Trabajo final de carrera

Aplicaciones web para trabajo colaborativo

Aplicación web para la corrección automática de pruebas

Memoria

Alumno: Lidia San Emeterio Arroyo
Consultor: Ferran Prados Carrasco
Curso: 2012/2013-1

Índice de contenido

1.- Introducción, motivaciones, propósito y objetivos del proyecto

La elección del área "Aplicaciones web para trabajo colaborativo" para la realización de este trabajó se basó en la experiencia, tanto personal como profesional, en este área. Entre todas las propuestas se seleccionó la "Aplicación web para la corrección automática de pruebas" por una motivación personal surgida de una de esas ideas que los informáticos suelen tener en sus raros (y escasos) momentos ociosos. Uno de esos momentos de inspiración en los que la exposición y planteamiento mental de una idea brillante, en la que se van dibujado esquemas de bases de datos totalmente normalizados, diagramas de clases e interfaces de usuario sorprendentes concluye con un "algún día que tenga tiempo lo haré".

La idea original surgió unos diez años atrás preparando el examen teórico de conducir y practicando con tests cuya corrección era puramente manual (dando lugar en alguna que otra ocasión a errores) o, gracias a la Dirección General de Tráfico y su página web, automática. En aquella época, los conocimientos personales disponibles en programación web se limitaban a HTML, CSS y una vaga idea de Javascript. Así, dadas estas herramientas, resultaba complicado implementar alguna solución para mejorar el sistema de práctica utilizado... sin embargo, era fácil definir claramente cuáles eran los puntos débiles del sistema sobre los que, aquella hipotética "aplicación", establecería una diferencia.

- Los tests disponibles para practicar eran estáticos, no importaba el momento en que se repasasen siempre tenían las mismas preguntas en el mismo orden. De esta forma, frecuentemente, existían tests en los que se cometían muchos fallos y otros en los que no, normalmente los mismos. También ocurría habitualmente que un test en el que no se cometían fallos (a pesar de poder habido en él preguntas contestadas bien "por casualidad") no se volvía a repetir y, por tanto, muchas de esas preguntas no se volverían a practicar.

- A pesar de que las preguntas correspondían de manera aleatoria a las diferentes áreas de conocimiento no había una posibilidad de elegir preguntas de un área o áreas concretas para poder realizar tests únicamente de ese área.

- Aún habiendo hecho cientos de tests, no existía una manera automática de saber en qué área se cometían más o menos fallos para poder repasarla. De igual forma, se desconocía qué preguntas (o qué tipo de preguntas) se habían fallado en más ocasiones para prestar más atención en ellas si aparecían en otra ocasión. Este caso se daba habitualmente con preguntas mal formuladas u opciones poco claras.

Curiosamente, desde la perspectiva del presente, al analizar aplicaciones de tests en ámbitos muy diferentes, siguen siendo aplicables los puntos anteriores. Por este motivo, en el desarrollo de este trabajo, el objetivo principal ha sido incluir una implementación a todas las mejoras descritas en la misma aplicación. Y, además, a pesar de que la idea primigenia surgiera de los exámenes de conducir, hacerla extensible y adaptable a otros entornos. Estos objetivos se definen en detalle en el apartado siguiente.

1.1.- Objetivos

Tal y como se ha avanzado en el apartado anterior, como resultado del desarrollo de este trabajo se ha implementado una aplicación web dinámica para la corrección automática de pruebas. Este sistema da solución a algunas de las carencias habituales en sistemas de evaluación automática. Además, esta aplicación estará enfocada especialmente a entornos colaborativos en los que se utilicen tests como herramienta de evaluación.

Los objetivos que se plantean para esta aplicación son los siguientes:

- Creación de un entorno colaborativo simple con tres roles básicos: usuarios administradores, usuarios que se ocupan de evaluar tanto creando exámenes como preguntas (profesores) y usuarios que realizan los tests (alumnos).
- Adaptación del entorno para permitir trabajar a los usuarios sobre un campo de conocimiento concreto (por ejemplo: una asignatura, un aula) que pueda tener diversas áreas diferentes. La orientación de esta "asignatura" será la preparación de un examen final de un formato determinado a través de exámenes previos/parciales y autoevaluaciones.
- Implementación de un sistema de estadísticas para permitir a los usuarios un seguimiento de sus progresos facilitando tanto la evaluación como el estudio.

2.- Estudio de viabilidad

El presupuesto de este proyecto se considera cero ya que todo el software que se ha utilizado es gratuito, incluso el servidor en el que se ubica la aplicación en Internet es totalmente gratuito (000webhost.com). Las horas de trabajo tampoco son candidatas a incluirse en el presupuesto ya que forman parte de la asignatura. En cualquier caso, lo que en un proyecto laboral sería la cifra final que el cliente pagaría en función de las horas dedicadas, en este proyecto se convertirá en la nota final de la asignatura.

A continuación se detallan los requisitos del entorno de trabajo en el que se llevará a cabo del desarrollo:

2.1.- Hardware

La máquina en la que se ha desarrollado el trabajo tiene las siguientes características:

- Ordenador: HP Pavilion dv6 Notebook PC; procesador Intel Core i7 Q720 @1.6GHz; RAM 4GB.
- Sistema operativo: Windows 7 64bits.

2.2.- Servidor

Localmente se ha utilizado un servidor Apache 2.2 con PHP 5.3 y MySQL 5.0.

El servidor en el que está alojada la aplicación online (http://tfc-uoc.freeiz.com/tfc-uoc) es un servidor Apache 2.0 con PHP 5.2.17 y MySQL 5.1.57.

2.3.- Software

Principalmente se ha utilizado la siguiente lista de software para el desarrollo:

- Eclipse (versión Helios, edición especial para el desarrollo en PHP).
- phpMyAdmin: Navegador visual de bases de datos.
- Subversion: Para el control de versiones del código.
- Navegador web: Principalmente se ha utilizado Firefox para desarrollar, aunque se realizarán también pruebas en Chrome, Internet Explorer 8, Safari y Opera. Exceptuando Internet Explorer, todos son navegadores, actualmente, se actualizan automáticamente por lo que no es posible especificar una versión concreta para ellos.
- Paquete OpenOffice para todas las tareas de documentación.

3.- Metodología

En las fases de planificación, análisis y diseño la metodología de trabajo ha sido guiada por la elaboración de las PAC. El trabajo realizado en ellas ha sido la guía y base para la fase final: la implementación.

La implementación se ha guiado por la estructura de datos definida en la PAC 2 (diagrama de clases y modelo E/R) que ha servido para llevar a cabo la definición de las tablas de la base de datos y la estructura de la aplicación. A la hora de desarrollar se ha seguido el "guión" establecido por los casos de uso (también definidos en la PAC 2). La estrategia de desarrollo para cada uno de estos casos ha seguido el siguiente "esquema":

1. Revisión del caso de uso descrito.
2. Implementación de las funciones necesarias para llevarlo a cabo y optimización de las consultas a la base de datos asociadas mediante la función EXPLAIN de MySQL y añadiendo los índices necesarios.
3. Implementación de un test (o tests) unitario para comprobar el funcionamiento.
4. Revisión del prototipo, diseño e implementación de la página asociada al caso de uso.
5. Tests adicionales.

4.- Planificación

En este apartado se describe la solución propuesta a través del desglose de tareas que se han llevado a cabo y el diagrama de tiempo asociado a ellas. A continuación se detallan las tareas tal y como quedaron recogidas en la PAC 1 ya que no ha habido grandes desviaciones sobre la planificación original.

4.1.- Definición de tareas

Las PAC han sido las tareas principales. Para el desglose de subtareas se han utilizado los puntos definidos por el

consultor en el enunciado de cada una de las PAC.

Realización de la PAC 1 (Plan de trabajo)

Tarea 1.1.- Elección de la propuesta

Tarea 1.2.- Redacción de la PAC 1

Redacción del plan de trabajo. El plan de trabajo incluye, entre otros puntos, la definición de los objetivos y las tareas del proyecto. La redacción de este documento ha servido para hacerse una idea global del proyecto y reflexionar sobre las diferentes componentes que han acabado formando parte de la aplicación final.

Realización de la PAC 2 (Especificación y análisis)

Tarea 2.1.- Definición del marco del problema

Análisis detallado del problema, valorando las soluciones así como sus ventajas e inconvenientes.

Tarea 2.2.- Análisis de requerimientos no funcionales

Análisis de los requerimientos de rendimiento, distribución, seguridad y usabilidad de la aplicación.

Tarea 2.3.- Definición de actores

Definición de los diferentes actores que intervienen en el sistema y las relaciones que pueden tener con la aplicación.

Tarea 2.4.- Diagrama de clases

Descripción del diagrama de clases, describiendo las diferentes clases que se implementarán así como las relaciones entre ellas.

Tarea 2.5.- Modelo E/R y modelo relacional de la base de datos

Diseño de la base de datos que da soporte al sistema. Elaboración de los correspondientes modelos entidad-relación y relacional para reflejar la estructura de datos.

Tarea 2.6.- Definición de casos de uso

Revisión de los actores de la tarea 2.3 y definición de los distintos casos de uso.

Tarea 2.7.- Fichas de casos de uso

Elaboración de las fichas correspondientes a los casos de uso.

Realización de la PAC 3 (Diseño)

Tarea 3.1.- Diagramas de actividad

Definición de los diagramas de actividad, describiendo los distintos flujos de trabajo de la aplicación.

Tarea 3.2.- Diagramas de secuencia

Creación de los diagramas de secuencia necesarios para reflejar las interacciones entre los distintos objetos que forman parte del sistema.

Tarea 3.3.- Prototipo de las interfaces

Diseño de prototipos para las interfaces (pantallas) más importantes de la aplicación para danr una idea muy genérica del diseño final.

Tarea 3.4.- Análisis de requisitos no funcionales

Revisión de los requerimientos no funcionales ya descritos en la tarea 2.2 y redacción de los requisitos de usabilidad, seguridad y accesibilidad.

Realización de la PAC 4 (Codificación, memoria y presentación virtual)

Tarea 4.1.- Implementación de la aplicación

Codificación de la aplicación en su totalidad realizando las pruebas de funcionamiento pertinentes en cada momento.

Tarea 4.2.- Redacción de la memoria

Esta tarea se ha realizado paralelamente a la anterior, en la medida de lo posible, ya que muchos de los apartados que contiene ya han sido descritos en los documentos elaborados para las PAC anteriores.

Tarea 4.3.- Confección de la presentación virtual

Elaboración de presentación virtual que condensa todo el trabajo realizado accesible (y comprensible) a cualquier interlocutor.

4.2.- Diagrama de tiempo

Es importante tener en cuenta que compaginando vida laboral y académica ha sido difícil definir una "jornada laboral" exacta para el TFC por lo que en el diagrama de tiempo, aunque la duración se especifica en "días", éstos se consideran días de dedicación no días completos. Habitualmente esta dedicación, de lunes a viernes, ha sido de una media de 3 horas al día y los fines de semana de 7 horas de media cada día. En los días festivos, puentes o días de vacaciones a lo largo del semestre la dedicación ha sido mayor.

A continuación se incluye el diagrama de tiempo ya descrito en la PAC 1:

	Nombre	Duración	Inicio
1	**Tarea 1: Realización de la PAC 1 (Plan de trabajo)**	**8 days**	**19/09/12 8:00**
2	Tarea 1.1.- Elección de la propuesta	1 day	19/09/12 8:00
3	Tarea 1.2.- Redacción de la PAC 1	4 days	25/09/12 8:00
4	**Tarea 2: Realización de la PAC 2 (Especificación y análisis)**	**11 days**	**30/09/12 8:00**
5	Tarea 2.1.- Definición del marco del problema	2 days	30/09/12 8:00
6	Tarea 2.2.- Análisis de requerimientos no funcionales	2 days	2/10/12 8:00
7	Tarea 2.3.- Definición de actores	2 days	4/10/12 8:00
8	Tarea 2.4.- Diagrama de clases	2 days	6/10/12 8:00
9	Tarea 2.5.- Modelo E/R y modelo relacional de la base de datos	2 days	8/10/12 8:00
10	Tarea 2.6.- Definición de casos de uso	2 days	10/10/12 8:00
11	Tarea 2.7.- Fichas de casos de uso	2 days	12/10/12 8:00
12	**Tarea 3: Realización de la PAC 3 (Diseño)**	**7 days**	**14/10/12 8:00**
13	Tarea 3.1.- Diagramas de actividad	2 days	14/10/12 8:00
14	Tarea 3.2.- Diagramas de secuencia	2 days	16/10/12 8:00
15	Tarea 3.3.- Prototipo de las interfaces	4 days	18/10/12 8:00
16	Tarea 3.4.- Análisis de requisitos no funcionales	1 day	22/10/12 8:00
17	**Tarea 4: Realización de la PAC 4 (Codificación, memoria y presentación virtual)**	**54,125 days**	**22/10/12 8:00**
18	Tarea 4.1.- Implementación de la aplicación	50 days	22/10/12 8:00
19	Tarea 4.2.- Redacción de la memoria	45 days	23/10/12 8:00
20	Tarea 4.3.- Confección de la presentación virtual	5 days	28/12/12 9:00

5.- Marco de trabajo y conceptos previos

A pesar de que las pruebas tipo test se utilicen en muchos y variados contextos, en el desarrollo de este trabajo nos hemos centrado especialmente en uno de ellos: el ámbito educativo en entornos virtuales colaborativos. Generalmente los tests se utilizan como herramienta de evaluación de unos conocimientos determinados. Además también se utilizan como herramienta de estudio y repaso de conocimientos adquiridos o de preparación a un test final con unas características determinadas. Por ejemplo, supongamos una asignatura en la que el examen final consiste en cincuenta preguntas tipo test, con una única respuesta válida, agrupadas en bloques de diez preguntas donde cada una cubre un área diferente. Dado este entorno, es de esperar que los alumnos se preparen para esta prueba realizando tests que tengan la misma estructura. Es en este punto donde encontramos el problema que se ha centrado en resolver este trabajo.

Dada la hipotética asignatura descrita en el ejemplo anterior, lo habitual sería que los alumnos tuvieran disponibles muchos tests con exactamente la misma estructura para preparar el examen final. En general, estos tests, aunque pudieran corregirse de forma automática, serían estáticos (contendrían el mismo número de preguntas en el mismo orden y la misma estructura de áreas). Además, por muchos tests que el alumno realizase, inevitablemente acabaría dándose por satisfecho si no cometiera más de un número de fallos. Sin embargo, no tendría la posibilidad de crear un test “personalizado” en el que elegir qué área (o áreas) quiere repasar y con cuántas preguntas. De igual manera, al ir realizando tests, ni alumnos ni profesores, tendrían una visión global de su progreso (en qué áreas debería mejorar, qué preguntas se responden erróneamente con mayor frecuencia, etc.).

Así, descrito el marco del problema, este trabajo pretende desarrolla una herramienta web dinámica orientada a los entornos virtuales de aprendizaje en la que los alumnos pueden prepararse para un examen final tipo test de manera flexible. Además, los profesores pueden conocer el progreso de sus alumnos detallado y realizar pruebas para que repasen sus puntos más débiles.

6.- Requisitos del sistema

6.1.- Requisitos no funcionales

Consideraremos en todo momento la aplicación final que se ha desarrollado como un prototipo. Por lo tanto, los requerimientos no funcionales que se describan a continuación tienen en cuenta este hecho.

Rendimiento

La aplicación permite la concurrencia de usuarios (puede haber diversos profesores y alumnos utilizando la aplicación al mismo tiempo). El tiempo de respuesta del sistema para estos usuarios es siempre rápido (del orden de milisegundos).

Seguridad

El acceso a la aplicación está restringido (es necesario tener un usuario y contraseña). Asimismo, existen diferentes roles (administrador, profesor, alumno) que otorgan una serie de permisos determinados. También se ha controlado

que los usuarios no puedan acceder a elementos sobre los que no tienen privilegios (por ejemplo: un usuario, si no es administrador, sólo puede editar su propio usuario).

La aplicación se distribuye en módulos asociados a cada uno de los elementos (aulas, áreas, usuarios, exámenes, pruebas de evaluación y preguntas) cada módulo puede tener diversas acciones asociadas (por ejemplo: ver, editar, listar, etc.) y, además, estas acciones pueden llevarse a cabo sobre entradas determinadas (identificadas por un ID). El acceso se controla, en función de los roles, a estos tres niveles: módulo (module), acción (action) y entrada (recordid).

Usabilidad

La interfaz de usuario es sencilla e intuitiva para usuarios familiarizados con cualquier otro tipo de aplicación web. Es importante considerar que la idea de este prototipo es que estuviera integrado en un entorno virtual colaborativo más amplio, fuera del alcance de este trabajo, y con muchas otras funcionalidades que el usuario ya estará acostumbrado a utilizar habitualmente.

La aplicación tiene un diseño básico y sencillo para no afectar a la usabilidad. La navegación se basa en una cabecera en la que se incluye un acceso al perfil de usuario y otro a la desconexión del sistema. Todas las páginas tienen un menú superior y, en algunos casos, un menú adicional (submenú). Los datos en todas las páginas consisten en formularios o datos tabulados.

Accesibilidad

Para este trabajo, considerándolo un prototipo, no se han tenido en cuenta requisitos específicos de accesibilidad y quedarán propuestos como líneas futuras del trabajo. Aún así se ha analizado la aplicación con tests de accesibilidad online superando en todos los casos los tests para la especificación WCAG 2.0 (Level AA). La aplicación puede prescindir del uso de Javascript (incluso del tema) y ser accesible.

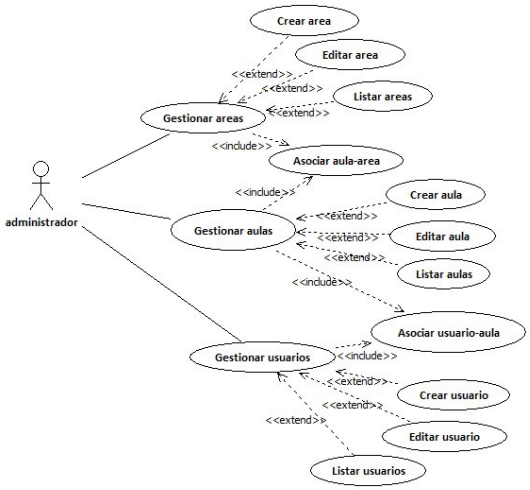
6.2.- Requisitos funcionales

Los requisitos funcionales del sistema están relacionados con los actores que intervienen en él así como los casos de uso que se definen para ellos. A continuación detallamos actores y casos de uso.

Actores

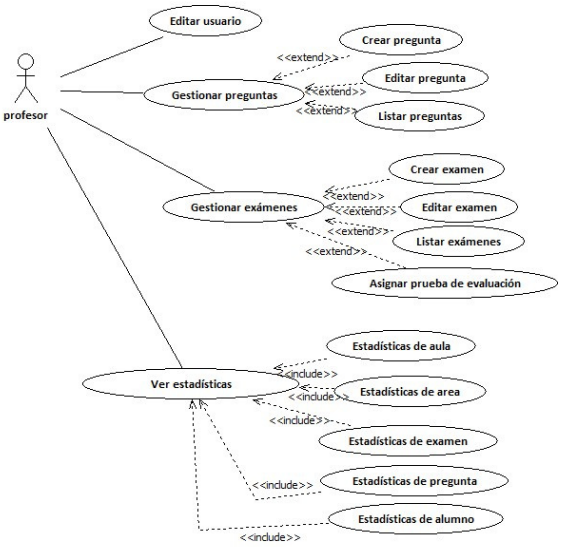
Administrador

El administrador se encarga de añadir usuarios (profesores, alumnos), gestionar áreas y gestionar aulas.



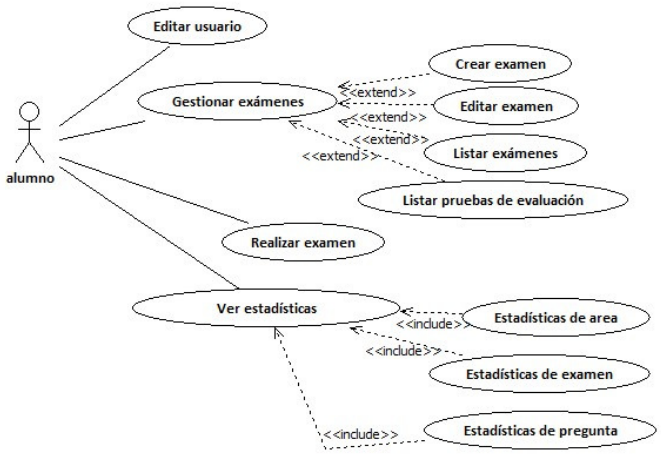
Profesor

Un profesor está al cargo de una o varias aulas formadas por un conjunto de alumnos. En estas aulas podrá crear tests que los alumnos deberán realizar como forma de evaluación (“prueba de evaluación”). También tendrá acceso a un sistema de estadísticas. Los profesores también pueden incorporar preguntas al sistema.



Alumno

Un alumno forma parte de una o varias aulas. Dentro del aula tiene un espacio donde puede crear tests personalizados para practicar para la prueba final. En el aula también tiene acceso a los tests que tendrá que realizar como parte de la asignatura (las "pruebas de evaluación" que asigna el profesor al aula). Al igual que los profesores, los alumnos tienen acceso a las estadísticas de su progreso en la asignatura.



Fichas de casos de uso

Nota: En la descripción de los casos de uso, así como en el resto del trabajo, consideramos un área como una de las "partes" que pueden formar parte de los conceptos que se tratan en un aula (o en una asignatura) y que pueden ser evaluables. Consideraremos siempre el concepto "aula" y "asignatura" como sinónimos, utilizando mayormente el término "aula" al ser más general.

Crear área

Resumen de funcionalidad: Añade un área a la base de datos.

Casos de uso relacionados: Editar área, listar áreas, asociar aula-área.

Precondición: El área no existe en la base de datos.

Postcondición: El área se incluye en la base de datos.

El administrador crea un área dándole un nombre.

Editar área

Resumen de funcionalidad: Modifica los datos de un área.

Casos de uso relacionados: Crear área, listar áreas, asociar aula-area.

Precondición: El área existe en la base de datos.

Postcondición: El área se actualiza en la base de datos.

El administrador puede cambiar el nombre de un área.

Listar áreas

Resumen de funcionalidad: Lista las áreas existentes en la base de datos.

Casos de uso relacionados: Crear área, listar áreas, asociar aula-área.

Precondición: Haber accedido al sistema como administrador.

Postcondición: Se muestra la lista de áreas.

Se muestra la lista completa de áreas.

Asociar aula-área

Resumen de funcionalidad: Asocia un área a un aula.

Casos de uso relacionados: Editar área, listar áreas, editar aula.

Precondición: El área y el aula existen en el sistema.

Postcondición: El área se asocia al aula.

Al asociar un área a un aula ésta estará disponible al crear exámenes en ese aula.

Crear aula

Resumen de funcionalidad: Añade un aula a la base de datos.

Casos de uso relacionados: Editar aula, listar aulas, asociar aula-área, asociar usuario-aula.

Precondición: El aula no existe en la base de datos.

Postcondición: El aula se incluye en la base de datos.

El administrador crea un aula dándole un nombre.

Editar aula

Resumen de funcionalidad: Modifica los datos de un aula.

Casos de uso relacionados: Editar aula, listar aulas, asociar aula-área, asociar usuario-aula.

Precondición: El aula existe en la base de datos.

Postcondición: El aula se actualiza en la base de datos.

El administrador puede cambiar el nombre de un aula.

Listar aulas

Resumen de funcionalidad: Lista las aulas existentes en la base de datos.

Casos de uso relacionados: Crear aula, listar aulas, asociar aula-área, asociar usuario-aula.

Precondición: Existen aulas en el sistema.

Postcondición: Se muestra la lista de aulas.

Se muestra la lista completa de aulas.

Asociar usuario-aula

Resumen de funcionalidad: Asocia un usuario a un aula determinada.

Casos de uso relacionados: Crear aula, editar aula, listar aulas, asociar aula-área

Precondición: El usuario y el aula existen en la base de datos.

Postcondición: El usuario queda asociado a un aula.

El administrador puede asociar un usuario a un aula.

Crear usuario

Resumen de funcionalidad: Crea un nuevo usuario en la base de datos.

Casos de uso relacionados: Editar usuario, listar usuarios, asociar usuario-aula.

Precondición: El usuario no existe en la base de datos.

Postcondición: El usuario se incluye en la base de datos.

El administrador crea un usuario añadiendo su nombre, apellidos y rol (profesor o alumno).

Editar usuario

Resumen de funcionalidad: Modifica los datos de un usuario.

Casos de uso relacionados: Crear usuario, listar usuarios, asociar usuario-aula.

Precondición: El usuario existe en la base de datos.

Postcondición: El usuario se actualiza en la base de datos.

Se puede acceder a los datos del usuario y modificarlos. El administrador puede modificar el rol, los usuarios únicamente su nombre y apellidos.

Listar usuarios

Resumen de funcionalidad: Lista los usuarios existentes en el sistema.

Casos de uso relacionados: Crear usuario, editar usuario, asociar usuario-aula.

Precondición: Existen usuarios en el sistema.

Postcondición: Se muestra la lista de usuarios.

Se muestra la lista completa de usuarios.

Crear pregunta

Resumen de funcionalidad: Añade una pregunta a la base de datos.

Casos de uso relacionados: Editar pregunta, listar preguntas.

Precondición: La pregunta no existe en la base de datos.

Postcondición: La pregunta se incluye en la base de datos.

El profesor crea una pregunta completando sus datos (área a la que pertenece, texto, opciones y respuesta correcta).

Editar pregunta

Resumen de funcionalidad: Modifica una pregunta existente en la base datos.

Casos de uso relacionados: Crear pregunta, listar preguntas.

Precondición: La pregunta existe en la base de datos y no forma parte de ningún examen que ya se haya realizado.

Postcondición: La pregunta se actualiza en la base de datos.

El profesor puede modificar los datos de una pregunta existente.

Listar preguntas

Resumen de funcionalidad: Lista las preguntas existentes en el sistema.

Casos de uso relacionados: Crear pregunta, editar pregunta.

Precondición: Haber accedido al sistema como profesor.

Postcondición: Se muestra la lista de preguntas.

Se muestra la lista completa de preguntas existentes en el sistema.

Crear examen

Resumen de funcionalidad: Permite al usuario crear un examen nuevo.

Casos de uso relacionados: Editar examen, listar exámenes, realizar examen.

Precondición: El examen no existe en la base de datos.

Postcondición: El examen se incluye en la base de datos.

El usuario puede crear un examen dándole un nombre, añadiendo comentarios y eligiendo el número de preguntas de cada área. Los profesores, además, pueden elegir qué preguntas en concreto formarán el examen.

Editar examen

Resumen de funcionalidad: Modifica los datos o forma de un examen.

Casos de uso relacionados: Crear examen, listar exámenes.

Precondición: El examen existe en la base de datos y nunca se ha realizado.

Postcondición: El examen se actualiza en la base de datos.

El usuario puede editar la forma de una examen (nombre, comentarios, áreas) siempre que este examen sea editable (un examen es editable si no ha sido realizado por ningún usuario ni asignado a ningún aula)

Listar exámenes

Resumen de funcionalidad: Lista los exámenes existentes en el sistema correspondientes a un usuario.

Casos de uso relacionados: Crear examen, editar examen.

Precondición: Haber accedido al sistema como profesor o alumno.

Postcondición: Se muestra la lista de exámenes.

Se muestra la lista completa de exámenes creados por el usuario que ha accedido al sistema.

Asignar prueba de evaluación

Resumen de funcionalidad: Añade un asociación entre un examen y un aula (prueba de evaluación).

Casos de uso relacionados: Crear examen, listar exámenes, editar examen.

Precondición: El examen y el aula existen en la base de datos.

Postcondición: El examen seleccionado se asigna a un aula como prueba de evaluación.

El profesor puede asignar alguno de los exámenes que haya creado a una de sus aulas como prueba de evaluación.

Estadísticas de aula

Resumen de funcionalidad: Muestra las estadísticas de un aula determinada.

Casos de uso relacionados: Estadísticas de área, estadísticas de examen, estadísticas de pregunta, estadísticas de alumno.

Precondición: El aula tiene alumnos que han realizado exámenes.

Postcondición: Se muestran las estadísticas de un aula.

Se muestran las estadísticas del aula seleccionada.

Estadísticas de área

Resumen de funcionalidad: Muestra las estadísticas de un área determinada.

Casos de uso relacionados: Estadísticas de aula, estadísticas de examen, estadísticas de pregunta, estadísticas de alumno.

Precondición: Existen alumnos que han realizado exámenes con preguntas de algún área.

Postcondición: Se muestran las estadísticas de los exámenes por área.

Se muestran las estadísticas agrupadas por área.

Estadísticas de examen

Resumen de funcionalidad: Muestra las estadísticas de un examen determinado.

Casos de uso relacionados: Estadísticas de área, estadísticas de aula, estadísticas de pregunta, estadísticas de alumno.

Precondición: El examen existe en la base de datos y algún alumno lo ha realizado.

Postcondición: Se muestran las estadísticas de un examen.

Los alumnos pueden ver sus estadísticas (resultados) en un examen determinado, los profesores pueden ver las estadísticas de las pruebas de evaluación.

Estadísticas de pregunta

Resumen de funcionalidad: Muestra las estadísticas de preguntas.

Casos de uso relacionados: Estadísticas de área, estadísticas de examen, estadísticas de aula, estadísticas de alumno.

Precondición: Existen usuarios que han realizado algún examen.

Postcondición: Se muestran las estadísticas agrupadas por pregunta.

Se muestran las estadísticas por pregunta: preguntas con mayor/menor porcentaje de acierto.

Estadísticas de alumno

Resumen de funcionalidad: Muestra las estadísticas de un alumno determinado.

Casos de uso relacionados: Estadísticas de área, estadísticas de examen, estadísticas de pregunta, estadísticas de aula.

Precondición: El alumno ha realizado algún examen.

Postcondición: Se muestran las estadísticas de un alumno.

Se muestran las estadísticas del alumno seleccionado.

Listar pruebas de evaluación

Resumen de funcionalidad: Lista las pruebas de evaluación asignadas al aula a la que pertenece el alumno.

Casos de uso relacionados: Realizar examen.

Precondición: Haber accedido al sistema como alumno.

Postcondición: Se muestra la lista de pruebas de evaluación asignadas.

Se muestra la lista completa de exámenes de pruebas de evaluación asignadas al aula al que pertenece el alumno.

Realizar examen

Resumen de funcionalidad: El alumno puede realizar un examen creado por él o asignado.

Casos de uso relacionados: Crear examen, listar pruebas de evaluación.

Precondición: El examen existe en la base de datos.

Postcondición: El alumno finaliza el examen y se muestran los resultados.

El alumno realiza el examen contestando las preguntas y viendo sus resultados al finalizar.

7.- Estudio y decisiones

Desde el primer momento se decidió desarrollar la aplicación utilizando PHP y MySQL por ser el entorno de desarrollo utilizado, personal y profesionalmente, en los últimos cinco años. En un primer momento se valoro desarollar esta aplicación como un módulo de Drupal, debido a la experiencia en este sistema de gestión de contenidos. Sin embargo, finalmente, se decidió implementarlo como una solución aparte desarrollando todo el código que compone la aplicación. Utilizar Drupal hubiera supuesto un gran número de ficheros y tablas en la base de datos en las que el propio desarrollo de la aplicación se "perdería", también complicaría la descripción del diagrama de clases y la estructura de datos ya que para poder verla con nitidez sería necesario estar mínimamente familiarizado con Drupal.

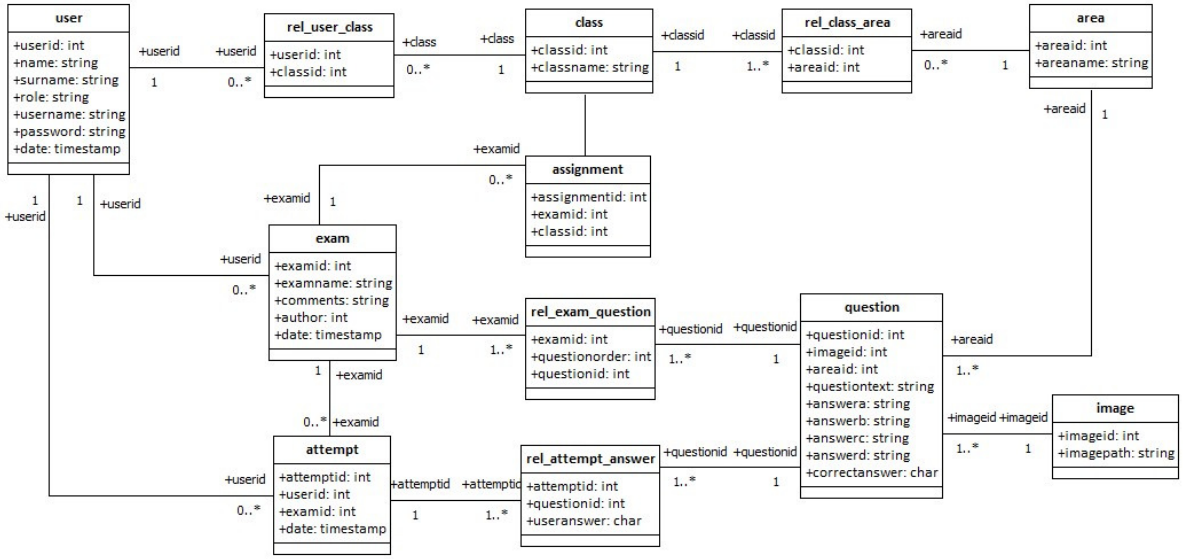
Así, se ha optado por crear una aplicación basada en una serie de ficheros .php en la que las diferentes operaciones están controladas por una librería de funciones. Aún así, se han utilizado la experiencia adquirida en los últimos años en Drupal, Moodle y desarrollo de otras aplicaciones para construir la aplicación de la manera más limpia y sencilla posible. Entre otros, se pueden considerar como "herencia de Drupal", los siguientes elementos de esta aplicación:

- Utilización de mod_rewrite para crear URL's "amigables".
- Utilización del sistema de menus definiendo sus rutas y control de accesos.
- Creación de un index.php a modo de plantilla y otros .php para cada módulo que generar el contenido (vistas) de cada página.
- Utilización de un sistema para dibujar tablas y formularios fácilmente.

8.- Análisis y diseño del sistema

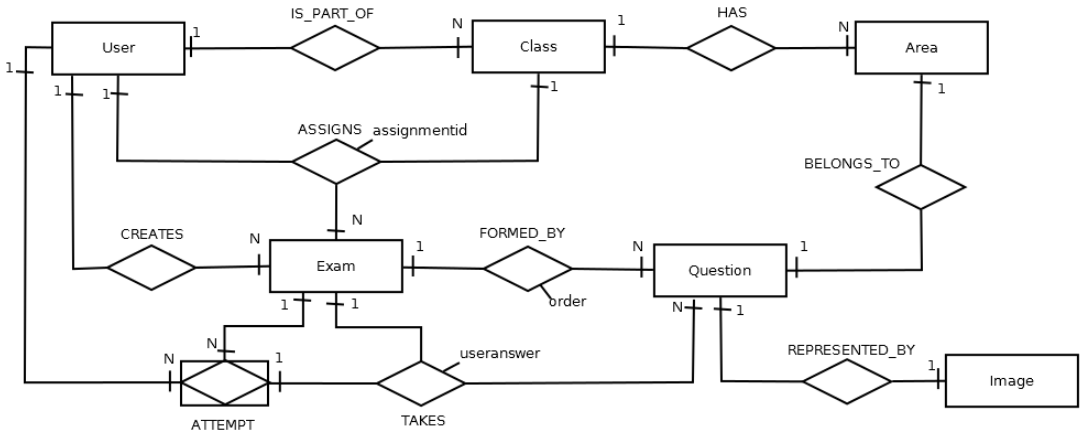
8.1.- Diagrama de clases

El diagrama de clases descrito en la PAC 2 ha sufrido únicamente una ligera modificación. En la tabla user se han añadido los campos username y password. También se ha cambiado el nombre del campo order de la tabla rel_exam_question por questionorder (para evitar problemas con la palabra order que es una palabra reservada en SQL). A continuación se incluye el diagrama modificado:



8.2.- Modelo E/R y modelo relacional de la base de datos

Modelo E/R



Area (areaid, areaname)

Attempt (attemptid, userid, examid, date)

Class (classid, classname)

Exam (examid, examname, comments, author, date)

Image (imageid, imagepath)

Question (questionid, imageid, areaid, questiontext, answera, answerb, answerc, answerd, correctanswer)

User (userid, name, surname, username, password, role, date)

Modelo relacional

Area (areaid, areaname)

Attempt (attemptid, userid, examid, date)

{userid} Clave foránea de user(userid).

{examid} Clave foránea de exam(examid).

Class (classid, classname)

Exam (examid, examname, comments, author, date)

{author} Clave foránea de user(userid).

Image (imageid, imagepath)

Question (questionid, imageid, areaid, questiontext, answera, answerb, answerc, answerd, correctanswer)

{imageid} Clave foránea de image(imageid).

{areaid} Clave foránea de area(areaid).

User (userid, name, surname, role, date)

Assigns (assignmentid, examid, classid)

{examid} Clave foránea de exam(examid).

{classid} Clave foránea de class(classid).

Formed_by (examid, questionid, questionorder)

{examid} Clave foránea de exam(examid).

{questionid} Clave foránea de question(questionid).

Has (classid, areaid)

{classid} Clave foránea de class(classid).

{areaid} Clave foránea de area(areaid)

Is_part_of (userid, classid)

{userid} Clave foránea de user(userid).

{classid} Clave foránea de class(classid).

Takes (attemptid, questionid, useranswer)

{attemptid} Clave foránea de attempt(attemptid).

{questionid} Clave foránea de question(questionid).

9.- Implementación y pruebas

El desarrollo se ha llevado a cabo sin grandes problemas, la experiencia confirma una vez más que si un proyecto se ha analizado y diseñado con detalle la implementación siempre resulta mucho más rápida y el tiempo se limita casi exclusivamente al tiempo que se tarde en escribir todas las funciones y estructuras analizadas y diseñadas.

Una parte importante de la implementación, que además han servido de test unitarios, ha sido la elaboración de dos ficheros .php que llenan las tablas de la aplicación con datos aleatorios (la estructura de ficheros se detalla en el ANEXO). Este proceso ha servido para probar unitariamente todos los casos de uso y además para llenar la aplicación de datos sin tener que utilizar cada uno de los usuarios para generar esta información. Aún así, también se han realizado en todo momento pruebas de funcionamiento “de verdad” sobre el propio navegador con algunos de los usuarios.

10.- Implantación y resultados

Como ya se he comentado en el apartado 7 de este documento, la aplicación se ha implementado utilizando una librería de funciones. No se han considerado necesario un desarrollo orientado a objetos por ser una aplicación sencilla. La única página a la que realmente se accede en la aplicación es a un index.php que actúa a modo de plantilla, recibiendo los parámetros que necesita de la URL (formateados de forma “amigable” gracias al módulo mod_rewrite de apache). En función del módulo (module), acción (action), entrada (recordid) y usuario logeado se componen los elementos que forman parte de la página (menús, formularios, tablas, etc).

Cada uno de los .php asociados a cada módulo se ocupa de procesar la información correspondiente a las acciones de ese módulo. Así, por ejemplo, accediendo a http://<url-de-la-aplicacion>/user/edit/25 se accede al módulo “user” (se incluirá user.php), con la acción “edit” y la entrada (recordid) 25 (en este caso el ID de usuario).

La aplicación controla también los accesos a nivel de módulo (por ejemplo: al módulo “area” sólo pueden entrar los administradores), a nivel de acción (a la acción “view” del módulo “exam” sólo pueden entrar los profesores) y a nivel de entrada (a una entrada particular de un examen sólo puede entrar el autor o los alumnos del aula asignada en caso de una prueba de validación).

11.- Conclusiones

Pienso que la aplicación, a pesar de ser muy sencilla (no hay que olvidar que es únicamente un prototipo), cumple los objetivos que se definieron para el proyecto en la PAC 1 y también los objetivos necesarios para poder superar la asignatura “Trabajo final de carrera”.

He hecho todo lo posible para que este trabajo sea suficientemente aceptable para que, después de todos estos años de equilibrios imposibles entre mi vida laboral, personal y académica, pueda dar por satisfactoriamente finalizada (al menos por un tiempo merecido) mi trayectoria académica.

Me gustaría también destacar en este apartado que el código de la aplicación está desarrollado intentando que el nombre de variables y funciones sea autoexplicativo y sencillo de entender. He tenido que prescindir de una documentación exhaustiva (usando notación phpDoc para las funciones por ejemplo) debido a un problema en mi muñeca derecha (agravado en el último año y del que tengo que ser intervenida) que me hace no poder escribir todo lo rápido que podría y optar por escribir el código primero para que la aplicación pueda funcionar en lugar de documentación y código al mismo tiempo (que es la buena práctica que acostumbraba a seguir). Este problema, y la falta de tiempo material, es algo también me ha hecho entregar este trabajo quizás de manera un poco justa y con bastantes cosas que podrían ser mejorables.

12.- Trabajo futuro

Existen diversas vías de mejora de este trabajo. A continuación se destacan algunas de ellas:

- Añadir en las preguntas la opción de incluir una imagen (a pesar de que la estructura de datos esta preparada para ello finalmente no se llegó a incluir en la implementación final).
- Añadir paginación a todas las páginas con registros o, como mínimo a la página de las preguntas.
- Añadir un sistema dinámico (via Javascript) de ordenar las tablas por diferentes campos (sort).
- Añadir un sistema dinámico para buscar preguntas y elegirlas para que forman parte de un examen (posibilidad de filtrar por área, texto de la pregunta o las respuestas, etc.). Permitir también editar este tipo de exámenes (por ejemplo reordenando las preguntas).
- Añadir la acción de borrar en todos los módulos con el debido control. Por ejemplo una pregunta no podría ser nunca borrada si forma parte de un examen, al borrar un examen se tendrían que borrar sus intentos, etc.
- Añadir imágenes a los usuarios para hacer la aplicación más personal.

13.- Bibliografía

1. **PHP.** Referencia del lenguaje PHP: http://www.php.net

2. **MySQL**. Referencia del lenguaje MySQL: http://www.mysql.com

3. **Stack Overflow.** Ayuda para dudas diversas: http://www.stackoverflow.com

4. **Web accessibility checker**. Tests de accesibilidad: http://achecker.ca/checker / http://wave.webaim.org

5. **Markup Validation Service**. Validador HTML: http://validator.w3.org

6. **Web Developer**. Extensión de Firefox: http://chrispederick.com/work/web-developer/

14.- ANEXO

14.1.- Estructura de ficheros

/files: Directorio preparado para subir diversos archivos de la aplicación (imágenes de preguntas, avatares de usuarios), tal y como se ha detallado anteriormente, esto ha quedado propuesto como una línea de futuro.

/files/questionimages: Directorio preparado para subir las imágenes de las preguntas.

/includes: Directorio que incluye todos los ficheros que controlan la aplicación:

area.php, assignment.php, class.php, exam.php, login.php, question.php, stats.php, user.php: Acciones/Vistas para cada una de las acciones de los módulos correspondientes.

config.php: Configuración de la aplicación, es necesario modificarlo en función del entorno.

dbconnect: Conexión a la base de datos.

functions.php: Librería de funciones.

/scripts: Directorio preparado para scripts de Javascript (incluye un fichero script.js vacío y la última versión de la librería Jquery). Finalmente no se ha utilizado Javascript pero se ha conservado el directorio.

index.php: Único fichero al que se accede en la aplicación y que muestra los datos correspondientes en función de los parámetros recibidos en la URL gracias a la configuración de .**htaccess**.

testdata1.php: Fichero que genera datos para áreas, aulas y usuarios (así como las asociaciones entre ellos). En las primeras líneas del fichero aparece la configuración para elegir cuántos elementos crear de cada tipo. Este fichero también ha servido en el proceso de desarrollo para realizar pruebas unitarias.

testdata2.php: Fichero que genera datos para preguntas, exámenes y pruebas de evaluación (así como las relaciones entre ellos). Al igual que testdata1.php incluye líneas de configuración y también ha servido para pruebas unitarias.

14.2.- Ficheros de datos

En el fichero estructura.sql se incluye únicamente la estructura de las tablas (con un DROP TABLE IF EXISTS).

En el fichero estructura_y_datos.sql se incluye la estructura y datos aleatorios generados con los ficheros de test (con un DROP TABLE IF EXISTS).

14.3.- Manual de usuario

Nota: Los usuarios genéricos se describen en función de los datos cargados en la aplicación entregada. Estos datos se incluyen en el fichero estructura_y_datos.sql.

Para todos los roles haciendo click en el nombre de usuario en la parte superior derecha se edita el usuario.

Administrador

Usuario genérico: admin – admin

Aulas: Lista de aulas. Haciendo click sobre el nombre de las aulas, áreas y usuarios se pueden editar. Desde el submenu se puede crear un aula nueva.

Áreas: Lista de áreas. Haciendo click sobre el nombre de las áreas se pueden editar. Desde el submenu se puede crear una nueva área.

Usuarios: Lista de usuarios. Haciendo click sobre el nombre de los usuarios y aulas se pueden editar. Desde el submenú se puede crear un nuevo usuario.

Profesor

Usuarios genéricos: teacherX – teacher (donde X es un número 1 al 2)

Exámenes: Lista de exámenes. Cada uno con sus opciones correspondientes. Desde el submenu se puede crear un examen nuevo.

Aulas: Lista de aulas en la que el usuario es profesor.

Preguntas: Lista de preguntas. Haciendo click sobre ellas se pueden editar. Desde el submenu se puede crear una pregunta nueva. Marcando los checkbox se pueden seleccionar preguntas para crear un examen (el botón de creación está al final) .

Estadísticas: Estadísticas de los diferentes módulos.

Alumno

Usuario genérico: studentX – student (donde X es un número del 1 al 6)

Exámenes: Lista de exámenes. Cada uno con sus opciones correspondientes. Desde el submenu se puede crear un examen nuevo.

Pruebas de evaluación: Lista de pruebas de evaluación.

Estadísticas: Estadísticas de los diferentes módulos.