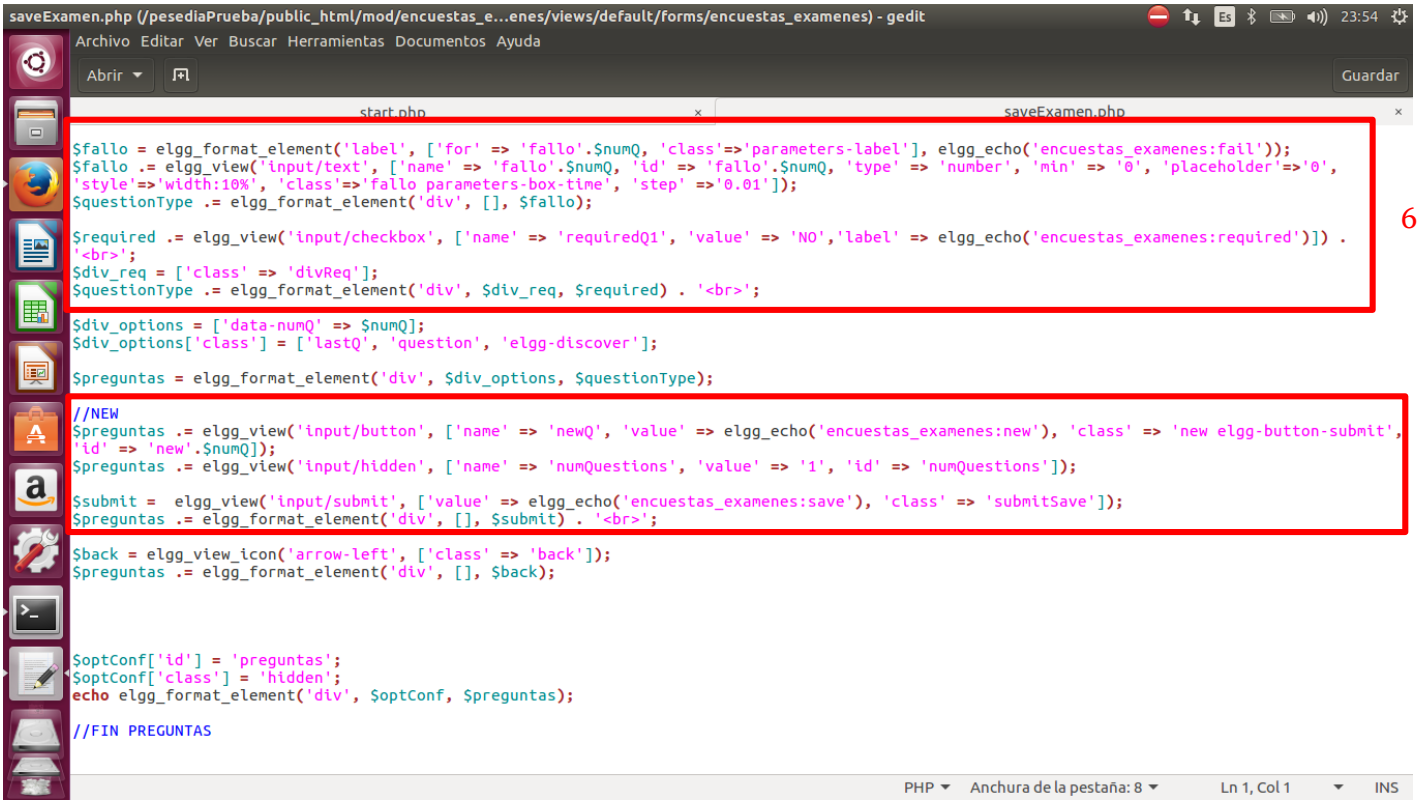

Figura 37: archivo /views/default /forms/saveExamen.php 4

6- Cada pregunta del examen también tiene opciones de configuración específicas, como la nota de la pregunta (en caso de acierto o de fallo), si la pregunta es obligatoria o no, o si se aceptará una expresión regular como respuesta válida en caso de una pregunta de texto.

7- Además, en el fragmento de código mostrado en la figura 37 también se pueden ver dos botones: uno para crear una nueva pregunta (función que lleva a cabo el archivo Javascript) y el botón de publicación del examen, del tipo *input/submit*, que es el que ejecutará la acción correspondiente.

- ***object/*** - Como se ha tratado en varias ocasiones, este *plugin* hace uso de diferentes entidades del tipo *elggObjet*. En el *plguin* de encuestas y exámenes se ha necesitado crear vistas específicas para las entidades con subtipo *encuesta* y *examen*. Para ello, Elgg permite hacer uso de una función llamada *elgg_view_entity*. En este directorio se han guardado dos archivos: *encuesta.php* y *examen.php*, los cuales contienen dos tipos de vista cada uno accesibles a través de *elgg_view_entity*. Estas vistas son: vista completa (*full*) o vista resumida (*summary*). La vista completa es la que se muestra cuando se visualiza una encuesta o un examen, y la vista resumida es la que se muestra al listar todas las encuestas y exámenes al acceder al *plugin*, donde se muestra el título de la encuesta o examen, su autor, su descripción y algunos datos más, tal y como se puede ver en las figuras 38 y 39.

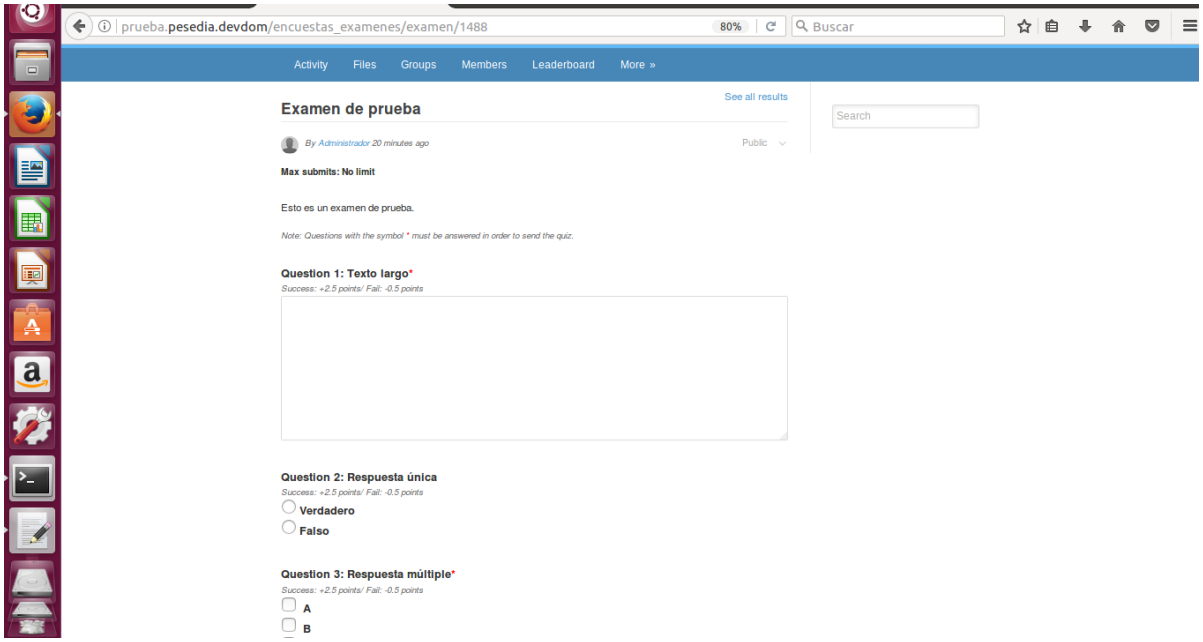

Figura 38: Captura de un examen "full"

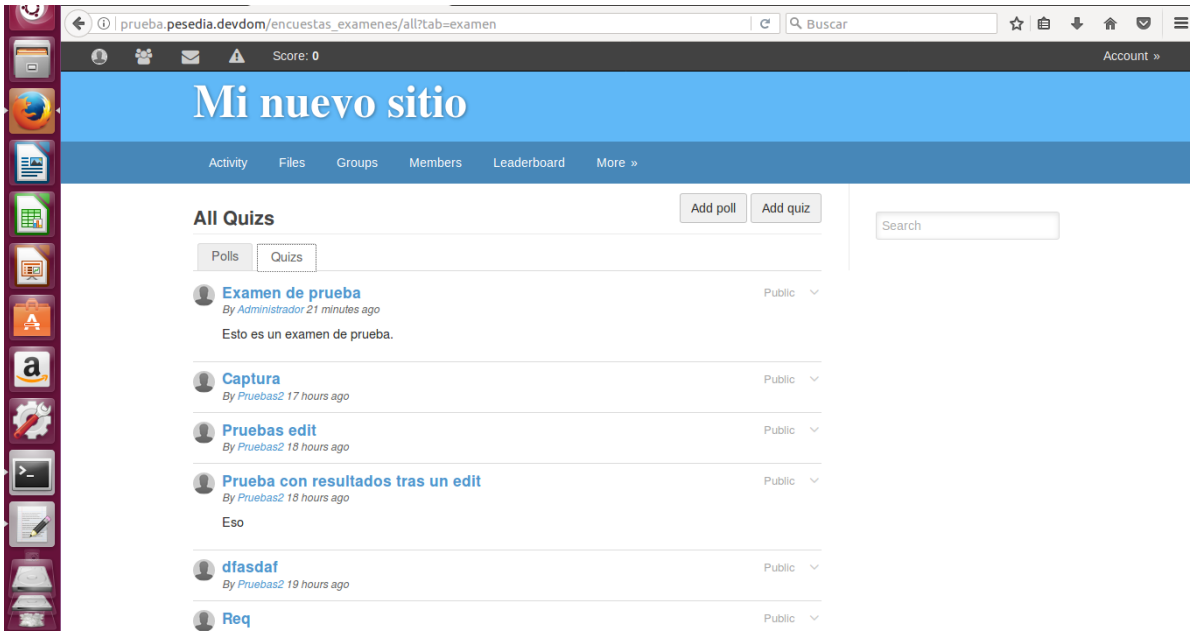

Figura 39: Página principal del plugin donde se muestran varios exámenes "summary"

Se muestra a continuación parte del código que permite ver en la pantalla lo mostrado en las figuras anteriores:

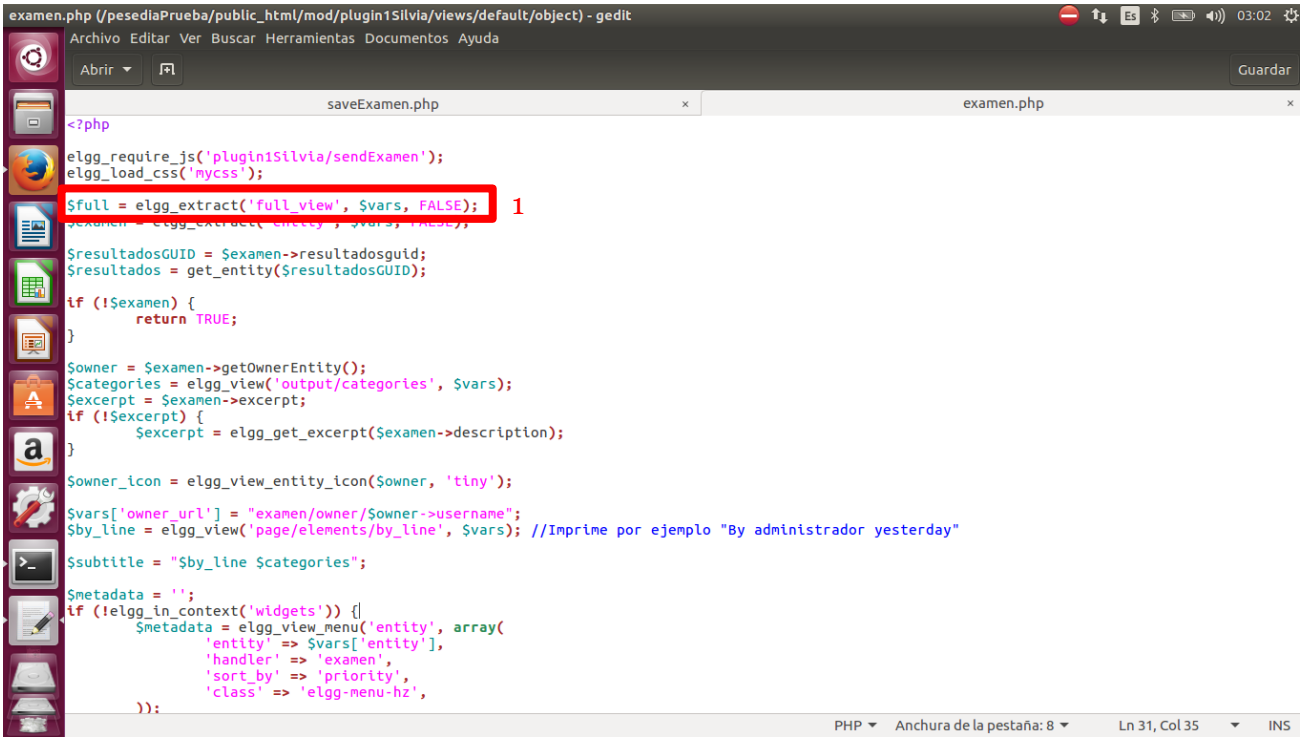

Figura 40: archivo /views/default/object/examen.php 1

1- Esta línea de código recupera qué tipo de vista deberá ser mostrada del examen (en este caso), y prepara ciertos elementos que son comunes en ambas vistas, como la fecha de creación del examen, su autor o la foto de perfil del usuario.

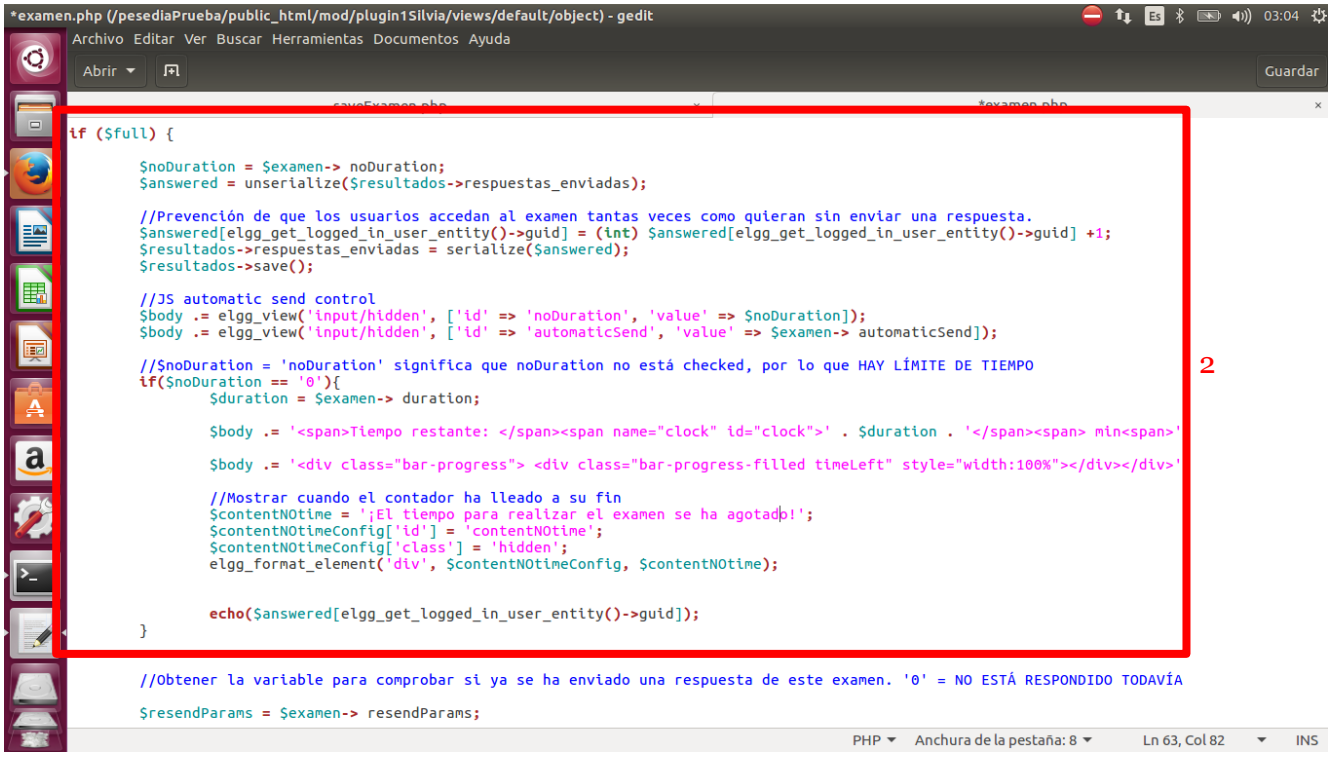

Figura 41: archivo /views/default/object/examen.php 2

2- En esta captura se muestra parte del código que se ejecuta para dar lugar a la vista completa del examen, para ello, opciones de configuración como el tiempo límite para realizar el examen o el envío automático del examen en caso de que el tiempo límite se agote son recuperadas y utilizadas para mostrar en la vista.

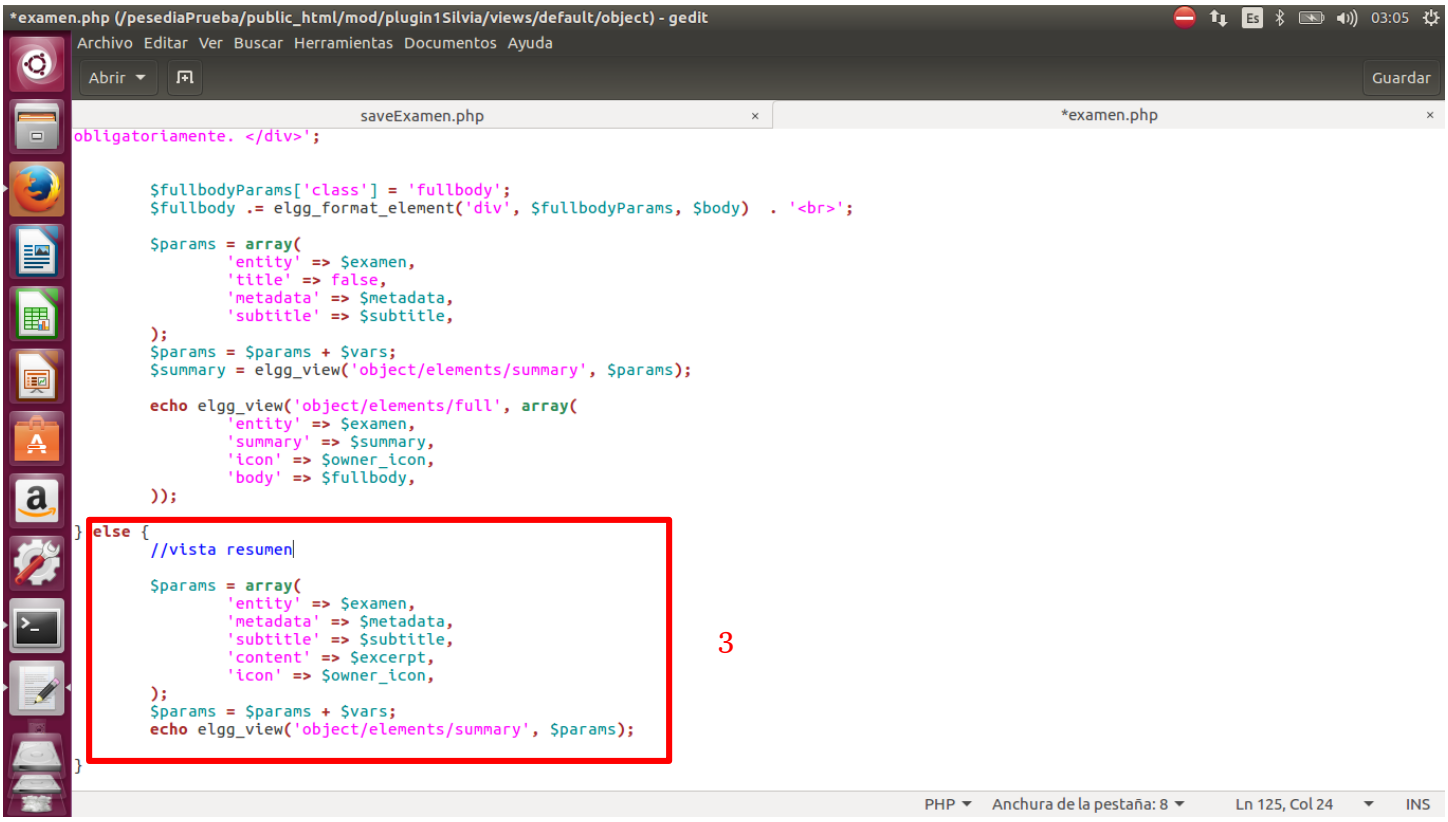

Figura 42: archivo /views/default/object/examen.php 3

3- En caso de mostrar el resumen únicamente cuando se listan todos los exámenes, la información que necesitamos es mucho menor que al mostrar la vista completa.

- ***resources/encuestas_examenes*** – En este directorio se encuentran las páginas que constituyen las vistas de nuestro *plugin*. Elgg facilita el acceso a estas vistas a través de la función *elgg_view_resource()*, que es utilizada por el fichero *start.php* numerosas veces para mostrar el contenido adecuado según la URL. Los archivos contenidos en el directorio son los siguientes:
    - *all.php:* El código contenido en este archivo es el encargado de mostrar la primera vista del *plugin* cuando se accede a él a través del menú de Elgg.
    - *allFilterEncuesta.php y allFilterExamen.php:* Es posible filtrar las encuestas y exámenes por su autor accediendo al enlace directo con su nombre de usuario que aparece junto a una encuesta o examen, estos archivos son los encargados de implementar este filtro y su vista correspondiente.

- *viewEncuesta.php y viewExamen.php*: Esta vista muestra las opciones de las que dispone el usuario referentes a una encuesta o examen: acceder a él, ver el resultado enviado, ver su corrección (en caso de que sea un examen) y editar, eliminar, reiniciar o visualizar los resultados (en caso de ser el creador de la encuesta o examen.)

- *fillEncuesta.php y fillExamen.php:* Vista encargada de mostrar los formularios *vote.php* (para encuestas) y *send.php* (para exámenes) los cuales permiten rellenar la encuesta o examen respectivamente y enviar los resultados al servidor.

- *addEncuesta.php y addExamen.php:* Al igual que la vista anterior, estos archivos contienen el código para mostrar los formularios *saveEncuesta.php* y *saveExamen.php*, que permiten añadir una nueva encuesta o examen respectivamente.

- *editEncuesta.php y editExamen.php:* Archivos encargados de visualizar los formularios editEncuesta.php y editExamen.php, que permiten la edición de encuestas y exámenes.

- *eliminarEncuesta.php y eliminarExamen.php:* Archivos encargados de llevar a cabo el borrado de encuestas y exámenes. No cuentan con vista propia, únicamente con un mensaje de confirmación.

- *reiniciarEncuesta.php y reiniciarExamen.php:* Al igual que los archivos anteriores, estos son los encargados de reiniciar los resultados de encuestas y exámenes tras mostrar un mensaje de confirmación.

- *myresultEncuesta.php y myresultExamen.php:* Estos archivos son los encargados de mostrar al usuario que responde la encuesta o examen los resultados que ha enviado y, en caso de los exámenes, la nota obtenida si el creador lo ha decidido así.

- *resultsEncuesta.php y resultsExamen.php:* Esta vista permite al creador del examen acceder a los resultados obtenidos hasta el momento en la encuesta o examen.

- *correctionExamen.php:* En caso de que el creador del examen lo haya decidido así, el usuario que lo responda podrá acceder a esta vista para ver la corrección de las respuestas del examen que ha enviado.

A continuación se pretende mostrar ciertas partes del código de algunos de estos archivos que se ha considerado interesante de mención.

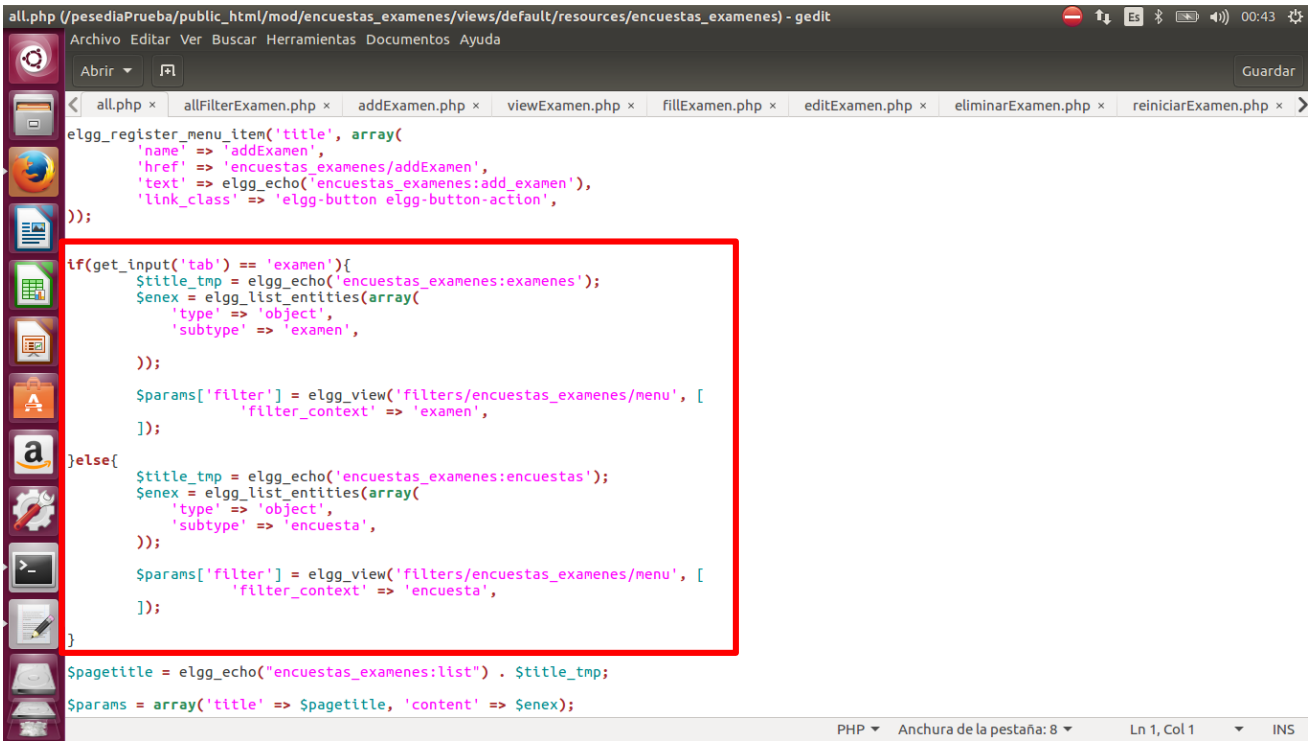

Figura 43: archivo /views/default/resources/encuestas_examenes/all.php 1

En la página principal del *plugin* (tal y como se puede ver en la figura 39) aparece un listado de todas las encuestas y exámenes en un menú con dos pestañas (una para cada tipo de contenido). El código de la figura muestra precisamente cómo se ha implementado dicho filtrado, recuperando para ello todos los objetos del tipo encuesta o examen dependiendo de la pestaña del menú en la que se encuentra el usuario.

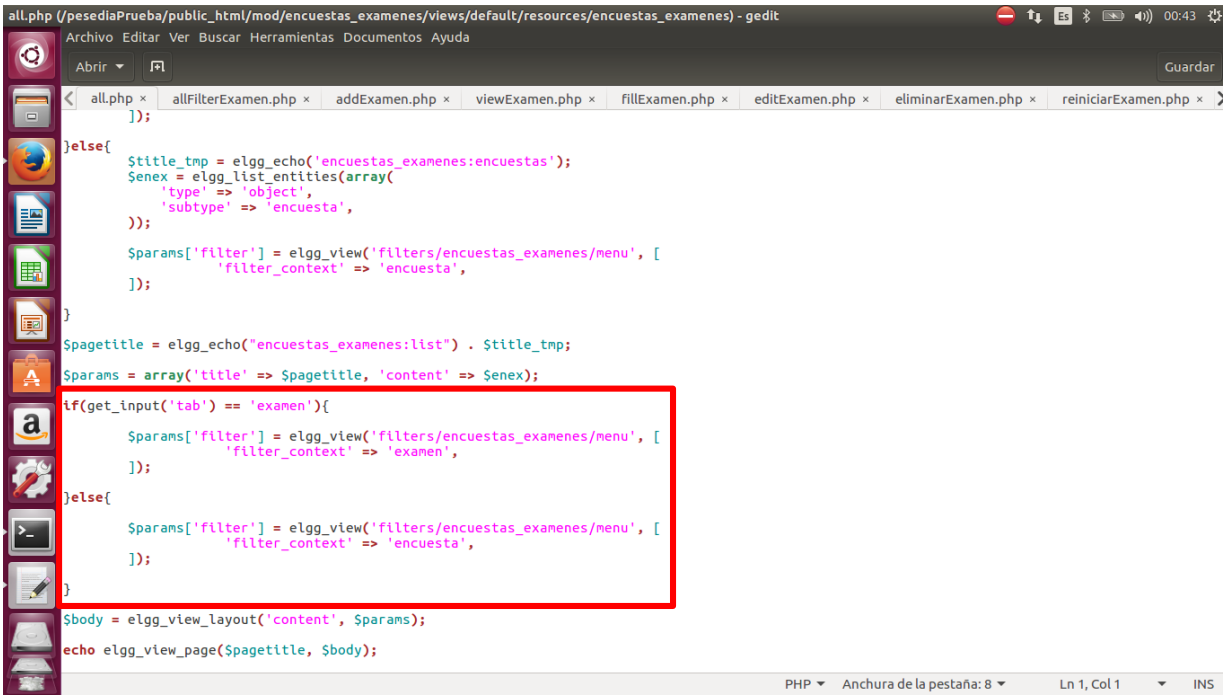

Figura 44: archivo /views/default/resources/encuestas_examenes /all.php 2

Poder mostrar estos listados que se mencionaban bajo la captura 43 no requiere únicamente de recuperar el contenido de la base de datos, sino que también se ha tenido que reescribir con un tipo de menú de pestañas ya disponible en Elgg. Detalles de este menú serán aportados cuando se haga referencia al directorio que contiene el código sobrescrito *(/views/default/filters*), en este mismo apartado de la memoria.

Figura 45: archivo /views/default/resources/encuestas_examenes/addExamen.php

En esta captura se muestra el archivo *addExamen.php* en su totalidad. Como puede observarse, el código que se implementa tiene como principal objetivo ofrecer la vista del formulario *saveExamen.php* y situarla dentro de las vistas básicas *layout* y *page*. El código del resto de archivos encargados de mostrar formularios es muy similar al de esta captura.

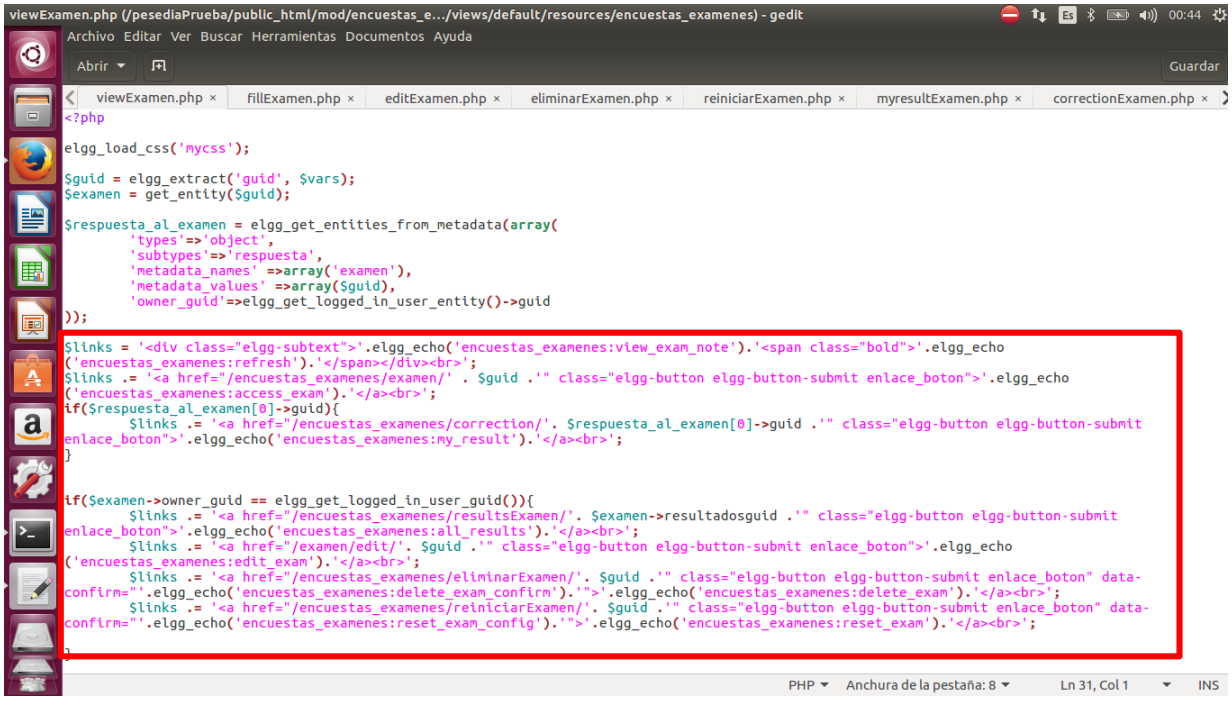

Figura 46: archivo /views/default/resources/encuestas_examenes/viewExamen.php

El código de *viewExamen.php* (al igual que *viewEncuesta.php*) consta de una serie de enlaces HTML que ofrece todas las opciones disponibles para el examen (o la encuesta). Su interfaz puede verse en la figura 65.

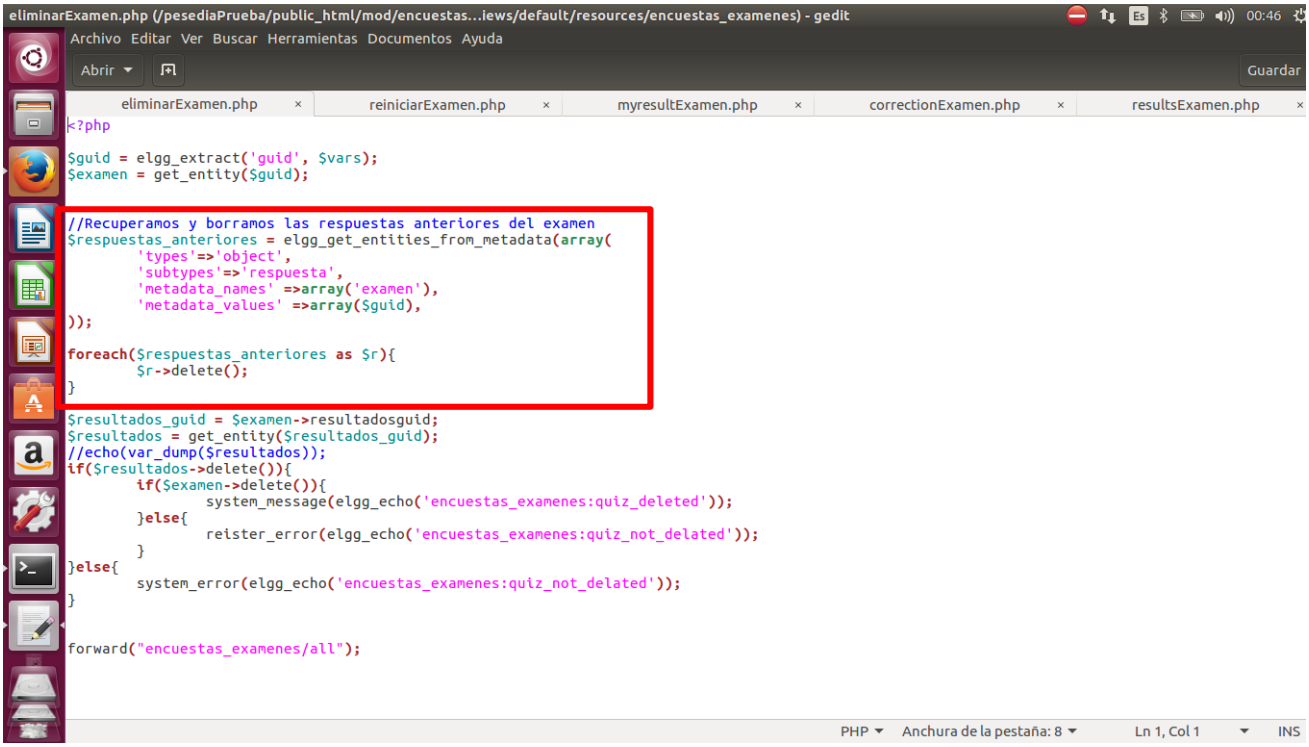

Figura 47: archivo /views/default/resources/encuestas_examenes/eliminarExamen.php

Para eliminar y reiniciar una encuesta o examen en primer lugar se recogen todas las respuestas asociadas (*respuesta* y *voto*), y luego, se eliminan definitivamente de la base de datos.

Posteriormente se eliminan los objetos *resultadosEncuesta* o *resultadosExamen* y los propios *encuesta* o *examen*. Al reiniciar, se omite este último paso, y únicamente se eliminan definitivamente las respuestas, modificando *resultadosEncuesta* y *resultadosExamen* para eliminar también los resultados almacenados en ellos.

Figura 48: Captura de confirmación para eliminar un examen

El código de eliminar o reiniciar una encuesta o examen no viene una vista propia, únicamente muestra el mensaje de confirmación que se puede ver en la figura 48. En este caso, Pesedia está configurada en inglés y es por eso que todos los textos del *plugin* están en este idioma, pero como ya se ha mencionado en la fase de Análisis (y tal y como se aclara al hablar del directorio *languages/*), también está disponible la traducción al español.

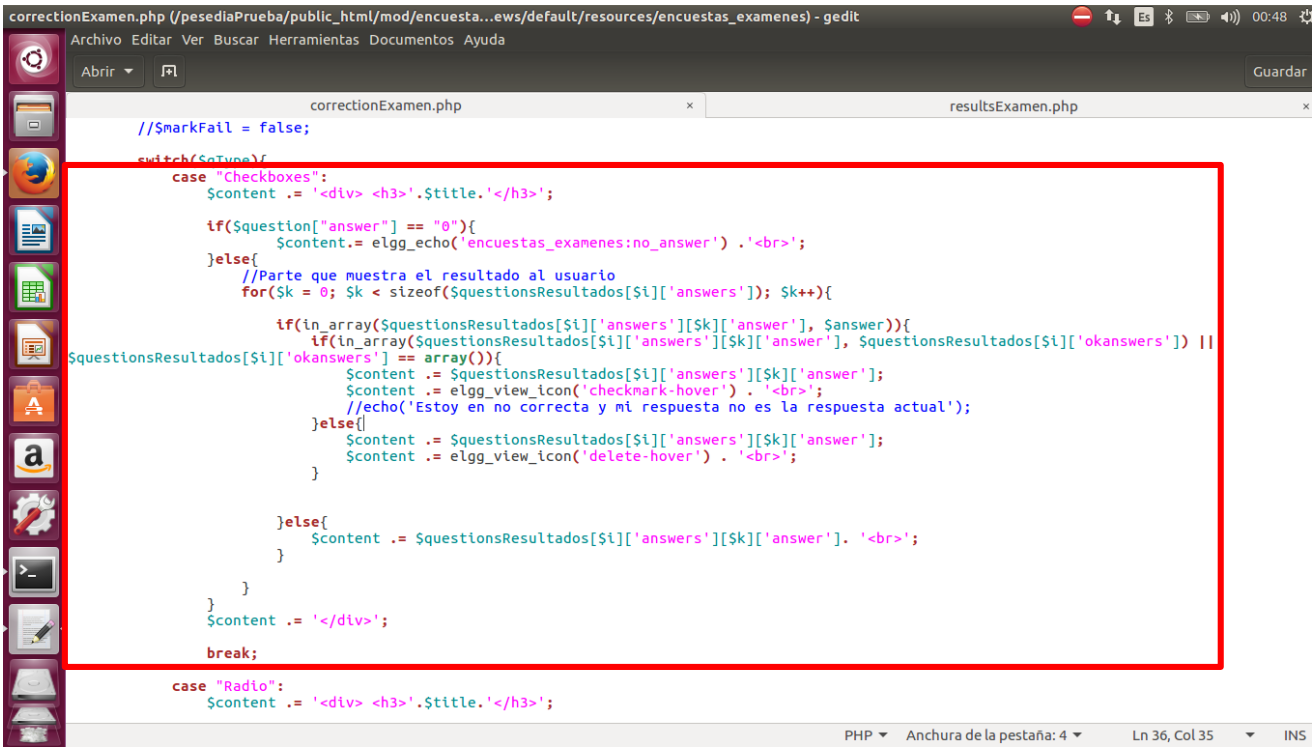

Figura 49: archivo /views/default/resources/encuestas_examenes/correctionExamen.php

Tanto *myresultsExamen.php* (y *myresultEncuesta.php*) como *correctionExamen.php* implementan el código que permite no solo comprobar si las respuestas introducidas por el usuario son correctas o no, si no también calcular la nota del examen y mostrar todos estos resultados al usuario. En el fragmento de código seleccionado podemos ver cómo se llevan a cabo estas comprobaciones de si la respuesta del usuario es la correcta o no, y cómo en caso de que si lo sea se introducirá un símbolo de acierto (*icon/checkmark*) y en caso de que no lo sea un símbolo de error (*icon/delete*).

- ***filters/encuestas_examenes*** – Para poder generar el menú de pestañas que se puede ver al entrar al *plugin* de Encuestas y Exámenes, ha sido necesario reescribir la vista del menú "*filter*" que ya viene por defecto en el *core* de Elgg. Es por ello que en este directorio hay un único archivo llamado *menu.php*, con el nuevo código del menú *filter*, tal y como se puede ver en la captura de abajo. Esta vista es llamada por el archivo *views/default/resources/all.php*, y aunque pueda parecer que este fragmento de código podría ser introducido directamente en él, no es así. Esto se debe a que se ha hecho un reemplazamiento del código del *core* en el

plugin de encuestas y exámenes, y para ello, el directorio y el nombre del archivo de la vista debe coincidir exactamente con el contenido que se quiere sustituir en el *core* de Elgg.

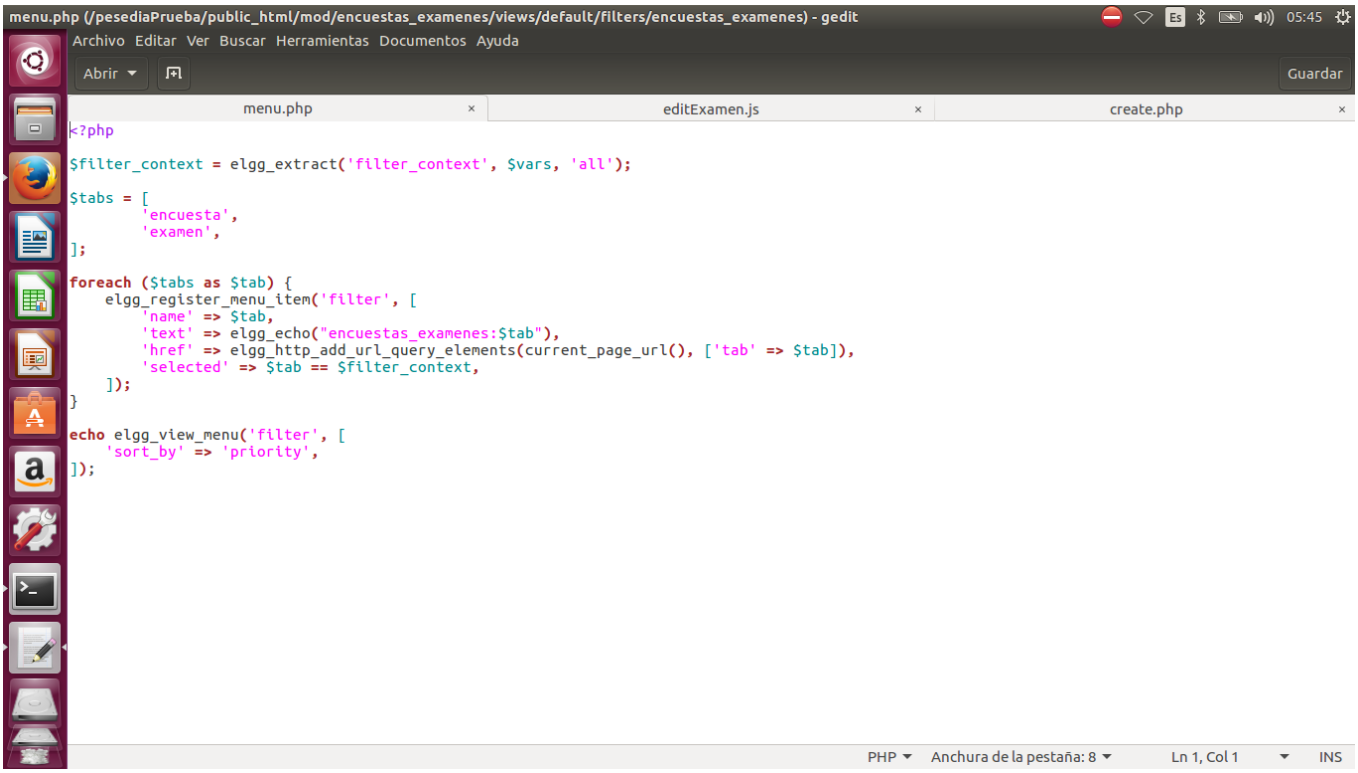

Figura 50: archivo /views/default/filters/encuestas_examenes/menu.php

- ***js/encuestas_examenes*** – Esta carpeta guarda los archivos Javascript del proyecto. Todos los archivos Javascript están asociados a formularios, ya que la función principal de su uso ha sido dinamizar los formularios de creación de encuestas y exámenes, así como también validar si un examen está listo para ser enviado o no (si se enviará la respuesta automáticamente cuando el tiempo de respuesta límite se agote, si todas las preguntas del examen obligatorias han sido respondidas, si el examen no se envía vacío, etc). Los archivos situados en este directorio son pues los siguientes, cada uno asociado a su formulario de idéntico nombre.
    - *saveEncuesta*.js y *saveExamen.js*
    - *vote.js y send.js*
    - *editEncuesta.js y editExamen.js*

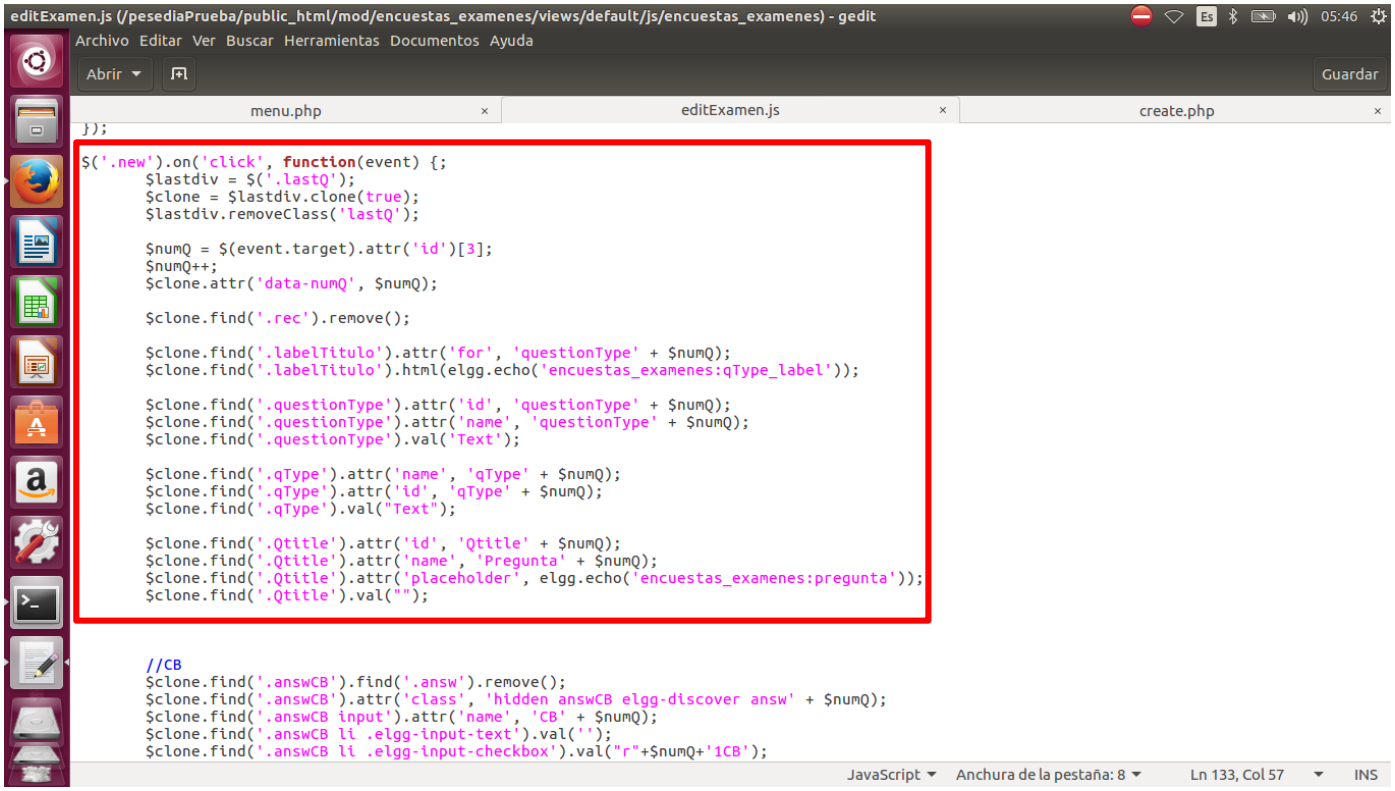

Figura 51: archivo /views/default/js/encuestas_examenes/editExamen.js 1

Se muestra en la figura 51 parte del código que se ejecuta al añadir una nueva pregunta a, en este caso, un examen. Como se puede ver, lo que el *plguin* hace realmente es clonar la pregunta anterior y modificar sus identificadores, así como reiniciar sus datos.

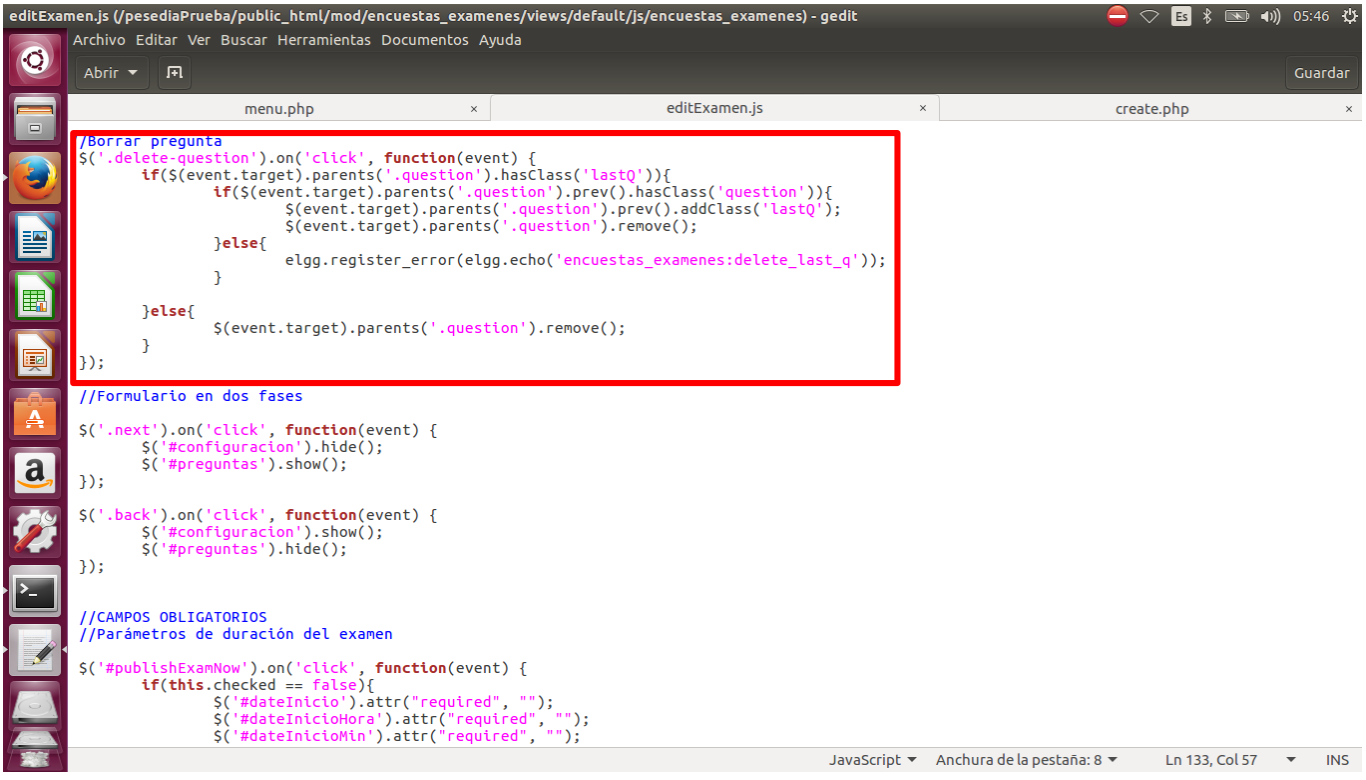

Figura 52: archivo /views/default/js/encuestas_examenes/editExamen.js 2

He aquí la parte del código encargada de borrar una pregunta durante la creación del examen (o encuesta), otra de las partes de la dinamización del formulario de creación del examen (o encuesta).

- ***river/object*** – Elgg denomina *river* al flujo de contenido que se muestra en el muro de un usuario formado por publicaciones y novedades del resto de usuarios. En el caso del *plugin* de encuestas y exámenes, cada vez que un usuario cree nuevo contenido (y siempre que el que lo visualice tenga permisos para ello), aparecerá notificado en el *river*. Para ello existen dos archivos diferentes en el *plugin*:
    - *examen/create*.php: Notifica mediante el *river* que un nuevo examen ha sido creado, muestra su descripción e indica el autor.

    - *encuesta/create*.php: Al igual que el archivo anterior, pero notifica cuando una nueva encuesta es creada.

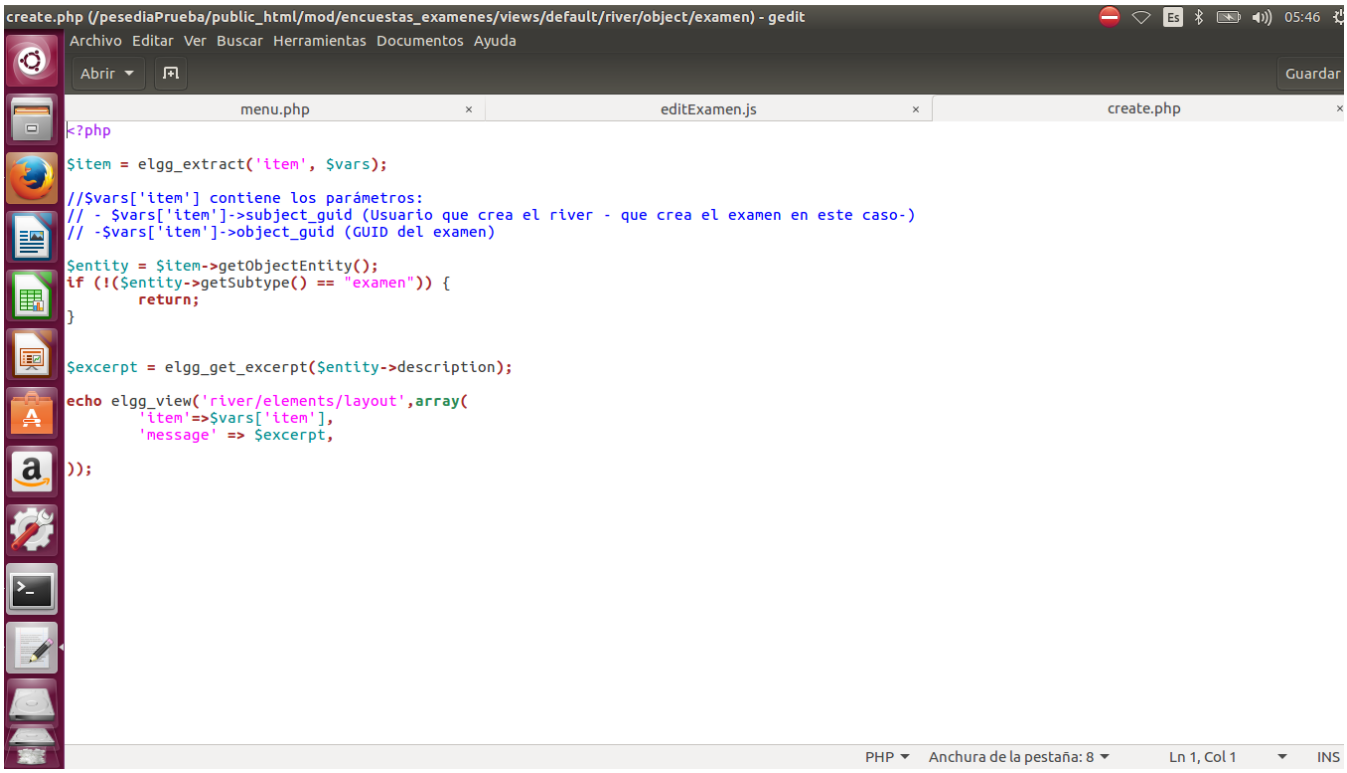

Figura 53: archivo /views/default/river/object/examen/create.php

- ***mycss.css:*** Este archivo CSS no está contenido en ninguno de los directorios anteriores, sino que está en *views/default* directamente. Contiene la hoja de estilo del *plugin*.

actions/

Es en este directorio donde se sitúan todos los archivos PHP que contienen las acciones asociadas a los formularios del *plugin*. Como se ha visto anteriormente, en el archivo *start.php* ya se han registrado todas las acciones y se han asociado a su formulario correspondiente, por lo que llegados a este punto se puede profundizar acerca de qué acciones se han implementado para el *plugin* de encuestas y exámenes, y para qué sirven.

Cabe destacar en primer lugar que estas acciones se desencadenan cuando se ejecuta un botón del tipo "*submit*" a través del código PHP, HTML o Javascript, así como también se destaca que estos archivos no generan ninguna vista en el *plugin* más allá que un mensaje informativo o un mensaje de error.

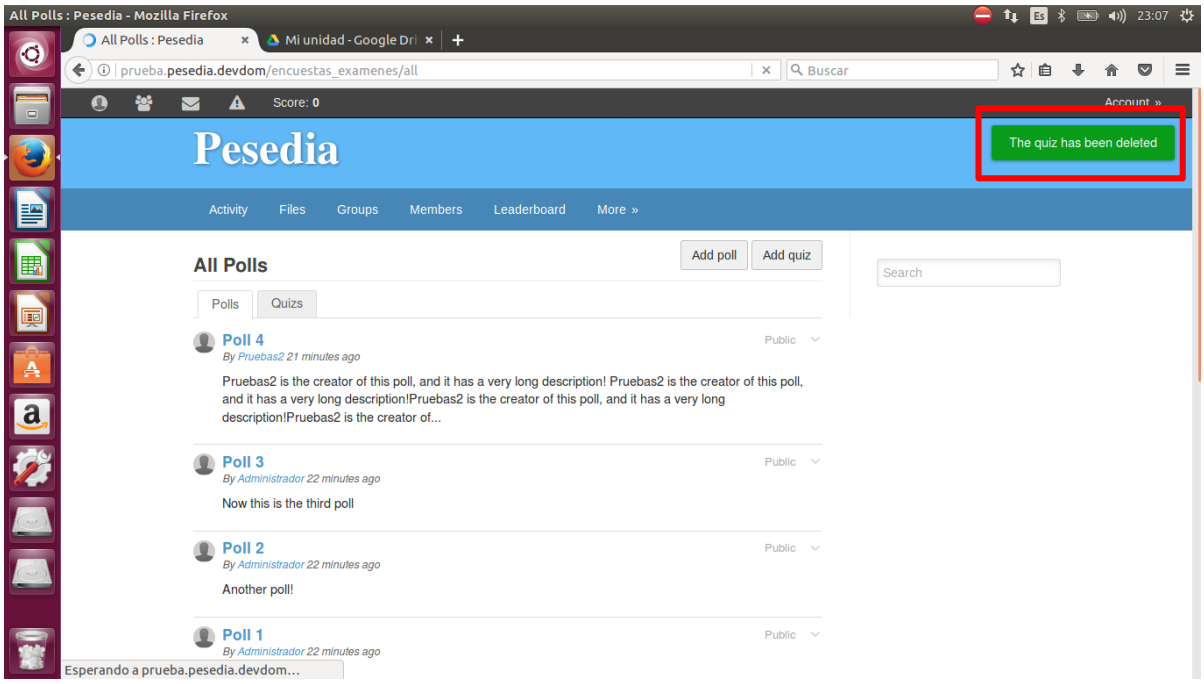

Figura 54: Mensaje tras eliminar un examen correctamente

Hay seis acciones en el *plugin* de encuestas y exámenes, y son:

- *saveEncuesta.php* y *saveExamen.php* – Cuando un usuario crea una encuesta o examen esta acción es la encargada de crear las entidades del tipo *encuesta* y *resultadosEncuesta* (para las encuestas) y *examen* y *resultadosExamen* (en caso de los exámenes).

- *vote.php* y *send.php* – Estas acciones son las encargadas de crear los objetos *voto* y *respuesta*, donde se guardan los resultados de una encuesta o examen realizada por un usuario. El código de esta acción también elimina de la base de datos respuestas anteriores para el mismo examen que haya enviado el mismo usuario.

- *editEncuesta.php* y *editExamen.php* – Al editar una encuesta o examen pueden ocurrir dos cosas que afecten al comportamiento de estas acciones. En primer lugar, es posible que únicamente se editen las opciones de configuración, o la configuración referente a cada pregunta del examen, en este caso, estos cambios se guardarán y los resultados enviados por los usuarios hasta el momento quedarán intactos. En cambio, si se editan las respuestas a las preguntas (se añaden, cambian o eliminan respuestas) el examen se reiniciará; es decir, se eliminarán todas las respuestas enviadas por los usuarios hasta el momento, y se reiniciarán los objetos *resultadosEncuesta* y *resultadosExamen*.

Se muestran a continuación algunas capturas de fragmentos de código del archivo *saveExamen.php*, donde se pretende mostrar cuáles han sido las pautas de implementación a la hora de crear entidades:

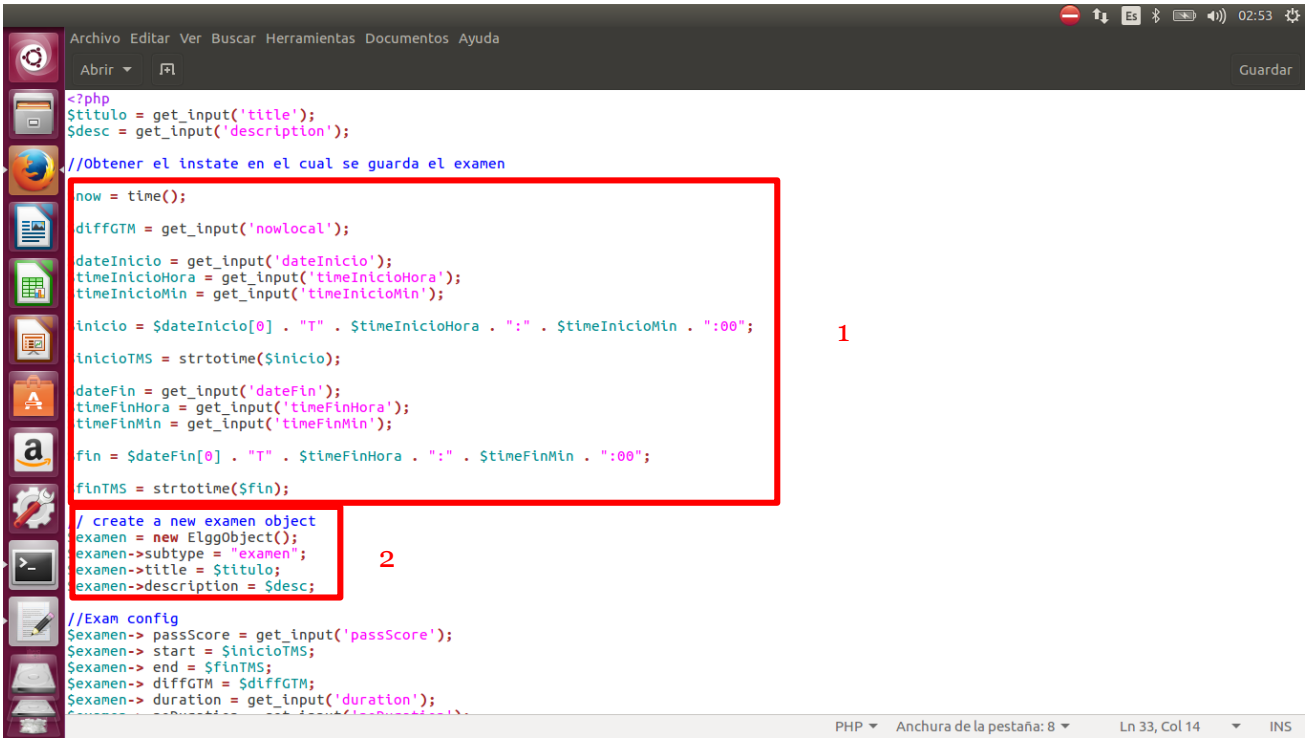

Figura 55: archivo /actions/saveExamen.php 1

1- En primer lugar podemos ver cómo está remarcado en color rojo un fragmento de código cuya finalidad es añadir las fechas de apertura y cierre del examen en un formato adecuado para ser manipulado por Elgg a los objetos que vamos a crear. Trabajar con fechas y horas no es algo sencillo, y en este proyecto se ha abordado de la siguiente manera: Si bien el código ejecutado en el lado del servidor (todo el código PHP) supone que la hora se encuentra en un formato GTM+0, a la hora de ser utilizada por cualquier usuario, esta se adapta a la hora local de su dispositivo a través de código Javascript. La implicación de esto es a la hora de que un usuario cree un examen en su formato de hora local, (por ejemplo, la zona

horaria de España es GTM+2) la hora de creación guardada en el servidor estará en el formato GTM+0.

2- También se puede ver en el código cómo se crea el objeto y se le añade el subtipo *examen*, así como se añaden también las dos primeras propiedades del examen, el título y la descripción, que han sido recogidos previamente a través de la función *get_input($html_name)*, la cual nos permite recuperar del HTML generado por el formulario cualquier elemento a través de su atributo "*name*".
Recuperar las respuestas requiere de algo más de complejidad, ya que estas son personalizadas por el creador de cada examen, y cada una de ella tiene a su vez múltiples propiedades que son guardadas en un mapa ordenado diferente para las entidades con subtipo *examen* y *resultadosExamen*.

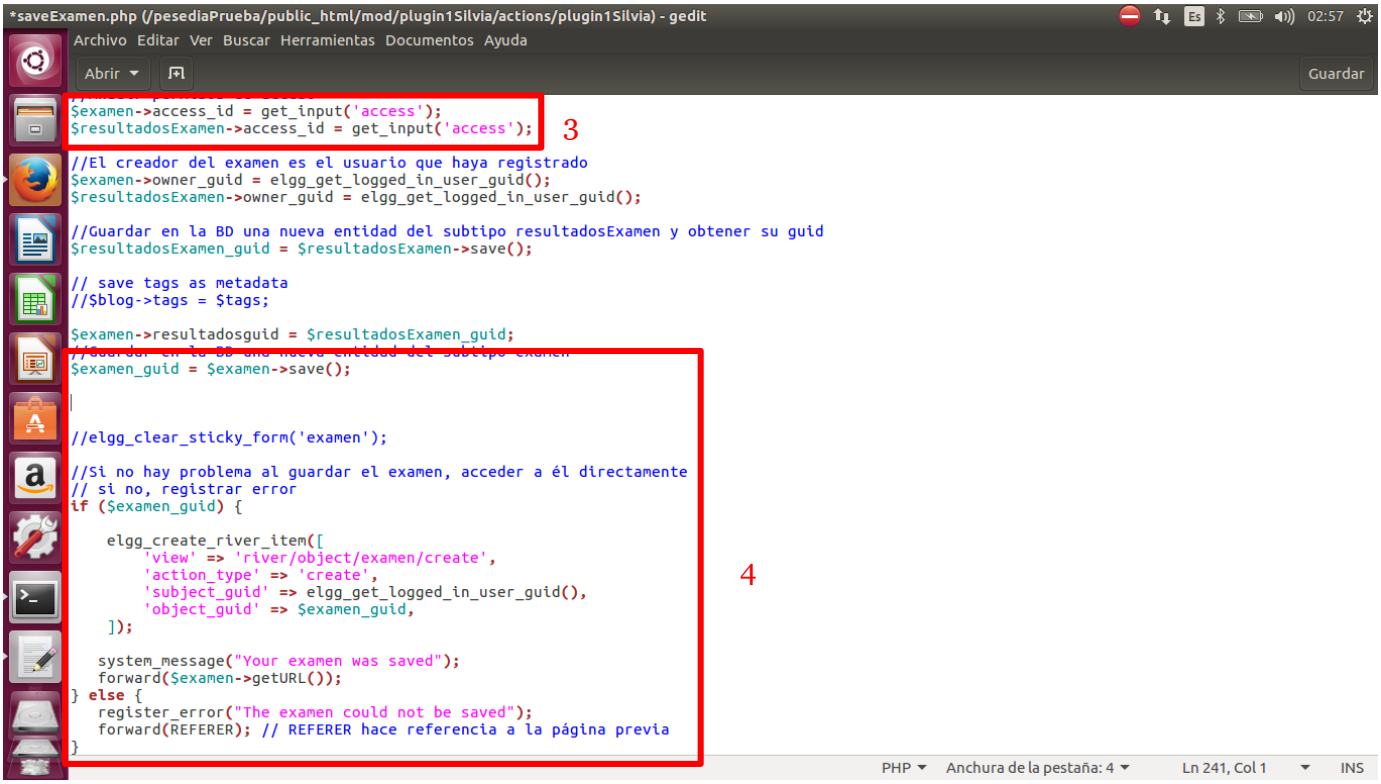

Figura 56: archivo /actions/saveExamen.php 2

3- Durante la configuración de la encuesta o examen el creador de esta puede elegir quién podrá verla. Para ello, Elgg cuenta con un tipo de input especial llamado *input/access*, el cual permite elegir al usuario entre las siguientes opciones:

- Amigos (el examen/encuesta solo estará disponible para usuarios amigos del creador)
- Privado (solo el creador puede ver el examen/encuesta)
- Público (todos los usuarios, tanto registrados como no registrados pueden ver el examen/encuesta)
- Usuarios registrados (todos los usuarios registrados pueden ver el examen/enceusta)

En este caso, el *input/access* se encuentra en el formulario *saveExamen* y tiene el "*name= access*" en el HTML, por lo que al asociarlo al *access_id* del objeto, queda definido así cuáles serán los permisos de acceso a este.

4- A través de la función *save()* el objeto queda guardado en la base de datos. En caso de que haya habido algún error, se mostrará por pantalla el mensaje "El examen no se guardó correctamente" y habrá una redirección a la página anterior, y en caso de que todo funcione correctamente, se mostrará el mensaje "El examen ha sido guardado correctamente" y se redireccionará al propio examen. También se añadirá al muro de Pesedia (llamado por Elgg *river*) la publicación de que un nuevo examen ha sido creado mediante la función *elgg_create_river_item*.

languages/

Este directorio contiene las traducciones del *plugin*. Cada archivo que contiene el directorio es para un lenguaje diferente, y sus nombres siguen la norma ISO 639-1. Por el momento el *plugin* está disponible en español e inglés, por lo que el directorio contiene los archivos:

- *es*.php
- *en*.php

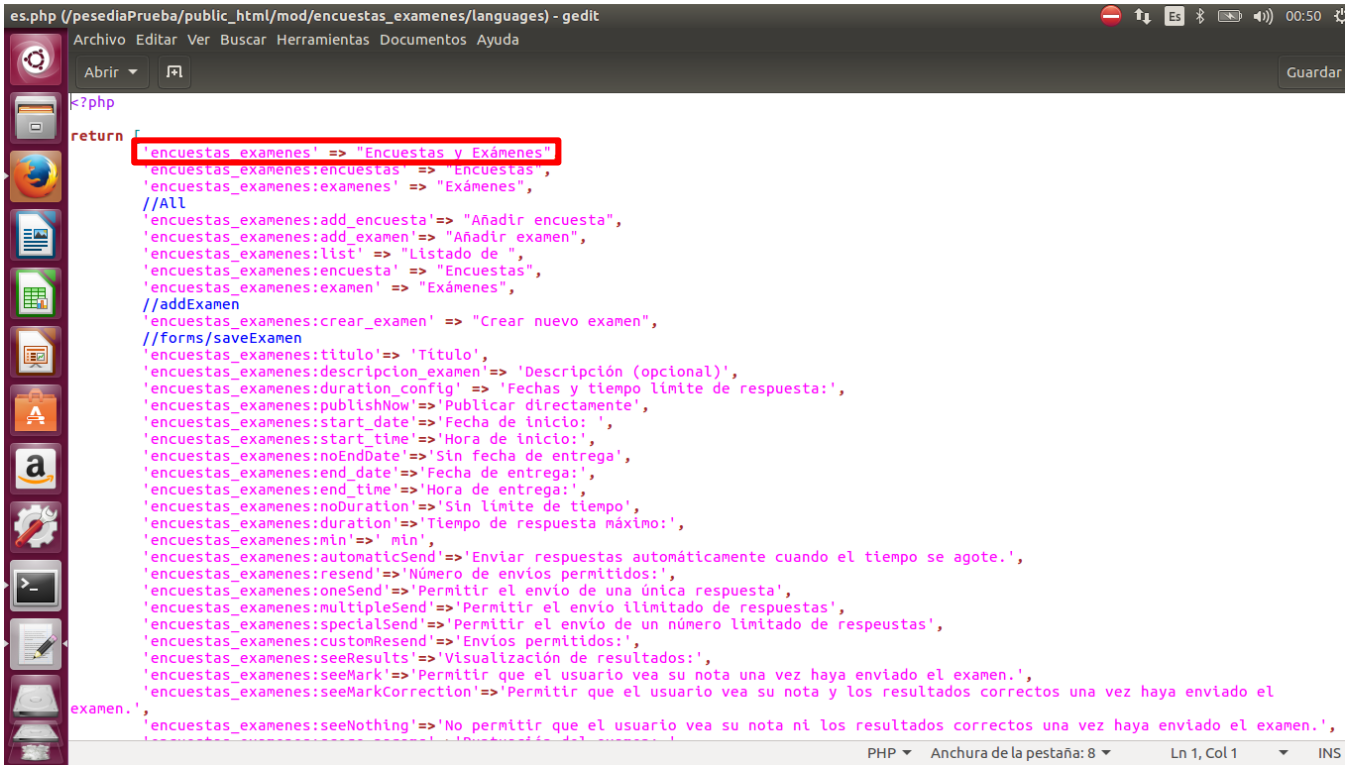

Figura 57: archivo /languages/es.php

Como se puede ver, las traducciones están formadas por un par "*clave* => *valor*". La clave se encuentra en el código del *plugin*, a través de la función *elgg_echo("encuestas_examenes:clave")*. El nombre del *plugin* antes de la clave se escribe para no sobrescribir ningún valor del *core* de Elgg, o de otros *plugins*. El valor es lo que se muestra al usuario, y es lo que cambia de idioma según el archivo .php

7. Integración del plugin

7.1 Instalación y despliegue de Elgg

Cabe destacar en primer lugar que durante todo el proyecto se ha utilizado el sistema operativo Ubuntu 16.04 como servidor debido a la capacidad que este ofrece para realizar una instalación personalizada tanto de Elgg como del resto de herramientas que se precisan para su funcionamiento: servidor local, base de datos, PHP, editores de texto, consola, etc.

Los pasos que se han seguido para poder realizar la instalación de Elgg han sido los que se mencionan a continuación:

Desde el terminal Linux:

PASO 1: Instalar dependencias

1. Para poder instalar los distintos requerimientos de Elgg se va a hacer uso de la herramienta de gestión de paquetes apt, por lo que en primer lugar se ha procedido a actualizar el repositorio:

```
$ sudo apt-get update -y
$ sudo apt-get upgrade –y
```

PASO 2: Infraestructura LAMP

A continuación se pretende recrear la infraestructura LAMP, que consiste en la combinación de las siguientes herramientas: el sistema operativo Linux (el cual se ha instalado en primer lugar), el servidor web Apache, el gestor de base de datos MySQL y PHP.

1. Instalación de Apache2

```
$ sudo apt-get install -y apache2
```

2. Instalación del SGBD mySQL

```
$ sudo apt-get install -y mysql-client
## Se ha de crear aquí el usuario root y asignarle una
contraseña que luego deberá emplearse en la instalación de
phpMyAdmin
$ sudo apt-get install -y mysql-server
```

3. Instalación de PHP nativo

```
$ sudo apt-get install php
```

4. Instalación del software phpMyAdmin para un manejo simplificado y cómodo de la base de datos.

```
$ sudo apt-get install -y phpmyadmin
```

5. Instalación de PHP 5.6

```
$ sudo apt-get install -y php5.6 php5.6 -mysql php5.6 -gd php5.6
-imap php5.6 -ldap php5.6 -odbc php5.6 -xml php5.6 -xmlrpc
php5.6 -curl php5.6 -mbstring php5.6 -mcrypt
```

PASO 3: Configurar el sistema

Ya tenemos instaladas todas las herramientas necesarias para el funcionamiento de Elgg, así pues, el siguiente paso es configurarlas.

1. Creación de la base de datos mySQL, donde *$db_passwd* y *$db_name* son la contraseña y el nombre de la base de datos respectivamente.

```
$ mysql -u root -p$db_passwd -e "CREATE DATABASE $db_name;"
$ mysql -u root -p$db_passwd -e "FLUSH PRIVILEGES;"
$ sudo service mysql restart
```

2. Configuración de Apache2

```
## Crear el documento 'pesedia.conf' en el directorio
'/etc/apache2/sites-available/'  ->
## Añadir el sitio pesedia.devdom a ‘etc/hosts’
$ a2ensite pesedia.conf
```

```
$ service apache2 reload
```

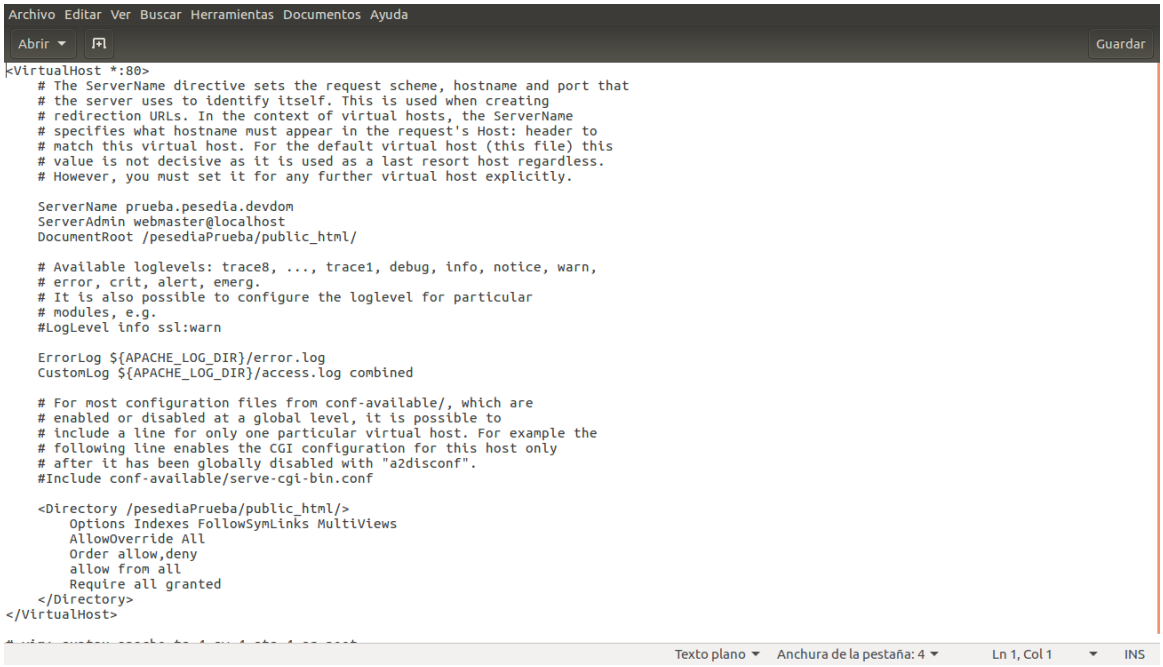

Figura 58: archivo etc/apache2/sites-available

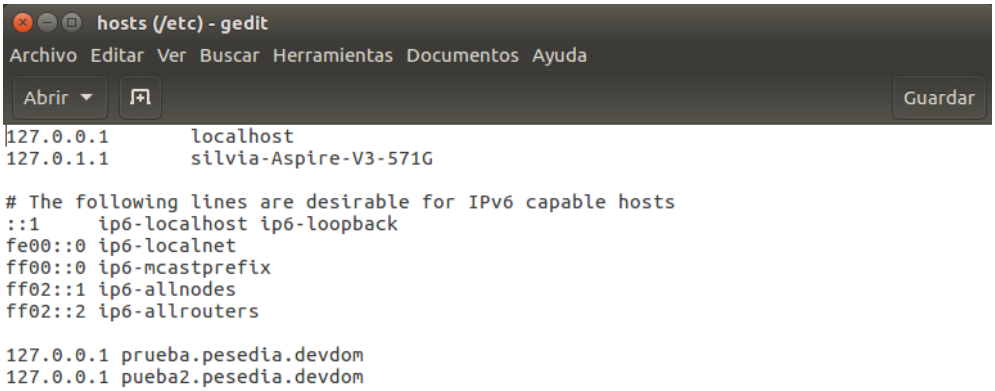

Figura 59: archivo etc/hosts

PASO 4: Instalación de Elgg

Ya tenemos todo preparado para la instalación de Elgg, por lo que es ahora cuando podemos descargar Elgg desde nuestro navegador e instalarlo finalmente en el servidor. En este caso se ha trabajado con la versión 2.3 de Elgg, ya que es la más reciente.

1. Descargar Elgg

```
$ mkdir /pesedia
$ cd /pesedia
#Tras haberlo descargado en la carpeta Downloads desde el
navegador previamente
$ unzip –q ~/silvia/Downloads/elgg-2.3.zip
```

2. Crear la estructura de directorios recomendad por Elgg y asignar los permisos correspondientes:
   - */pesedia/public_html/*, directorio donde estará almacenada la instalación de Elgg, toda la lógica y vistas de la red social.
   - */pesedia/pdata/*, directorio donde la herramienta Elgg almacenará datos de la red social que no pueden ser almacenados en la base de datos (iconos de perfil, imágenes, documentos, etc.). Por motivos de seguridad, esta carpeta debe encontrarse fuera de *public_html* de manera obligatoria.

```
$mv elgg-2.3/ public_html
$mkdir pdata
$chown –R www-data:www-data .
$chmod –R ug+rw .
$chmod –R o-rwx .
```

PASO 5. Configuración de Pesedia

Ya se puede acceder desde nuestro navegador a la URL prueba.pesedia.devdom (que es sobre la que se ha estado trabajando) y configurar Elgg asignándole tanto la base de datos como la dirección del directorio en el que se guardarán los archivos que no serán almacenados en la base de datos (directorio pdata).

Por último, poder trabajar en un entorno idéntico al de un sitio Pesedia, tras la instalación se ha procedido a instalar, activar y/o desactivar los *plugins* que utiliza Pesedia.

8. Pruebas

A lo largo del desarrollo del proyecto se han llevado a cabo diferentes tipos de prueba para comprobar en todo momento que se ha estado trabajando sobre algo consistente y minimizar los cambios de última hora. Se podría decir que se ha llevado a cabo un diseño y evaluación iterativa del proyecto, con el fin de poder identificar lo antes posible si el usuario tendrá algún tipo de dificultad a la hora de utilizar el *plugin* y también posibles errores de implementación.

8.1 Prueba de usabilidad del prototipo

La finalidad de evaluar el prototipo realizado con la herramienta JustInMind, cuyas capturas se pueden ver en la fase de Diseño ha sido, principalmente, determinar si un usuario sin conocimiento específico sobre informática era capaz de realizar cada una de las tareas detallada en los casos de uso. Si bien para la mayoría de las tareas el usuario pudo llevar a cabo una simulación correcta de los pasos a seguir en el *plugin*, opciones como editar, eliminar y reiniciar la encuesta no fueron encontradas fácilmente, así que se decidió añadir un menú de opciones específico para cada examen al acceder a él (tal y como se puede ver en la figura 22). Tras esta nueva interfaz y llevar a cabo la misma prueba con un nuevo usuario (de nuevo sin conocimientos informáticos avanzados) sí que se pudo evaluar como satisfactoria la evaluación del prototipo.

8.2 Prueba de usabilidad del *plugin*

Una vez la implementación del *plugin* se hubo considerado finalizada se ha realizado una prueba de usabilidad con un usuario creador de contenido que aunque (al igual que con el prototipo) no gozaba de conocimientos técnicos de informática sí que se dedica a la docencia. Si bien todas las tareas especificadas en los casos de uso se llevaron a cabo de manera efectiva; es decir, el usuario pudo realizar las tareas de manera autónoma y en un tiempo razonable, sí que se descubrieron varios fallos que podrían ser considerados“de seguridad” referidos al número de envíos de una encuesta y examen, y se plantearon las siguientes preguntas:

- ¿Ha de considerarse “envío” el acceso a la encuesta o examen?
  La respuesta a esta pregunta, considerando el punto de vista de un profesor, que es al fin y al cabo el tipo de persona al que va dirigido el *plugin* fue un sí, y es que, de no ser considerado como envío el acceso a un examen, el usuario podría entrar a él tantas veces como guste, burlando así el número máximo de envíos. A raíz de esto, también surgió la siguiente pregunta.

- ¿Ha de considerarse refrescar la página un envío del examen?
  La respuesta, al igual que con la pregunta anterior, también ha sido sí, y es que en caso contrario se podría burlar no solo el problema planteado en la pregunta anterior, sino también que el usuario tenga la capacidad de reiniciar los tiempos límites del examen.

Así pues, se ha incluido en la interfaz un aviso indicando que tanto el acceso a la encuesta o examen como refrescar la página durante la encuesta o examen restan un envío al límite de envíos permitidos. Depende de cómo el creador lo haya configurado, los resultados introducidos por el momento se enviarán al servidor automáticamente o se perderán.

8.3 Pruebas de funcionalidad

En la fase de Análisis se han descrito los requisitos funcionales que ha de cumplir el *plugin*. Con estas pruebas, se pretende comprobar que efectivamente estos requisitos se cumplen correctamente.

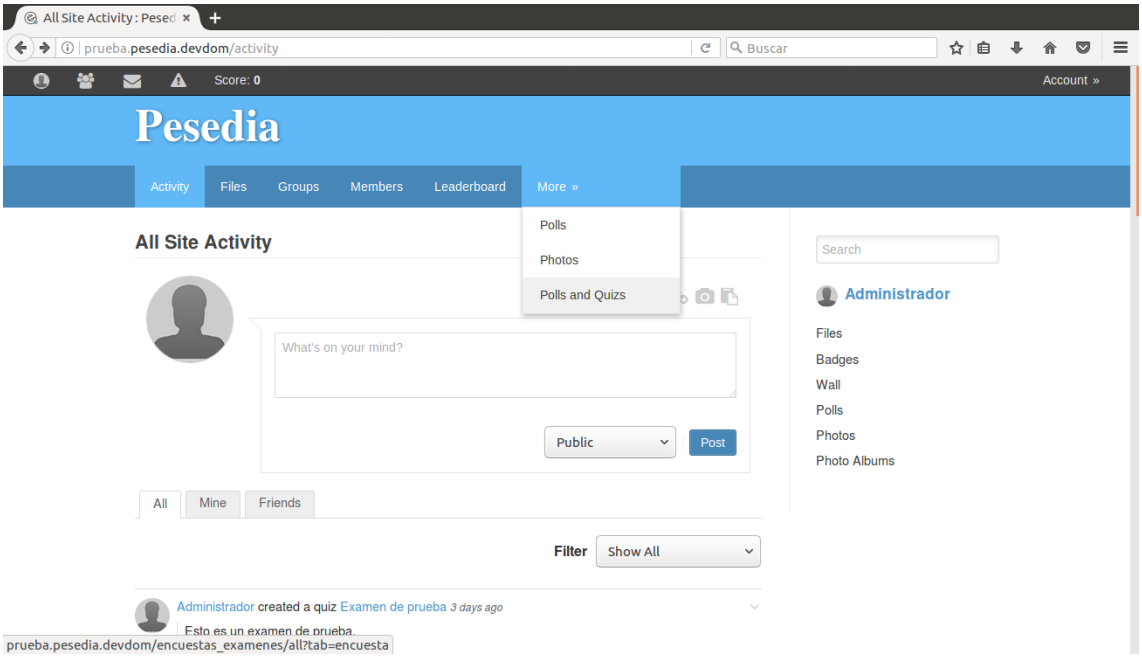

Figura 60: Acceder al plugin Encuestas y Exámenes

A través de esta primera captura podemos ver cómo se accede al *plugin* desde el menú principal de Elgg, cumpliendo así el FR-1.

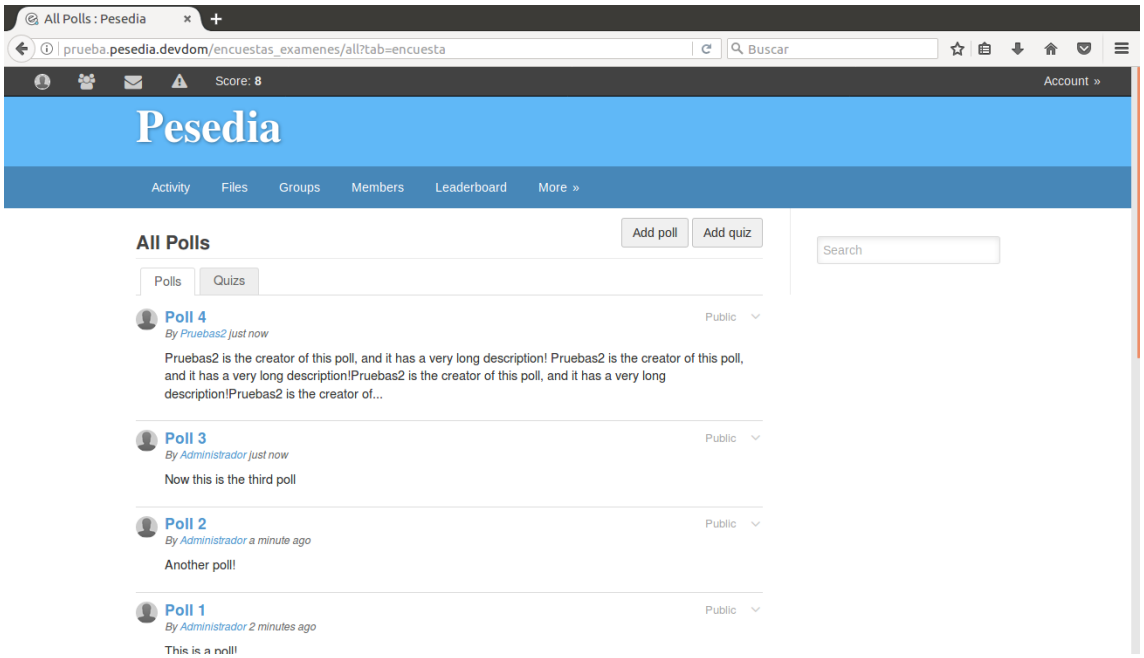

Figura 61: Página principal del plugin

En esta captura podemos ver el listado de encuestas (ya que es la pestaña de encuestas la que está seleccionada). Se puede apreciar que hay dos usuarios creadores de encuestas en la captura: Administrador y Pruebas2, y que todas las encuestas son públicas. La pestaña de exámenes funciona de igual manera, cumpliendo así el FR-2.

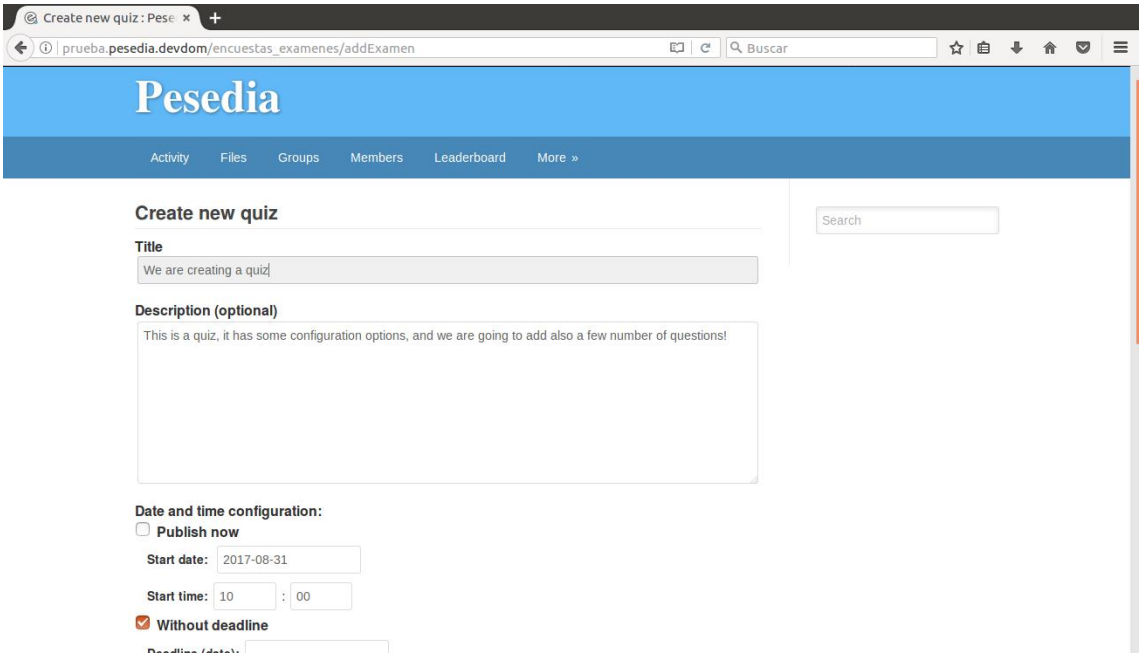

Figura 62: Crear examen 1

El FR-3 trata de añadir una encesta o examen. En la evaluación el usuario de prueba ha podido añadir ambos sin problema. La captura muestra la creación de un examen cuya

fecha de publicación es el día 31 de agosto del año 2017, a las 10:00 de la mañana y que no tiene fecha límite de envío (FR-4).

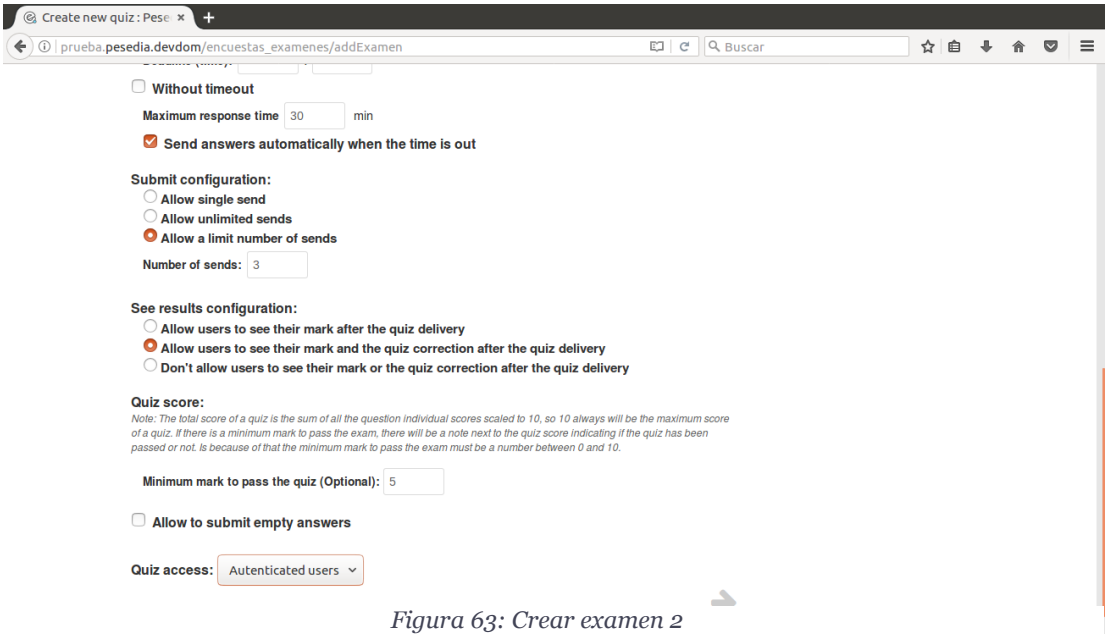

Figura 63: Crear examen 2

Continuando con las opciones de configuración del examen, esta captura muestra que el tiempo límite para realizar el examen serán treinta minutos, y que las respuestas se enviarán de manera automática al servidor en caso de que el tiempo expire (FR-5). También se puede ver cómo se ha puesto un número máximo de tres envíos del examen (FR-6), y que se permitirá ver al usuario tanto la corrección de este como su nota obtenida (FR-17).

Por último en lo referente a la configuración de, la nota mínima para aprobar será de 5 puntos sobre 10 (FR-9), y el examen estará disponible únicamente a usuarios autenticados (FR-18), no permitiéndose el envío de exámenes vacíos (FR-12).

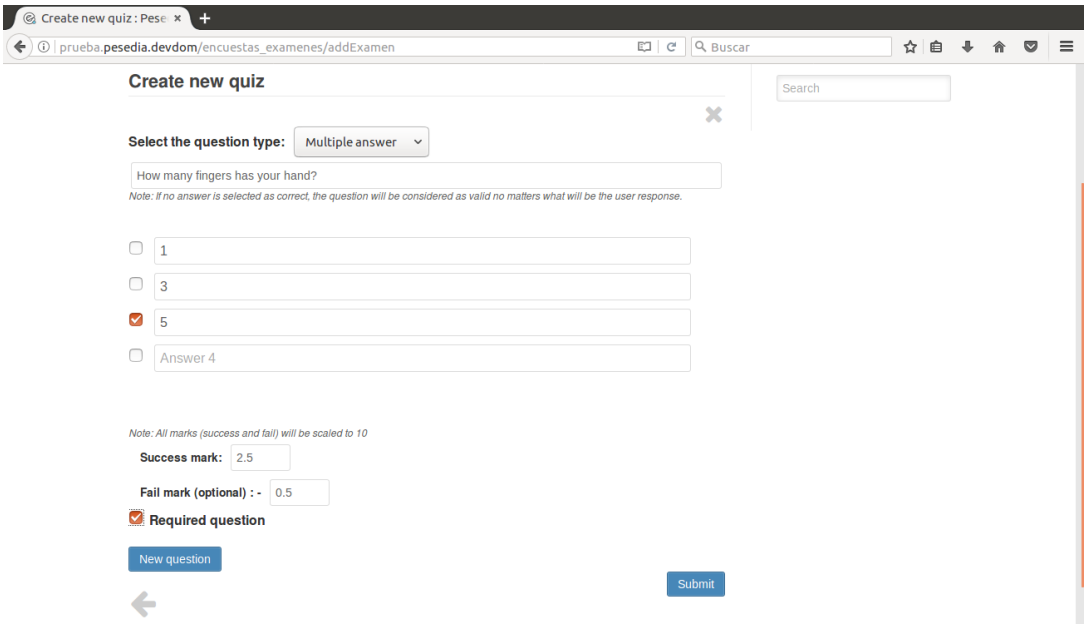

Figura 64: Crear examen 3

Procediendo ahora a la creación de preguntas, en esta captura se muestra una de ellas (aunque en las pruebas realizadas se han hecho exámenes y encuestas con preguntas de todo tipo, intercaladas entre sí) del tipo respuesta múltiple, la cual ha sido marcada como pregunta obligatoria para poder enviar el examen (FR-7) y se le ha puesto de un peso de 2.5 puntos en caso de acierto y 0.5 puntos en caso de fallo (FR-10). Puesto que en estos momentos solo hay una pregunta en el examen, estas puntuaciones serán escaladas a 10 (FR-8).

Para poder añadir más respuestas a la pregunta, el usuario ha de situarse en el último cuadro de texto de la respuesta, y uno nuevo se genera automáticamente (FR-11).

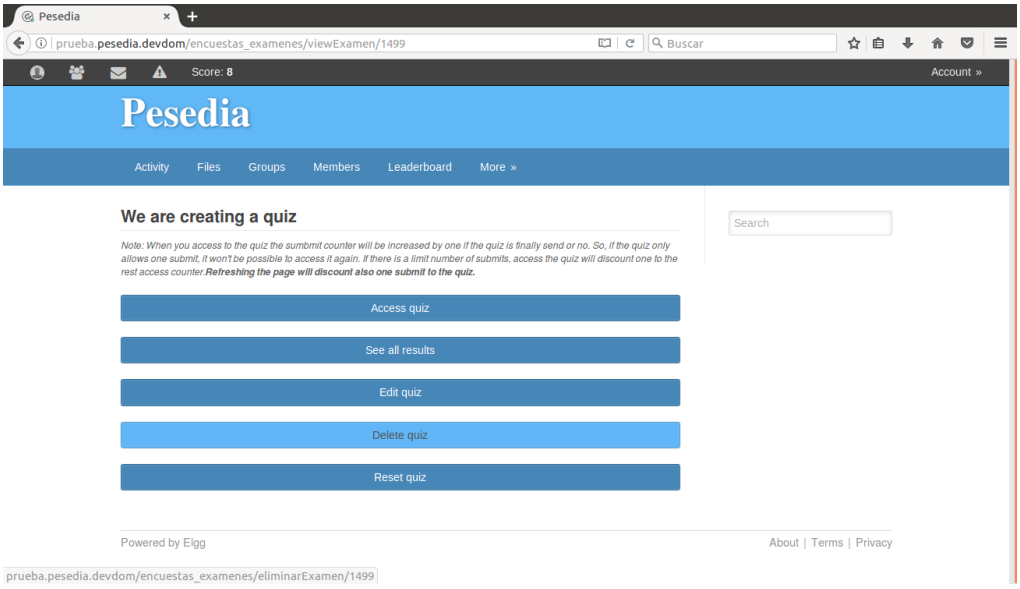

Figura 65: Vista general del examen para el creador

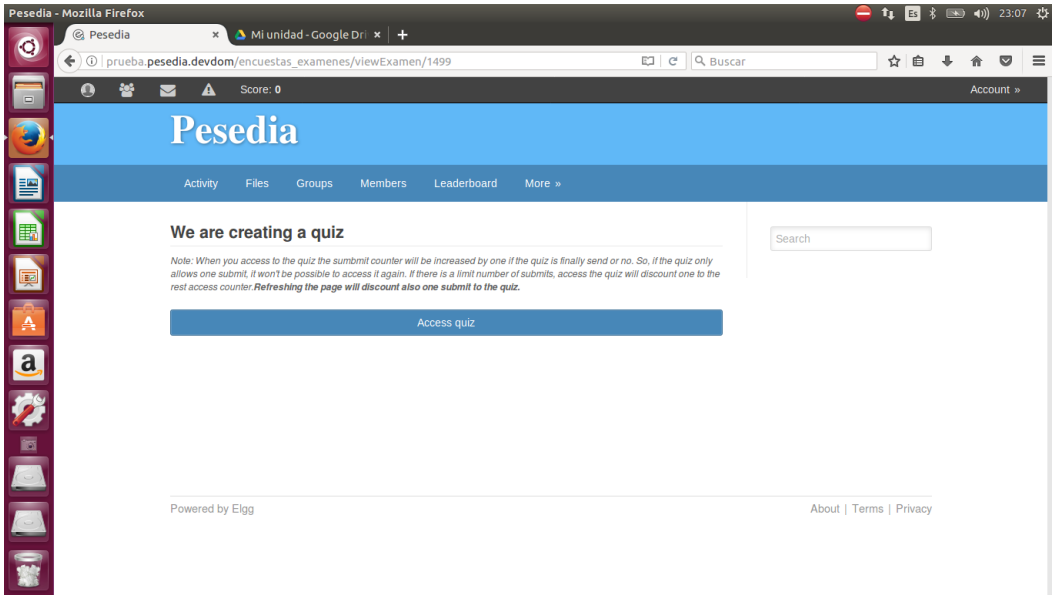

Figura 66: Vista general del examen para el usuario no creador

Una vez el examen está publicado, el creador del mismo podrá acceder a todas las opciones disponibles tal y como se puede ver en la figura 65, en cambio, cualquier otro usurario únicamente podrá entrar a responder el examen, como muestra la figura 66.

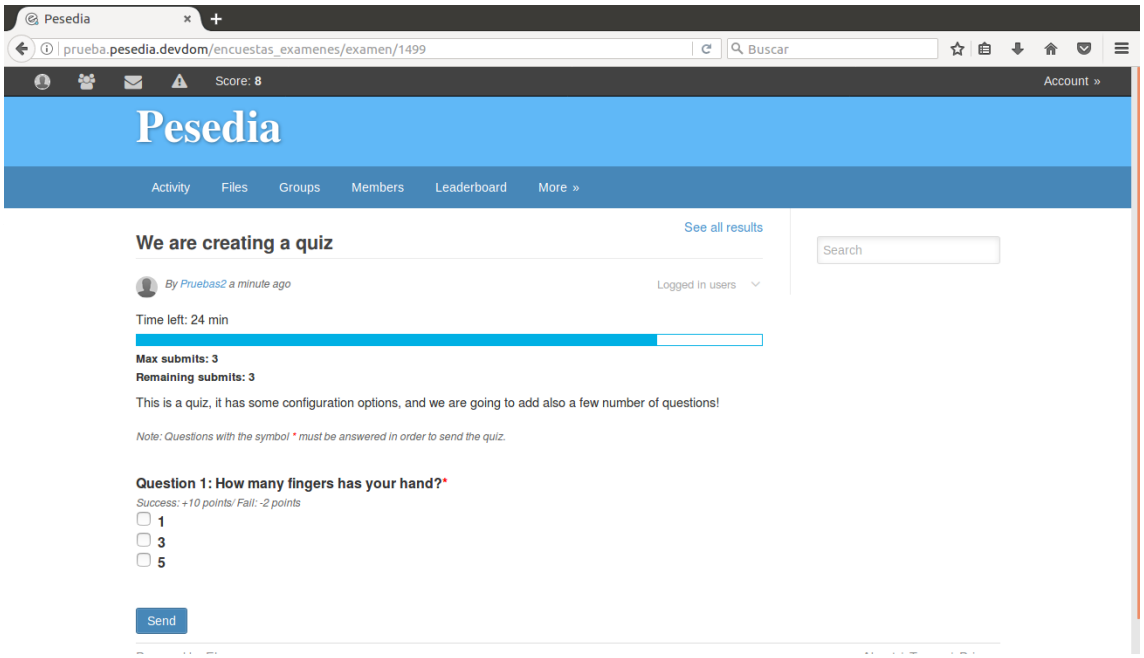

Figura 67: Responder examen

Tras enviar el examen el usuario podrá ver lo que ha enviado, así como la nota obtenida, ya que se ha configurado que se mostrará tanto la nota como el resultado.

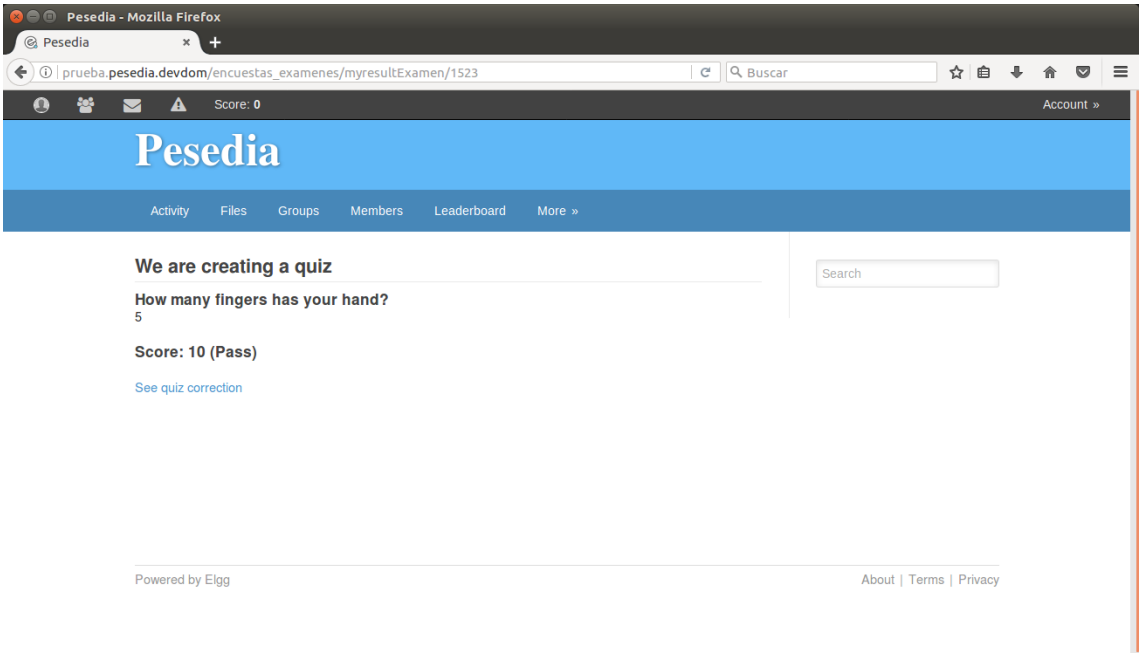

Figura 68: Mi resultado

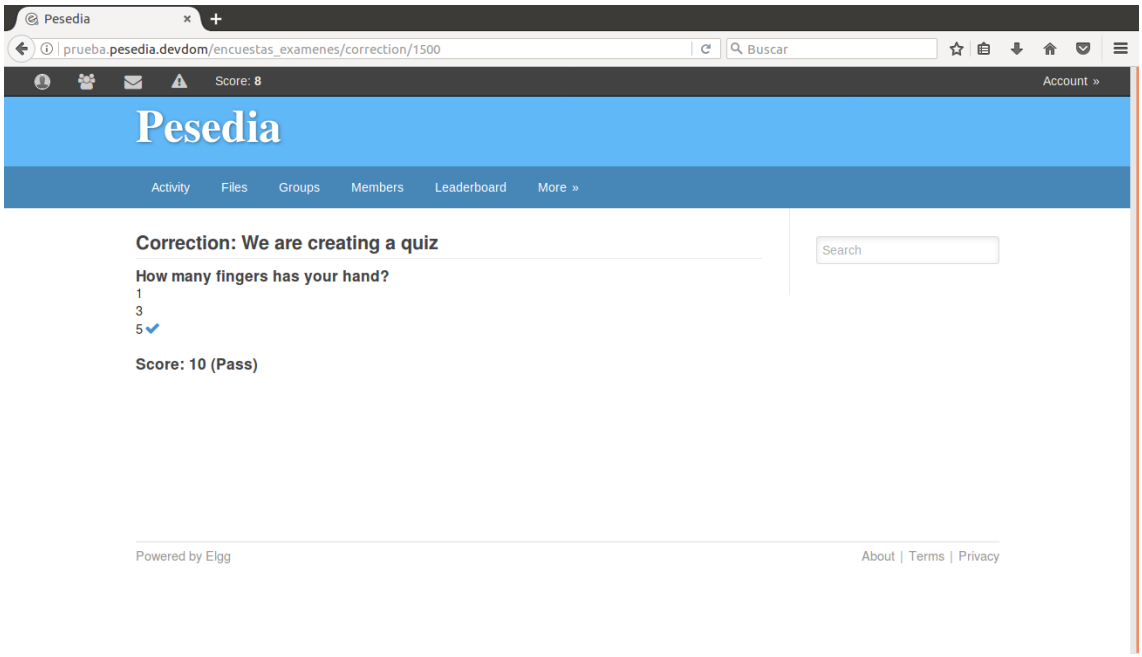

Figura 69: Ver corrección

En la figura 70 se muestra un examen que todavía no está disponible, puesto que su fecha de activación no ha llegado todavía, y por lo tanto, nadie puede ni verlo, ni enviar una respuesta aún.

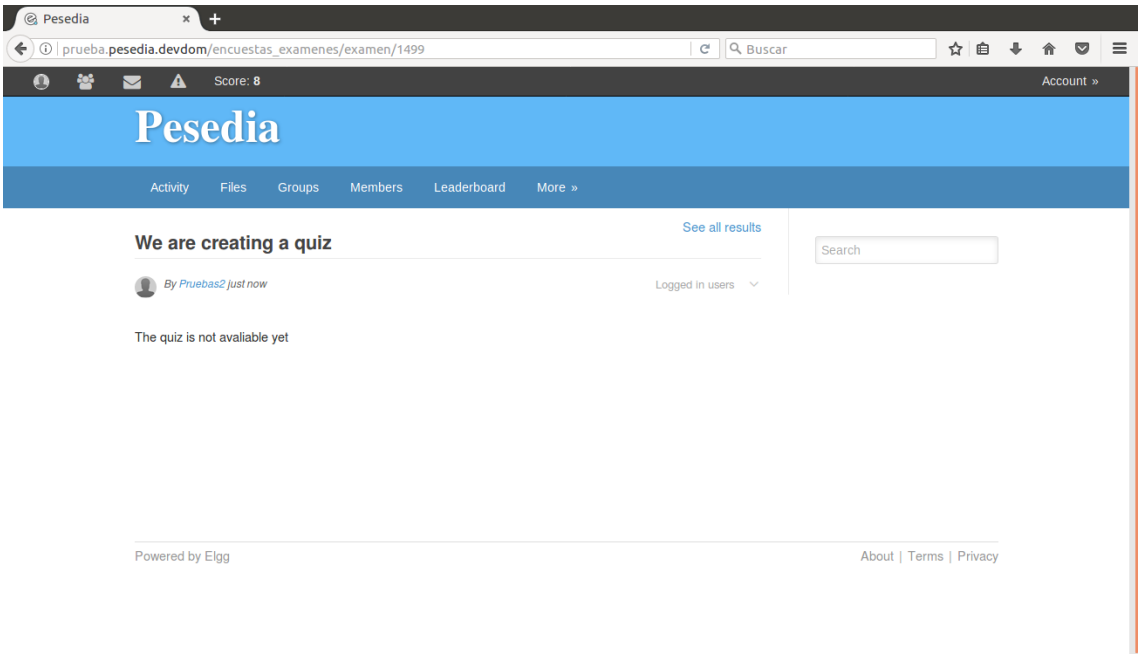

Figura 70: Examen no disponible

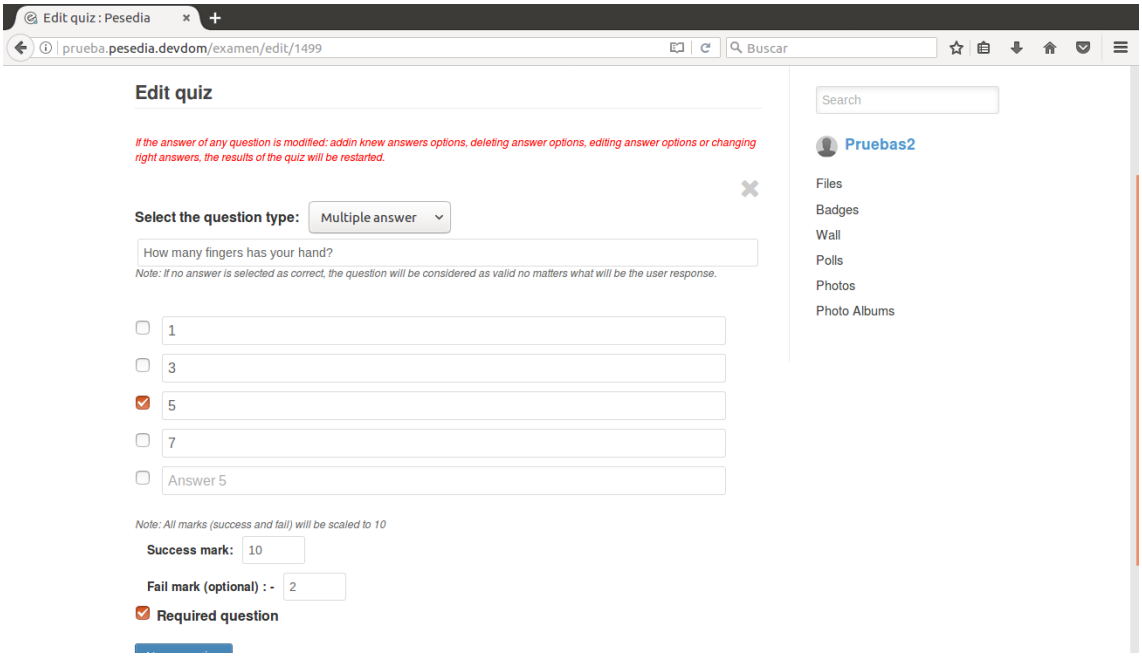

Figura 71: Editar examen

Como se puede ver al editar un examen, la interfaz es muy similar a la que aparece al crear un nuevo examen, no obstante, esta vez hay un aviso que indica que si se modifica alguna opción de las respuestas de una pregunta o si se añaden nuevas preguntas, el examen se reiniciará (con las encuestas ocurre igual), cumpliendo así el FR-13.

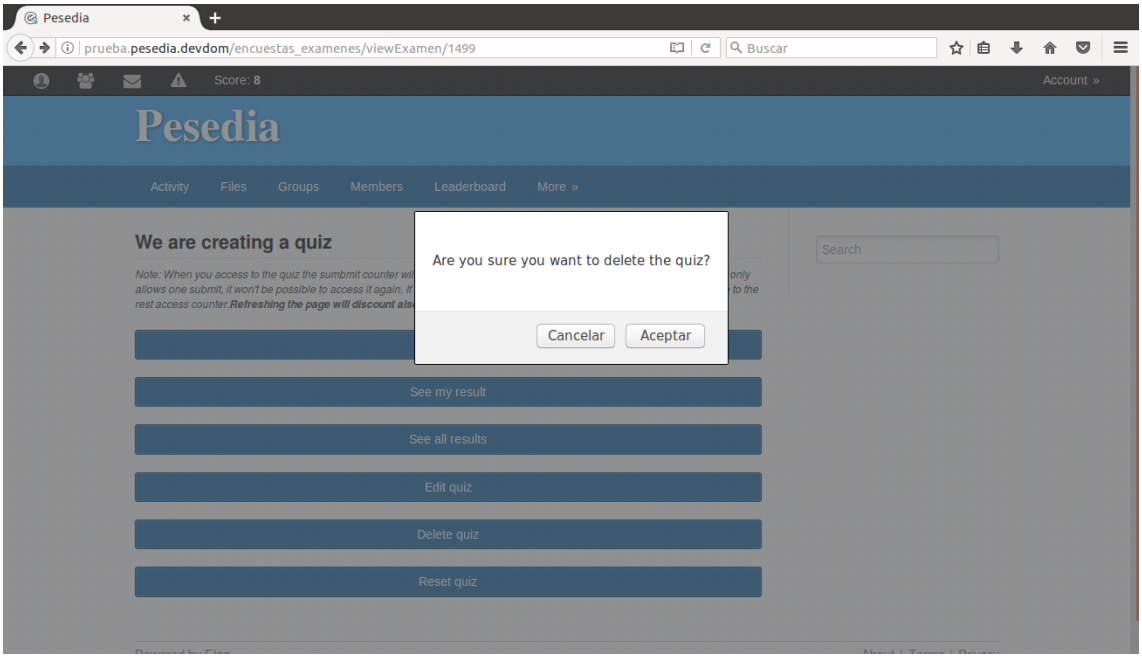

Figura 72: Eliminar examen

Eliminar y reiniciar un examen son dos tareas que el creador de la encuesta o examen puede realizar cuando desee, sin importar si la fecha límite del examen ha llegado o si se encuentra todavía en curso (FR-14 y FR-15).

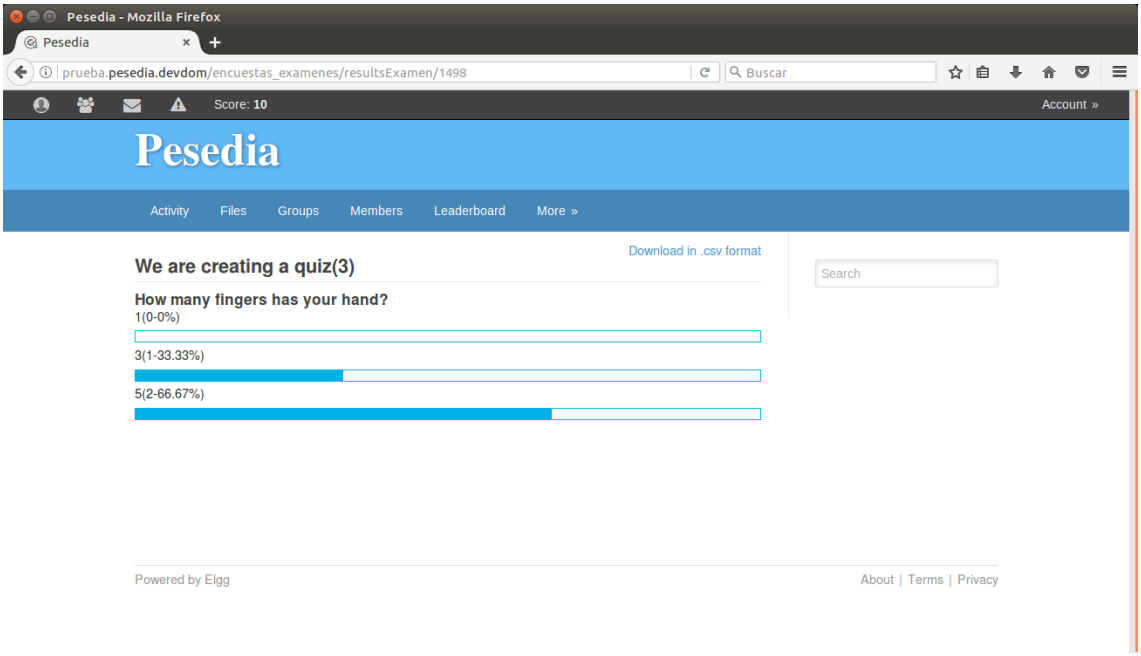

Figura 73: Ver resultados

Por último, la figura 73 muestra la interfaz en la que se pueden ver todos los resultados obtenidos hasta el momento, así como el enlace de descarga del archivo .csv con estos mismos resultados.

Todos los casos de uso se satisfacen tal y como queda estipulado en la especificación de requisitos, dándose estas pruebas de funcionalidad como válidas.

9. Conclusiones

El objetivo de este proyecto era crear un *plugin* que permitiese a los usuarios de Pesedia crear y responder tanto encuestas como exámenes. Las primeras debían de disponer no solamente de un conjunto variado de opciones de configuración que se definió en la fase de especificación de requisitos, sino también de diferentes tipos de pregunta y otras opciones de configuración específicas para las preguntas. Los exámenes debían ofrecer todo lo anterior, con la gran diferencia de que existen respuestas correctas a las preguntas, y estas se pueden puntuar.

Al principio de comenzar el proyecto se desconocían (o conocían de manera básica), los componentes y tecnologías necesarias para la implementación del *plugin* (como PHP, Javascript, jQuery, HTML5, CSS, MySQL, o Apache) y la propia plataforma Elgg, pero el estudio y aprendizaje llevado a cabo a lo largo del proyecto ha permitido adquirir todos los conocimientos necesarios para poder desarrollar el *plugin* de encuestas y exámenes, además de conseguir llevar a cabo una especificación de requisitos completa previamente.

Todos los detalles acerca de las necesidades del proyecto mentados en la especificación de requisitos son cumplidos por el *plugin*, considerándose pues válido para su futuro uso.

Trabajar con Elgg ha supuesto un verdadero reto que ha permitido descubrir una herramienta realmente potente y versátil para crear un entorno de red social. Una de las partes más decisivas el proyecto ha sido sin duda alguna trabajar correctamente con el modelo de datos de Elgg, y también una de las partes más difíciles. Poder comprender el funcionamiento interno de Elgg y la manera de que el *plugin* de encuestas y exámenes cumpliese con las expectativas y necesidades de un *plugin* de Elgg ha sido una tarea crucial para asegurar las buenas prácticas recomendadas por la documentación de la plataforma, aparte de un buen funcionamiento.

Una vez finalizado el proyecto, el *plugin* pasará al grupo de investigación Tecnología Informática - Inteligencia Artificial de la Universitat Politècnica de València, por lo que trabajar en una parte de un gran proyecto ha supuesto una gran responsabilidad que ha permitido seguir un caso real de desarrollo de *software*.

A modo de conclusión final, merece la pena mencionar que este proyecto ha permitido poner en práctica muchas de las disciplinas adquiridas como fruto de haber cursado la titulación, siendo el primer paso de muchos en el futuro profesional de la autora.

10. Bibliografía

(1) ACHOUR, M. ET AL. (1997). *PHP Manual. The PHP Group.* <http://php.net/manual/en/> [consulta: 1 de septiembre de 2017].

(2) ALEMANY BORDERA, J. (2016) PESEDIA. Red social para concienciar en privacidad. Universitat Politècnica de Valéncia.

(3) AMBLER, S.W. UML *2 Class Diagrams: An Agile Introduction. Agile Modeling.* <http://www.agilemodeling.com/artifacts/classDiagram.htm> [consulta: 1 de septiembre de 2017].

(4) ÁLVAREZ M. A. (2014) Qué es MVC. Desarrolloweb <https://desarrolloweb.com/articulos/que-es-mvc.html> [consulta: 1 de septiembre de 2017].

(5) ELGG (2013) *Elgg documentation* <http://learn.elgg.org/en/2.3/index.html> [consulta: 1 de septiembre de 2017].

(6) FACEBOOK (2016*) Facebook: About Privacy* <https://www.facebook.com/about/privacy> [consulta: 1 de septiembre de 2017].

(7) HJULSTAD, H. (2001) ISO / TC 37 / SC 2 / WG 1 "*Coding systems*". ISO / TC 37 / SC 2 / WG 1.

(8) INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (2011). *29148-2011 - Systems and software engineering -- Life cycle processes -- Requirements engineering.* <https://standards.ieee.org/findstds/standard/29148-2011.html> [consulta: 1 de septiembre de 2017].

(9) JACOBSON, I., JONSSON, P., CHRISTERSON, M. AND OVERGAARD, G. (1992) Ingeniería de Software Orientada a Objetos - Un acercamiento a través de los casos de uso. Addison Wesley Longman, Upper Saddle River, N.J.

(10) MCGEE, L. *Browsers, media players. World Wide Web Consortium* <https://www.w3.org/standards/agents/browsers> [consulta: 1 de septiembre de 2017].

(11) NIELSEN, J., Y MOLICH, R. (1990). *Heuristic evaluation of user interfaces,* Proc. ACM CHI'90 Conf. *(Seattle, WA, 1-5 April), 249-256.*

(12) PÉREZ PORTO, J Y MERINO, M. (2015) Definicion.de: Definición de *plugin* <http://definicion.de/plugin/> [consulta: 1 de septiembre de 2017].

(13) RIEHLE, D. (2000), *Framework Design: A Role Modeling Approach, Swiss Federal Institute of Technology.*

(14) SCHILDT, H. Java, Manual de referencia (7 edición). Mc Graw Hill. p. 105. ISBN 0-07-226385-7.