Sistemas Multimedia Basados en Internet

Entregables hito 0

nahiara latorre gómez, antonio martínez galvañ

**Diseño integral de la arquitectura técnica: especificación de los elementos del backend, tecnologías y relaciones entre ellos**

***Lenguajes y frameworks***

El lenguaje que vamos a utilizar para el desarrollo de nuestra aplicación web es **PHP** por ser un lenguaje abierto y libre, con una sintaxis sencilla, fácil de aprender y cumple los estándares básicos de programación orientada a objetos.

De todos los frameworks que tiene PHP hemos decidido usar **Symfony** para ayudar al desarrollo de nuestra web.

Otras de las razones por las que hemos elegido PHP como lenguaje de desarrollo web es que queremos hacer nuestra base de datos con MySQL y PHP dispone de paquetes totalmente autoinstalables tanto para UNIX(LAMPP) como para Windows(XAMPP), los cuales son fáciles de instalar y configurar. De esta forma tendríamos en nuestros ordenadores un servidor local con acceso a base de datos, evitando así posibles problemas de compatibilidades.

***Sistema de gestión de Base de Datos***

Como ya hemos adelantado en el apartado anterior el sistema de gestión de base de datos que vamos a usar es **MySQL,** ya que, entre otras características, es un DBMS multiplataforma, es relativamente fácil encontrar ayuda como tutoriales puesto que es usado por multitud de aplicaciones web, además, al igual que pasa con PHP, es fácil de aprender lo cual supone una ventaja para el programador y no es necesario un experto Administrador de Base de Datos. Y, por último, otra característica favorable es que se trata de un DBMS escalable.

***Servidor HTTP***

Para nuestro sistema vamos a usar el **servidor web Apache** ya que se trata de un servidor libre debido a que es un software Open Source con licencia GPL. Otras ventajas que tiene Apache y por las que nos hemos decantado por usar este servidor es que multiplataforma, es altamente configurable, es decir, tiene diseño modular y es extensible, y por último, soporta PHP, algo importante puesto que queremos usar dicho lenguaje para el desarrollo web.

***Sistemas de Colas***

Teniendo en cuenta que en nuestra base de datos seguramente habrán miles de usuarios y pueden estar realizando peticiones al mismo tiempo es necesario que implementemos un sistema de gestión de colas para que la respuesta del sistema de cara al usuario sea inmediata, aunque realmente el sistema tarde más tiempo. En cuanto a esta parte tenemos pensado mostrar páginas al usuario de que se está realizando la petición que ha hecho al sistema, es decir, crear tareas asíncronas.

**Especificación de elementos y tecnologías de seguridad y autenticación**

Para garantizar la seguridad de nuestro sistema, proveeremos al sitio de un certificado de seguridad SSL, el cual se puede o bien comprar a empresas que proporcionan este servicio, o se puede crear por nuestra parte, y protocolo TLS, para proteger los datos de los usuarios mediante el cifrado de los mismos.

La autenticación basada en nuestra página web será algo conocido → Usuario y Contraseña. Esta autenticación se entrega al usuario por parte del administrador del sistema, el cual crea el usuario asignando un rol específico, siendo imposible de modificar por cualquier rol que no posea los privilegios de administrador.

En la página de identificación de usuario dispondrá de un enlace para la recuperación de contraseñas de los usuarios, dicho enlace redireccionará al usuario a una página del sistema donde tendrá que seguir unas instrucciones para verificar que es usuario de la página. Dicha verificación será a través de su correo electrónico, y el sistema enviará una contraseña creada de forma aleatoria, una vez realizada dicha acción el usuario solo podrá acceder a la página web mediante esta contraseña. Los usuarios podrán cambiar su contraseña en su página de perfil.

En el caso de que decidiéramos implantar un servicio de e-mails entre padres y profesores, por ejemplo, y sobre el que pudiera circular información de carácter privado, dotaríamos a este servicio de un cifrado mediante PGP, un criptosistema híbrido que emplea clave pública muy empleado a la hora de dotar de confidencialidad al servicio de correo.

**Especificación API Rest y del modelo de datos**

User: operaciones realizadas sobre los usuarios

|  |  |
| --- | --- |
| POST/user | Petición para crear usuarios |
| GET/user/login | Login al sistema |
| GET/user/logout | Logout del usuario |
| DELETE/user/{username} | Borrar el usuario {username} |
| GET/user/{username} | Obtener todos los campos de datos del usuario {username} |
| PUT/user/{username} | Actualizar la información del usuario |
| GET/user/conversaciones | Conseguir el listado completo de conversaciones del usuario |
| POST/user/conversaciones | Crear una nueva conversación |
| POST/user/conversaciones/{conversacion\_id}/mensajes | Enviar un mensaje en la conversación {conversación\_id} |

Tablon: operaciones realizadas sobre el tablón de anuncios

|  |  |
| --- | --- |
| POST/asignatura | Crear una nueva asignatura |
| DELETE/asignatura/{id} | Borrar una asignatura |
| PUT/asignatura/{id} | Actualizar una asignatura |
| GET/asignatura/{id} | Conseguir todos los datos de una asignatura por su id |
| GET/asignatura/findByName | Conseguir todos los datos de una asignatura por su nombre |
| POST/asignatura/{id\_asignatura}/material | Subir un material de una asignatura |
| GET/asignatura/{id\_asignatura}/material/{id} | Conseguir todos los datos de un material id de la asignatura id\_asignatura |
| GET/asignatura/{id\_asignatura}/material/findByName | Conseguir todos los datos de un material de cierta asignatura buscando por nombre de material |
| DELETE/asignatura/{id\_asignatura}/material/{id} | Borrar el material id de la asignatura id\_asignatura |
| DELETE/asignatura/{id\_asignatura}/material/findByName | Borrar un material por su nombre de la asignatura id\_asignatura |

Curso: operaciones que se realizan sobre cada nivel educativo

|  |  |
| --- | --- |
| POST/curso | Crear un curso |
| DELETE/curso/{id} | Borrar un curso |
| PUT/curso/{id} | Actualizar un curso |
| GET/curso/{id} | Conseguir todos los datos de un curso por su id |
| GET/curso/findByName | Conseguir todos los datos de un curso por su nombre |

Informe: operaciones realizadas sobre el informe/expediente de un alumno

|  |  |
| --- | --- |
| POST/informe | Crear un informe/expediente |
| DELETE/informe/{id} | Borrar un informe |
| PUT/informe/{id} | Actualizar un informe |
| GET/informe/{id} | Conseguir todos los datos de un informe por su id |
| GET/informe/findByName | Conseguir todos los datos de un informe por su nombre |

Foro: operaciones realizadas sobre los temas creados en los foros

|  |  |
| --- | --- |
| POST/foro | Crear un tema en el foro |
| DELETE/foro/{id} | Borrar un tema en el foro |
| GET/foro/{id} | Conseguir todos los campos de un tema del foro por id |
| GET/foro/findByName | Conseguir todos los campos de un tema del foro por nombre |
| POST/foro/{foro\_id}/reply | Publicar un comentario en un tema del foro |
| DELETE/foro/{foro\_id}/reply/{reply\_id} | Borrar un comentario en un tema del foro |
| PUT/foro/{foro\_id}/reply/{reply\_id} | Editar un comentario en un tema del foro |

Debido a nuestra falta de experiencia a la hora de documentar APIs y a la gran variedad de herramientas para crear y especificar APIs Rest, no estamos seguros de qué framework emplear. No obstante, tras cierta investigación de frameworks habitualmente empleados para esta tarea, nos hemos decidido por emplear [Swagger](http://swagger.io), un framework gratuito con editor online que trabaja tanto sobre un lenguaje de especificación muy utilizado, como es JSON, como sobre YAML, más fácilmente leíble por humanos, lo que facilita que un usuario nuevo de la API se familiarice con una mayor rapidez con esta.