SISTEMA DE VERIFICACIÓN DE CERTIFICADOS ACADÉMICOS BASADO EN BLOCKCHAIN

VNiVERSiDAD D SALAMANCA

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Septiembre 2022

Autor:

Santiago Vicente Valle

Tutores:

Pablo Chamoso Santos

Diego Valdeolmillos Villaverde

Juan Manuel Corchado Rodríguez

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

DEFENSA DEL TRABAJO FIN DE:
Grado en Ingeniería Informática

Título

Sistema de verificación de títulos académicos basado en blockchain
(Referencia: ARHPY)

Title*

Blockchain-based academic diploma verification system

FACULTAD DE CIENCIAS

Datos del estudiante

Nombre y apellidos: Santiago Vicente Valle
DNI: 70925482Z
Correo electrónico: Santivv@usal.es
Teléfonos: 628486196, 923286767
Dirección: C/Rabindranath Tagore, nº6
37185 Villamayor (Salamanca)

Presenta el trabajo para su defensa en la convocatoria de Septiembre del curso académico 2022/2023

Los tutores consideran que el trabajo es Apto para ser presentado (nombre y firma)

Valdeolmillos Villaverde, Diego	
Corchado Rodríguez, Juan Manuel	
Chamoso Santos, Pablo	

Firma del estudiante

Fecha 06/09/2022

(*) Obligatoriamente se debe cumplimentar el título en inglés para su inclusión en el SET (Suplemento Europeo al Título)

Sr./a. PRESIDENTE/A DE LA COMISIÓN DE TRABAJOS FINALES GRADO EN INGENIERÍA INFORMÁTICA

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

DEFENSA DEL TRABAJO FIN DE:
Grado en Ingeniería Informática

Título

Sistema de verificación de títulos académicos basado en blockchain
(Referencia: ARHPY)

Title*

Blockchain-based academic diploma verification system

FACULTAD DE CIENCIAS

Datos del estudiante

Nombre y apellidos: Santiago Vicente Valle
DNI: 70925482Z
Correo electrónico: Santivv@usal.es
Teléfonos: 628486196, 923286767
Dirección: C/Rabindranath Tagore, nº6
37185 Villamayor (Salamanca)

Presenta el trabajo para su defensa en la convocatoria de Septiembre del curso académico 2022/2023

Los tutores consideran que el trabajo es Apto para ser presentado (nombre y firma)

Valdeolmillos Villaverde, Diego	
Corchado Rodríguez, Juan Manuel	
Chamoso Santos, Pablo	

Firma del estudiante

Fecha 06/09/2022

(*) Obligatoriamente se debe cumplimentar el título en inglés para su inclusión en el SET (Suplemento Europeo al Título)

Sr./a. PRESIDENTE/A DE LA COMISIÓN DE TRABAJOS FINALES GRADO EN INGENIERÍA INFORMÁTICA

RESUMEN

El presente trabajo describe un sistema de verificación diseñado como una plataforma web encargada de almacenar certificados académicos que suban los distintos usuarios que utilicen la plataforma, de forma que se mantengan organizados internamente para que cada poseedor pueda acceder, mantener y compartir estos cuando desee.

La idea surge de mantener una estructura centralizada con todos los títulos académicos expedidos por las diferentes entidades para que en el caso de compartir alguno de ellos se pueda realizar a través de la plataforma de manera sencilla y el receptor sea capaz de visualizar el certificado recibido, y con ellos comprobar los datos que proporciona, además de su veracidad.

Una de las principales consignas del sistema es que aprovechando que trabaja con certificados académicos sea compatible con la principal herramienta online de código abierto para el aprendizaje 'Moodle', en la cual se basan la gran mayoría de plataforma educativas (por ejemplo, Studium), por lo que está diseñada para poder interactuar con alguna de estas plataformas que tenga los servicios web activados y permita la expedición y almacenamiento de certificados. Durante el desarrollo del tema se explicará cómo se realizó y su funcionamiento, también se explicará sus posibles utilidades y ventajas que puede conllevar su uso.

La plataforma web se centra en 3 puntos: primero, introducción de un certificado nuevo en la base de datos, un usuario que tenga un certificado académico puede subirlo a la plataforma y este quedará guardado junto a un conjunto de caracteres o clave asignado para su identificación inequívoca; segundo, comprobación de un certificado mediante su clave previamente creada; y por último, el almacenamiento de los certificados de cada usuario, donde el poseedor de estos puede visualizar la información de todos sus certificados, visualizar individuamente y compartirlos con otros usuarios de la plataforma para que estos también los puedan visualizar.

Como resultado se busca conseguir una plataforma dinámica que pueda usar cualquier usuario en Internet, por lo que no requiere ningún tipo de conocimiento avanzado del tema, y sea una herramienta complementaria a las plataformas de aprendizaje online donde centralizar y compartir los títulos personales de cada usuario.

Palabras clave: certificados, plataforma web, Moodle, blockchain

SUMMARY

This verification system is designed as a web platform responsible for storing academic certificates uploaded by the different users who use the platform, so that they are kept internally organized so that each holder can access, support and share them whenever they want.

The idea arises from maintaining a centralized structure with all the academic titles issued by the different entities so that in the case of sharing any of them, it can be done through the platform in a simple way and the recipient is able to view the certificate received, and with them check the data you provide, in addition to its veracity.

One of the main slogans of the system is that, taking advantage of the fact that it works with academic certificates, it is compatible with the main open source online tool for learning 'Moodle', on which the vast majority of educational platforms are based (for example, Studium), so it is designed to be able to interact with any of these platforms that have web services activated and allow the issuance and storage of certificates. During the development of the subject, it will be explained how it was carried out and its operation, its possible utilities and advantages that its use may entail will also be explained.

The web platform focuses on 3 points: first, introduction of a new certificate in the database, a user who has an academic certificate can upload it to the platform and it will be saved together with a set of characters or password assigned for identification unequivocal; second, verification of a certificate using its previously created key; and finally, the storage of the certificates of each user, where the holder of these can view the information of all their certificates, view them individually and share them with other users of the platform so that they can also view them.

As a result, it seeks to achieve a dynamic platform that any user on the Internet can use, so it does not require any type of advanced knowledge of the subject and is a complementary tool to online learning platforms where to centralize and share the personal titles of each user.

Keywords: certificates, web platform, Moodle, blockchain.

Índice

Ilustraciones

1. INTRODUCCIÓN

Existen muchas formas de obtener un certificado/título, como por ejemplo cursos, grados u otras opciones que, tras finalizar su plan establecido, expiden un título que permite reconocer a la persona del cual es titular que ha conseguido superar con éxito el curso. El ejemplo más cercano y muy claro en este caso es la obtención del título del grado de ingeniería informática por parte de sus alumnos. Estos títulos pueden ser de 2 tipos: los impresos a papel, apreciados por todo usuario ya que es la mejor forma de constatar de forma material su propiedad, pero no es lo más útil; y los certificados digitales.

Todo certificado tiene como objetivo constatar una información o evento sobre una entidad o persona, así como existen certificados de firma digital que permiten reconocer la identidad de una persona física. Este Trabajo de Fin de Grado se centrará en los certificados académicos expedidos de manera digital.

La meta es conseguir una plataforma web que actúe como una base de datos de certificados académicos para cualquier usuario, es decir, cualquier persona que posea algún certificado pueda guardarlo en la plataforma o simplemente quiera comprobar la existencia de alguno de otra persona. Con esto se permite compartir la información de los certificados propios con otros los cuales quiera el propietario del certificado. Más adelante veremos la forma que tienen estos de compartirlo.

Dado que se usarán certificados académicos, la propia plataforma permitirá compatibilizar su uso con certificados que se expidan desde una plataforma Moodle, permitiendo su conexión con estas.

Ilustración 1: Moodle

Moodle es una herramienta de código abierto que permite la creación de plataformas educativas online escrita en PHP. Es la herramienta de referencia para la creación de estas plataformas debido a su capacidad de personalización, junto con sus múltiples plugin que le permiten añadir nuevas funcionalidades. En la actualidad la gran mayoría de plataformas educativas que existen están creadas con esta herramienta lo que proporcionará a la plataforma una mayor versatilidad.

1.1. Estructura de la memoria

La memoria de este trabajo se dividirá en diferentes secciones en las que se explicarán los aspectos más relevantes del TFG.

Cada uno de los apartados resumirá una parte de la realización del proyecto y con el fin de que quede claro el método de creación y, por supuesto, el resultado final

- Conceptos teóricos
- Objetivos del sistema
    - Objetivos de software
    - Objetivos personales
- Técnicas y herramientas utilizadas
- Aspectos relevantes y desarrollo del trabajo
- Trabajos relacionados
- Líneas de trabajo futuras
- Conclusiones
- Referencias

A esta memoria principal se le adhieren 3 cuatro anexos donde se describirán con más detalle los resultados y el proceso:

- Anexo 1 - Planificación del proyecto: explicación teórica del proceso de creación del proyecto, planificación de las tareas, estimación de costes…
- Anexo 2 - Requisitos del proyecto: descripción de los requisitos necesarios del sistema.
- Anexo 3 - Manual de usuario: descripción del manejo de la plataforma web desde la perspectiva de un usuario cualquiera.

2. CONCEPTOS TEÓRICOS

2.1. Web services

2.1.1. *Web services* de Moodle

Funciones que proporciona Moodle para acceder a este, de manera que permite realizar diversas operaciones como recoger información de la plataforma, todo ello de forma externa. En otras palabras, es una API REST que permite la conexión a Moodle.

2.1.2. API (*Application Programming Interface*)

Interfaz de programación de aplicaciones que permite la comunicación entre dos componentes a través de unas definiciones y protocolos establecidos. La arquitectura de las API está basada en un cliente-servidor, donde gracias a las funciones el cliente pregunta por información al servidor y este le responde.

2.1.3. API REST

Las API REST es un tipo de API que tiene como características que no almacenan el estado del cliente en el servidor y únicamente devuelven datos planos. Algunas de sus funciones más comunes son GET, PUT, DELETE, POST, etc.

2.1.4. Token

Código generado por una plataforma Moodle para el acceso de un usuario registrado de esa plataforma a los servicios web que esta pretende dar. Cada *token* en único para el usuario y plataforma y permite el acceso a las funciones definidas durante su generación.

2.2. PHP

2.2.1. Variables superglobales

Variables de PHP con un alcance global entre scripts, es decir no es necesario declararlas siempre que se vayan a usar. Un ejemplo de estas son las variables de sesión que se describirán a continuación.

2.2.2. Variables de sesión

Conjunto de variables que se utilizan en PHP con información sobre el script actual. Este tipo de variables se consideran superglobales, ya que sin necesidad de declararlas estarán disponibles en cualquier parte del programa. Para acceder a estas variables es necesario comenzar la sesión en el script con *session_start()*, y se declaran de la siguiente forma: $_SESSION[‘variable’] = valor.

En este trabajo se usarán para asegurar la sesión de usuario logueado, guardar su nombre de usuario e id asociado a su cuenta en la base de datos y, en caso de necesidad, un código de error.

Ilustración 2: Variables de sesión

Métodos GET y POST

Métodos usados en PHP para el paso de datos de un script a otro. También son funciones de las APIs que sirven con el mismo propósito, pasar información entre el cliente y el servidor que se están comunicando.

2.2.3. Blob type (binary large object)

Tipo de dato utilizado en SQL que es capaz de almacenar cadenas binarias de hasta 2gb. En el proyecto se utilizarán para guardar archivos binarios generados a partir de los certificados que los usuarios suban a la plataforma.

2.3. CSV

Abreviatura de código seguro de verificación. Código utilizado para la identificación inequívoca de un certificado académico. Este código es asignado una vez que un archivo es subido a la plataforma y está compuesto por 23 caracteres, los 3 primeros fijos y el resto generados de manera aleatoria.

```php
function generate_csv($db){

    $uniquecodefound = false;
    $seed = str_split('abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789');
    $code = 'SVC';

    shuffle($seed);
    foreach (array_rand($seed, 29) as $k) $code .= $seed[$k];

    while (!$uniquecodefound) {
        if ($result = $db->query("SELECT csv FROM certificate_files")){
            while ($fila = $result->fetch_row()) {
                if ($fila[0] == $code){
                    $uniquecodefound = false;
                    break;
                } else {
                    $uniquecodefound = true;
                }
            }
        }

        if ($uniquecodefound) {
            $code = 'SVC';
            shuffle($seed);
            foreach (array_rand($seed, 29) as $k) $code .= $seed[$k];
        }
    }

    return $code;
}
```

Ilustración 3: Generación CSV

2.4. SCRUM

Scrum es una metodología ágil que se basa en la organización en equipos autoorganizados, desarrollo del trabajo en iteraciones incrementales en las que al terminarlas se replanteará que objetivos se han cumplido, cuales no y se continuará con la siguiente iteración hasta el fin del proyecto. Esto a su vez permite que se pueda ir añadiendo funcionalidades extra al trabajo que no estuvieran contempladas al principio.

2.5. Blockchain

Blockchain es una estructura de datos casi imposible de falsificar usada en redes de nodos descentralizadas. Blockchain surgió con Bitcoin como método de asegurar la fiabilidad de las transacciones de dicha criptomoneda.

2.5.1. Smart contract

Script que define unas reglas para las transacciones y que está diseñado para ejecutarse automáticamente una vez esas reglas se hagan ciertas.

3. OBJETIVOS DEL PROYECTO

3.1. Objetivos de software

El objetivo principal es realizar de una plataforma web que permita la organización en conjunto de todos los certificados académicos independientemente de la entidad certificadora que haya expedido el título, para ello se persigue la capacidad de aceptar la subida de archivos y su posterior almacenamiento junto con información relevante sobre el mismo.

Para ello, el sistema tiene como uno de sus objetivos mantener una base de datos interna en la que se almacenen los certificados, información relevante sobre ellos, datos de los usuarios registrados, así como de sus cuentas en la plataforma; o también información sobre las distintas instituciones certificadoras de las cuales se han recibido certificados.

También se busca que la plataforma sea capaz de distinguir la persona que ha subido el título y por tanto le asocie su posesión. Para conseguirlo, se establecerá una gestión de usuarios efectiva que permita la creación de una cuenta mediante un registro y el sistema se encargue almacenar la información, asegure la privacidad de los datos de cada usuario, sobre todo la contraseña, la cual se almacenará encriptada y nunca se mostrará como es realmente.

Establecimiento de conexión con una plataforma de aprendizaje online basada en Moodle mediante los servicios web que está posee para la gestión externa de los certificados que pueda expedir la institución que administre la plataforma en cuestión.

3.2. Objetivos personales

Mis principales objetivos personales con este TFG es recoger experiencia profesional para mi futuro ya que es el primer proyecto a gran escala que realizo. Soy consciente de que en la vida laboral de un informático es muy importante saber gestionar proyectos de un tamaño considerable y conseguir dividir el trabajo en partes más pequeñas que se puedan resolver con más simpleza, haciendo más sencillo llegar al resultado final.

Conseguir realizar este proyecto poco a poco y reuniendo todas esas pequeñas tareas para conseguir al final la plataforma funcional es mi meta.

4. TÉCNICAS Y HERRAMIENTAS UTILIZADAS

4.1. Moodle

Herramienta para la creación de plataformas de aprendizaje online de código abierto, cuyo enlace para la descarga se encontrará en el apartado de referencias al final del documento. En el proyecto, se creará una plataforma Moodle con el módulo de actividad custom certificate instalado para la administración de certificados y la conexión con los *web services*.

Moodle es de las herramientas académicas online más usadas en Internet y por ello, la gran mayoría de webs dedicadas a la enseñanza se basan en él, ya que proporciona una gran cantidad de recursos diferentes para los profesores y los alumnos para facilitar la enseñanza.

4.2. XAMPP

Aplicación de software libre que está basada en la simulación de un servidor Apache de manera local, junto con una base de datos asociada MySQL e intérpretes para los lenguajes PHP y Perl.

Herramienta muy útil para la creación de un servidor web ya que permite su testeo en local sin necesidad de desplegarlo ni de asignarle un dominio.

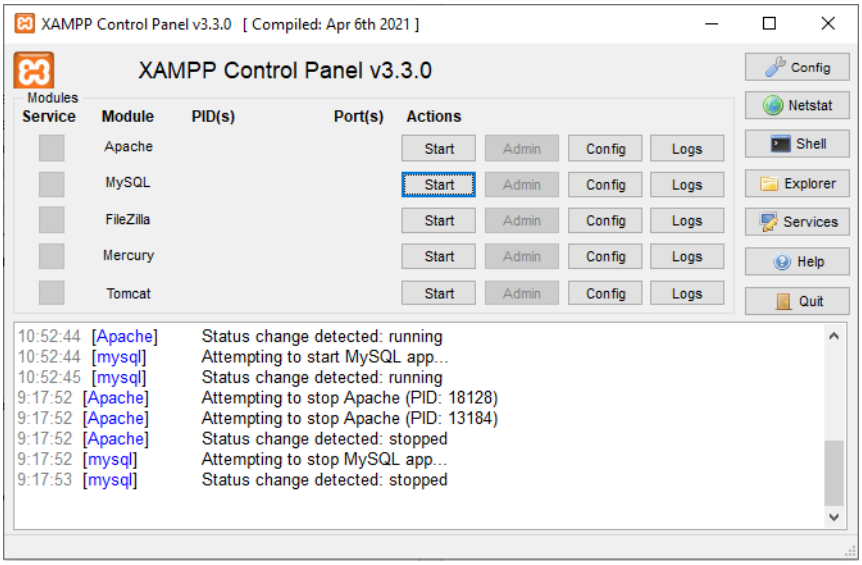


Ilustración 4: XAMPP

4.3. VS Code

Entorno de desarrollo para la edición de código desarrollado por Microsoft. Permite una gran cantidad de opciones y programar en casi cualquier lenguaje, todo ello gracias al sistema de extensiones que le permiten añadir una mayor funcionalidad al programa.

4.4. BD de phpMyAdmin

El sistema de gestión de la base de datos que incluye XAMPP como se ha mencionado antes está basado en bases de datos MySQL relacionales que se administran desde el portal phpMyAdmin. Allí tienes la opción de ver todas las bases de datos creadas, gestionarlas o crear alguna nueva.

A esta herramienta se accede directamente desde XAMPP, primero activando el servicio de MySQL y luego pulsando el botón 'admin'.

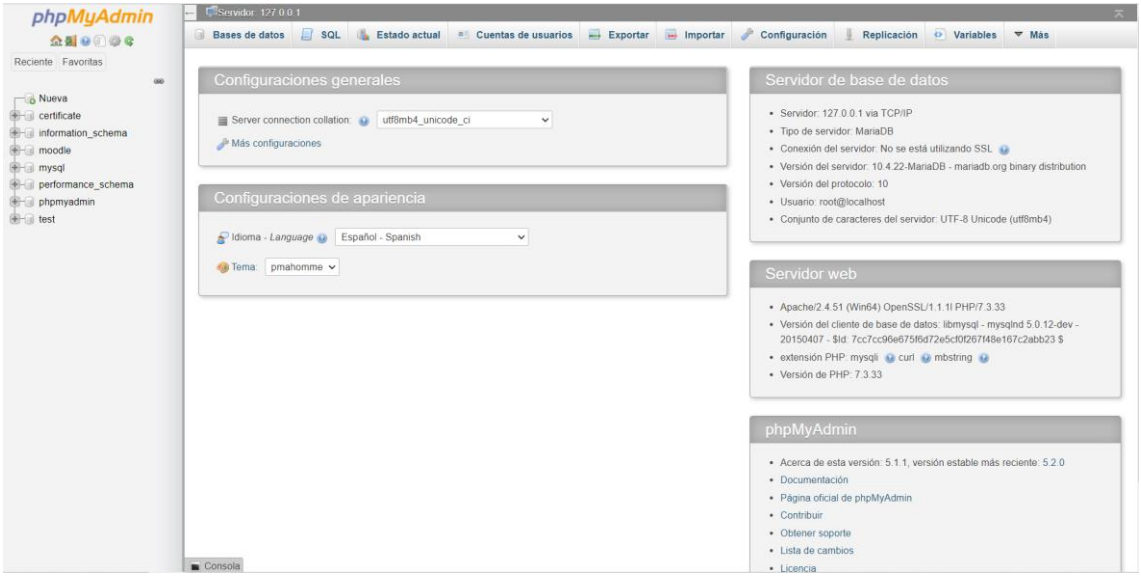


Ilustración 5: phpMyAdmin

4.5. Bootstrap

Es una biblioteca para HTML, CSS y JavaScript que proporciona al programador una gran cantidad de opciones para el diseño de una página web de forma responsiva. Desde su página web se pueden comprobar que tiene una gran cantidad de ejemplos para diferentes tipos de páginas web, además de también muchas plantillas para la creación de estas.

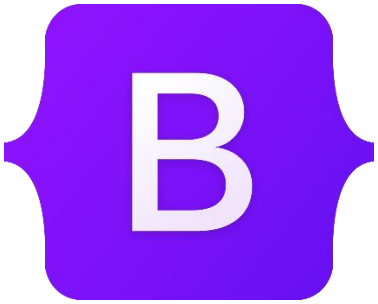


Ilustración 6: Bootstrap

4.6. Remix IDE

Es un entorno de desarrollo para la edición de código Solidity, el cual es utilizado para la creación de smart contracts, y que permite compilar y depurar este tipo de contratos.

4.7. Lenguajes

4.7.1. PHP

Lenguaje de programación para el desarrollo web que fue creado en 1995 y se estableció muy rápido como uno de los principales lenguajes para la creación de páginas web. Durante los últimos años ha ido caído en decadencia con la aparición de otras herramientas mejores con el mismo fin (JavaScript junto con algún *framework* como Vue.js o React.js, TypeScript).

4.7.2. JavaScript

Lenguaje de programación orientado a objetos creado en 1995 y usado para el desarrollo de funciones interactivas de las páginas web.

4.7.3. HTML

Lenguaje de etiquetas de hipertexto que permite construir la parte visual de una página web sin estilo, como, por ejemplo, títulos, textos, tablas, imágenes…

4.7.4. CSS

También llamado Hojas de estilo en cascada debido a su traducción desde el inglés, permite añadir el estilo al código HTML, es decir, ya que HTML maneja lo que se ve en la página web, CSS se encarga del cómo se ve y con ello, proporcionar una estética mejor a la web.

4.7.5. Solidity

Lenguaje de programación orientado a objetos diseñado para la creación de contratos inteligentes que puedan ser implementados en la *blockchain*. Cabe destacar que, en sus inicios, Solidity se creó para la red Ethereum, pero ahora se puede utilizar en diferentes redes *blockchain*.

5. ASPECTOS RELEVANTES Y DESARROLLO DEL PROYECTO

5.1. Planificación temporal

La planificación del proyecto se dividirá en *sprints*, tal y como indica la metodología Scrum, donde cada sprint representará un paquete importante del proyecto. En total habrá 6 *sprints*:

- *Sprint* 1: Inicio del proyecto y Moodle
    - Requisitos del sistema
    - Instalación
    - Modificación de CSV y base de datos
    - Modificación funciones Web Services
- *Sprint* 2: Conexión a Moodle mediante Web Services
    - Obtención del token
    - Conexión al Moodle
- *Sprint* 3: Gestión del almacenamiento de los certificados
    - Almacenamiento de la información de un certificado
    - Introducción del certificado
    - Gestión interna de los certificados y frontend
    - Funciones extra
- *Sprint* 4: Gestión de los usuarios
    - Almacenamiento de la información de los usuarios
    - Gestión de las sesiones
    - Establecimiento de los roles
- *Sprint* 5: Administradores y certificados
    - Funciones de administrador: usuarios
    - Funciones de administrador: certificados
    - Funciones de administrador: entidades
    - Creación de certificados propios
- *Sprint* 6: Blockchain
    - Creación del smart contract
    - Implementación en la plataforma

A esto habría que añadirle un *Sprint* 0 que se correspondería con conseguir los conocimientos necesarios para realizar el proyecto y familiarizarse con las herramientas y tecnologías necesarias del trabajo.

5.2. Diseño de la base de datos

Para el almacenamiento de los datos del sistema se usa una base de datos relacional MySQL, la cual viene integrada en la propia herramienta de XAMPP. La información guardada será en su mayoría accesible desde la plataforma para su comprobación y modificación por los usuarios.

Existen 4 tablas:

- ***Certificates_files***: se almacena la información de los certificados gestionados desde la plataforma, consta de 9 columnas
- ***Cert_entities***: esta tabla contiene la información de las entidades certificadoras que han aparecido en la plataforma, guardando su información para evitar que todos los certificados subidos de la misma entidad sean considerados de distintas por diferencias léxicas.
- ***Data***: contiene únicamente tres filas con información correspondiente a las conexiones: el nombre de dominio de la plataforma, el nombre de dominio del Moodle al que se conectará y el token que necesita para conectarse.
- ***Users***: la table '*users*' contiene a todos los usuarios registrados de la plataforma junto con toda la información precisada por estos.

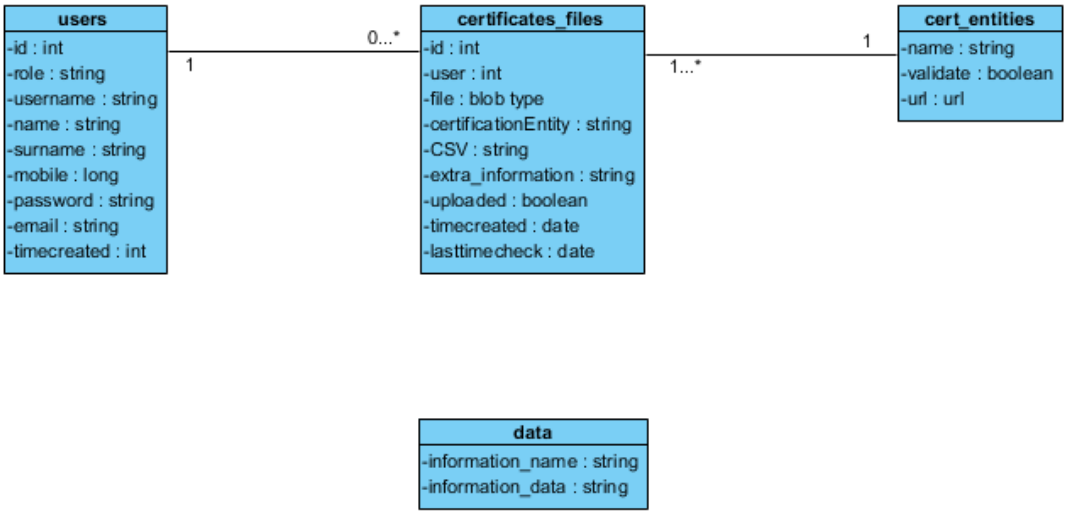


Ilustración 7: Base de datos

5.2.1. *Certificates_files*

- *Id*: número entero identificativo de cada certificado en la base de datos
- *User*: número entero identificativo del usuario al que pertenece el certificado
- *File*: variable que guarda el archivo binario del certificado subido/generado
- *certificationEntity*: entidad certificadora que el usuario ha indicado que ha expedido el certificado. En caso de ser generado por la propia plataforma será "Sistema verificador".
- CSV: código que identificara inequívocamente al certificado en la plataforma.
- *Extra_information*: demás información no catalogada que el usuario propietario del certificado quiera mostrar.
- *Uploaded*: variable booleana que indica si la información del certificado está compartida o no en la *blockchain*.
- *Timecreated*: número entero que indica la fecha de subida/creación del certificado en cuestión.
- *Lasttimecheck*: número entero que indica la fecha de última visualización del certificado en cuestión.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
1	**id** 🔑	bigint(20)			No	*Ninguna*		AUTO_INCREMENT
2	**user**	int(11)			No	*Ninguna*		
3	**file**	longblob			No	*Ninguna*		
4	**certificationEntity**	varchar(100)	utf8mb4_general_ci		No	*Ninguna*		
5	**csv**	varchar(50)	utf8mb4_general_ci		No	*Ninguna*		
6	**extra_information**	varchar(255)	utf8mb4_general_ci		Sí	*NULL*		
7	**uploaded**	tinyint(1)			No	0		
8	**timecreated**	bigint(20)			No	current_timestamp()		
9	**lasttimecheck**	bigint(20)			Sí	current_timestamp()		

Ilustración 8: Certificates_files

5.2.2. *Cert_entities*

- *Name*: nombre de la entidad certificadora.
- *Validate*: variable booleana que indica si un administrador ha validado la existencia de la entidad certificadora.
- *Url*: dirección web de la entidad certificadora, solo si se precisa.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
1	**name**	varchar(255)	utf8mb4_general_ci		No	*Ninguna*		
2	**validate**	tinyint(1)			No	0		
3	**url**	varchar(255)	utf8mb4_general_ci		Sí	*NULL*		

Ilustración 9: Cert_entities

5.2.3. *Data*

- *Information_name*: nombre del dato que se desea guardar.
- *Information_data*: dato que se desea guardar.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
1	**information_name**	varchar(255)	utf8mb4_general_ci		No	*Ninguna*		
2	**information_data**	varchar(255)	utf8mb4_general_ci		Sí	*NULL*		

Ilustración 10: Data

5.2.4. Users

- *Id*: número entero identificativo de cada usuario en la base de datos
- *Role*: rol obtenido por el usuario dentro de la plataforma, puede ser 'user' o 'admin', mediante el cual se determinará los privilegios de cada usuario.
- *Username*: nombre de usuario elegido por el mismo
- *Name*: nombre verdadero del usuario
- *Surname*: apellidos del usuario
- *Mobile*: número de teléfono del usuario
- *Password*: contraseña para la cuenta del usuario, esta se guarda hasheada para mayor privacidad.
- *Email*: cuenta de correo del usuario
- *Timecreated*: número entero que indica la fecha del registro del usuario en la plataforma.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
1	**id** 🔑	bigint(20)			No	*Ninguna*		AUTO_INCREMENT
2	**role**	varchar(30)	utf8mb4_general_ci		No	user		
3	**username**	varchar(20)	utf8mb4_general_ci		No	*Ninguna*		
4	**name**	varchar(30)	utf8mb4_general_ci		No	*Ninguna*		
5	**surname**	varchar(70)	utf8mb4_general_ci		No	*Ninguna*		
6	**mobile**	int(20)			Sí	*NULL*		
7	**password**	varchar(255)	utf8mb4_general_ci		No	*Ninguna*		
8	**email**	varchar(60)	utf8mb4_general_ci		No	*Ninguna*		
9	**timecreated**	bigint(20)			No	current_timestamp()		

Ilustración 11: Users

5.3. Actores

En la plataforma se distinguen tres tipos de actores: usuario no logueado, definido con el propio nombre y usuario logueado, los cuales se dividirán en usuarios corrientes y administradores que se diferenciarán según el valor que tenga en la columna 'role' en la tabla de 'users' de la base de datos.

- Usuario no logueado: usuario sin iniciar sesión en la plataforma.
- Usuario logueado: usuario que ha iniciado sesión en la plataforma.
    - Usuario corriente: usuario logueado sin permisos adicionales.
    - Administrador: usuario logueado con permisos adicionales (acceso a todos los certificados existentes en la plataforma, permisos para la gestión de todos los usuarios de la plataforma y para la modificación de las entidades de certificación).

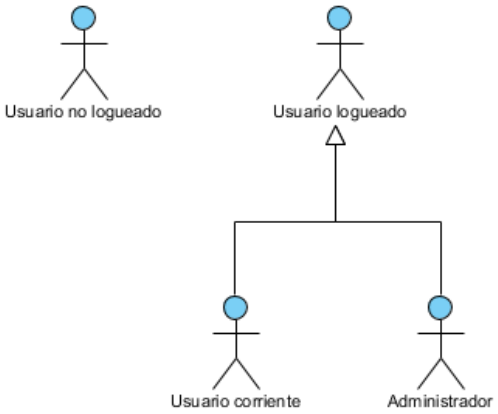


Ilustración 12: Actores

5.4. Moodle

5.4.1. Módulos de actividad de Moodle

El acrónimo Moodle proviene de '*Modular Object-Oriented Dynamic Learning Environment*' que traducido al español significa 'Ambiente de Aprendizaje Modular Dinámico Orientado a Objetos', lo que nos indica que tiene una estructura modular que puede ir aumentando según se vaya desarrollando.

Los módulos de actividad son parte de esa estructura modular que permiten mayor variedad de funcionalidades en los cursos, por ejemplo: introducción de archivos, chats, URLs, etc. El paquete de Moodle que se descarga de la página oficial contiene varios módulos de actividad preinstalados con gran parte de la funcionalidad básica que se necesita en los cursos online.

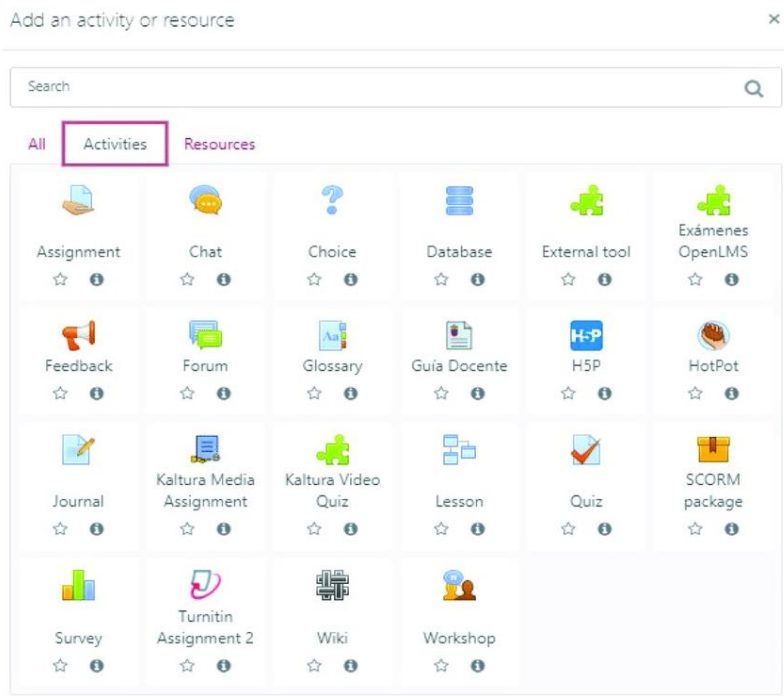


Ilustración 13: Módulos de actividad

Estos se almacenan en la carpeta /mod del Moodle en cuestión y guardan una estructura de ficheros similar:

- Carpeta /*backup*: en esta carpeta se encuentra archivos que indicaran la manera de que se ejecutará la recuperación de una copia de seguridad
- Carpeta /*db*: se encontrarán diversos archivos de configuración donde se declaran, entre otras cosas, las capacidades que tiene el módulo de actividad. Entre los archivos más importantes destacaré:
    - install.xml: se declara la/s tabla/s que usará el módulo de actividad. Este archivo se ejecutará una vez al instalarse el propio módulo.
    - *services.php*: contiene un array con cada una de las definiciones de las funciones que proporciona el módulo a los *web services* de Moodle (nombre de la clase, nombre del método, capacidades que necesita usar el servicio, etc.)
- Carpeta /*lang*: esta carpeta a su vez contendrá tantas carpetas como idiomas quiera incluir el administrador del módulo nombradas con la abreviatura del idioma (por ejemplo, inglés es 'en' o español 'es') y cada carpeta contendrá un fichero llamado como el nombre del módulo con extensión .php donde se indicarán todas las cadenas de texto que se utilizarán en el módulo.

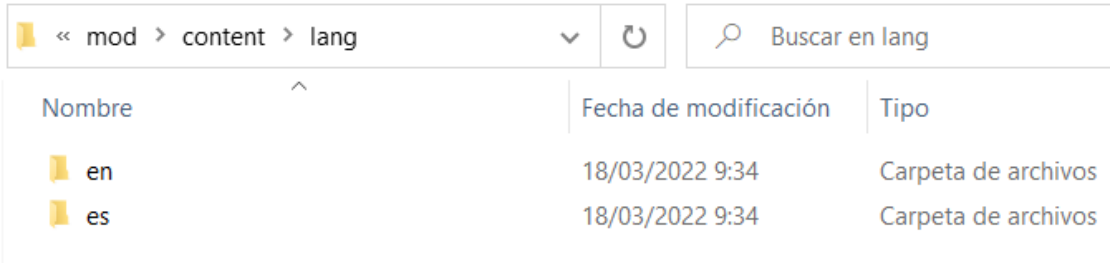


Ilustración 14: Carpeta /lang

- Carpeta /*pix*: se almacena el icono que se visualizará al lado del módulo de actividad en el Moodle.

Además de ficheros de configuración que no se contienen en ninguna carpeta dentro del módulo:

- lib.php: declaración de las funciones principales del módulo, sobre todas las opciones que este proporciona. Las más importantes son:
    - *add_instance*: creación de una nueva instancia del módulo.
    - *update_instance*: modificación de una instancia ya existente del módulo.
    - *delete_instance*: eliminación de una instancia del módulo.
- *mod_form.php*: se estable el formulario donde se indicarán los datos necesarios para crear una nueva instancia.
- *index.php*: fichero encargado de recoger la información de la base de datos de la instancia a la que se desea acceder. Todo fichero *index.php* debe contener al principio lo siguiente:

Ilustración 15: Principio de index.php

Inicia el fichero *config.php* de Moodle donde se guarda toda la información para la conexión a la base de datos, recoge el parámetro 'id' desde la URL y busca en la base de datos una instancia con ese 'id'.

- *view.php*: fichero encargado de mostrar el contenido de la instancia a la que se accede.
- *version.php*: indica información básica del módulo como la versión mínima requerida de Moodle para su correcto funcionamiento, la versión para la que se ha creado, el nombre del propio módulo.

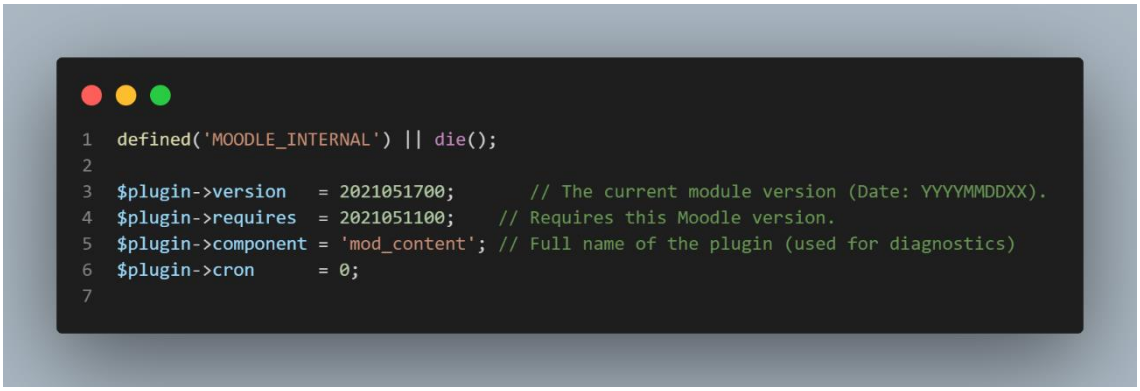


Ilustración 16: Ejemplo de version.php

5.4.2. Módulo custom certificate

Este módulo en concreto permite a un administrador de un curso crear plantillas de certificados académicos que posteriormente un alumno con acceso a esa actividad podrá usar para descargarse el propio certificado con sus datos concretos, por lo que permite a un curso online distribuir certificaciones de participación o de haberlo superado.

El enlace para la descarga del módulo custom certificate se encontrará en el apartado de referencias.

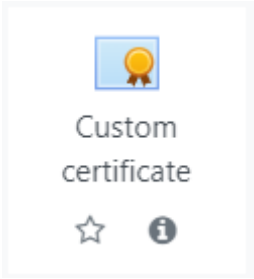


Ilustración 17: Módulo custom certificates

5.4.3. Modificaciones de módulo custom certificate

Para adecuar el módulo a las necesidades del proyecto, se han realizado varias modificaciones en el código fuente del módulo de actividad. Se seguirá la estructura básica de los módulos de actividad mencionada con anterioridad. Las modificaciones realizadas son:

- Actualización de la base de datos del módulo para que en la tabla de los certificados se incluya una columna para el código CSV que se asignará a cada certificado.
    - Dentro de la carpeta de módulo de actividad, en */db/install.xml*. En este archivo se modifica la tabla *customcert_issues* para la inclusión de la columna *code* para el CSV. Después de esta modificación para que se represente en la base de datos hay que desinstalar y volver a instalar el módulo de actividad entero.

```
<TABLE NAME="customcert_issues" COMMENT="Stores each issue of a customcert">
      <FIELDS>
        <FIELD NAME="id" TYPE="int" LENGTH="10" NOTNULL="true" SEQUENCE="true"/>
        <FIELD NAME="userid" TYPE="int" LENGTH="10" NOTNULL="true" DEFAULT="0" SEQUENCE="false"/>
        <FIELD NAME="customcertid" TYPE="int" LENGTH="10" NOTNULL="true" DEFAULT="0" SEQUENCE="false"/>
        <FIELD NAME="code" TYPE="char" LENGTH="40" NOTNULL="false" SEQUENCE="false"/>
        <FIELD NAME="hash" TYPE="text" LENGTH="big" NOTNULL="false" SEQUENCE="false"/>
        <FIELD NAME="emailed" TYPE="int" LENGTH="1" NOTNULL="true" DEFAULT="0" SEQUENCE="false"/>
        <FIELD NAME="timecreated" TYPE="int" LENGTH="10" NOTNULL="true" DEFAULT="0" SEQUENCE="false"/>
      </FIELDS>
      <KEYS>
        <KEY NAME="primary" TYPE="primary" FIELDS="id" COMMENT="Primary key for customcert_issues"/>
        <KEY NAME="customcert" TYPE="foreign" FIELDS="customcertid" REFTABLE="customcert" REFFIELDS="id"/>
      </KEYS>
    </TABLE>
```

Ilustración 18: install.xml

    - En el fichero */classes/certificate.php*, añadir a la función *issue_certificate()* una variable con el nuevo código CSV del certificado antes de insertarlo en la base de datos.

- Creación de una función para la generación del código CSV y posterior asignación a la hora de guardar el certificado en la base de datos.
    - En el mismo fichero */classes/certificate.php*, se crea la función *generate_csv()* para la creación del código CSV. Esta función ya está mostrada con anterioridad en la definición del CSV. La función genera un código de 32 dígitos, de los que los tres primeros serán fijos y los 29 restantes son caracteres aleatorios para asegurar que este sea único.

- Creación de las funciones para comprobar el código CSV y extracción del certificado de la base de datos de Moodle para ser utilizadas a través de los *web services* y poder conectarse desde la plataforma externa a Moodle.
    - Funciones creadas en */classes/external.php* que se describirán en el apartado siguiente.
    - A cada función de estas hay que añadirle 2 más correspondientes a los parámetros recibidos (<nombre de la función>*_parameters*) y los devueltos (<nombre de la función>*_returns*).

- Modificación estética de la plantilla en la generación de los certificados.
    - En la función *customcert_add_instance()* del fichero lib.php del módulo de actividad, se añaden los nombres del curso, del alumno que ha recibido el certificado, las fechas de comienzo y fin del curso, las diferentes secciones del curso y la fecha de expedición del certificado.

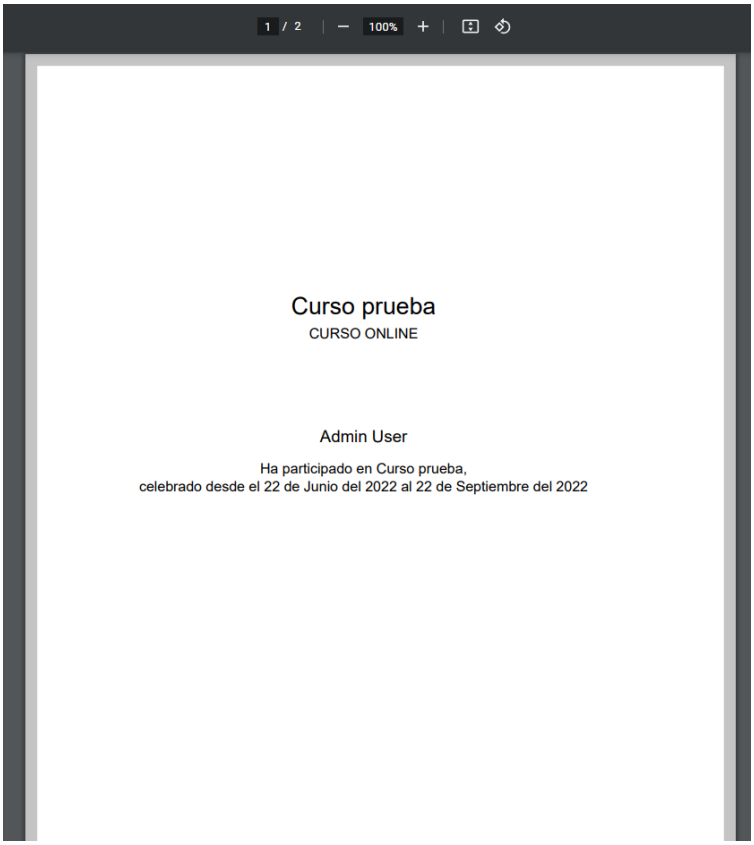


Ilustración 19: Ejemplo de certificado de Moodle

5.4.4. *Web services*

Los *web services* de Moodle proporcionan una conexión API desde cualquier plataforma externa a las funciones que tenga declarada Moodle para su uso a través de estos. En el caso de este trabajo se utilizarán algunas de las funciones del módulo de actividad *custom certificates*, las cuales han sido creadas con el propósito que se necesita para el funcionamiento de la plataforma.

Las funciones creadas son:

- *mod_customcert_check_csv*: comprueba si el CSV introducido se encuentra asignado en la base de datos del módulo custom certificate a algún certificado.

```php
public static function check_csv($csv) {
        global $DB;

        $params = [
            'csv' => $csv
        ];
        self::validate_parameters(self::check_csv_parameters(), $params);

        $params_sql = array('csv' => $csv);
        //Obtención de los datos de la tabla 'customcert_issues'
        $sql = $DB->get_record_sql("SELECT ci.id AS id, ci.userid AS userid, ci.customcertid AS customcertid, ci.code AS code, ci.timecreated AS
        timecreated FROM mdl_customcert_issues ci WHERE ci.code = :csv", $params_sql);
        $name = $DB->get_record_sql("SELECT u.username AS username FROM mdl_user u WHERE u.id = :userid", array('userid' => $sql->userid));

        $array['id'] = $sql->id;
        $array['userid'] = $sql->userid;
        $array['customcertid'] = $sql->customcertid;
        $array['code'] = $sql->code;
        $array['timecreated'] = $sql->timecreated;
        $array['user'] = $name->username;

        return $array;
    }
```

Ilustración 20: mod_customcert_check_csv

- *mod_customcert_certificate*: devuelve los datos del certificado comprobado anteriormente con la función *mod_customcert_check_csv*.

```php
public static function certificate($csv) {
        global $DB;

        $params = [
            'csv' => $csv
        ];
        self::validate_parameters(self::certificate_parameters(), $params);

        $params_sql = array('csv' => $csv);
        //Obtención de los datos que posteriormente se le pasaran a la función generadora del duplicado del certificado
        $sql = $DB->get_record_sql("SELECT ct.id, ct.name, ct.contextid, ci.userid FROM mdl_customcert_issues ci JOIN
        mdl_customcert c ON ci.customcertid=c.id JOIN mdl_customcert_templates ct ON c.templateid=ct.id WHERE ci.code = :csv", $params_sql);

        return $sql;
    }
```

Ilustración 21: mod_customcert_certificate

5.5. Plataforma web

5.5.1. Frontend – Backend

Para la creación de la plataforma web se ha usado una combinación de 4 lenguajes de programación, los cuales ya han sido definidos anteriormente, que permiten realizar diferentes funciones y gracias a los puntos fuertes de cada uno de ellos crear la plataforma web con la mejor funcionalidad posible.

- **HTML y CSS**: dedicados al *frontend* de la plataforma, junto con Bootstrap son las herramientas utilizadas para toda la parte visual de la plataforma. Como partes a destacar:
    - *navbar.php* y *footer.html*: son 2 archivos únicamente basados en el frontend, en los cuales se crea los *navbar* y *footer* de la plataforma que aparecerán en casi todas las páginas.

Ilustración 22: Navbar

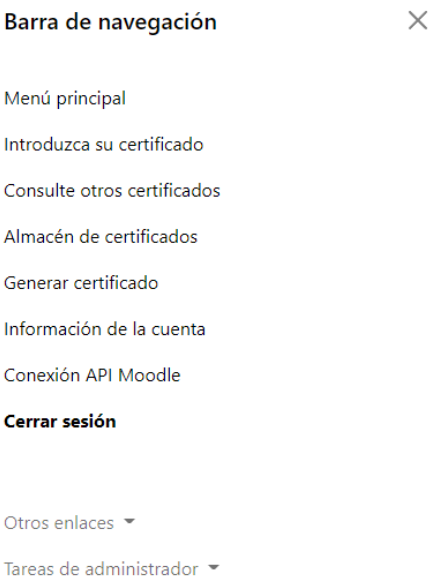


Ilustración 23: Barra lateral

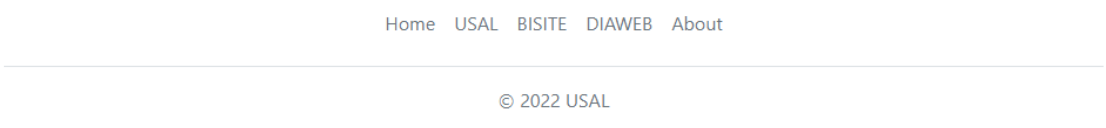


Ilustración 24: Footer

- *style.css*: script de estilo donde se declara, entre otras cosas, el color de los diferentes objetos de la plataforma, su localización dentro de la página…

- **PHP**: utilizado para el *backend*. Una de sus grandes fortalezas es el paso de datos entre los diferentes ficheros, mediante métodos GET, POST o mediante las variables de sesión. También permite de forma bastante sencilla con las funciones *query*.
    - *params.php* y *database.php*: recoge los datos necesarios para poder conectarse a la base de datos y al Moodle que se le indique.

- **JavaScript**: utilizado para el *backend* al igual que PHP, pero utilizado para funciones más sencillas de programar respecto de PHP, por ejemplo, la visualización de certificados
    - *connect.php*: visualización del PDF del certificado, funciones extra para la gestión de las páginas del PDF o el zoom sobre ellas y a la hora de generar un código QR.

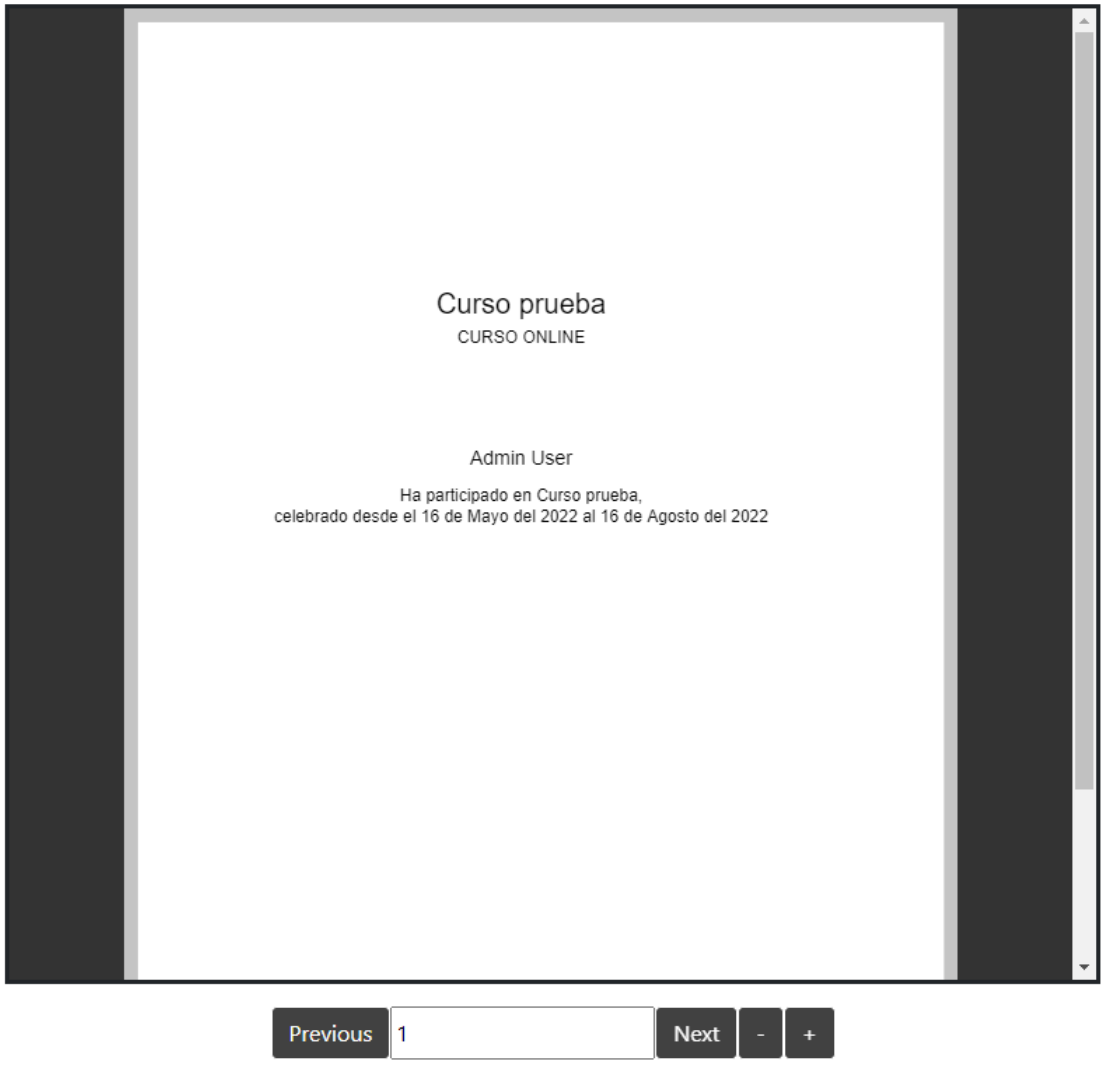


Ilustración 25: Visualización con JavaScript

5.5.2. Bootstrap y estilo

Bootstrap es una biblioteca multiplataforma de código abierto que contiene una gran variedad de plantillas de objetos para diseño web que se utiliza sobre HTML y CSS. Permitirá al proyecto conservar una estética responsiva y adaptarse a los diferentes dispositivos desde los que se pueda acceder sin que aparezcan disposiciones extrañas de los objetos de la página web.

Para incluir la librería en el proyecto se importará está directamente desde un sitio web de la siguiente manera:

```
<!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">
```

Ilustración 26: Adicción de Bootstrap

A su vez para ofrecer una mejor estética a las páginas web se crea una hoja de estilo en cascada (CSS) común para todas ellas, en donde indicando según la propiedad *class* o *id* de las etiquetas HTML se las definirá entre otras cosas el color que utilizarán y la disposición o anclaje dentro de la web.

Colores utilizados:

Ilustración 27: Paleta de colores

5.5.3. Conexión a Moodle

Para la conexión a Moodle des de la plataforma web se usa el archivo *curl.php* que permitirá hacer una petición al servidor de Moodle. Mediante la función post de *curl.php* se podrá acceder a la función de los *web services* que se especifique. En las siguientes imágenes se muestra un ejemplo:

Ilustración 28: Inicio de curl

Ilustración 29: Conexión a Moodle

5.5.4. Html2pdf

Para la generación de PDF desde 0 en la plataforma se usa la librería de PHP llamada HTML2PDF que funciona como un conversor de HTML a PDF, permitiendo así escribir dentro de un PDF vacío y luego generarlo como se desee.

```php
$content = ob_get_clean();
    require_once(dirname(__FILE__).'\vendor\autoload.php');
    use Spipu\Html2Pdf\Html2Pdf;
    use Spipu\Html2Pdf\Exception\Html2PdfException;
    use Spipu\Html2Pdf\Exception\ExceptionFormatter;

    try
    {
        $html2pdf = new Html2Pdf('L', 'A4', 'es', true, 'UTF-8', 3);
        $html2pdf->pdf->SetDisplayMode('fullpage');
        $html2pdf->writeHTML($content, isset($_GET['vuehtml']));
        $pdfContent = $html2pdf->Output('PDF-CF.pdf', 'S');



        $html2pdf->Output('PDF-CF.pdf');
    }
    catch(HTML2PDF_exception $e) {
        echo $e;
        exit;
    }
```

Ilustración 30: Html2pdf

El código HTML que se convertirá a PDF se escribirá dentro de una etiqueta <page>.

Para instalar esta librería se utilizará el sistema de gestión de paquetes de PHP, *Composer*, y junto a ello, en una terminal en la carpeta del proyecto se escribirá el comando '*composer require spipu/html2pdf*', entonces se incluirá una línea en el fichero *composer.json* con la librería y se creará una carpeta llamada *vendor* con los archivos necesarios para el funcionamiento de esta.

5.5.5. QRious

QRious es una librería JavaScript para la generación de códigos QR para formato web, permite mediante la clase QRious incluir en la página web un código QR con los datos que le desees incluir.

```
new QRious({
                element: document.querySelector("#codigo"),
                value: "https://localhost/show_pdf.php?csv="+"<?php echo $resp->code ?>", // La URL o el texto
                size: 200,
                backgroundAlpha: 0, // 0 para fondo transparente
                foreground: "#000", // Color del QR
                level: "H", // Puede ser L,M,Q y H (L es el de menor nivel, H el mayor)
            });
```

Ilustración 31: QRious

5.5.6. Ficheros con parámetros de configuración

Conjunto de ficheros que proporcionan cabeceras al resto de archivos. Son necesarios en la gran mayoría de ficheros de la plataforma, por lo que en vez de incluir su código cada vez que se necesita, se han separado en estos ficheros y se los añade, asemejando la función a un *include*:

- *params.php*: incluye los datos de la conexión a la API de Moodle, el dominio del servidor y la inicialización de curl.
- *database.php*: aporta la inicialización de la conexión a la base de datos de la plataforma.

```
/*Declaración de la conexión a la base de datos (dirección del servidor,
usuario, contraseña, nombre de la bas de datos)*/
$db = new mysqli('localhost', 'root', '', 'certificate');

if ($db->connect_errno) {
    echo "Error número $db->connect_errno conectando a la base de datos.<br>Mensaje: $db->connect_error.";
    exit();
}
```

Ilustración 32: Conexión a la base de datos

5.5.7. Ficheros de elementos fijos

Estos ficheros tienen

- *navbar.php*: incluye la barra de navegación de la plataforma web y todos los enlaces a las diferentes funciones de la web.
- *footer.html*: incluye el pie de página de la plataforma web junto con enlaces externos.
- *index.php*: página por defecto del servidor, es decir la página a la que se redirige si únicamente escribes como URL el dominio del servidor. Corresponde con el formulario de login de la plataforma donde pide al usuario que introduzca su usuario y contraseña, los cuales enviará mediante el método POST a otro fichero para su verificación.
  El estilo del formulario está basado en uno de los ejemplos que proporciona Bootstrap.

```
<form method="post" action="login.php">

                <div class="mb-3">
                    <label for="username" class="form-label">Nombre de usuario</label>
                    <input type="username" class="form-control" id="InputUser" aria-describedby="userHelp" name="user">
                    <div id="userHelp" class="form-text">Este sitio web gestiona la información personal según la Ley de Protección de Datos</div>
                </div>
                <div class="mb-3">
                    <label for="password" class="form-label">Contraseña</label>
                    <input type="password" class="form-control" id="InputPassword" name="password">
                </div>

                <!-- 2 column grid layout -->
                <div class="row mb-4">
                    <div class="col-md-6 d-flex justify-content-center">
                        <a href="forgot_password.php">He olvidado mi contraseña</a>
                    </div>
                </div>

                <!-- Submit button -->
                <button type="submit" class="btn btn-primary btn-block mb-4">Entrar</button>

                <!-- Register buttons -->
                <div class="text-center">
                    <p>No estoy registrado: <a href="signup_form.php">Hazlo aquí</a></p>
                </div>
            </form>
```

Ilustración 33: Formulario con método POST

- *signup_form.php*: similar al fichero *index.php*, pero este se encarga del registro de un usuario en la plataforma. Pide información completa sobre la persona que quiere registrarse. Por lo demás usa el mismo funcionamiento que se explica en el punto anterior.
- *menu.php*: página de inicio de la plataforma, para acceder a ella se necesita haber iniciado sesión. Desde esta página se puede acceder a todas las funciones que permite la plataforma.

5.5.8. Ficheros de sesión y usuarios

- *login.php*: fichero *backend* asociado a index.php. Recibe por método POST los parámetros de nombre de usuario y contraseña, se conecta la base de datos y comprueba con los usuarios ya existentes ambos datos. En caso de que exista alguna coincidencia positiva inicia la sesión del usuario y redirige a *menu.php*. En caso contrario muestra un mensaje de error.

```php
//Comprueba si la contraseña introducida coincide con el encriptado
        if(password_verify($password, $obj->password)){
            session_start();
            //Se guarda la información del usuario logueado en las variables de sesión
            $_SESSION['user'] = $obj->username;
            $_SESSION['userid'] = $obj->id;

            //Variable sobre el role del usuario
            $_SESSION['role'] = $obj->role;

            $db->close();

            //Redirecciona al menu
            header("Location: http://localhost/menu.php");
            die();
        }
```

Ilustración 34: Verificación de la contraseña

- *signup.php*: fichero *backend* asociado a *signup_form.php*. Recibe por método POST toda la información del usuario, comprueba que no exista un usuario duplicado y que la contraseña se haya introducido ambas veces igual e introduce al nuevo usuario en la base de datos.
- *logout.php*: fichero *backend* que cierra la sesión iniciada por el usuario y redirige a *index.php*.
- *forgot_password.php*: formulario para la recuperación de la contraseña en caso de que un usuario no logueado haya olvidado la contraseña de su cuenta.

- *change_password.php*: fichero *backend* asociado a *forgot_password.php*. Recibe por método POST el correo asociado a la cuenta y la nueva contraseña introducidos por el usuario y tras comprobar su existencia actualiza la información del usuario con la nueva contraseña.

```php
//Comprueba que la contraseña se ha introducido correctamente ambas veces y que no exista ya un usuario igual
    if (($password === $password2) && $obj->count) {
        $password = password_hash($password, PASSWORD_BCRYPT);
        $sql = "UPDATE users SET password=\"$password\" WHERE username=\"$username\"";
        $stmt = mysqli_prepare($db, $sql);

        //Ejecutamos la sentencia SQL
        if (!mysqli_stmt_execute($stmt)){
            echo '<p class="text">Error al crear el nuevo usuario. ERROR: '.mysqli_stmt_error($stmt).'</p>';
        }

        //Redirecciona al menu
        header("Location: http://localhost/index.php");
        die();
    }
```

Ilustración 35: Actualización de una nueva contraseña

- *user.php*: muestra al usuario la información que tiene la plataforma sobre él con la intención de que este sea capaz de modificarla. Llama a la base de datos mediante una consulta *query* para la obtención de la información.
- *change_user.php*: fichero *backend* asociado a *user.php*. Recoge por método POST la información del usuario en caso de que este haya pinchado en el botón de aplicar cambios de *user.php*, entonces actualiza la información en la base de datos.

5.5.9. Ficheros de certificados

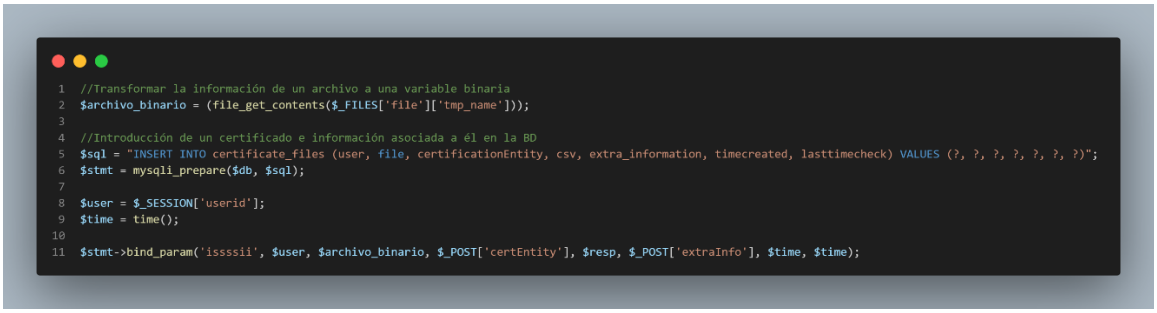
- *file.php*: formulario para la introducción de un nuevo certificado, donde se pedirá al usuario el fichero del certificado, la entidad certificadora que lo expidió e información extra que quiera incluir. También se incluye la programación de un *datalist* para la búsqueda de opciones al escribir la entidad certificadora.

```
<script>
    // Obtener la referencia del elemento datalist
    var list = document.getElementById('datalistOptions');

    <?php
    /*Recoge todas las fila de la tabla de certificados cuyo user coincida con
    el id del usuario qua ha iniciado sesión*/
    if ($result = $db->query("SELECT name FROM cert_entities WHERE validate=1")) {
        // Crea las celdas
        while ($obj = $result->fetch_object()){
    ?>

            var option = document.createElement('option');
            option.value = "<?php echo $obj->name; ?>";
            list.appendChild(option);

    <?php
        }
    }

    $db->close();
    ?>

</script>
```

Ilustración 36: Datalist

- *gen_csv.php*: función de generación del código CSV.

- *put_certificate.php*: fichero backend asociado a *file.php*. Recibe el certificado incluido y su información por método POST y gracias a la variable superglobal *$_FILES* y lo guarda en la base de datos como archivo binario.



```
//Transformar la información de un archivo a una variable binaria
$archivo_binario = (file_get_contents($_FILES['file']['tmp_name']));

//Introducción de un certificado e información asociada a él en la BD
$sql = "INSERT INTO certificate_files (user, file, certificationEntity, csv, extra_information, timecreated, lasttimecheck) VALUES (?, ?, ?, ?, ?, ?, ?)";
$stmt = mysqli_prepare($db, $sql);

$user = $_SESSION['userid'];
$time = time();

$stmt->bind_param('issssii', $user, $archivo_binario, $_POST['certEntity'], $resp, $_POST['extraInfo'], $time, $time);
```

Ilustración 37: Inserción de un nuevo certificado

- *input_csv.php*: formulario para la introducción de un código CSV con la finalidad de verificar el certificado asociado a ese código.
- *connect.php*: fichero dedicado a la visualización del certificado que se desee. El certificado se extrae de una base de datos conociendo su código CSV, el cual recibe por método POST si proviene de *input_csv.php* o por parámetro URL si proviene de *my_certificates.php*.
  Una vez recogido el CSV, el script lo comprobará en la base de datos de la plataforma y en la base de datos del Moodle de prueba al que se conectará con los web services.
  Si no encuentra el CSV mostrará un mensaje informativo.
  Si lo encuentra, muestra una tabla con la información relevante del certificado, debajo de esto aparecerá un visualizador de PDFs donde se podrá verlo, con opciones para pasar de páginas y aumentar y disminuir el zoom, y al final de la página el código QR para verlo en toda la ventana.

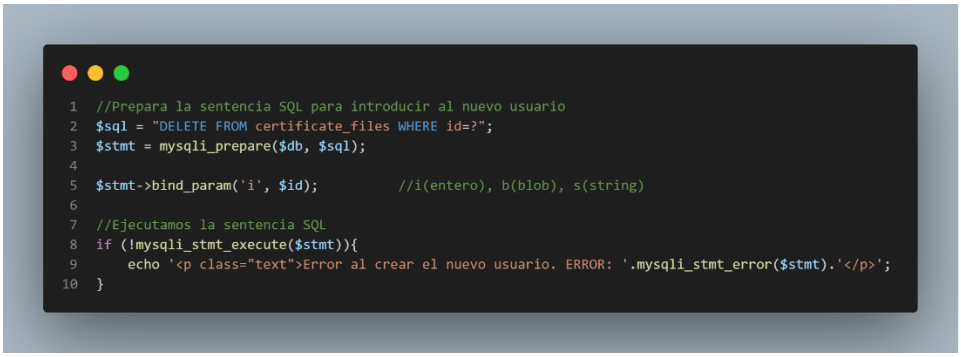
```
//Introducción del certificado en pdf en la ventana de visualización
function render() {
    myState.pdf.getPage(myState.currentPage).then((page) => {

        var canvas = document.getElementById("pdf_renderer");
        var ctx = canvas.getContext('2d');

        var viewport = page.getViewport(myState.zoom);

        canvas.width = viewport.width;
        canvas.height = viewport.height;

        page.render({
            canvasContext: ctx,
            viewport: viewport
        });
    });
}
```

Ilustración 38: Visualización de un PDF

```
pdfjsLib.getDocument('./my.pdf').then((pdf) => {

    myState.pdf = pdf;
    render();

});
```

Ilustración 39: Inserción de un PDF en la ventana

- *show_pdf.php*: fichero para la visualización de un certificado PDF en la ventana completa. Es utilizado como enlace proporcionado en un código QR.
- *my_certificates.php*: fichero que muestra el almacén de todos los certificados que un usuario tiene introducidos en la plataforma.
  Crea una tabla dinámica con JavaScript donde se muestran la información de los certificados y proporciona 3 opciones más al usuario: ver, redirige a connect.php con el código CSV del certificado que se quiere visualizar; subir, compartir la información del certificado en la blockchain; y eliminar, elimina el certificado y su información de la base de datos.
- *delete_certificate.php*: fichero *backend* asociado a la función de eliminar de *my_certificates.php*, actualiza la base de datos para eliminar el certificado en cuestión.



```php
//Prepara la sentencia SQL para introducir al nuevo usuario
$sql = "DELETE FROM certificate_files WHERE id=?";
$stmt = mysqli_prepare($db, $sql);

$stmt->bind_param('i', $id);          //i(entero), b(blob), s(string)

//Ejecutamos la sentencia SQL
if (!mysqli_stmt_execute($stmt)){
    echo '<p class="text">Error al crear el nuevo usuario. ERROR: '.mysqli_stmt_error($stmt).'</p>';
}
```

Ilustración 40: Eliminación de un certificado

- *pdf.php*: fichero para la generación de un certificado propio de la plataforma, es decir, creación de un PDF gracias a la librería Html2pdf.

5.5.10. Ficheros de administradores

- *admin_certs.php*: fichero que crea una tabla con la información de todos los certificados que se encuentran en la base de datos de la plataforma, independientemente del propietario del certificado. Permite las opciones de eliminar y visualizar el certificado. Solo pueden acceder a él usuarios con el rol '*admin*'.
- *admin_users.php*: fichero que crea una tabla con la información de todos los usuarios que están registrados en la plataforma. Permite las opciones de eliminar usuario y cambiar su rol. Solo pueden acceder a él usuarios con el rol '*admin*'.
- *admin_certentites.php*: fichero que crea una tabla con la información de las entidades certificadoras de las que hay constancia en la plataforma. Permite las opciones de eliminar entidad y verificar entidad, con ello se mostrará como opción la próxima vez que un usuario intente introducir un nuevo certificado. Solo pueden acceder a él usuarios con el rol '*admin*'.
- *admin_functions.php*: fichero *backend* que implementa las funciones de los 3 ficheros comentados anteriormente. Las funciones implementadas son: eliminar certificado, eliminar usuario, cambiar rol de usuario, eliminar entidad y verificar entidad. En todas estas funciones se actualizará la base de datos de la plataforma con la nueva información.

5.5.11. Otros ficheros

- *moodle_API.php*: formulario que implementa la comprobación y actualización de la información de conexión a la API de Moodle
- *change_moodle_API.php*: fichero *backend* asociado a *moodle_API.php*. Ejecuta la función en caso de que el usuario quiera modificar la información del formulario y actualiza la base de datos.

5.6. Blockchain

Blockchain proporciona una red de datos rápida y descentralizada capaz de registrar y almacenar transacciones en una cadena de bloques de forma segura.

La tecnología de la *blockchain* permitiría a este trabajo guardar de forma segura y pública parte de la información de los certificados (código CSV asignado en la plataforma, entidad certificadora e ID del usuario) que los usuarios subirán a la plataforma. Serviría de forma redundante de guardar y compartir esta información, pero añadiendo la complejidad y la innovación que provoca la *blockchain*.

5.6.1. Web3.php

Web3.php es una librería escrita en PHP que permite la interacción mediante código con la red de Ethereum.

Para su instalación será necesario el gestor de paquetes *composer* que ya se ha utilizado antes. Primero se tiene que modificar el fichero *composer.json* de la carpeta del proyecto y añadir la siguiente línea de código:

Ilustración 41: Estabilidad composer

Por último, en una terminal del ordenador se ejecutará el comando '*composer require web3p/web3.php dev-master'* y la instalación de la librería habrá concluido. Como resultado el fichero *composer.json* debería quedar de la siguiente manera:

Ilustración 42: composer.json

Esta librería sería la que permite conectarla plataforma con la red blockchain y también ejecutar el Smart contract que se le indique.

5.6.2. Smart contract

El smart contract se desarrolla en el entorno de desarrollo web remix IDE, que proporciona un editor de código, un compilador y opciones para ejecutar.

En el smart contract se declaran las siguientes variables:

```
event NewMessage(string csv, string username);

struct Data {
    string csv;
    string certificationEntity;
    string usernameID;
    uint fecha;
}

Data[] public messages;
```

Ilustración 43: Variables smart contract

Una estructura de datos que almacenará los datos que los usuarios podrán compartir sobre sus certificados (código CSV asignado al certificado, entidad certificadora que ha expedido el certificado, ID del usuario propietario y la fecha en la que se ha compartido), un array de las estructuras de datos llamada Data y la declaración del evento que se emitirá cada vez que se introduzca nueva información.

```
function createMessage(string memory _csv, string memory _certEntity, string memory _user) public {
    Data memory newMessage = Data(_csv, _certEntity, _user, block.timestamp);
    messages.push(newMessage);
    emit NewMessage(_csv, _user);
}
```

Ilustración 44: Función createMessage

La función createMessage almacenará en los datos que recibe por parámetros, es decir, los datos del certificado y emitirá el evento.

5.7. Paquetes

El proyecto se divide en 4 paquetes: gestión de usuarios, gestión de certificados, gestión de entidades y gestión de la conexión a Moodle. Los cuatro paquetes conforman las cuatro partes principales.

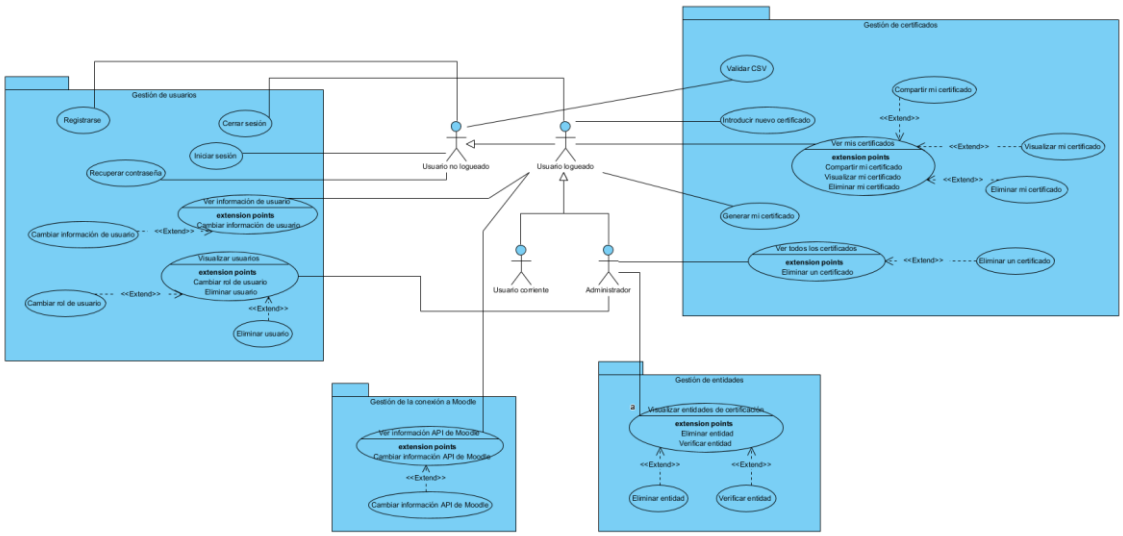


Ilustración 45: Diagrama de casos de uso

En el diagrama también se puede ver la disposición de los actores participantes. El actor Usuario no logueado tiene acceso a los casos de uso que se encuentran antes de iniciar una sesión en la plataforma (iniciar sesión, registrarse y recuperar contraseña) y además el caso de uso 'validar CSV' para que cualquier persona puede comprobar un certificado. El actor Administrador tiene acceso a Visualizar usuarios, Ver todos los certificados, Visualizar entidades de certificación y a los casos de uso que extienden de estos. Para el resto de los casos de uso se accede desde el actor Usuario logueado o en su defecto, actores que hereden de él.

5.7.1. Gestión de usuarios

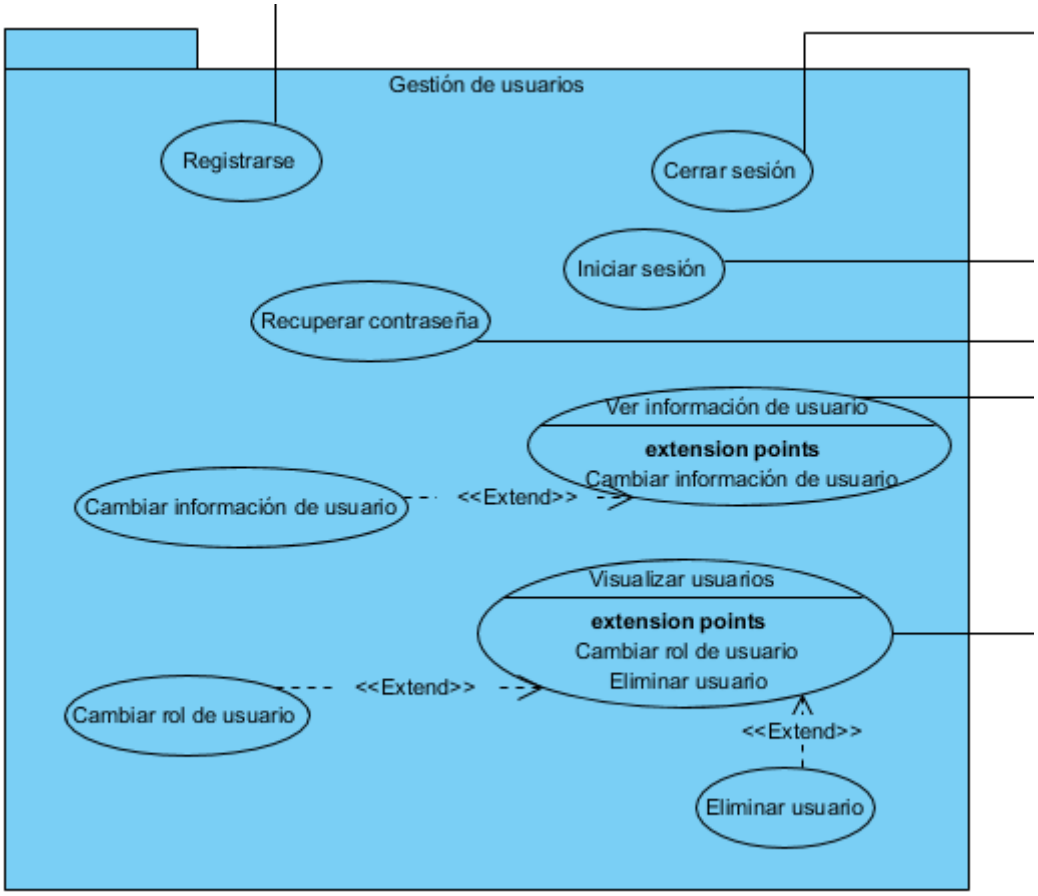


Ilustración 46: Paquete gestión de usuarios

El paquete 'gestión de usuarios' se encargará, como su nombre indica, de los usuarios que se estén registrados en la plataforma, así como de la información referente a ellos y de las sesiones en la plataforma.

Los casos de uso que se incluyen en este paquete son: Iniciar sesión, Registrarse, Recuperar contraseña; accedidos desde un actor usuario no logueado, Cerrar sesión, Ver información de usuario, Cambiar información de usuario; accedidos desde un actor usuario logueado, Visualizar usuarios, Cambiar rol de usuario y Eliminar usuario; accedidos desde un actor administrador.

5.7.2. Gestión de certificados

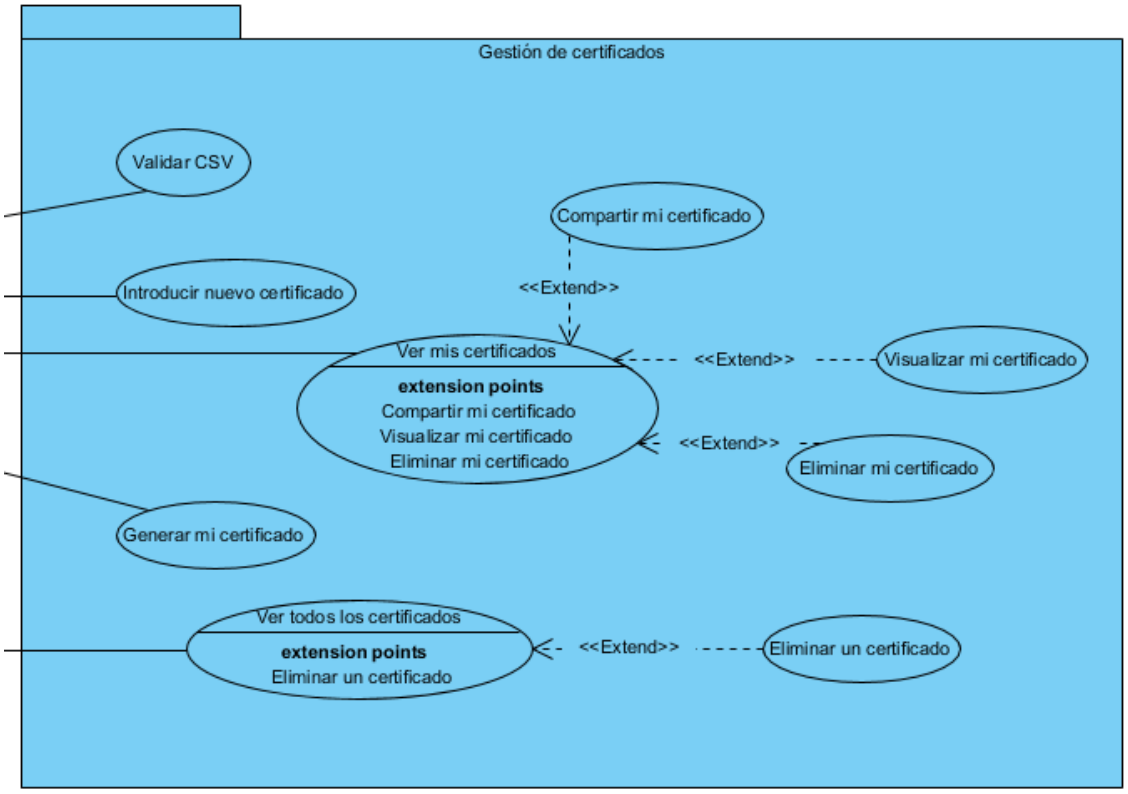


Ilustración 47: Paquete gestión de certificados

El paquete 'gestión de certificados' se encargará de los certificados subidos y creados desde la plataforma para su manejo correcto, correspondiéndoles con un código CSV y su visualización por parte de cualquier usuario.

Los casos de uso que se incluyen en este paquete son: Validar CSV; accedido desde un actor usuario no logueado, Introducir nuevo certificado, Ver mis certificados, Compartir mi certificado, Visualizar mi certificado, Eliminar mi certificado, Generar mi certificado; accedidos desde un actor usuario logueado, Ver todos los certificados y Eliminar un certificado; accedidos desde un actor administrador.

5.7.3. Gestión de la conexión a Moodle

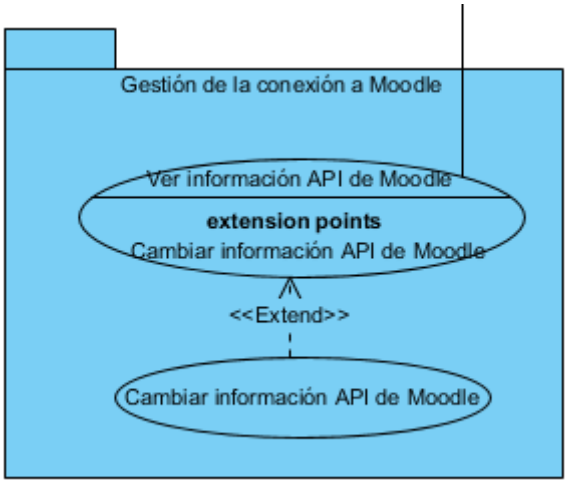


Ilustración 48: Paquete gestión de la conexión a Moodle

El paquete 'gestión de la conexión a Moodle' se encargará de manejar la información necesaria para que la plataforma se pueda conectar con el Moodle indicado por el usuario.

Los casos de uso que se incluyen en este paquete son: Ver información API de Moodle y Cambiar información API de Moodle; accedidos desde un actor usuario logueado.

5.7.4. Gestión de entidades

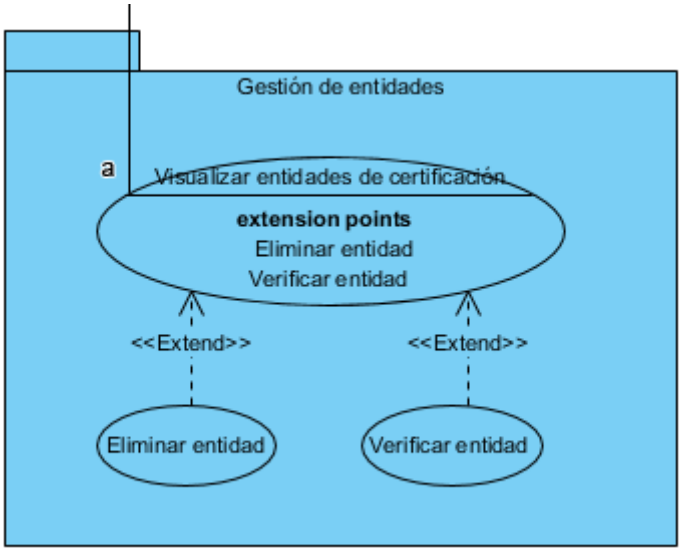


Ilustración 49: Paquete gestión de entidades

El paquete 'gestión de entidades' se encargará de controlar las diferentes entidades que aparecerán al subir certificados a la plataforma y permitirán a los usuarios tener las opciones para completar la información de sus certificados.

Los casos de uso que se incluyen en este paquete son: Visualizar entidades de certificación, Eliminar entidad y Verificar entidad; accedidos desde un actor administrador.

6. TRABAJOS RELACIONADOS

Este Trabajo de Fin de Grado, como ya se ha descrito a lo largo de la memoria, lleva a cabo una plataforma que actúa como base de datos para certificados y su posterior verificación por terceras personas.

Este tipo de funciones se asemeja bastante a las aplica la Sede Electrónica de la USAL que es capaz de recoger documentos, como trámites o certificados, para registrarlos y asignarlos un código de verificación para que cualquier persona que posea ese código pueda acceder y visualizar el documento.

La sede electrónica admite una gran variedad de documentos de relevancia dentro de la Universidad de Salamanca, en cambio la plataforma desarrollada se centra exclusivamente en los certificados de tipo académico, lo que permite especializarse más en ese ámbito.

La Universidad de Salamanca no es el único organismo que posee una herramienta del estilo, por ejemplo, la seguridad social, educación, la junta de Castilla y León y otras muchos organismos estatales o autónomos también tienen una sede electrónica donde guarda y administra todos los documentos que le competen.

7. LÍNEAS DE TRABAJO FUTURAS

Para este Trabajo de Fin de Grado existen múltiples líneas futuras de desarrollo dependiendo de las funciones que se deseen añadir. El proyecto desarrollado se ha realizado con un tiempo y recursos muy limitado por lo que puede considerarse un esbozo de lo que puede llegar a ser. Con más tiempo y trabajo, manteniendo una planificación ordenada y objetivos ambiciosos, se puede llegar a conseguir un producto muy interesante, innovador y con una gran finalidad.

Como mejora principal está el despliegue de la plataforma en el servidor que no tenga un ámbito únicamente local y abrir el uso de la plataforma al público. Para ello, habría que asegurar la estabilidad de la plataforma para la conexión de múltiples usuarios a la vez, evitar posibles usos fraudulentos con métodos de seguridad como recaptcha de Google para la subida de certificados y el registro de nuevos usuarios.

Otro tipo de mejoras pueden ser:

- Almacenamiento completo de datos de los certificados en la blockchain, evitando así el uso de una base de datos en un servidor centralizado. Sería necesario asegurar la confidencialidad de los datos encriptando los importantes, como los datos del usuario propietario del certificado o de la información confidencial de este.
- Posibilidad de visualizar los perfiles de usuarios. Cada usuario debería de ser capaz de elegir que certificados deja públicos para que cualquiera que acceda al perfil pueda comprobar.
- Añadir la capacidad de firmar digitalmente cualquier tipo de documento.

8. CONCLUSIONES

Para concluir con la memoria de este trabajo expondré las conclusiones a las que he llegado una vez finalizado este Trabajo de Fin de Grado.

Creo conveniente destacar que el esfuerzo que se le dedica a un trabajo de este calibre es bastante considerable y es muy dedicado toda la búsqueda de información, planificación, programación, testeo, documentar… Un punto para tener en cuenta es que estos proyectos son mucho más sencillos y se consigue llegar a mejores resultados si se realiza bajo un grupo de trabajo multidisciplinar en el que se puedan repartir las tareas. También provee al proyecto de diversos puntos de vista, con lo que se consiguen muchas más ideas para aumentar la envergadura del proyecto y, sobre todo, más capacidad analítica, por ejemplo, a la hora de resolver los problemas que vayan surgiendo.

Respecto al resultado del proyecto, creo que este establece una base funcional clara de la funcional total que se puede llegar a conseguir desarrollando más adecuadamente el proyecto, es decir con más recursos y tiempo.

En cuanto a las herramientas utilizadas considero que actualmente PHP puro no es la mejor opción para el desarrollo web, primero porque existen *frameworks* que facilitarían al programador una tarea más sencilla, y segundo, porque hay otros lenguajes de programación web más dominantes en el sector, como es caso de TypeScript o algunos de los *frameworks* de JavaScript. Por otro lado, el entorno de desarrollo, en este caso Visual Studio Code, utilizado junto con otros programas si que me parece que cumplen con bastante éxito su función. XAMPP que permite la simulación del servidor ha sido muy útil durante todo el desarrollo, además de que tenga incluida una base de datos relacional le aporta un mayor valor.

La blockchain es una tecnología innovadora que aporta a mejorar la plataforma y aunque, esta tecnología no es sencilla, esta complejidad es lo que le aporta mayor valor, por lo que pienso que puede llegar a integrarse en mayor medida.

Para finalizar con la memoria, lo más importante para mi personalmente pienso que ha sido la experiencia conseguida realizando un proyecto desde cero, sobre todo los puntos negativos y errores, ya que me van a permitir aprender de ellos y poder evitarlos en un futuro.

9. REFERENCIAS

- Moodle. (2022, Marzo). Moodle downloads. https://download.moodle.org/
- Moodle. (2021). Moodle Custom certificate. https://moodle.org/plugins/mod_customcert
- The PHP Group. (2001). Manual de PHP. https://www.php.net/manual/es/index.php
- Oracle. (2022). The BLOB type and TEXT types. https://dev.mysql.com/doc/refman/8.0/en/blob.html
- GitHub. (2014, Mayo). moodlehq/sample-ws-clients/PHP-REST/curl.php. https://github.com/moodlehq/sample-ws-clients/blob/master/PHP-REST/curl.php
- Moodle. (2020, Octubre). Web services. https://docs.moodle.org/dev/Web_services
- Stackoverflow. (2018, Febrero). How to enable web services on Moodle? https://stackoverflow.com/questions/48928475/how-to-enable-web-services-on-moodle
- Moodle. (2022, Febrero). Creating a web service client. https://docs.moodle.org/dev/Creating_a_web_service_client
- Codigonautas. (2021, Abril). Cómo crear un visor de PDF JavaScript. https://codigonautas.com/como-crear-un-visor-de-pdf-javascript/
- Stackoverflow. (2012, Abril). How to get the number of pages of a .PDF uploaded by user? https://stackoverflow.com/questions/10253669/how-to-get-the-number-of-pages-of-a-pdf-uploaded-by-user
- The PHP Group. (2003). Subida de ficheros. https://www.php.net/manual/es/features.file-upload.php
- Kiuvox. (2016, Febrero). Cómo guardar archivos en MySQL desde PHP. https://kiuvox.com/como-guardar-archivos-en-mysql-desde-php/
- Stackoverflow. (2012, Octubre). How do I generate a pdf-file from a binary file? https://stackoverflow.com/questions/13006505/how-do-i-generate-a-pdf-file-from-a-binary-file
- The PHP Group. (2006). Getcwd. https://www.php.net/manual/es/function.getcwd.php
- The PHP Group. (2020). Password_hash. https://www.php.net/manual/es/function.password-hash.php
- The PHP Group. (2020). Password_verify. https://www.php.net/manual/es/function.password-verify.php
- Forobeta. (2016, Octubre). ¿No funciona password_verify? https://forobeta.com/temas/no-funciona-password_verify.523239/
- DelftStack. (2021, Julio). Crear tabla usando JavaScript. https://www.delftstack.com/es/howto/javascript/create-table-javascript/
- Bootstrap. (2011, Agosto). Bootstrap: https://getbootstrap.com/
- MDBootstrap. (2021). Login form. https://mdbootstrap.com/docs/standard/extended/login/
- Parzibyte's blog. (2021, Junio). Generar códigos QR con JavaScript. https://parzibyte.me/blog/2021/06/26/generar-codigos-qr-javascript/
- AWS. (2022, Marzo). ¿Qué es una API? https://aws.amazon.com/es/what-is/api/
- Adobe Color. Rueda cromática. https://color.adobe.com/es/create/color-wheel
- Ethereum Remix. Remix IDE. https://remix-project.org/

- QuickNode. (2022). How to connect to Ethereum using PHP. https://www.quicknode.com/guides/web3-sdks/how-to-connect-to-ethereum-using-php
- Infura inc. (2022). Blockchain APIs. https://infura.io/
- Alchemy. (2022, Marzo). What is an ABI of a Smart Contract? Examples and Usages. https://www.alchemy.com/overviews/what-is-an-abi-of-a-smart-contract-examples-and-usage#:~:text=The%20Application%20Binary%20Interface%20(ABI,the%20application%20and%20the%20user.
- Programmer Think. (2022, Febrero). Web3 – overview of smart contract. https://programmer.ink/think/web3-overview-of-smart-contract-php-implementation-of-eth-4.html
- Github. (2022, Marzo). Web3p/web3.php. https://github.com/web3p/web3.php