Proyecto Fin de Carrera

Aplicación móvil de la plataforma web Moodle

Mobile application of the Moodle web platform

Para acceder al Título de

INGENIERO EN INFORMÁTICA

Autor: Rubén Gutiérrez López
Director: Carlos Blanco Bueno

Septiembre - 2013

Agradecimientos

Cualquiera que haya hecho alguna vez cualquier tipo de trabajo, sabe perfectamente que sin la ayuda y sin el consejo de algunas personas, es imposible sacarlo adelante, sobre todo cuando el proyecto es de la magnitud de un proyecto fin de carrera. Por este motivo considero que este apartado es uno de los más importantes del informe.

Para empezar quisiera dar las gracias a mi profesor de proyecto, Carlos Blanco, por hacer caso de mis peticiones y llevar mi proyecto fin de carrera. No solo eso, además me proporcionó una idea bastante interesante como es la realización de una aplicación móvil de Moodle. Además tengo que resaltar que siempre fue muy atento conmigo y que nunca dejo de contestarme a ningún correo, ni nunca me negó la entrada a su despacho.

Por otro lado, sin mi familia y sin el apoyo económico que estos me han dado, no estaría cursando la carrera. Para mí ser informático es un sueño y por lo tanto, ellos han hecho posible que pueda estar frente a mi última prueba antes de conseguir el tan ansiado título.

Tampoco puedo olvidarme que compañeros de clase tales como Daniel Quevedo y Eduardo Calzado, sus consejos siempre fueron bien recibidos.

Y para terminar, me gustaría dar las gracias a todos los profesores que me han estado educando a lo largo de mi carrera para que pueda desenvolverme perfectamente en proyectos como estos. Sin su ayuda y sin su paciencia seguramente no tendría los conocimientos que tengo a día de hoy.

A todos ellos, siempre formarán parte de mi memoria y contarán con mi eterna gratitud. Gracias por todo.

Contenido

Índice de imágenes

Índice de tablas

Resumen

Prácticamente la totalidad de universidades cuentan con herramientas virtuales para el apoyo al aprendizaje y enseñanza de los alumnos, de entre las cuales Moodle es probablemente la más extendida. La Universidad de Cantabria ofrece servicios y soporte para Moodle, de forma que la mayoría de titulaciones y asignaturas cuentan con espacios virtuales en los que profesores y alumnos pueden estar informados sobre los eventos de la asignatura (clases, fechas de entrega, etc.), consultar materiales, entregar tareas o prácticas, interactuar mediante foros, etc.

Mediante estos sistemas el alumno puede estar al día de los eventos de cada asignatura y organizar mejor su tiempo de trabajo. Sin embargo, al tener cada asignatura un calendario propio los alumnos no tienen una visión general de su carga de trabajo. Muy a menudo el propio alumno ha de ir consultando curso a curso y mientras confecciona a mano un calendario integrado para saber cuáles son sus entregas semanales o mensuales y de este modo gestionar mejor su tiempo.

A este hecho se une el de que el estudiante medio tiene una agenda demasiado apretada, y en la mayoría de los casos no dispone del tiempo suficiente para consultar los cursos, lo que deriva a que el alumno participa menos e incluso puede que se olvide de mirar los eventos que el curso tiene preparados para él.

Con el objetivo de facilitar la autogestión de tiempo de trabajo y estudio al alumno este proyecto pretende crear una aplicación móvil de consulta rápida para la plataforma Moodle. Esta aplicación por un lado le facilitará al alumno el acceso rápido a información importante de cada asignatura, y por otro lado mostrará de forma integrada los eventos siguientes de todas las asignaturas en las que está matriculado, por lo que no será necesario acceder a cada curso de forma individual a la hora de adquirir información.

En cuanto a las tecnologías utilizadas para el desarrollo de este proyecto, se ha optado por Phonegap como Framework de desarrollo, ya que permite compilar la aplicación para las principales plataformas móviles (Android, iOS, WindowsPhone, etc.) y se han utilizado los lenguajes HTML, CSS, JavaScript y PHP.

Palabras Clave

- Herramienta de apoyo a la enseñanza,
- Gestión de tiempo de trabajo,
- Moodle,
- Calendario,
- Foro,
- Información,
- Alumnos,
- Profesores,
- Aplicaciones móviles.

Abstract

Almost all universities have virtual tools to support learning and teaching of students, among which Moodle is probably the most widespread. The University of Cantabria provides services and support for Moodle, so that most degrees and courses have virtual spaces where teachers and students can be informed about the events of the course (lectures, deadlines, etc.), check materials/notes, turn in assignments or practices, interact through forums, etc.

Through these systems, students can keep abreast of events in each subject and better organize their working time. However, as each course has its own schedule, it is difficult for students to have an overview of their workload. Very often, students themselves have to consult each course separately, to then draw up a full schedule to know what their weekly or monthly installments are, and thus, better manage their time.

This is coupled with the fact that the average student usually has a pretty tight agenda and, in most cases, does not have enough time to see the courses, which results in the student participating less and may even forgetting to look at the available events the course has for him/her.

In order to facilitate self-management of work-study time, this project aims to create a quick reference mobile application for Moodle platform. This application, on the one hand, will provide students quick access to important information from each subject and, on the other, will seamlessly show the upcoming events in all subjects in which they are enrolled, so they will not need to access each course individually at the time of acquiring information.

In terms of technologies utilized for the development of this project, Phonegap has been chosen as development Framework, since it allows to compile the application to major mobile platforms (Android, iOS, WindowsPhone, etc.). Finally, HTML, CSS, JavaScript and PHP have been used as programming languages.

Introducción y objetivos

Sistemas de apoyo a la enseñanza

Desde hace tiempo, estamos envueltos en un entorno global donde la tecnología participa de forma activa en muchas de las labores que desempeñamos en nuestras vidas. En cuestión de muy pocos años, los avances tecnológicos han sido tales que, actualmente podemos considerar que tenemos una gran dependencia de ellos.
Estas mejoras han afectado a todos los sectores, sobretodo en el campo de la educación, contando a día de hoy con tecnologías que ayudan a potenciar y mejorar dicho sector.

Como en la mayoría de universidades y escuelas de España y del resto de países del mundo, la Universidad de Cantabria se apoya para el aprendizaje de sus alumnos, en plataformas e-learning, las cuales, ofrecen una serie de herramientas útiles para facilitar una mejor interacción entre profesor y alumno. La aplicación de dicha tecnología no es más que un añadido a las herramientas de comunicación y aprendizaje, no implica que alumno y profesor no puedan interactuar físicamente. El tipo de aprendizaje empleado en la Universidad de Cantabria es presencial, o dicho en inglés, BendedLearning (B-learning), que traducido al español sería algo así como aprendizaje mixto: combinando el encuentro virtual con el encuentro físico.

Las plataformas e-learning, también conocidas como campus virtuales o LMS (Learning Management System), son espacios virtuales de aprendizaje que facilitan una educación basada en la distancia. En estos espacios virtuales se realizan actividades tales como el intercambio de archivos, la participación en foros, chats…, además de muchas otras actividades adicionales.

Este tipo de plataforma web ofrece varias ventajas en el ámbito educativo. Podemos destacar que es una herramienta flexible, ya que permite al profesor moldear su curso en función de sus características o estrategias de enseñanza. Además, tienen un fácil acceso, sólo necesitas acreditarte con un usuario y una contraseña. También podemos hacer referencia que al ser una aplicación de código abierto, existen varias plataformas gratuitas.

Suele tener una interface gráfica bastante sencilla o intuitiva, por lo que, una persona con pocos conocimientos no tendrá muchos problemas para manejarse dentro de ella.

Como explicamos anteriormente, esta plataforma elimina las barreras de la distancia. Un profesor puede proporcionar recursos tales como apuntes, notas, evaluaciones…, como un alumnos puede enviar sus informes, trabajos…, sin tener que estar en el mismo lugar. El único requisito es tener acceso a Internet.

Actualmente, existen una gran cantidad de plataformas e-learning. Estas pueden ser comerciales o de código abierto. Las dos plataformas más utilizadas dentro del ámbito universitario español son WebCT, seguida de cerca por Edustan (ambas comerciales).

También existen .LRN (dot learn), Docebo, Blackboard y eCollege, de código abierto. A nivel europeo se encuentra ILIAS, muy utilizada en entornos de formación empresarial, especialmente en Reino Unido y países del norte.

Sin embargo, en la actualidad, está ganando mucha fuerza en universidades y escuelas otra plataforma de licencia gratuita, Moodle.

Moodle

La Universidad de Cantabria utiliza Moodle como plataforma e-learning. A su vez emplea otras como Blackboard, pero Moodle ha sufrido un crecimiento tan grande en comparación con las demás, que ha llegado a convertirse en la única plataforma web de carácter educativo.

Moodle, es una aplicación web de carácter educativo de distribución libre. Fue creado por Martin Dougiamas, el cual fue administrador de otra plataforma e-learning, WebCT. Su idea fue crear un entorno educativo donde el profesor, en base a sus conocimientos y habilidades, moldea el lugar a su técnica de enseñanza, en vez de simplemente publicar y transmitir la información a los estudiantes.

Actualmente cuenta con la versión 2.5 estable. Su instalación es sencilla, simplemente requiere una plataforma que soporte PHP y la disponibilidad de una base de datos.

después solamente hay que cargar la web mediante un navegador web (actualmente soporta la mayoría de ellos).

Para poder explicar las diferentes características básicas de Moodle las dividiremos en tres ramas:

Nivel General	
Interoperabilidad	Se puede utilizar bajo diversos sistemas operativos como Windows, Mac, Linux, etc.
Escalable	Se puede utilizar tanto en organizaciones pequeñas como en grandes.
Personalizable	Se puede configurar en función a los requerimientos del usuario, pudiendo activar muchas funcionalidades.
Económico	Se puede utilizar tanto en organizaciones pequeñas como en grandes.
Seguro	Cuenta con mecanismos de seguridad que permite que el entorno sea seguro y libre de amenazas.

Tabla 1: Nivel General.

Nivel Pedagógico	
Flexible	Se personaliza a lo que el profesor quiera, por lo que, puede usarse bajo diferentes técnicas de enseñanza.
Monitoreo	Permite un seguimiento del alumno.

Tabla 2: Nivel Pedagógico.

Nivel funcional
Facilidad de uso
Gestión de perfiles de usuario.
Presentar cualquier contenido digital, ya sea texto, audio, imagen o video.
Gestión de tareas y actividades.
Implementación de foros para una participación mayor del alumnado.
Inclusión de nuevas funcionalidad, es decir, actualización en base a nuevas necesidades.

Tabla 3: Nivel Funcional.

Objetivos

Tal como lo hemos mencionado, parece que todos son ventajas, pero vemos una clara limitación, la cual trataremos de solventar. Los estudiantes universitarios suelen tener una gran cantidad de créditos, es decir, se han matriculado en muchas asignaturas por lo que tienen una gran carga lectiva y muchas horas de trabajo. Esto limita el tiempo que dispone el alumno para consultar las tareas, actividades u otros eventos que dispone en dicha plataforma.

Es por esto, por lo que surge la idea de crear una aplicación para Smartphones que sirva al alumno como guía de bolsillo para poder economizar su tiempo y disponer de todo lo relevante en el momento.

El objetivo principal es agrupar todas las asignaturas. En lugar de tener que acceder de forma individual una a una, el alumno puede tener una visión conjunta de todos los eventos.

Se pretende agrupar todos los cursos a los que pueda tener acceso el usuario, para así facilitar la navegación en la plataforma y tener una accesibilidad más directa a todos los contenidos.

Puesto que el uso de Smartphones en España es bastante elevado, queremos aprovechar al máximo este sector y por lo tanto, crear una aplicación multiplataforma (Android, IOs, etc.), para que puedan utilizarla todos los alumnos con independencia del dispositivo móvil que tengan, que contenga la siguiente información:

- Un calendario genérico en el que se agrupen todos los eventos de los cursos en los que está matriculado un alumno, permitiendo así que el alumno observe de forma integrada su carga de trabajo y pueda gestionar mejor su tiempo.

- Un apartado de información que muestre al usuario algunas características importantes de los cursos, como recursos o participantes.

- Un foro donde estén agrupados todos los mensajes de todos los cursos, simulando la bandeja de entrada de una plataforma de correo.

- Además de un apartado exclusivo para profesores que sirva para mostrar algunas de las estadísticas de cada asignatura, referentes a la participación de cada alumno en las distintas actividades, consultas de material, accesos, etc.

Con esto, tanto alumno como profesor, podrán llevar un seguimiento de sus cursos con las ventajas que esto conlleva.

Metodologías

Dentro del desarrollo de un proyecto software, hay que tener en cuenta uno de los aspectos más importantes: la elección de la metodología de desarrollo software, o lo que es lo mismo, como se va a realizar dicho proyecto.

Existen diferentes tipos de metodologías. Cada cual se adapta mejor dependiendo del tipo de proyecto, por lo que a la hora de establecer el tipo de metodología que se quiere utilizar, hay que tener en cuenta factores como la complejidad de los requisitos (si pueden cambiar con el tiempo o no), cantidad de personas con las que se trabaja, dominio de las herramientas software, etc.

En mi caso, he escogido un tipo de metodología ágil por los siguientes motivos:

- Permite realizar cambios con rapidez y en cualquier fase del proyecto.
- Tiene un desarrollo iterativo.
- Se realizan revisiones con bastante frecuencia
- Mayor control del esfuerzo que te falta por hacer, no del que llevas haciendo.
- Timeboxing, es decir, una asignación inflexible de tiempos a cada iteración.
- Además no se emplea tanto tiempo en la documentación.

Ahora bien, dentro de la categoría de las metodologías agiles, existen un gran número de subcategorías, cada una con una característica única que la hace diferente al resto. Para este proyecto, hemos escogido dentro de la metodología ágil, la metodología SCRUM.

El eje central de la metodología SCRUM es la iteración y la descomposición de las etapas en pequeños pasos, por lo que, en vez de centrarse en un todo, se trabaja en pequeños componentes. Además, lo prioritario es crear la funcionalidad más importante para el cliente y luego poco a poco, se van creando otras funcionalidades complementarias o menos importantes.

El clico de vida de este tipo de metodología se muestra reflejado en la siguiente ilustración:

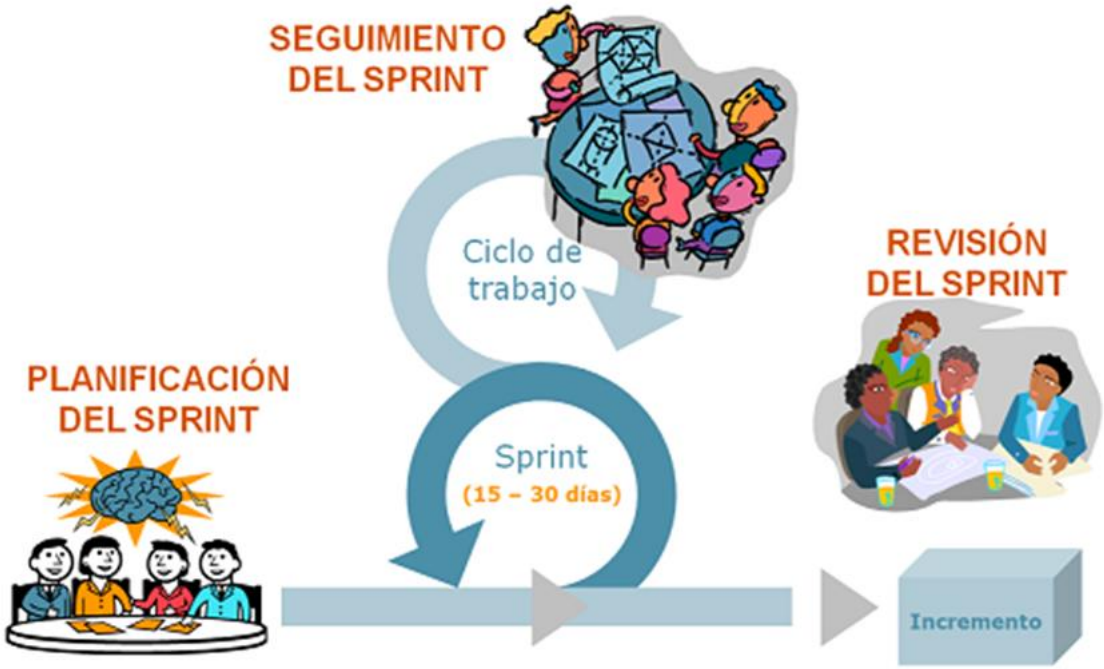

Imagen 1: Ciclo de vida basado en la metodología SCRUM.

Este ciclo de vida está compuesto por las siguientes fases:

1. **Planificación del sprint:** Es la reunión previa para planificar los objetivos que se pretenden alcanzar durante el sprint. Estos objetivos se elaboran en base a un tiempo establecido, que es lo que va a durar el sprint.
2. **Sprint:** Es el periodo por el cual se desarrollaran los objetivos previamente establecidos en la planificación.
3. **Seguimiento del sprint:** Durante el sprint es necesario llevar un registro del trabajo realizado para así tener un mayor control del tiempo y la calidad del trabajo.
4. **Revisión del sprint:** Es la última etapa del ciclo, donde se analizan los resultados obtenidos al final del proyecto.

Normalmente los sprints se realizan en paralelo, en varios grupos de trabajo, para llegar con mayor rapidez a los objetivos finales. En este caso, puesto que solo hay una única persona trabajando en el proyecto, se llevan a cabo las iteraciones de forma individual.

Presentación del problema y análisis de requisitos

El objetivo final de este proyecto es el de crear una aplicación móvil basada en consultas a la base de datos de la plataforma Moodle. El usuario final puede tener acceso a información acerca de cualquiera de las asignaturas en las que está matriculado. Dicha información consta de un calendario donde están agrupados todos los eventos de todas las asignaturas. Además, dispone de una bandeja de entrada en la cual los mensajes de todos los foros están presentes. Por otra parte, se puede consultar información del curso acerca de los recursos subidos o de los participantes inscriptos en las diferentes materias. Finalmente, el profesor puede consultar mediante estadísticas todas las actividades realizadas por los alumnos dentro de sus cursos.

Requisitos funcionales

A la hora de crear la aplicación hay que tener en cuenta cuales son los objetivos finales de esta y que requisitos debe cumplir para satisfacer las necesidades de nuestro cliente, en este caso, todos los alumnos y profesores matriculados en la Universidad de Cantabria.

A continuación, se muestra la información relativa a las funcionalidades asociadas a nuestra aplicación al finalizar el proyecto:

Información sobre asignaturas

La aplicación permite consultar los datos relacionados con las asignaturas en las que el alumno está inscrito. Estos son:

1. Información genérica de la asignatura.
2. Los recursos facilitados, tales como PDFs, archivos de texto, páginas web, etc., deben ser visualizados por el usuario o profesor.
3. Información detallada de los participantes del curso.

Calendario

El usuario puede visualizar un calendario donde están todos los eventos de todas las asignaturas en las cuales el usuario este matriculado. Además, los datos se muestran de tres formas diferentes:

1. Un calendario mensual, donde quedan plasmados los días en los que el usuario tenga algún evento.
2. Una lista de eventos. El usuario puede visualizar en ella todos los acontecimientos que tiene a lo largo del curso. Muestra toda la información relativa a dicho hecho.
3. Una lista de eventos semanal. En este caso, sólo se muestra la semana actual y todas las actividades que ocurren en ella.

Foros

En este caso, se detallan los mensajes de todos los foros en los que el usuario este matriculado. Estos indican:

1. Qué usuario lo ha mandado y a qué asignatura pertenece.
2. Una imagen del perfil del usuario para una identificación más rápida del mensaje.
3. La cabecera y el contenido del mensaje.
4. En el caso de que sea una contestación a otro mensaje, se pueden conocer los datos de este.

Actividad

Y por último, esta funcionalidad solo queda reservada para el usuario que tenga el rol de profesor. Este puede consultar las estadísticas de los alumnos en los diferentes cursos en los que desempeña su labor docente. La información se resume en los movimientos realizados por el alumno dentro de la asignatura: consultas de recursos, de temas del foro, número de veces que ha entrado dentro del curso, etc.

Requisitos no funcionales

Conocer las funcionalidades del proyecto es importante, pero también lo es identificar otros requisitos que hacen que el sistema funcione de las formas más óptima y correcta.

A estos requisitos se les conoce como no funcionales, y son:

1. La identificación del usuario ha de ser segura y fiable. Solo deben tener acceso a la aplicación aquellos usuarios que están registrados dentro de la plataforma Moodle.
2. El sistema debe ser óptimo. No debe bloquearse, ni permitir que un usuario pueda bloquear la plataforma Moodle permitiendo así que otros usuarios no se vean afectados.
3. Debe ser un sistema apto para diferentes dispositivos y sistemas operativos.

Diagrama de casos de uso

Diagrama general

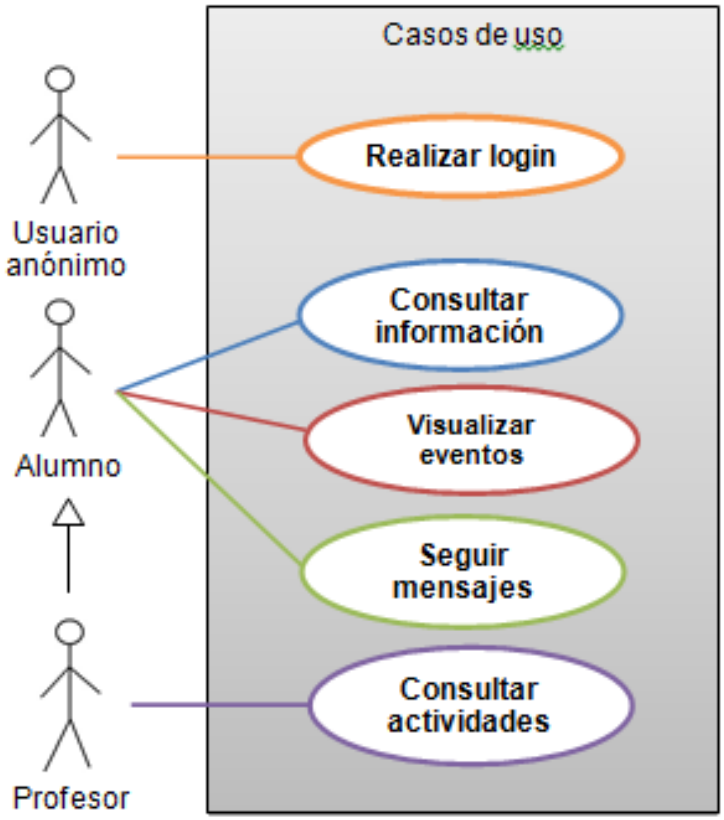

Imagen 2: Diagrama general de casos de uso.

Como se muestra en la anterior figura, correspondiente al diagrama de casos de uso de la aplicación.

Primero hay un usuario anónimo que puede identificarse dentro del sistema introduciendo un usuario y una contraseña.

Por otra parte, el alumno registrado tiene las funcionalidades necesarias para consultar información, observar el calendario y ver los foros. Igualmente, el profesor dispone de esas mismas características, incluyendo una nueva, consulta del registro de actividades de cada usuario dentro de un curso mediante estadísticas.

A continuación, desglosaremos cada caso para analizar con más claridad la funcionalidad de cada uno.

Diagrama de Información

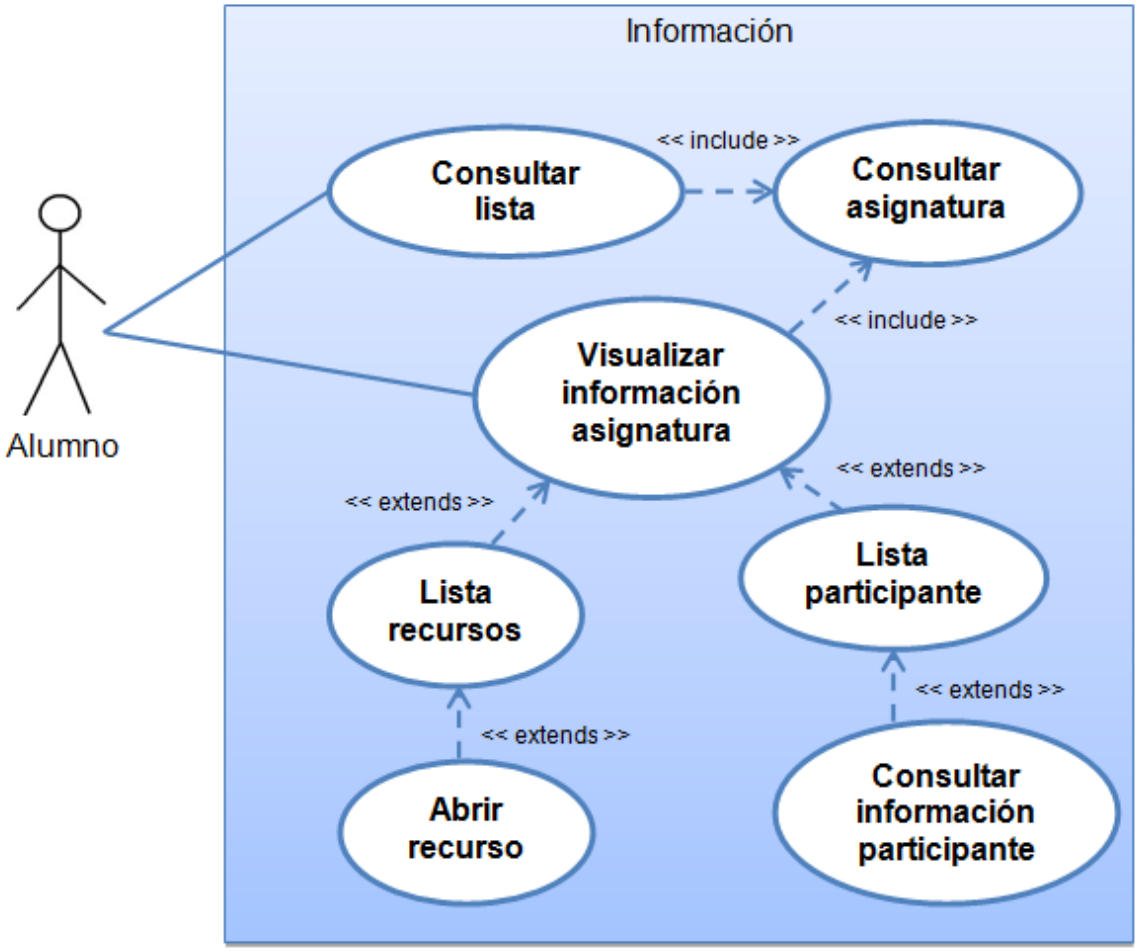

Imagen 3: Casos de uso de la funcionalidad de Información

En este primer caso, ilustra la Información referente a los cursos en los que está inscrito el usuario identificado. Dicho usuario, tanto alumno como profesor, podrá ver la lista completa de todas las asignaturas en las que está matriculado. Además, una vez seleccionado el curso, el usuario puede acceder a todos los recursos de este, así como conocer los datos de cada participante.

Diagrama de Calendario

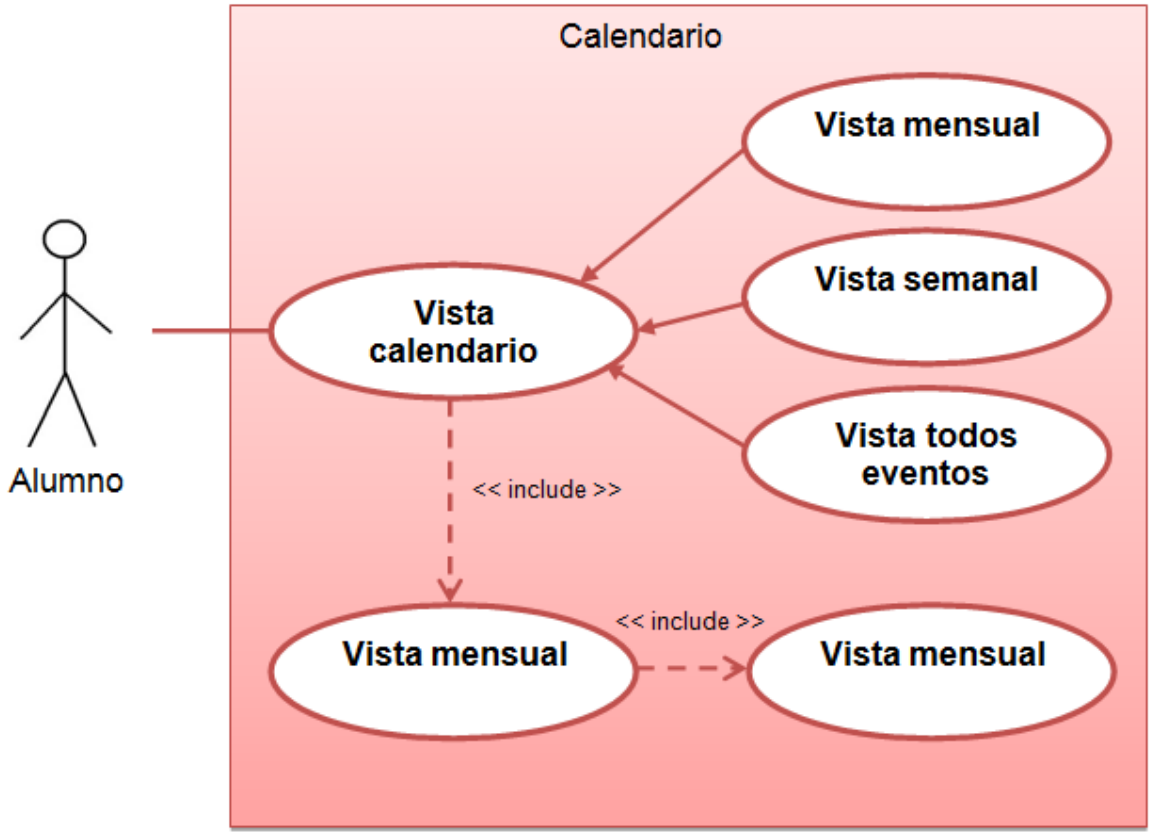

Imagen 4: Casos de usos de la funcionalidad de Calendario

En este segundo caso, el usuario puede visualizar de tres maneras diferentes los eventos que tiene en su agenda. La peculiaridad de esta funcionalidad es que agrupa todos los eventos de todos los cursos en una sola vista, en vez de tener que ir curso por curso.

La primera forma de visualización, es un calendario mensual donde se señala en que día hay un evento asignado. La segunda, es una lista con todos los eventos ordenados. Y por último, la consulta es semanal, es decir, la aplicación muestra la semana actual y los eventos de esta.

Diagrama de Foro

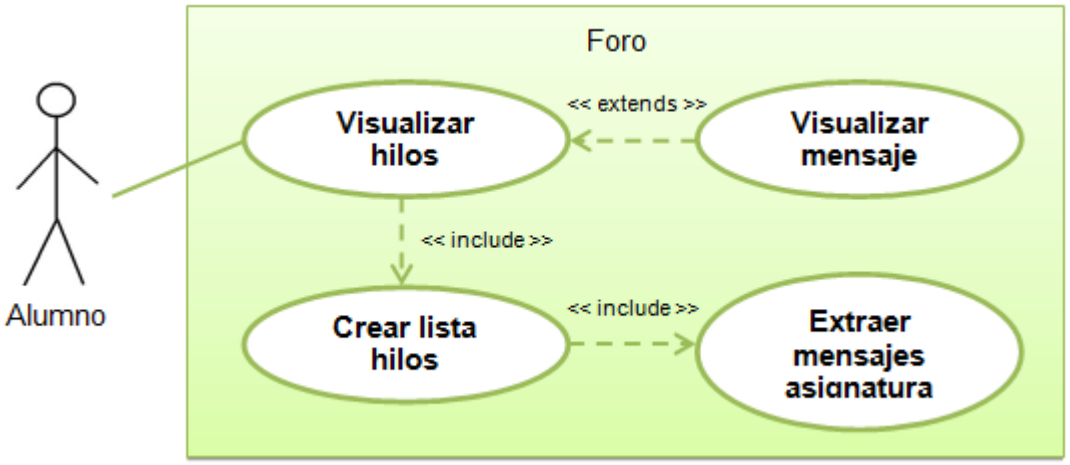

Imagen 5: Casos de uso de la funcionalidad de Foro.

En este caso de uso el usuario podrá ver en una bandeja de entrada todos los mensajes que se han ido generando en todos los cursos en los que el usuario está matriculado. En cualquier caso, el alumno o el profesor, con tan solo acceder al mensaje, podrá ver toda la información relativa a este.

Diagrama de Actividad

Imagen 6: Casos de uso de la funcionalidad de Actividad

Y por último, esta funcionalidad solo está disponible para el rol de profesor. Este podrá consultar todas las actividades realizadas por los alumnos dentro del curso en el que imparte clases.

Diseño

En este apartado vamos a describir el diseño de la aplicación. Según el estándar ISO 12207, se identifican dos tipos de diseño software: El diseño arquitectónico y el diseño detallado. A continuación, se pasa a explicar ambos diseños.

Diseño arquitectónico

Este tipo de diseño se caracteriza por ser de alto nivel. Describe la estructura, es decir, los componentes y sus relaciones. A la hora de hacer el diseño detallado es necesario realizar este paso previo. Además representa el enlace que hay entre la especificación de requisitos y el diseño.

El diseño arquitectónico escogido es un estilo caracterizado por usarse en sistemas distribuidos y su nombre es “arquitectura en tres capas”.

Arquitectura en tres capas

Este estilo se caracteriza por estructurar el diseño en tres capas fundamentales:

- Capa de presentación
- Capa de negocio
- Capa de datos

Esta arquitectura surge para poder llevar el desarrollo a varios niveles, es decir, en el caso de realizar algún cambio en alguna capa, el resto de capas no se pueden ver afectadas.

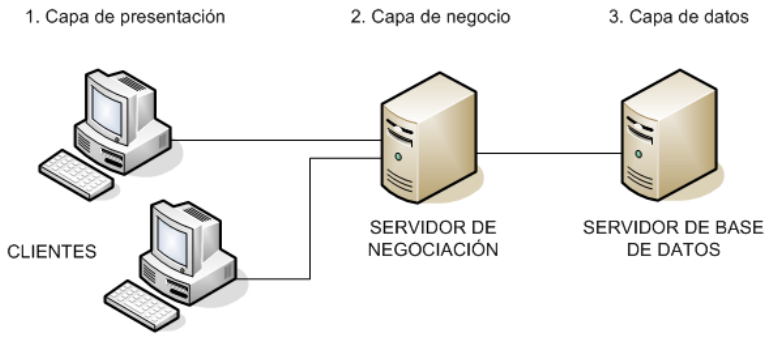

Imagen 7: Arquitectura en tres capas

Capa de presentación

- Interactúa con el usuario, es decir, comunica la información y captura los datos del usuario.
- Esta capa se comunica únicamente con la capa de negocio.
- Al ser la apariencia de nuestra aplicación (la interfaz gráfica) tiene la obligación de ser amigable, entendible y fácil de usar.

Capa de negocio

- Es donde están ubicados los programas.
- Se comunica con la capa de presentación para recibir las solicitudes y presentar los resultados.
- Se comunica con la capa de datos para solicitar al gestor de la base de datos almacenar o recuperar datos de él.

Capa de datos

- En esta capa es donde residen los datos.
- Se comunica solamente con la capa de negocio.
- Está formada por uno o más gestores de bases de datos. Estos gestores son los que realizan las labores de almacenar o recuperar información en base a unas solicitudes mandadas desde la capa de negocio.

Hemos optado por utilizar este tipo de arquitectura debido a que si en un posible proyecto futuro queremos crear una aplicación nueva, simplemente habría que cambiar la capa de aplicación y la de datos pudiendo reutilizar el modelo que hemos implementado.

En nuestra aplicación se pueden ver bien diferenciadas las tres capas:

- **Capa de presentación**: Está compuesto por un único archivo de extensión HTML. Además cuenta con otros archivos de extensión CSS que ayudan a mejorar la apariencia de la aplicación.

- **Capa de negocio:** Es un único archivo Javascript que se encarga de recoger las solicitudes enviadas por el usuario y de realizar el intercambio de datos con la base de datos.

- **Capa de datos:** Es la base de datos creada por Moodle. Mediante archivos de extensión PHP alojados en el servidor, realizamos las llamadas a la base de datos con la finalidad de obtener los datos necesarios.

Imagen 8: Arquitectura en tres capas de la aplicación.

Diagrama de componentes

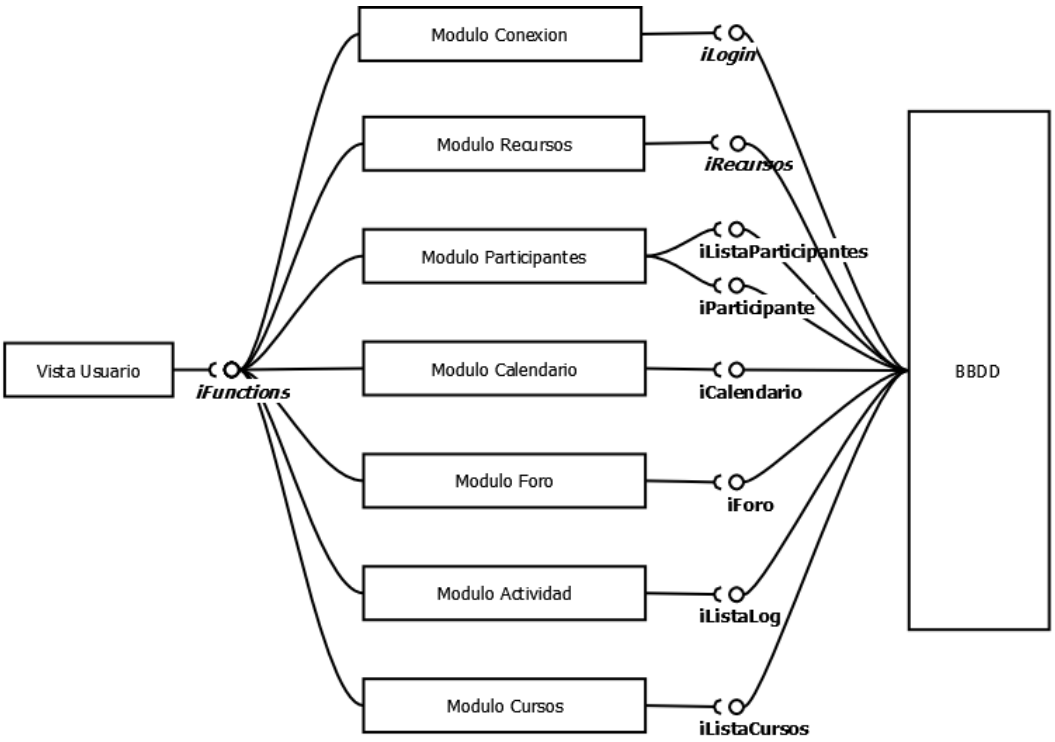

Imagen 9: Diagrama de componentes

Interfaces

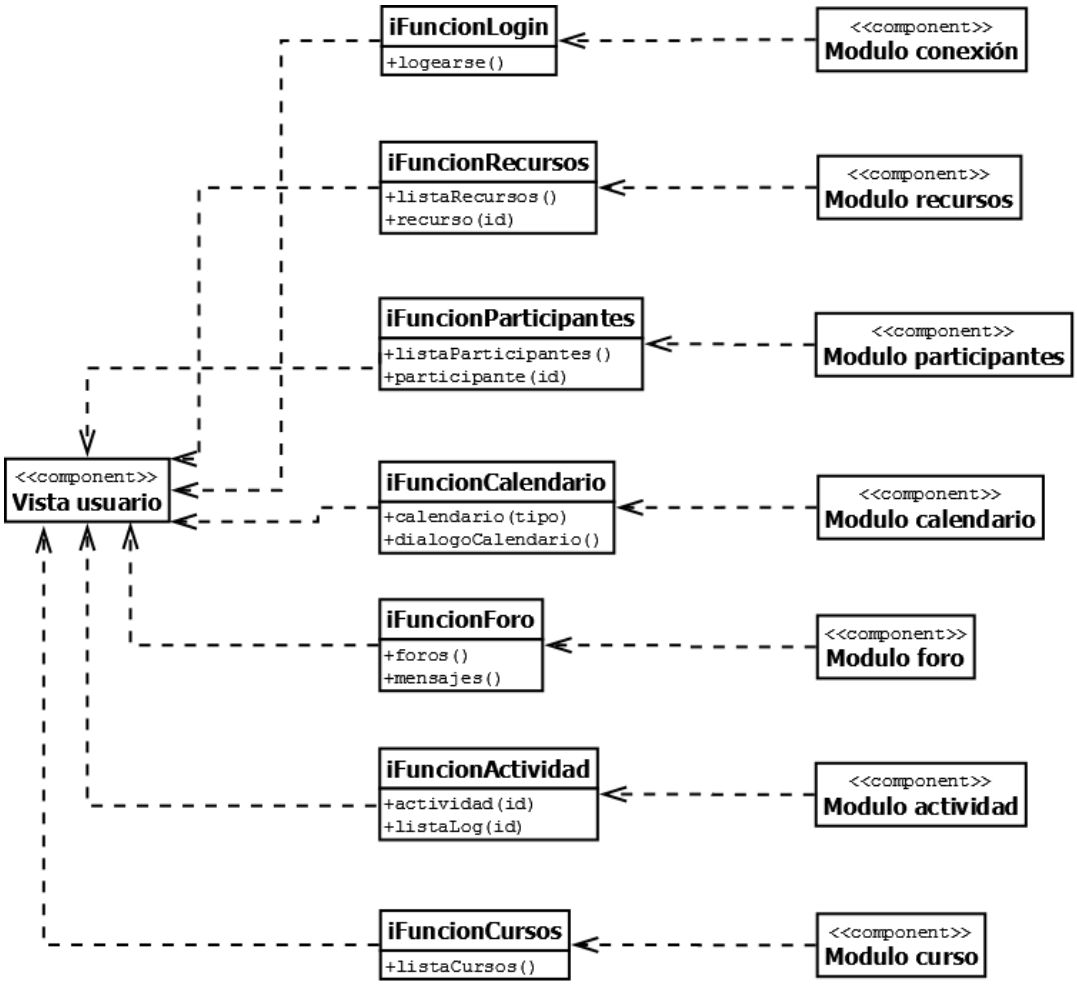

Imagen 10: Interfaces entre la capa de presentación y la capa de negocio

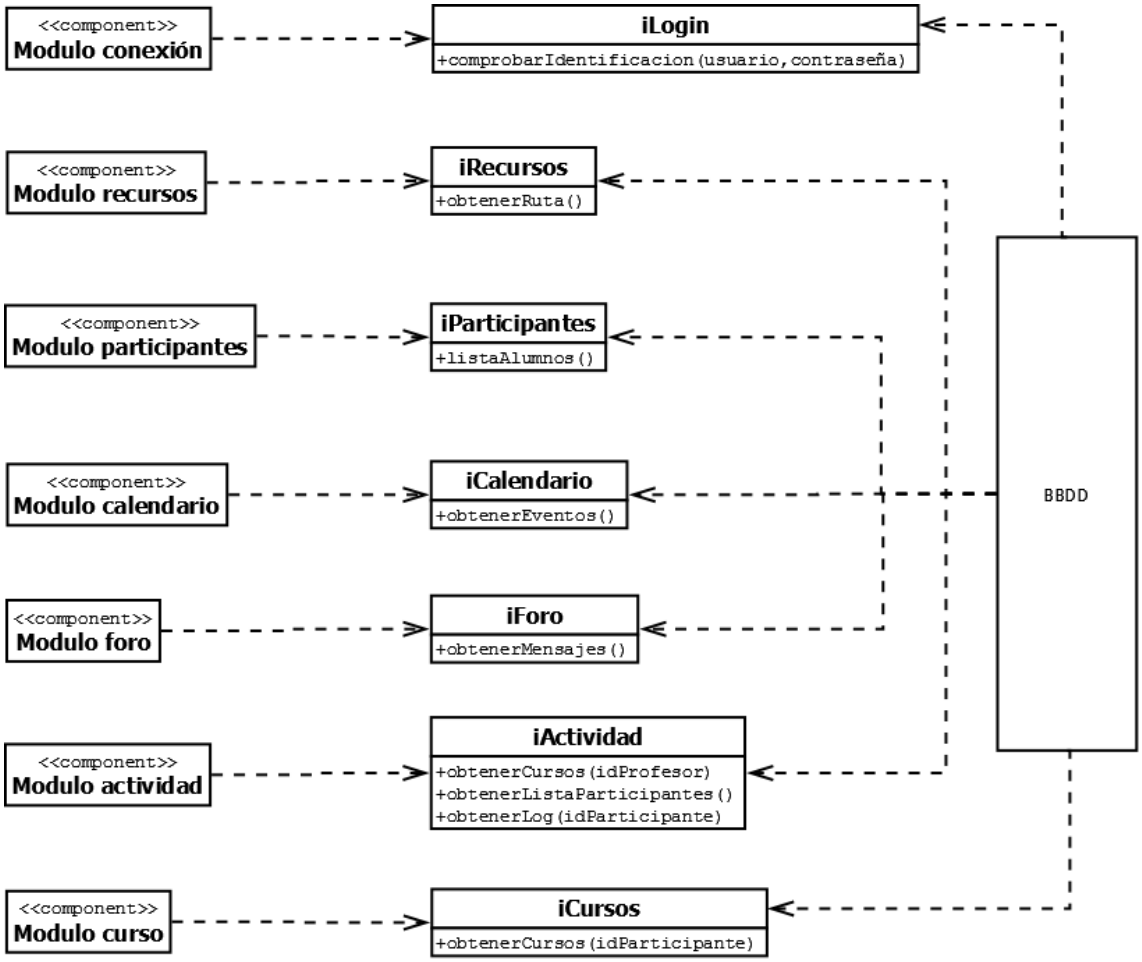

Imagen 11: Interfaces entre la capa de negocio y la capa de datos

Diseño detallado

Servidor y características de Moodle

Para poder diseñar nuestra aplicación es necesario comprender primero como está compuesta la base de datos de Moodle. Además, debemos conocer cómo y dónde va a estar instalada. Por lo tanto, dedicaremos la siguiente sección a describir todo esto.

Moodle está programado en PHP, lo cual hace que sea bastante flexible en lo referente al tiempo de máquina en la que se ejecuta. Además, los principales sistemas operativos como Unix, Mac OSX, Windows, etc. soportan este lenguaje permitiendo así una gran independencia en cuanto al tipo de servidor en el que se va a ejecutar la aplicación.

La base de datos de Moodle es única y ofrece a sus usuarios una total abstracción de su capa, permitiendo así elegir entre diferentes motores de bases de datos. En la actualidad, la última versión de Moodle es la 2.5 (14 de mayo 2013) y es capaz de soportar los siguientes motores:

- MySQL
- PostgreSQL
- MSSQL
- Oracle
- SQLite

En las primeras versiones de Moodle solamente se podía utilizar los motores de MySQL y PostgreSQL pero a medida que la plataforma evolucionaba, se han ido incluyendo nuevos motores.

Hay que destacar que también ha evolucionado en lo referente a los requerimientos de la máquina del cliente. Moodle, al ser una plataforma web, soporta multitud de navegadores en sus distintas versiones. Los más utilizados: Google Chrome, Mozilla Firefox, Internet Explorer, Safari y Opera, son capaces de soportar dicha plataforma.

El lugar de instalación de la plataforma debe ser un servidor, para que los futuros usuarios puedan acceder a la herramienta sin problema. En nuestro caso, el tipo de servidor web es un tipo de servidor local. Esto supone la ventaja de poder realizar pruebas sin limitaciones ya que no tenemos acceso a la plataforma Moodle de la Universidad de Cantabria.

El servidor de ámbito local escogido recibe el nombre de WAMP, cuyas siglas significan:

- **Windows**, como sistema operativo
- **Apache**, como servidor web
- **MySQL**, como gestor de bases de datos
- **PHP**, como lenguaje de comunicación.

Existen más servidores locales, cada uno con sus características. Sin embargo, este servidor se ajusta perfectamente a nuestras necesidades. Moodle soporta una base de datos con un gestor MySQL. Además el sistema operativo en el que trabajamos es Windows. Y por último, utilizamos como lenguaje de comunicación PHP, para conectar la capa de datos con la capa de negocio.

Base de datos de Moodle

Una vez realizada la selección del servidor y concretadas las características de Moodle, podemos pasar a estudiar la base de datos que contiene dicha plataforma.

La base de datos consta de 205 tablas. Están divididas en módulos, es decir, se agrupan en base al módulo al que describen, donde definen sus propiedades y relaciones con el resto de módulos dentro de los diferentes contextos de la herramienta.

Conocer cada módulo es obligatorio para poder comprender el almacenamiento de los datos y así conseguir realizar las consultas de forma satisfactoria. A continuación, se detalla cada Módulo, qué representan y cómo tienen los datos almacenados.

Tabla mdl_context

Es una de las tablas más importantes de Moodle. Su función principal es la de relacionar los diferentes módulos que forman la base de datos. Está compuesta por los siguientes campos.

- **ID:** Clave primaria de la tabla. Identifica el contexto del recurso al que hace referencia**.**
- **Contextlevel:** Es el tipo de recurso. Dependiendo del valor de este campo, se indica el nivel del contexto. Estos valores vienen definidos en el fichero accesslib.php (/lib/acceslib.php), y por defecto contiene los siguientes valores:
  - 10 - Sistema
  - 20 - Vacío
  - 30 - Usuario
  - 40 - Categoría
  - 50 - Curso
  - 60 - Grupo
  - 70 - Modulo
  - 80 - Bloque
- **Instanceid:** Identificador del recurso que origina el contexto. Con este campo se explica perfectamente el campo contextlevel. Por ejemplo, nosotros tenemos un recurso con instanceid = '7' y contextlevel='50'. Esto significa que el recurso que origina este contexto es el curso con id=7.
- **Path:** representación de la ruta en jerarquía, es decir, de que contexto superior es subcontexto el que viene definido en esta fila.
- **Depth:** Profundidad dentro de la jerarquía de contextos.

El campo instanceid, es clave ajena de muchas tablas al mismo tiempo, pero esta clave quedará definida en base al valor que tome el campo contextlevel. Esto quiere decir que evitamos tener que crear muchos campos ajenos con sus respectivas restricciones que deberían asegurar que solo uno de esos campos tenga valor no nulo.

Tabla mdl_user

Esta tabla contiene los datos referentes a las personas que están registradas en Moodle. Esta información puede encontrarse en la sección de perfil del usuario. Además, aparecen todos los usuarios independientemente del rol o de la importancia que estos tengan. Los campos más importantes están descritos a continuación:

- **Id**: Identificador del usuario. Es uno de los campos más importantes a la hora de buscar datos relacionados con el usuario.

- **Username**: Nombre de usuario. Este es el nombre que tiene que utilizar el usuario a la hora de acceder a la plataforma de Moodle.

- **Password**: Contraseña del usuario. Esta contraseña está cifrada con una encriptación basada en el algoritmo MD5 (Message-Digest Algorithm 5). Además, para poder desencriptar la contraseña, es necesario acceder a una semilla, la cual está ubicada en el archivo “config.php”. Con la unión del algoritmo MD5 y la semilla, podemos descifrar la contraseña y por lo tanto poder verificar si el usuario ha introducido los datos correctamente.

- **Picture:** Este campo es importante para saber si un usuario tiene una imagen de perfil. En la aplicación lo consultaremos para poder colocar la imagen adecuada. Los valores que pueden tomar este campo son 0 (no tiene imagen) o 1 (el usuario ha subido una imagen). Para obtener dicha imagen habrá que acceder a otras tablas.

- **Otros campos significativos:** Otros campos importantes son firstname (nombre real del usuario), lastname (apellido del usuario), e-mail, pagina web personal, Skype, etc. Estos campos adicionales completan el resto de datos acerca del usuario. Son importantes a la hora de querer saber más acerca de la persona registrada.

Tabla mdl_role

Para conocer todos los posibles roles que puede adquirir un usuario, es necesario consultar esta tabla. Cada rol posee su campo de identificación y su nombre. El campo más importante para nuestra aplicación es el campo archetype en el cual está definido el nombre técnico de cada rol.

Tabla mdl_role_assignments

Esta tabla permite conocer qué rol tiene asignado cada usuario relacionando ambas tablas. Para ello utiliza los siguientes campos:

- **Roleid:** Identificador del rol.
- **Userid:** Identificador del usuario.
- **Contextid:** Este campo es bastante importante. Es el identificador de contexto. Nos ayuda a la hora de relacionar un usuario con cualquier otro módulo de la plataforma. Más tarde se explicará mejor este campo cuando expliquemos la relación entre un usuario y un curso.

Mediante el uso de claves ajenas, esta tabla relaciona usuarios y roles. Asigna a cada id de usuario, un id del rol. Al ser claves ajenas, un mismo usuario puede adquirir varios roles, es decir, un mismo usuario puede ser profesor y alumno a la vez.

Tabla mdl_course

Indica los cursos que hay creados. De la misma forma que la tabla de usuarios, esta tabla dispone de los siguientes campos:

- **Id:** Identificador del curso. Mediante este campo podremos relacionar el curso con otras tablas.
- **Fullname:** Nombre completo del curso. Este campo sirve para indicar que nombre figura dentro de la plataforma y es la manera más sencilla de identificar el curso.
- **Shortname:** Nombre corto del curso. Este campo cumple la misma función que el anterior pero con un nombre más breve para una identificador más rápida.

Para conocer qué profesor está impartiendo las lecciones y qué alumnos son los que las están recibiendo, hay que tener en cuenta las tablas anteriormente descritas.

Gracias al campo contextid de la tabla mdl_role_assignments podemos relacionar al usuario con el curso. ¿Cómo es posible esto? Sencillo. Obtenemos el contextid del usuario (tabla mdl_role_assignments). Todos los id de la tabla mdl_context que sean igual que contextid son módulos relacionados con el usuario. Ahora identificamos los cursos gracias a contextlevel=50 y obtenemos la relación deseada.

Como hemos explicado anteriormente, la tabla mdl_context es fundamental a la hora de relacionar módulos.

Tabla mdl_event

Gracias a esta tabla podemos conocer todos los eventos que hay creados dentro de la plataforma. Sus características son relativamente sencillas como se pueden ver a continuación:

- **Id:** Identificador el evento.
- **Name:** Nombre que recibe el evento.
- **Description:** Descripción del evento.
- **Courseid:** Identificador del curso al que pertenece el evento.
- **Userid:** Usuario que ha creado el evento.
- **Timestart:** Fecha en la que se creó el evento.
- **Timeduration:** Fecha en la cual, el evento finaliza.

El campo courseid es el más importante. Gracias a él podemos relacionar el evento con el curso. Además la suma de los campos Timestart + Timeduration da como resultado la fecha de entrega del evento.

Tabla mdl_forum_discussions

Esta tabla hace referencia a las discusiones registradas dentro de la plataforma. En ningún momento hace referencia a los post creados dentro de dichos temas, simplemente indica que hay un hilo iniciado. Para cumplir con esta finalidad, la tabla cuenta con los siguientes campos:

- **Id:** Identificador del tema.
- **Course:** Curso donde está creada la discusión.
- **Name:** Nombre de la discusión.
- **Firstpost:** En la tabla donde están referenciados todos los post creados sobre el tema, este campo indica cual fue el primer post.
- **Userid:** Quien fue el creador del tema.
- **Timemodified:** Actualización de la última vez que se escribió en este tema.

Tabla mdl_forum_posts

Esta tabla sirve para complementar la anterior. Conociendo los temas que están creados podemos hallar en esta tabla todas las respuestas referentes a estas discusiones. Los siguientes campos muestran como están estructurados los mensajes:

- **ID:** Identificador del post.
- **Discussion:** Indica a que discusión pertenece el post. Es la clave ajena que relaciona esta tabla con la tabla mdl_forum_discussions.
- **Parent:** Este campo es bastante importante. Su misión es organizar los post, es decir, indica a que post contesta el post actual ya que en Moodle se permite contestar a cualquier mensaje.
- **Userid:** Usuario que creó el post.
- **Created:** La fecha en la que se creó el post.
- **Modified:** La última vez que se modificó el mensaje.
- **Subject:** Cabecera del mensaje enviado.
- **Message:** Contenido del mensaje.

En definitiva, relacionando estas dos tablas podemos obtener toda la información necesaria para navegar por los foros y consultar todos los mensajes enviados.

Tabla mdl_log

En esta tabla quedan registradas todas las acciones realizadas por todos los usuarios dentro de Moodle. En los siguientes campos se muestra como esta almacenada esta información:

- **ID:** Identificador de la acción realizada por el usuario.
- **Time:** Fecha exacta en la que se realizó dicha acción.
- **Userid:** Identificador del usuario.
- **Ip:** IP desde donde se cometió la operación.
- **Course:** Curso donde la acción fue hecha.
- **Module:** Categoría de la Acción que se ha realizado, es decir, si dicha acción ha sido realizada sobre el modulo curso (como por ejemplo puede ser, ver curso) en este campo aparecerá el módulo course.
- **Action:** Tipo de acción realizada. Las acciones más típicas son consultar curso (view), logearse (login), crear discusión (add discussion) y muchas otras más.
- **url:** Dirección web donde la acción tuvo lugar.
- **info:** Identificador del afectado por la acción. Esto quiere decir que si la acción es por ejemplo, course view (ver curso), el identificador indicará que curso ha sido visto por el usuario.

Tabla mdl_log_display

Con la anterior tabla es suficiente para poder mostrar en la aplicación las acciones realizadas por el usuario, sin embargo, Moodle muestra los datos de otra forma más correcta y más simplificada. Los siguientes campos que se muestran a continuación aportarán una información más detallada de esta tabla:

- **ID:** Identificador del tipo de dato.
- **Module:** Es el mismo campo que la tabla anterior. En ella se muestra que modulo ha sido afectado por la acción.
- **Action:** También muestra (igual que en la anterior tabla) las acciones realizadas por los usuarios.
- **Mtable:** En este campo, se muestra que tabla se ve afectada según qué acción ha realizado el usuario.
- **Field:** Indica que campos de la tabla son los que se van a mostrar.
- **Compontent:** Modulo donde se realizó la acción.

Por lo tanto, gracias a esta tabla podemos obtener una visualización más correcta, gracias a que muestra los campos de las tablas afectadas en vez del identificador.

Tabla mdl_resource

A la hora de consultar los recursos que nos proporciona cada curso, es necesario recurrir a esta tabla. En ella están todos los recursos que se han ido suministrando en todos los cursos de la plataforma. Sin embargo, en esta tabla solo podemos obtener el nombre del recurso. Los campos de esta tabla son los siguientes:

- **ID**: Identificador del recurso.
- **Course**: Identificador del curso al cual pertenece el recurso.
- **Name**: Nombre del recurso.
- **Intro**: Descripción detallada del recurso, proporcionada previamente por el usuario que lo subió a la plataforma.
- **Timemodified**: Indica cuando se subió o se modificó el archivo.

Tabla mdl_files

Esta tabla es una de las más importantes debido a que en ella se encuentra la localización de todos los ficheros subidos a Moodle. Tiene una forma particular de almacenar los archivos, sobre todo desde la actualización 2.0. En versiones anteriores se utilizaba otro método más básico pero menos seguro.

La ruta donde se almacenan los archivos es: C:/wamp/moodledata/filedir/XX/YY donde los campos XX e YY los obtenemos del campo contentHash de la tabla mdl_files. Los dos primeros caracteres de este campo corresponden a los valores XX y los dos siguientes a los valores YY. De esta forma hemos conseguido obtener la ruta donde están almacenados los ficheros de la plataforma Moodle.

De esta forma, mediante la dirección del archivo podemos forzar a que el usuario lo descargue o que lo abra.

Herramientas utilizadas

Debemos tener presentes los objetivos finales del proyecto para poder elegir de forma correcta las herramientas que nos facilitarán el camino del éxito.

Por suerte para nosotros, el importante uso de los Smartphone hoy en día, ha provocado que el número de aplicaciones móviles se haya incrementado notablemente en los últimos tiempos. A su vez, las plataformas para el desarrollo software de dichas aplicaciones móviles ha ido avanzado conforme esto ha evolucionado.

Antes, si una persona quería crear una aplicación móvil, debía conocer obligatoriamente el código nativo del sistema operativo y por lo tanto, programar en esa versión especifica de la plataforma. Esto requiere una gran cantidad de tiempo si la persona quiere programar una aplicación multiplataforma. Actualmente, ya no hace falta estar programando de esta forma, sino que existen frameworks que nos facilitan notablemente el desarrollo de aplicaciones móviles multiplataforma. Los dos más importantes son Titanium Appcelerator y PhoneGap. Nosotros emplearemos este último a la hora de crear nuestra aplicación.

Phonegap

PhoneGap es un framework para el desarrollo de aplicaciones nativas de sistemas operativos móviles, haciendo uso de tecnologías web como HTML5, CSS3 y JavaScript. Es posible desarrollar aplicaciones para los siguientes sistemas operativos:

- Android.
- iOS.
- Windows Phone.
- BlackBerry OS.
- Web OS.
- Symbiam.
- Bada.

Al tratarse en realidad de una página web, tenemos acceso al DOM y podemos usar otros frameworks web como por ejemplo jQuery Mobile o Sencha Touch, que nos permiten realizar un diseño aparentemente nativo de forma sencilla con los componentes visuales típicos del HTML.

La página web oficial de Phonegap nos facilita un archivo comprimido con una carpeta perteneciente a cada sistema operativo. En esta carpeta, hay una librería Javascript y otra en el lenguaje nativo que usa la plataforma para desarrollar aplicaciones. En el caso de Android, por ejemplo, existe una librería en Javascript usada para el desarrollo de aplicaciones web, permitiendo el acceso al hardware del dispositivo a través de APIS. Además, garantiza el proceso de compilación, convirtiendo a Phonegap en puente de acceso a herramientas nativas del sistema operativo y del hardware del dispositivo tales como el acelerómetro, la cámara, los contactos, eventos, la geolocalización, las redes o el almacenamiento, entre otros.

Además Phonegap facilita en su página web varios tutoriales o guide started para configurar todo el entorno de trabajo de forma satisfactoria y así poder trabajar sin problemas.

Es necesario conocer los siguientes lenguajes de programación para poderles compilar luego con Phonegap.

HTML 5.0

Es una colección de estándares para el diseño y desarrollo de páginas web. Representa la manera en que se presenta la información en el explorador de internet y la forma de interactuar con ella

Es el lenguaje más utilizado a la hora de crear páginas web. Se utiliza para describir y traducir la estructura y la información en forma de texto, así como para complementar el texto con objetos tales como imágenes.

La forma de escribir el código, es entre etiquetas rodeadas por corchetes angulares (<,>). Y la estructura del programa es siempre la misma: una cabecera y un cuerpo. En la cabecera es donde se encuentra toda la información que no se va a mostrar en el navegador web, cosas como las referencias a otros ficheros o las propiedades que tendrá la página. Por otra parte en el cuerpo se describirá como será visualmente la página.

La última versión es la 5.0 donde se introducen algunos cambios significativos con respecto a las otras versiones:

- **Simplificación:** El nuevo código ofrece nuevas formas, más sencillas, de especificar algunos parámetros y piezas de código.
- **Contenido multimedia:** Reproducción de audio y video sin necesidad de plugins.
- **Almacenamiento de datos del lado del cliente:** Una diferencia fundamental entre las aplicaciones de escritorio y web era la necesidad de éstas últimas de procesar la información y consultas en bases de datos siempre en un servidor, haciendo que las aplicaciones sean más lentas y siempre requeridas de una conexión a Internet constante. HTML5 permite almacenar y procesar información en el cliente, convirtiendo una aplicación web en una aplicación mucho más parecida a una de escritorio.
- **Efectos y nueva versión de hojas de estilo CSS:** La nueva versión de HTML se acompaña de una nueva versión de las hojas de estilo CSS, el CSS3. Muchas de las cosas que hasta ahora solo podrían lograrse insertándolas como imágenes, pueden realizarse con código. Esto no solo se traduce en una mejora de la velocidad y rendimiento de un sitio, sino también en nuevas e ilimitadas opciones de diseño.
- **Geo-localización:** Los sitios web pueden conocer la ubicación física de la persona que lo visita.

CSS 3.0

CSS es el acrónimo de **C**ascading**S**tyle **S**heets (es decir, hojas de estilo en cascada). Es un lenguaje de estilo que define la presentación de los documentos HTML. Abarca cuestiones relativas a fuentes, colores, márgenes, líneas, altura, anchura, imágenes de fondo, posicionamiento avanzado y muchos otros temas. Si HTML se usa para estructurar el contenido, CSS se usa para formatear el contenido previamente estructurado.

A medida que la Web fue ganando popularidad, los diseñadores empezaron a buscar posibilidades para añadir formato a los documentos en línea. Para satisfacer esta reclamación, los fabricantes de los navegadores (en ese momento, Netscape y Microsoft) inventaron nuevas etiquetas HTML, entre las que se encontraban, por ejemplo, <font>, que se diferenciaba de las etiquetas originales HTML en que definían el formato y no la estructura.

Esto también llevó a una situación en la que las etiquetas estructurales originales, por ejemplo, <table>, se usaban cada vez más de manera incorrecta para dar formato a las páginas en vez de para añadir estructura al texto. Muchas nuevas etiquetas que añadían formato, por ejemplo, <blink>, sólo las soportaban un tipo determinado de navegador. "Necesitas el navegador X para visualizar esta página" se convirtió en una declaración de descargo común en los sitios web.

CSS se inventó para remediar esta situación, proporcionando a los diseñadores web con sofisticadas oportunidades de presentación soportadas por todos los navegadores. Al mismo tiempo, la separación de la presentación de los documentos del contenido de los mismos, hace que el mantenimiento del sitio sea mucho más fácil. Por lo que una misma web, con un mismo contenido, podía cambiar totalmente su apariencia reemplazando simplemente el fichero CSS por otro diferente, lo que supuso una enorme ventaja para diseñadores web.

JavaScript

JavaScript se ha convertido en el lenguaje de programación del lado del cliente más utilizado hoy en día gracias a su sencillez, rapidez, ligereza y compatibilidad con la mayoría de los navegadores actuales.

Es un lenguaje que permite a los desarrolladores crear acciones en sus páginas web. Además, no requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos.

Permite crear efectos en las páginas web para introducir contenidos dinámicos, y elementos de la página que tengan movimiento, cambien de color o cualquier dinamismo. Además, permite tratar todos los eventos o acciones producidos por el usuario.

Javascript es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros.
Con el surgimiento de lenguajes como PHP del lado del servidor y Javascript del lado del cliente, surgió Ajax en acrónimo de (Asynchronous Javascript And XML). El mismo es una técnica para crear aplicaciones web interactivas. Este lenguaje combina varias tecnologías:

- HTML y Hojas de Estilos CSS para generar estilos.
- Implementaciones ECMAScript, uno de ellos es el lenguaje Javascript.
- XMLHttpRequest es una de las funciones más importantes que incluye, que permite intercambiar datos asincrónicamente con el servidor web, puede ser mediante PHP, ASP, entre otros.

Debemos tener en cuenta que aunque Javascript sea soportado en gran cantidad de navegadores nuestros usuarios pueden elegir la opción de Activar/Desactivar el Javascript en los mismos.

JQueryMobile

jQuery Mobile es un framework desarrollado por jQuery que combina HTML5 y jQuery para la creación de portales web móviles. Nos permite generar aplicaciones cuya apariencia será siempre la misma independientemente del dispositivo desde el que acceda un usuario siempre que acepte HTML5.

Este framework nos provee de ciertas herramientas que nos hacen la tarea de crear una página mucho más sencilla. Con unas pocas asignaciones de atributos HTML podremos generar increíbles interfaces muy usables y accesibles.

La sintaxis es parecida a la de Javascript, pero han ampliado el número de funciones y métodos para dar cabida a todas las nuevas funcionalidades de HTML 5 cmo la geolocalización, acelerómetro o eventos que detecten el control con el dedo sobre la pantalla. Cada componente tiene sus propias funcionalidades Javascript y sus propios métodos de control. En la documentación se pueden consultar los métodos y eventos que soportan.

Implementación

Como hemos definido antes, nuestra aplicación está compuesta por tres partes bastante diferenciadas entre ellas. Estas son: la parte visual, la parte funcional y la parte de consultas a la base de datos.

Parte visual

Esta parte está programada en el lenguaje HTML y junto a sus hojas de estilo CSS forman el aspecto visual de la aplicación. La información más importante referida al diseño está contenida en un único fichero llamado index.html. La aplicación móvil tiene un formato web estructurado en un sistema basado en multipáginas, es decir, reunimos todas las páginas en un único archivo facilitando así las llamadas entre ficheros.

El archivo index.html se divide en dos partes:

- **Head**: Aquí se encuentran las referencias a otros ficheros, que utiliza la página web para que todo funcione de manera óptima. Los ficheros más importantes son las librerías de JQueryMobile, el archivo JavaScript y la hoja de estilos CSS que proporciona una mejora visual a nuestra aplicación.

- **Body**: Es la parte donde se encuentran todas las páginas de la aplicación. Además, aquí se definen todos los elementos que necesita el usuario para interactuar con el sistema. Gracias a las librerías de JQueryMobile, podemos crear elementos predefinidos que nos facilitará el trabajo a la hora de crear la aplicación.

El resultado conjunto de lo mencionado anteriormente se refleja en las siguientes imágenes:

Imagen 12: Identificación.

Imagen 13: Menú principal del alumno.

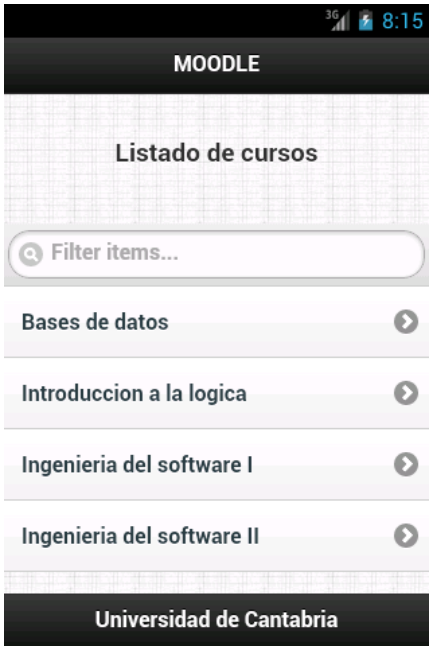

Imagen 14: Pantalla cursos.

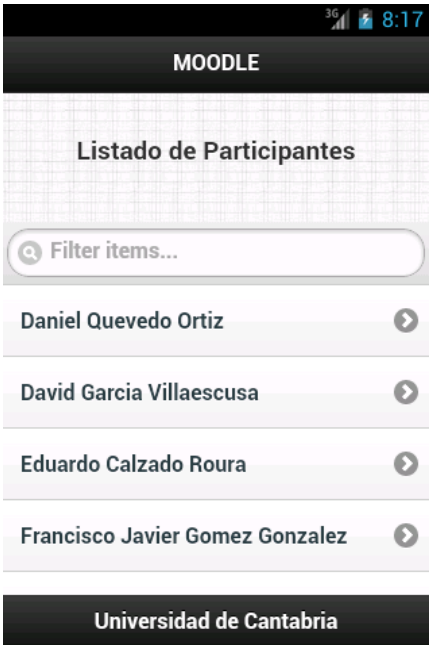

Imagen 15: Pantalla participantes.

Imagen 16: Perfil del participante.

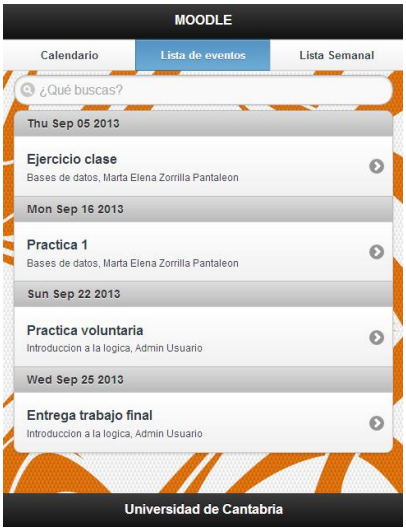

Imagen 17: Lista de eventos del calendario.

Imagen 18: Foro.

Imagen 19: Registro de actividades.

Parte funcional

Es la encargada de manejar todos los elementos de la parte visual que interactúan con el usuario final de la aplicación. Además, tiene otras funcionalidades tales como la de comunicarse con el servidor realizando llamadas a archivos alojados en él. Dichos archivos tienen la extensión .php y facilitan la conexión entre la parte funcional y la base de datos. En resumen, es el nexo que hay entre ambas partes, la visual y la base de datos.

Por otra lado, es fundamental que esta capa esté conectada con la visual, es decir, con el archivo .html, para así poder recoger y mostrar los datos que necesita el usuario en todo momento. Mediante funciones JavaScript, el usuario podrá manejar botones, listas, cambiar de página, etc. En definitiva, JavaScript puede controlar tanto elementos estáticos como dinámicos, lo cual es fundamental para datos que pueden cambiar en función de la decisión del usuario.

Otro punto a tener en cuenta, es la conexión entre un archivo de extensión JavaScript con otro en formato php. Esta unión es posible gracias al lenguaje Ajax, con el cual es posible realizar cambios sobre páginas sin necesidad de recargarlas. Este lenguaje es capaz de intercambiar datos con los archivos .php alojados en el servidor. Es fundamental para poder recoger los datos necesarios de la base de datos.

En nuestra aplicación solo hay un único archivo JavaScript encargado del manejo de todas las funciones. El nombre de este archivo es functions.js. Se compone de dos partes diferenciadas:

- El tratamiento de elementos. Son funciones que únicamente esperan a que un usuario interactúe con algún elemento de la parte visual y así actuar en consecuencia.
- Funciones que intercambian datos con los archivos alojados en el servidor. No todas las funciones realizan esto pero si la gran mayoría. El proceso de vida de estas funciones es el siguiente: obtener los datos del usuario, mandar dichos datos como parámetros al servidor, recoger su respuesta e insertarla en la parte visual.

Llamadas a la base de datos

Una vez detalladas la parte visual y la parte funcional, queda definir esta última. Es la encargada de realizar las consultas a la base de datos de Moodle. Los archivos cuya finalidad es realizar esta labor, son los que tienen la extensión .php. Puesto que la base de datos se encuentra alojada dentro del servidor, estos ficheross, deben estar también dentro de este.

El lenguaje de programación PHP tiene unas librerías implementadas con funciones que permiten interactuar con una base de datos con un gestor MySQL. Estos archivos se encargan de recoger los parámetros enviados desde el documento JavaScript, tratarlos en función de las necesidades y devolver unos nuevos.

Para realizar una conexión con la base de datos es necesario tener en cuenta lo siguiente:

- **Cabecera**: En ella se encuentran todos los permisos necesarios para poder establecer la conexión con la base de datos.

- **Mysql_connect**: Es una función que viene implementada en las librerías de PHP. Este método es necesario para conectarse con la base de datos alojada dentro del servidor. Retorna true o false dependiendo de si la conexión se ha realizado con éxito o no.

- **Mysql_select_db**: Una vez realizada la conexión, es necesaria seleccionar la base de datos que queremos utilizar. Esta función esta implementada para realizar esta acción.

Una vez realizada la conexión con éxito, el siguiente paso es recibir los parámetros, en el caso de que fueran necesarios. Para ello se utiliza el método $_GET seguido del nombre de la variable de la cual queremos obtener su valor.

Para poder realizar consultas con la pase de datos, PHP dispone de los siguientes métodos:

- **Mysql_query**: Este método es el que realiza la consulta con la base de datos. Es necesario pasarle dos parámetros:
  - Una cadena de texto con la consulta que se quiere realizar
  - La variable $conexion inicializada anteriormente con las funciones encargadas de realizar la conexión con éxito.

- **Mysql_num_fields**: Este método es importante para saber si la consulta ha sido satisfactoria o no. Podremos conocer el número de valores retornados por lo que si este ha sido cero, la consulta no ha encontrado ningún valor.

- **Mysql_fetch_row**: Esta función nos ayuda a obtener los valores en función de las filas. Existe una función parecido para retornar los valores en columnas, pero esta es más utilizada. Es importante para leer los datos devueltos por la consulta.

La clasificación de los ficheros PHP del sistema es la siguiente:

- Existe un fichero .php para cada función del documento JavaScript que quiera realizar una consulta a la base de datos. En cada fichero se trata los datos de una manera específica y única. Lo que tienen en común estos ficheros son los métodos necesarios para realizar la conexión y las consultas con la base de datos.

- Además, existe otro fichero donde todas las funciones creadas están agrupadas. Cada fichero del grupo anterior, utiliza este fichero .php para realizar su cometido.

Base de datos de Moodle

Puesto que la aplicación realiza consultas a la base de datos de Moodle, es necesario que la plataforma contenga los datos necesarios de los cursos, foros, alumnos, etc. Para ello, simplemente entraremos a la plataforma Moodle desde nuestro Navegador web y mediante la cuenta de administrador, crearemos los usuarios, los cursos, los eventos, los foros y los recursos necesarios para poder realizar las pruebas oportunas.

Funcionamiento

A continuación mostraremos lo anteriormente descrito mediante código para poder ver con más claridad cómo funciona este sistema. El ejemplo empleado es como un usuario se identifica dentro del sistema para poder acceder a él.

```
<div data-role="page" id="inicio">
	<div data-role="header" data-position="fixed">
		<h1>MOODLE</h1>
	</div><!-- /header -->

	<div class="cuerpo" data-role="content">
		<form>
			<center><h3> Identificación </h3></center>

			<label for="text">Nombre de usuario:</label>
			<input data-clear-btn="true" name="login" id="login"
			value="type="text" placeholder="User" data-theme="e">

			<label for="password">Password:</label>
			<input data-clear-btn="true" name="password" id="password"
				value="" autocomplete="off" type="password"
					placeholder="Password" data-theme="e">
		</form>

		<a id="botonLogin" data-role="button">Entrar</a>
	</div><!-- /content -->

	<div data-role="footer" data-position="fixed">
		<h4>Universidad de Cantabria</h4>
	</div><!-- /footer -->
</div><!-- /page -->
```

El anterior fragmento de código corresponde a la parte visual de la identificación. La página consta de una cabecera, un cuerpo y un pie de página. En el cuerpo de la página web podemos observar dos input data. Estos campos a rellenar son los correspondientes al nombre de usuario y a la contraseña. Además hay un botón para realizar el login más abajo. En el momento en el que este se pulse, activará un evento controlado por el archivo JavaScript.

El código JavaScript que tratará de recoger estos datos y de mandarlos a la base de datos es el siguiente:

```
$(document).on("pageshow","#paginaLogin",function() {
	// Activar o desactivar boton login
	$('#usuario').off('click').on('click',function() {
	if ($("#password").val() != "") {
			$('#botonLogin').removeClass('ui-disabled');
		} else if ($("#password").val() == "" && $("#usuario").val() == "") {
			$('#botonLogin').addClass('ui-disabled');
		}
	});
	// Activar o desactivar boton login
	$('#password').off('click').on('click',function() {
		if ($("#usuario").val() != "") {
			$('#botonLogin').removeClass('ui-disabled');
		} else if ($("#password").val() == "" && $("#usuario").val() == "") {
			$('#botonLogin').addClass('ui-disabled');
		}
	});
	// Pulsar boton 'login'
		$('#botonLogin').off('click').on('click',function() {
			self.login();
		});
	});
```

Esta parte del código se encarga de verificar si el usuario ha pulsado en el botón Login. Además como añadido, esta tratado de tal forma que hasta que un usuario no haya rellenado uno de los dos campos, el botón no estará activo.

Una vez que todo está correcto y el usuario pulsa sobre el botón, se llama a la función login, la cual se describe a continuación:

```
// Función al pulsar el botón 'login'
login:function() {
	usuario = $("#usuario").val();
	var clave = $("#password").val();
	var valoresLogin = {'usuario': usuario, 'clave': clave};

	// AJAX -> Interactuar con el archivo .php del server
	$.ajax ({
		type: "get";
		dataType: "html",
		url: 'http://'+variableIP+'/login.php',
		data: valoresLogin,

		beforeSend : function () {
			console.log('Enviando...');
		},
		error : function (err) {
			console.log('Error'+err);
			alert("Error de conexion: "+err);
		},
		success : function (response) {
			console.log('Cambiando de pagina');
			var respLogin = response.split("*");
			if(respLogin[0].trim(" ") == 'correcto') {
				if(respLogin[1] == "student") {
					$("#botonActividad").hide();
				} else {
					$("#botonActividad").show();
				}
				$.mobile.changePage("#paginaMenu",{transition:"slide"});
			} else {
				alert("Usuario o password mal introducidos")
			}
		},
		complete : function (response, err) {
			console.log('Completado!! '+ response);
		}
	});
},
```

La función captura en las variables ‘usuario’ y ‘clave’ los valores recogidos de los campos definidos en el archivo .HTML. Además esos valores se guardan en una variable llamada valoresLogin que será enviada como parámetro al fichero login.php para poder realizar una consulta a la base de datos.

Como se puede apreciar en el código, dentro de la función definida en JavaScript, existe una parte correspondiente al lenguaje AJAX, la cual es la encargada de enviar y recibir datos con el servidor. Para ello, debemos definir cuatro variables para una correcta sincronización:

- **Type**: Tipo de petición que vamos a realizar. En nuestro caso colocamos GET ya que solo realizaremos funciones de consulta.
- **DataType**: Tipos de datos que la función espera de vuelta.
- **URL**: Es la dirección donde este alojado el fichero. Dentro del fichero JavaScript, está viene una variable llamada variableIP que contiene dicha información.
- **Data**: Parámetros que enviamos al fichero .php. En este caso el nombre del usuario y la contraseña.

Además, definimos cuatro tipos de funciones que se activarán en función de la situación.

- **BeforeSend**: Esta función se activa siempre antes de enviar los datos,
- **Error**: En este caso, la función se activará cuando la conexión no se ha podido realizar con éxito. Es bastante importante para poder localizar el fallo.
- **Success**: Esta es la más importante, ya que se activará en caso de que la llamada se haya realizado con éxito. Tiene como parámetro una variable llamada response donde viene todos los datos retornados por el fichero. Dentro de esta función es donde se realiza el posterior tratamiento de datos.
- **Complete**: En este caso, esta función saltará cuando el proceso de llamada a un fichero haya terminado independientemente de si ha tenido éxito o no.

Dentro de la función **Success,** en el caso de que la llamada tenga éxito, recogemos los datos, en nuestro ejemplo, una cadena de texto con dos valores: El primero verifica si el usuario se puede identificar satisfactoriamente y el segundo comprueba el rol de este. En el caso de que el usuario fuera un estudiante, ocultaremos el botón de la funcionalidad referente a la Actividad para evitar que pueda acceder a ella. Si todo ha ido correctamente, cambiaremos de página, al menú principal.

Para finalizar, la parte de código encargada de recibir los datos de la llamada Ajax y realizar la consulta a la base de datos es la siguiente:

```
<?php

	header("Access-Control-Allow-Origin: *");

	include("functions.php");

	if(!($conexion = mysql_connect("localhost", "root"))){
		echo("Error en conection");
		exit();
	}

	mysql_select_db("moodle24",$conexion);

	$user = $_GET['usuario'];
	$pass = $_GET['clave'];
	$semilla = 'MMl<G.Te ,}5`9rOCv!asiLTuK';
	$hash = md5($pass . $semilla);

	if($user == 'guest' || $user == 'admin') {
		echo "fallo*";
	} else {
		$queryUser = "Select username, password, id from mdl_user where username = '" .$user . "'";

		$consultaUsuario = mysql_query($queryUser, $conexion);

		if( mysql_num_fields($consultaUsuario) > 0 ){
			$datosUser = mysql_fetch_row($consultaUsuario);
			if($hash == $datosUser[1]) {
				$userid = $datosUser[2];
				echo "correcto*";
			}
		} else {
			echo "fallo*";
		}
		$sol = obtenerRol($userid, $conexion);
		echo $sol;
	}
?>
```

La primera línea se encarga de conceder los permisos oportunos para poder acceder al fichero. La segunda, sin embargo, incluye un archivo con todas las funciones definidas para poder usarlas posteriormente.

Como ya se explicó, utilizamos el método mysql_connect con el nombre del servidor y el usuario de la base de datos para poder conectarnos a ella. Retorna una variable $conexion que será importante para poder realizar las consultas más tarde. Por otra parte, el método mysql_select_db, nos ayuda a seleccionar la base de datos que queramos consultar.

Los valores que nos envía la llamada Ajax son recogidos en variables mediante la función $_GET. Estos valores son necesarios para poder realizar con éxito la llamada a la base de datos.

Este fichero está definido de tal forma que realizaremos una consulta a la tabla mdl_user para verificar si existen un usuario y una contraseña como las que ha introducido el usuario previamente. La contraseña, al estar encriptada, debemos tratarla de forma especial; mediante una semilla y un método de cifrado md5.

En el caso de que la comparación sea satisfactoria, retornamos “correcto”, en caso contrario, devolvemos “fallo”. Finalmente, llamamos a la función obtenerRol definida dentro del fichero functions.php (incluido arriba) para comprobar si el usuario es un estudiante o no.

La única manera posible de pasar los datos de un fichero a otro es mediante una cadena de texto. Esto genera un problema cuya solución pasa por colocar un asterisco detrás de cada “valor” para facilitar la separación de estos dentro del fichero JavaScript.

Pruebas

Para un correcto funcionamiento del software, es necesario pasar por una serie de pruebas que nos garantice un producto de garantía y sin fallos.

Debido a la metodología aplicada en nuestro proyecto, se han ido realizando pruebas por cada iteración garantizando que hasta que un módulo no esté funcionando perfectamente, no pasamos a crear otro.

El objetivo de cada prueba es realizar un test para todos los posibles casos en los que la aplicación pueda fallar para ser capaces de detectar los errores y así solucionarlos con la mayor rapidez posible.

Existen cuatro tipos de pruebas:

- Pruebas unitarias.
- Pruebas de integración.
- Pruebas de sistema.
- Pruebas de aceptación con el cliente.

Pruebas unitarias

Este tipo de prueba se centra en la búsqueda de errores de cada módulo de forma independiente. Para cada módulo se han ido realizando las pruebas oportunas. Puesto que la aplicación consiste únicamente en la consulta de información y no en la inserción de esta, hemos introducido valores diferentes en la base de datos de Moodle para poder comprobar varios casos distintos. Además, es necesario comprobar si los datos extraídos se visualizan correctamente dentro de la aplicación.

En nuestra aplicación, el único caso en el que un usuario introduce datos dentro de la aplicación es para la identificación del mismo. Por lo tanto, hemos tenido que probar varios casos para verificar que no existe ningún error dentro de este módulo. Para ello hemos probado lo siguiente:

- Usuario incorrecto.
- Contraseña mal introducida
- Usuario y contraseña mal introducidos

En todos estos casos, el sistema no permite la entrada al contenido de la aplicación. Además está controlado de tal forma que tampoco pueden entrar ni el administrador, ni el invitado.

Por otra parte, el resto de casos solamente se basan en consulta de datos. Para verificar estos casos, hemos optado por introducir diferentes valores dentro de las tablas para comprobar que la consulta se realiza correctamente.

El caso que más problemas ocasionó fue el de identificar el rol del usuario. La dificultad fue que un mismo usuario puede adquirir más de un rol y por lo tanto, había que tratar ese caso de forma especial. En lo referente a la aplicación, significa que si un usuario es profesor y alumno a la vez, en las funcionalidades de información del curso, calendario y foros, debían mostrarse los cursos en los que el usuario es alumno y profesor. Sin embargo, en la funcionalidad de la actividad solamente debía mostrar los cursos en los que el usuario imparte clases. Como añadido, existen dos tipos de profesores: 'EditingTeacher' y 'Teacher' por lo fue necesario tratar estos dos roles como uno solo.

Pruebas de integración

Una vez testeados los módulos por separado, es momento de hacer las pruebas de cada módulo dentro del sistema para ver si se ha conseguido integrar sin ningún tipo de error. Se ha ido realizando durante cada sprint, por lo que este tipo de prueba ha seguido un tipo incremental.

Pruebas de sistema

Antes de implementar la aplicación dentro del sistema es necesaria una serie de pruebas para comprobar que el este funciona correctamente. Es imprescindible realizarlo antes de la implementación ya que así es más fácil encontrar los errores y la

solución sería menos costosa debido a que no habría que tratar dos tipos de problemas.

Dentro de este tipo de prueba, se abarcan campos tales como:

- **Pruebas recuperación**: Si un usuario tiene algún tipo de problema con la aplicación, la plataforma Moodle no se verá afectada debido a que la aplicación no altera la plataforma en ningún momento.

- **Pruebas de seguridad**: Ningún usuario podrá acceder desde la aplicación a datos que no le corresponden. Solamente puedes identificarte con tu usuario y consultar tus datos sin alterar nada más.

- **Pruebas de rendimiento**: Se han realizado las pruebas en dos dispositivos con sistema operativo Android. El primer dispositivo es un Samsung Galaxy Mini de gama baja, mientras que el segundo es una Tablet Wolder de gama media. Se ha comprobado que en el primer dispositivo los tiempos de respuesta son más lentos que en el segundo pero a pesar de todo, la aplicación funciona con total normalidad en ambos dispositivos. Se deduce pues, que la aplicación funciona correctamente en cualquier dispositivo con sistema operativo Android.

Pruebas de aceptación

El objetivo final de esta prueba es validar que el sistema cumple con el funcionamiento exigido y esperado. Para ello, se proporciona la aplicación al cliente para que determine su aceptación desde el punto de vista funcional y de rendimiento.

En mi caso, como alumno de la Universidad de Cantabria, he ido probando personalmente la aplicación en cada iteración del proyecto. Además he proporcionado la aplicación a otros alumnos de la universidad para que ellos mismos califiquen sus funcionalidades.

Conclusiones

Llegados a la finalización del proyecto, sólo cabe plasmar las conclusiones obtenidas al finalizar el trabajo, reflejando todos los progresos, tanto negativos (obstáculos y barreras) como positivos (logros), que nos hemos ido encontrando en la realización del mismo.

También hay que hacer mención al resultado obtenido partiendo de unos objetivos iniciales marcados. Para ello utilizaremos una tabla para plasmar dicha evolución.

Objetivo	**Crear una aplicación basada en la consulta de la base de datos de Moodle.**
Conclusión	Un usuario registrado en la plataforma Moodle podrá identificarse en su aplicación móvil para acceder en tiempo real a sus datos relativos a: • **Información** relativa a los cursos en los que está matriculado. • Visualizar todos los **eventos** que tiene pendientes. • Consultar todos los **mensajes** de los foros en los que participa. • Si es profesor, ver la **actividad** de los alumnos dentro del curso en los que imparte clase.

Objetivo	**Poder soportar las últimas versiones de Moodle a partir de la 2.0 en adelante**
Conclusión	Se ha conseguido realizar este objetivo gracias a que la versión 2.0 no ha habido grandes cambios significativos.

Objetivo	**Que se pueda utilizar para cualquier plataforma Moodle**
Conclusión	En un principio estaba orientado a Moodle de la Universidad de Cantabria pero funciona y se adapta a cualquier plataforma independiente.

Objetivo	**El sistema debe ser multiplataforma, es decir, que funcione en varios sistemas operativos diferentes.**
Conclusión	Debido a que se ha empleado con PHONEGAP este objetivo se ha podido realizar con resultado final positivo.

Conclusiones personales

En mi opinión, es recomendable estructurar el problema para lograr tener un trabajo organizado y claro. Para ello, dividir la aplicación en módulos me parece una forma de facilitar el manejo y uso de dicha aplicación. También, el realizar el trabajo de forma iterativa ayuda mejor a detectar los errores pudiendo así solucionarlos con mayor facilidad.

Por otra parte, esta experiencia trabajando en el proyecto ha sido bastante útil para mi formación, ya que he sido capaz de desenvolverme dentro de un entorno nuevo partiendo desde cero. Además he tenido la habilidad para organizarme, adaptarme y buscarme la vida para poder sortear todos los obstáculos que he ido encontrando.

Trabajos futuros

Como sucede en cualquier tipo de proyecto, siempre hay partes que se pueden mejorar e incluso añadir al trabajo ya realizado.

Nuestra aplicación únicamente realiza consultas a la base de datos por lo que una futura línea de mejora podría ser la inserción de datos dentro de esta, es decir, que el usuario pueda introducir información dentro de Moodle.

En relación a lo anterior, una nueva funcionalidad podría ser la capacidad del usuario para responder a los mensajes del foro desde el terminal móvil. Con esto, se podrá contestar nada más consultar el mensaje consiguiendo que el usuario ahorrase tiempo y que además, no se le olvide responder por pereza a la hora de conectarse vía web. Esto fomentaría la participación de los participantes dentro de los cursos.

Dentro de esta línea, se podría añadir la capacidad de que el alumno pueda responder a eventos a través del terminal. El problema de esta funcionalidad es que, en el caso de tener que subir un archivo a la plataforma, dicho archivo deberá estar dentro del Smartphone o Tablet del usuario y esto rara vez pasa. Sin embargo resultaría útil para eventos en los que solo haya texto.

Además, Moodle cuenta con la capacidad de mandar mensajes privados entre usuarios registrados dentro de la plataforma. Poder ver y responder estos mensajes sería una mejora significativa a pesar de que esta función apenas es utilizada dentro del sistema.

Bibliografía

[1] Ingeniería del Software, 9ª Edición.
Sommerville, 2012
Addison-Wesley.

[2] Patrones de Diseño.
E. Gamma, 2002
Addison-Wesley

[3] UML 2.0 in a Nutshell
D. Pilone, 2005
O' Reilly

[4] Writting Effective Use Cases
Alistair Cockburn, 2000
Addison-Wesley

[5] UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design
J. Arlow e I. Neustadt, 2005
Addison-Wesley

[6] Software Architecture: Foundations, Theory and Practice
Richard N. Taylor, Nenad Medvidovic y Eric M. Dashofy, 2009

[7] PhoneGap Begginer's Guide
Andrew, Lunny, 2011
Packt Publishing

[8] Página oficial de Moodle
http://www.moodle.org

[9] Página oficial Phonegap
http://phonegap.com/

[10] Página oficial JQueryMobile
http://jquerymobile.com/

[11] Javascript, HTML, CSS y PHP
http://www.w3schools.com