**INTRODUCCIÓN**

* Ante la falta de un sistema automatizado que almacene la información relacionada con las actividades extraescolares del centro, decidimos por sugerencia de Domingo Pérez realizar este proyecto para solventar dicho problema.
* Dotando al centro de una aplicación mediante la cual tanto profesores como alumnos pueden contribuir a dicha tarea y exponer sus opiniones y comentarios sobre las propias actividades extraescolares.
* Dicho proyecto nos ofrece la oportunidad de enriquecer nuestro conocimiento y hábitos de programación.

**REQUERIMIENTOS FUNCIONALES**

Son aquellos que afectan directamente a la estructura, forma, interfaz, etc de la aplicación.

HARDWARE (Equipos físicos)

* Requisitos mínimos para Ubuntu 18.04 LTS
* Características del equipo servidor

SOFTWARE

* Software empleado en desarrollo y pruebas.
* Software empleado en producción.
  + **Servidor LAMP**: Un Servidor LAMP es el formado por las tecnologías Linux como sistema operativo base, Apache Web Service como servicio web, MySQL como sistema de gestión de base de datos y PHP como lenguaje de programación, aunque este último puede ser sustituido por Perl o Python.

**CONFIGURACIÓN DEL SERVIDOR**

* Una vez instalados todos los servicios necesarios mediante repositorio, procedemos a configurar apache, ya que los demás servicios no requieren de una configuración previa a su uso.
  + Creamos una regla en el firewall para permitir la salida.
  + Habilitar HTTPS en el servidor (conexión SSL/TSL)

- Creación de certificado autofirmado para hacer uso del servicio seguro de http.

- Redirigimos las peticiones que vengan por el puerto 80 hacia el 443 para forzar la conexión mediate https. Editamos el fichero /etc/apache2/sites-enabled/000-default.conf .

* Conflictos derivados de la migración del entorno de desarrollo al de producción.

Debido a que XAMPP crea todos los ficheros y carpetas con el usuario root, y apache emplea el usuario www-data, a la hora de subir texto u imágenes, no tenia permisos para ello, por lo que se optó por cambiar dichas carpetas de propietario en lugar de aplicar permisos totales para otros usuarios.

**REQUERIMIENTOS NO FUNCIONALES**

Son aquellos que aun siendo necesarios no afectan directamente a la aplicación.

Los requerimientos no funcionales se representan según el siguiente diagrama.

* **Usabilidad**: La aplicación dispone de una interfaz sencilla e intuitiva. Además, se implementa el sistema “Responsive web Design” para favorecer la experiencia del usuario en dispositivos móviles.
* **Rendimiento**: El servidor debe contar con un mínimo de tiempo de respuesta ante solicitudes del usuario, proporcionado por un hardware adecuado y una buena conexión a internet.
* **Seguridad**: Hace referencia tanto a las regulaciones legales, como el uso de cookies obligando a presentar un aviso para aceptar que la aplicación guarde información de usuario en el navegador, como a los siguientes aspectos esenciales si hablamos de seguridad informática.
  + **Confidencialidad e integridad**: la conseguimos a través del login y de las comprobaciones que hacemos en cada redirección.
  + **Disponibilidad**: depende del propio servidor y de su conexión a internet.
* **Costo**: hemos investigado por internet y preguntado en nuestras empresas de practicas el costo de la realización de una aplicación de esta envergadura.

**ORGANIZACIÓN**

* Para un mejor seguimiento del trabajo realizado, como las tareas acabadas, fallos localizados, sugerencias pensadas para añadir, etc, hacemos uso de una aplicación que funciona como una agenda donde creamos tarjetas y anotamos todo lo necesario para llevar un control del desarrollo de la aplicación. Dicha aplicación es “Trello”.
* Del mismo modo, usamos un repositorio de código para que ambos podamos subir la tarea que acaba de realizar y así el otro alumno puede disponer del proyecto actualizado al instante. Dicha aplicación es “GitHub”, la cual permite un control de versiones.
* De este modo hemos podido sincronizarnos eficientemente, evitando así que ambos trabajásemos en la misma tarea.

**PLANIFICACIÓN**

* La realización del proyecto se divide en una serie de etapas mostradas en la siguiente grafica de Gantt
* Según la planificación resultante, el proyecto comprende una duración de 4 meses (120 días). Dado que hemos sido dos personas trabajando en paralelo, se estima de media 50 horas semanales dedicadas al proyecto entre los dos.

|  |  |  |
| --- | --- | --- |
|  | Semana | Tiempo empleado (días) |
| Reuniones | 19-feb | 14 |
| Análisis | 05-mar | 7 |
| Diseño | 12-mar | 14 |
| Modelo de datos | 26-mar | 21 |
| Modelo Relacional | 26-mar | 7 |
| Modelo Lógico | 02-abr | 7 |
| Modelo Físico | 02-abr | 7 |
| Desarrollo | 26-mar | 70 |
| Cabecera | 26-mar | 14 |
| Índex | 02-abr | 7 |
| Cursos | 02-abr | 14 |
| Proyectos | 09-abr | 14 |
| Perfil de usuario | 09-abr | 7 |
| Cpanel Admin | 16-abr | 35 |
| Cpanel Profesor | 23-abr | 28 |
| Cpanel Alumno | 07-may | 28 |
| Documentación | 07-may | 28 |
| Formación | 21-may | 14 |

**ARQUITECTURA**

* El primer punto a tratar sobre la arquitectura empleada se basa en el modelo CLIENTE/SERVIDOR.
* En las aplicaciones web existen diversas variantes de la arquitectura básica según como se implementen las diferentes funcionalidades de la parte servidor, hemos optado por usar un único servidor para todos los servicios.

**ESTRUCTURA DEL SITIO WEB**

Estructura física

* La estructura física de un sitio web es la forma en que se almacenan los distintos recursos (ficheros) que forman un sitio web en el sistema de archivos del servidor web, en nuestro caso hemos optado por la siguiente estructura organizada por tipo de ficheros.

Estructura lógica

* La estructura lógica o de navegación define como un visitante se va a mover (navegar) de una zona a otra de un sitio web. La estructura lógica y la física son totalmente independientes, aunque se pueden definir de forma que una sea una proyección de la otra.
* Existen distintos tipos de estructuras lógicas de navegación. Las más comunes son la secuencial, en rejilla, en árbol y en red, aunque lo normal es que en un sitio web se de una combinación de todas ellas (estructura mixta).

La estructura creada busca cumplir con los siguientes aspectos.

* Regla de las tres pulsaciones (tres clics): Los usuarios buscan rapidez y efectividad.
* Evitamos los callejones sin salida: Páginas que no poseen enlaces para continuar la navegación.
* Los visitantes sabrán en todo momento dónde se encuentran y cómo volver a la página principal.
* Como añadido, creamos una barra de navegación o menú del sitio web con las categorías principales en que se ha organizado la información.
* La opción de búsqueda facilita a los visitantes la localización de la información que buscan.
* Indicamos en todo momento la información de contacto del centro.

**MODELO DE DATOS**

* En la realización de la base de datos se ha optado por el modelo relacional.
* La base de datos de la aplicación esta implementada mediante MySQL, debido a que permite utilizar diferentes lenguajes de programación, además de ser la más extendida a nivel mundial.
* Para la creación del modelo entidad-relación hemos utilizado la herramienta Día.
* Igualmente, para la creación del modelo físico optamos por la aplicación Toad Data Modeler.

**MODELO ENTIDAD-RELACIÓN**

**MODELO FÍSICO**

**TECNOLOGÍA Y LENGUAJES UTILIZADOS**

Para la creación del código fuente de la aplicación hemos optado por el uso de los siguientes lenguajes de programación.

* **HTML5**: Incorpora etiquetas con un significado semántico, lo que facilita su comprensión.
* **CSS3**: Permite controlar el estilo y el formato de múltiples páginas Web al mismo tiempo.
* **JavaScript**: Debido a su versatilidad hemos optado por su uso para que el usuario interactúe con la aplicación desde el front-end.
* **PHP**: lenguaje utilizado para todo lo relacionado con el back-end.
* **AJAX**: Utilizado para modificar ciertas secciones de la aplicación sin recargar toda la página.
* **MySQL y SQL:** lenguaje de consultas empleado para interactuar con la base de datos MySQL.

**LIBRERÍAS EXTERNAS**

* **JQUERY:** librería JavaScript usado en select múltiple entre otros.
* **BOOTSTRAP**: framework css creado por Twitter usado por ejemplo en la ventana modal de las cookies.
* **CLASS PHPMAILER**: clase php usada para el envio masivo de correos mediante el protocolo smtp (servidores Google)
* **CKEDITOR**: [editor de texto](https://es.wikipedia.org/wiki/Editor_de_texto) [HTML](https://es.wikipedia.org/wiki/HTML) de [código abierto](https://es.wikipedia.org/wiki/C%C3%B3digo_abierto) que proporciona funciones de procesador de texto en páginas web (cpanelalumno.php)
* **GOOGLE CHARTS**: Api de Google que crea gráficos estadísticos a través de consultas a la BBDD (cpanelindex.php)

**SEGURIDAD**

Para securizar la aplicación hemos seguido los siguientes métodos.

* **INICIO DE SESION:** Controlamos el acceso a los datos mediante un sistema de inicio de sesión que otorga privilegios dependiendo del rol del usuario.

Ejemplo de encriptado

* **USO DE VARIABLES DE SESIÓN Y NO DE COOKIES:** Debido a que el uso de cookies es bastante inseguro, optamos por el uso de variables de sesión para almacenar los datos del usuario.
* **CONTROL DE ATAQUES MEDIANTE URL:** Para evitar ataques por URL, comprobamos en cada redirección y cada carga de nueva página, que las credenciales URL de usuario coincidan con el usuario que ha iniciado la sesión.
* **PERIODO DE INACTIVIDAD:** Función que cierra la sesión automáticamente tras un tiempo de inactividad.
* **COPIAS DE SEGURIDAD:** Para evitar la pérdida de datos hemos implementado un sistema de copias de seguridad, el cual, realiza una copia semanal tanto de la base de datos como de los ficheros y carpetas que constituyen la aplicación.
* **PROTOCOLO HTTPS:** Decidimos establecer la conexión forzada a https.
* **INYECCIÓN SQL:** Para evitar ataques mediante inyección SQL, hemos implementado las siguientes medidas.
  + Usamos $\_POST en lugar de $\_GET en los formularios para validar las peticiones de los usuarios.
  + Para conectar con la BBDD usamos la conexión mysqli en lugar de mysql, la cual proporciona una barrera adicional contra la inyección sql.
  + Creamos limitaciones y reglas en los formularios para limitar las acciones de acuerdo a su uso.
  + En las entradas para introducir contraseñas, en ver de type=”text” usamos: type=”password”
  + En las entradas de texto se puede limitar la cantidad de caracteres usando el atributo “maxlength”.
  + En los formularios usados para subir archivos como imágenes, fotos, etc. puede limitarse el tipo de archivo a subir, basado en su extensión, así como regular su tamaño, por ello limitamos las imágenes a JPG, JPEG, BMP y PNG.
  + Convertir caracteres especiales con htmlentities (Convierte caracteres especiales en entidades HTML).
  + Como extra pedimos a un compañero que realiza su proyecto sobre auditoria de seguridad que intentara hacer una inyección sobre la web a través de SQLMAP.

**TEST REALIZADOS**

Además de las pruebas directas sobre el funcionamiento de la aplicación, hemos realizado test externos para verificar que el programa cumple con los requisitos funcionales y no funcionales decididos.

En este tipo de pruebas se encuentran los errores de funcionalidad, de diseño de la interfaz de usuario, problemas de acceso a la base de datos y errores de rendimiento.

Google PageSpeed Insights – PingDom - Nibbler

**ROLES DE USUARIO**

* **ROL INVITADO**

Los usuarios registrados pueden usar las mismas funciones que el usuario invitado, además de modificar su propio perfil y otras funciones explicadas a continuación.

* **ROL ALUMNO**
* **ROL PROFESOR**
* **ROL ADMINISTRADOR**

**CONCLUSIONES**

* **CONCLUSIONES FINALES**
  + La realización de este proyecto nos ha dotado de buenos hábitos de programación y nos ha enriquecido enormemente.
  + Se ha intentado realizar el proyecto de manera profesional y procurando en todo momento realizar una codificación clara, para facilitar la comprensión del código fuente.
  + Mención especial para lo aprendido en los centros de trabajo, donde hemos cogido grandes ideas y aumentado nuestro conocimiento.
  + Como conclusión final y global, se nos permite decir que la experiencia ha sido muy enriquecedora, aunque ha costado muchas horas de investigación y solución de errores surgidos.
* **PROBLEMAS ENCONTREADOS**
  + Los principales problemas no han sido otros que afrontar los distintos desarrollos que eran desconocidos para nosotros, ocasionando en algunas ocasiones los primeros errores.
  + Por un lado, la creación de la arquitectura HTML, MySQL, PHP, JavaScript, AJAX, etc no ha sido difícil de integrar, aunque también han surgido pequeños errores o inconvenientes.
  + Por ejemplo, integrar el envío automático de emails requirió algo más de trabajo y dedicación, igual ocurrió con el script de las cookies, etc.
* **OBJETIVOS CONSEGUIDOS**
  + Cubrir la necesidad del centro de documentar las actividades realizadas, objetivo que se cumplirá una vez implantado en el centro.
  + Adquisición de conocimientos, hábitos de trabajo, intento de desarrollo eficiente y ordenado, así como fomentar el trabajo en equipo.
* **POSIBLES AMPLIACIONES**
* Se puede implementar el sistema de copias de seguridad, tanto Backup como restauración desde una nueva sección en el panel de control del administrador de la aplicación.
* Incluir más información en forma de estadísticas, actualmente solo contempla los usuarios registrados, se podría ampliar con los proyectos y su número de participantes, número de proyectos que se realizan cada curso académico, etc.
* Añadir nuevos campos a la entidad cursos como fecha de inicio y fin, además de posibles relaciones directas con otras entidades, de ahí que se haya tratado como una entidad.
* Incluso un repaso al código fuente para optimizarlo.
* Posibilidad de incluir más idiomas como el francés.