PROYECTO FINAL DE CARRERA

ONLINE PASSWORD MANAGEMENT TOOL

MEMORIA

DEDICATORIA Y AGRADECIMIENTOS

EN PRIMER LUGAR DEDICAR ESPECIALMENTE A MI MADRE, A MI HERMANA CLARA Y ESPECIALMENTE A MI NOVIA LUCÍA QUE ADEMÁS DE SER MIS GRANDES APOYOS EN LA VIDA HAN SOPORTADO LAS INCONVENIENCIAS DE ESTE DURO TRABAJO, HORARIOS INVEROSÍMILES E INCÓMODOS, FRUSTRACIONES, ENFADOS Y LAMENTOS. QUE HAN COLABORADO EN TODO LO QUE HAN PODIDO, Y ME HAN APOYADO, DADO CARIÑO Y ÁNIMOS EN TODO MOMENTO. SÉ QUE SIN ELLOS YO NO SERÍA YO, NI ESTARÍA DONDE ESTOY.

A MIS AMIGOS EN ESPECIAL A GUILLERMO Y A NACHO, SIN SU AMISTAD Y AYUDA LA EXPERIENCIA VIVIDA EN ESTOS MESES NO HUBIERA SIDO NI REMOTAMENTE PARECIDA. GRACIAS POR HACERME VER LAS COSAS DE UN MODO TOTALMENTE DIFERENTE Y POR HABER ESTADO AHÍ.
GRACIAS DE CORAZÓN.

A TODOS MIS AMIGOS Y FAMILIARES POR HABER ESTADO A MI LADO TODO ESTE TIEMPO, HABERME APOYADO Y COMPRENDIDO SIEMPRE QUE LO HE NECESITADO.

ÍNDICE

JUSTIFICACIÓN DEL TFC

Actualmente el método de autentificación más extendida es el de usuario/contraseña. En la actualidad el número de aplicaciones que requieren una pareja de usuario/contraseña es cada vez más elevado.

Aunque sea una práctica muy poco recomendable, una gran mayoría de los usuarios usan la misma pareja de usuario/contraseña en las diferentes webs / redes sociales / aplicaciones.

A partir de este problema nos planteamos un sistema que sea capaz de almacenar todas nuestras contraseñas y al que podremos acceder a través de una contraseña maestra.

OBJETIVOS

El objetivo principal es desarrollar una aplicación web que nos permita a través de una llave maestra, acceder a un entorno seguro en el cual podamos almacenar información sensible. El uso primordial sería el de almacenar parejas de usuario / contraseña de diferentes servicios, pero también podría ser usado para almacenar números de tarjeta de crédito y PIN, combinaciones de cajas fuertes...
Las características básicas de este sistema son las siguientes:

- Poder ser accesible desde cualquier dispositivo con conexión a internet.
- Ofrecer al usuario la posibilidad de almacenar parejas de usuario/contraseña de los diferentes sistemas.
- Almacenar todos los datos de la aplicación cifrados. Haciendo imposible a los administradores de la misma el acceso a los datos de los usuarios.

- Mostrar la fortaleza de nuestra contraseña maestra.

- Encriptación en el lado del cliente. Nuestros datos no deben viajar en plano por la red.

El método que se ha seguido para el desarrollo del software ha sido incremental. En base a los requerimientos recogidos durante el análisis y diseño se libró una beta con parte de la funcionalidades El control de versiones se realizó a través de la aplicación GitX para MAC OS X.

PLANIFICACIÓN DEL PROYECTO

La definición de un ciclo de vida facilita el control sobre los tiempos en que es necesario aplicar recursos de todo tipo al proyecto. Todas las tareas del proceso de desarrollo de software deben ser planificadas, es decir, para cada una de ellas se debe establecer una fecha aproximada de inicio y otra de fin. Además, todas las tareas deben ser controladas a lo largo de todo el proceso de producción, esto es, se debe realizar un seguimiento continuo del proyecto informático. El trabajo se define con una duración de 4 meses y a continuación veremos el plan de trabajo

PAC 2 Análisis y diseño:

La primera actividad implica la recopilación, clasificación y análisis de toda la información relevante para llevar a cabo el proyecto.

Temporización: 06/10/2012 – 31/10/12 (25 días)

Objetivos: Profundizar sobre la criptografía basada en JavaScript, conocer las librerías disponibles y elegir las más adecuadas para realizar mi proyecto. Establecer de forma detallada funcionalidades del complemento web, para realizar un gestor de contraseñas online. Además obtener el análisis y el diseño de la aplicación a desarrollar. En una primera instancia, se había pensado que esta fase fuese un poco más extensa, pero nos requerirá más dedicación la actividad a continuación.

PAC 3 Implementación:

Implementación del complemento web y juegos de pruebas. Codificación y pruebas.

Temporización: 02/11/12 – 09/12/12 (27 días)

Objetivos:

- Implementar la aplicación en base al diseño creado.
- Generar juegos de pruebas que garanticen el correcto funcionamiento del gestor de contraseñas
- Demostrar que se cumplen con los requisitos establecidos.

III. PAC 4 Batería de pruebas y documentación

Documentación del producto desarrollado.

Temporización: 02/11/12 – 09/12/12 (27 días)

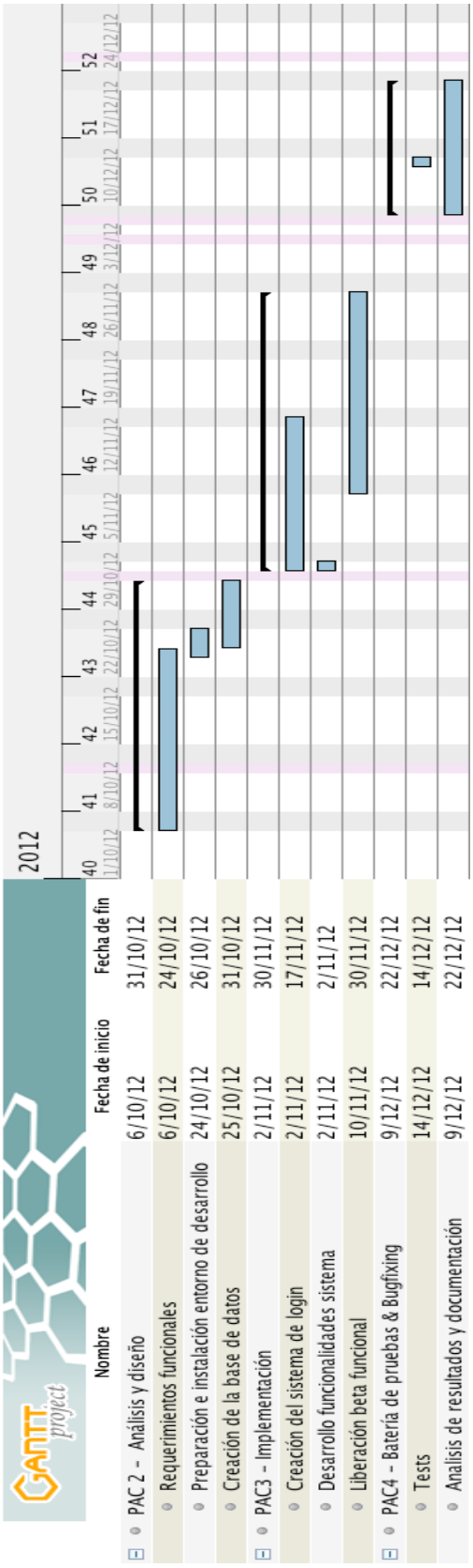
Objetivos:

- Comparación entre los artefactos de análisis y el resultado de la fase de implementación

Ejecución de la batería de pruebas

- Redacción de la documentación

Planificación temporal:



ARQUITECTURA DEL SISTEMA

En este apartado detallaremos los componentes tanto a nivel de software como de hardware que compondrán el sistema y los requisitos necesarios para el desarrollo y puesta en producción.

ARQUITECTURA DE HARDWARE

El siguiente diagrama representa la arquitectura de hardware necesaria para poder ejecutar la aplicación.

Al tratarse de una aplicación web podemos considerar que se puede acceder desde terminales externos a la red local del cliente o por lo contrario solo desde los terminales conectados en la red local del cliente.

Consideraremos que al tratarse de una aplicación interna no hay necesidad de ejecutarla desde terminales externos.

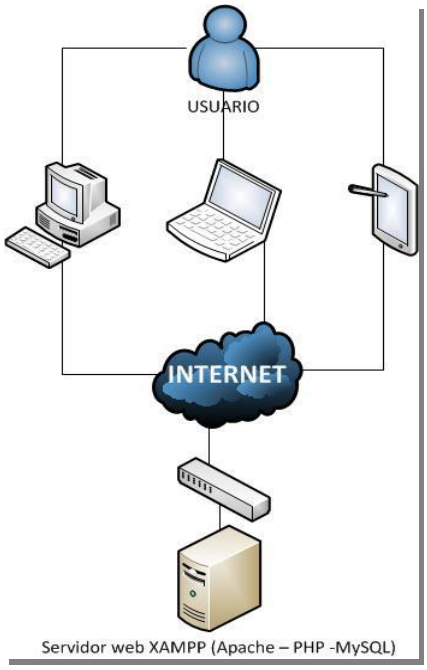


Fig. 1 Diagrama de Arquitectura de Hardware

ARQUITECTURA DE SOFTWARE

El siguiente diagrama representa la arquitectura del software que se empleará para el desarrollo de la aplicación.

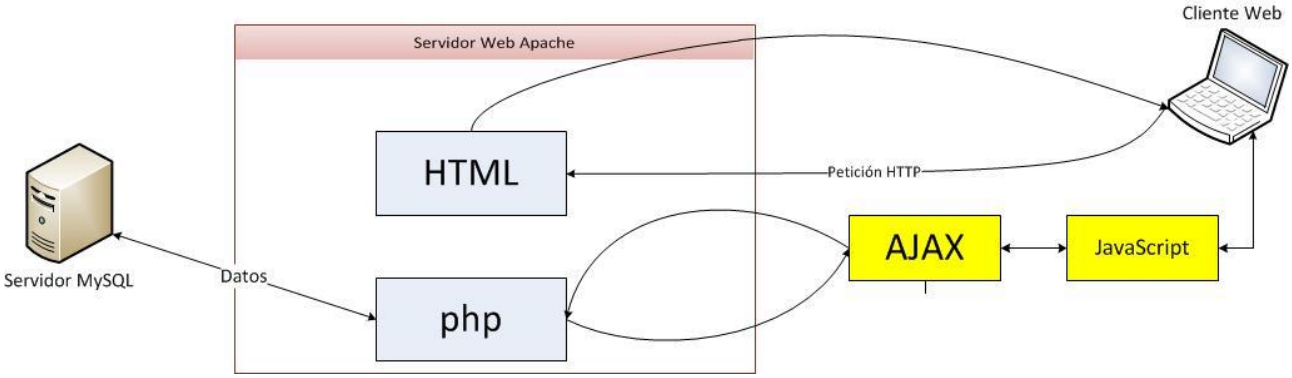


Fig. 2 Diagrama de Arquitectura de Software

La plataforma de desarrollo que se utilizará será **Aptana,** Un IDE basado en Eclipse y especializado para el desarrollo de aplicaciones web.

Concretamente consistirá en un proyecto web **PHP** combinado con tecnología **AJAX**.

Respecto a la capa de datos se utilizará el gestor de bases de datos **MySQL**

ANÁLISIS FUNCIONAL

En este apartado mostraremos los detalles y especificaciones fruto de la fase de análisis del proyecto.

Análisis de requerimientos / funcionalidades

Se desea un sistema que permita a un usuario, previamente autenticado en el sistema, almacenar una pareja de usuario / contraseña.

En esta primera versión los datos almacenados podrán ser únicamente visualizados por el usuario que los haya introducido. Por tanto, la aplicación debe permitir registrar usuarios mediante un nombre identificativo y una contraseña.

El sistema deberá encriptar en el lado del cliente todos los datos que son insertados posteriormente en la base de datos. Además el sistema deberá informarnos de la fortaleza de nuestras contraseñas.

En nuestro sistema se usan dos librerías. La Stanford JavaScript Crypto Library y la passpack.js que es una librería orientada a la creación de aplicaciones con tecnología Host-Proof Hosting.

El estándar de cifrado elegido es el AES ya que ambas librerías lo implementaban y podía integrarlas en mi aplicación sin problemas de compatibilidad.

¿Qué es AES (Advanced Encryption Standard)?

En 1997 el NIST (National Institute for Standars and Technology) convocó públicamente un concurso para la adopción de un nuevo estándar de cifrado en bloque simétrico que sustituyese al DES. El ganador del mismo fue el sistema llamado RIJNDAEL, desarrollado por V. Rijmen y J. Daemen (Univ. de Lovaina).

Describiremos brevemente como funciona este estándar.

Bloques y claves:

- Cifra bloques de longitudes 128, 192 o 256 bits.
- Claves de las mismas longitudes
- Los tamaños de ambas se fijan independientemente.
- Maneja toda la información en bytes. Luego, cifra mensajes de longitudes 16, 24 o 32 bytes con claves de longitud 16, 24 o 32 bytes.
- Mensajes y claves se manejan en forma de matrices con 4 filas.

Por tanto:

Los mensajes son matrices de $4 \times N_b$ bytes, siendo $N_b = 4, 6, 8$.

La clave es una matriz de $4 \times N_k$, siendo $N_k = 4, 6, 8$

Algoritmo de cifrado:

El cifrado consiste esencialmente en un numero variable de vueltas *r* (*r* =10, 12, 14) de un algoritmo básico que utiliza como entradas (para la vuelta *i*-esima) una clave de vuelta K_i del mismo tamaño que el mensaje y la matriz saliente de la vuelta anterior M_i-1.

Algoritmo:

Entrada el mensaje M y la clave K.

1. Se calculan las claves de vuelta: K_0, K_1,. . ., K_r (del mismo tamaño que M).
2. S:=M $\oplus$ K_0
3. Para *i* = 1, . . . , *r* hacer S := R[*i*]*(S, K_i)*
4. Devolver *S*

El número de vueltas *r* depende de N_b y N_k, sus valores vienen dados por la tabla:

	$N_k = 4$	$N_k = 6$	$N_k = 8$
$N_b = 4$	10	12	14
$N_b = 6$	12	12	14
$N_b = 8$	14	14	14

Algoritmo de las rondas:

Las rondas *R[i]* son todas iguales, excepto la última y consisten en aplicar 4 operaciones sucesivamente:
Algoritmo: Para una matriz de estado *S* y una clave K_i:

1. *S*:= ByteSub (*S*)

2. S:= ShiftRow(S).

3. S:= MixColumn(S).

4. S := AddRoundKey(S, K_i).

La última vuelta es igual que las demás, excepto que no se ejecuta la etapa 3. Nótese que la clave de vuelta solo se usa en la última etapa.

Operación ByteSub

Es una operación sobre cada una de las entradas (bytes) de la matriz *S* que consiste, para a = (a_1, . . . , a_8) ∈ F^8_2 = F_{256} en:

1. Si a ≠ 0 6 sustituir a por su inverso, a := 1/a.
2. Hacer un cifrado afín de Hill, a → a · A + b, siendo A una matriz binaria fija de tamaño 8 × 8 y b un vector fijo de 8 bits.

El efecto de la operación ByteSub es un cifrado por substitución mono alfabética usando como alfabeto A = F^8_2 La operación de invertir un elemento no es lineal, por lo que la sustitución en su conjunto no es lineal sobre F_2.

La matriz A y el vector b son los siguientes:

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \quad b = (1,1,0,0,0,1,1,0)$$

La operación ByteSub se puede ver como una S-caja y se puede implementar como tal o bien mediante la programación de las operaciones (admite simplificaciones importantes).

Representación y operaciones de ‘bytes´:

Representamos un byte como un polinomio binario de grado menor que 8:

$$\text{“8D”} = 10001101 = [1,0,1,1,0,0,0,1] \mapsto 1 + x^2 + x^3 + x^7$$

$$\text{“63”} = 01100011 = [1,1,0,0,0,1,1,0] \mapsto 1 + x + x^5 + x^6$$

La operación suma consiste en sumar coordenada a coordenada en F_2:

$$10001101 + 01100011 = 11101110$$

$$\text{“8D” + “63” = “EE”}$$

$$[1,0,1,1,0,0,0,1] + [1,1,0,0,0,1,1,0] = [0,1,1,1,0,1,1,1]$$

$$(1 + x^2 + x^3 + x^7) + (1 + x + x^5 + x^6) = x + x^2 + x^3 + x^5 + x^6 + x^7$$

El producto tiene alguna dificultad m as: consiste en multiplicar como polinomios y después tomar el resto por un polinomio fijo de grado $8,\ m(x)$ No vale cualquier polinomio m(x), el que se usa para nuestro caso es el polinomio $1+x+x^3+x^4+x^8$.. Por lo tanto, para multiplicar $10001100 = 1 + x^4 + x^5$ por $00011000 = x^3 + x^4$ hay que proceder como sigue:

1. Multiplicar como polinomios:
   $$(1 + x^4 + x^5)(x^3 + x^4) = x^3 + x^7 + x^8 + x^4 + x^8 + x^9 = x^3 + x^4 + x^7 + x^9$$
2. Reducir módulo $1 + x + x^3 + x^4 + x^8$:

$$x^3+x^4+x^7+x^9 \equiv x^3+x^4+x^7+x+x^2+x^4+x^5 \equiv x+x^2+x^3+x^5+x^7 \bmod (1+x+x^3+x^4+x^8)$$

Por tanto el resultado es [0, 1, 1, 1, 0, 1, 0, 1] = 10101110. O si preferimos: “8D” ×“63” = “AE”.

Con estas operaciones, convenimos en denotar a (F2)8 como F256.Se puede demostrar que con la operación de multiplicar anterior, cualquier elemento no nulo tiene un inverso; dicho en términos mas matemáticos, hemos dotado de estructura de cuerpo al conjunto de bytes. Todas las operaciones que se hacen sobre las distintas entradas de un estado se hacen con las operaciones descritas.

Operación ShiftRow:

Cada una de las filas de la matriz *S* se permuta por medio de una permutación que depende del número de columnas de *S*. Denotamos por la permutación circular a la izquierda:

$$\sigma(a,b,c,d) = (b,c,d,a); \quad \sigma(a,b,c,d,e,f) = (b,c,d,e,f,a);$$

El efecto de la operación sobre la fila *i*- esima de $S,\ f_i,$ es $\sigma^{c(i)}(f_i)$La primera fila queda inalterada, por lo que siempre es *c(0) = 0*. Los enteros *c(i), i = 1, 2, 3* dependen del número de columnas N_b de *M* y son:

	$c(1)$	$c(2)$	$c(3)$
$N_b = 4$	1	2	3
$N_b = 6$	1	2	3
$N_b = 8$	1	3	4

El efecto sobre la matriz *S* (para el caso en que $N_b = 4$) es entonces:

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{pmatrix} \mapsto \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{11} & a_{12} & a_{13} & a_{10} \\ a_{22} & a_{23} & a_{20} & a_{21} \\ a_{33} & a_{30} & a_{31} & a_{32} \end{pmatrix}$$

Operación MixColumn:

Opera sobre cada columna $c = (c_0, c_1, c_2, c_3)$ de la matriz M. Para ello se escribe la columna en forma polinómica: $c(T) = c_0 + c_1 T + c_2 T^2 + c_3 T^3$ y se hace la transformación:

$$c(T) := c(T)d(T) \bmod (T^4 + 1)$$

Siendo $d(T) = x + T + T^2 + (1 + x)T^3$ un polinomio fijo de grado 3.

La operación anterior se puede escribir en forma matricial como:

$$(c_0, c_1, c_2, c_3) \mapsto (c_0, c_1, c_2, c_3) \begin{pmatrix} x & 1 & 1 & 1+x \\ 1+x & x & 1 & 1 \\ 1 & 1+x & x & 1 \\ 1 & 1 & 1+x & x \end{pmatrix}$$

$$S \mapsto \begin{pmatrix} x & 1+x & 1 & 1 \\ 1 & x & 1+x & 1 \\ 1 & 1 & x & 1+x \\ 1+x & 1 & 1 & x \end{pmatrix} S$$

Por lo tanto la transformación es un cifrado de Hill en bloques de tamaño 4 sobre el alfabeto F256.

Operación AddRoundKey:

La matriz M se sustituye por la matriz $M \bigoplus K_i$

Calculo de las subclaves de vuelta:

A partir de la matriz de clave

$$K = \begin{pmatrix} k_{00} & k_{01} & \cdots & k_{0,N_k-1} \\ k_{10} & k_{11} & \cdots & k_{1,N_k-1} \\ k_{20} & k_{21} & \cdots & k_{2,N_k-1} \\ k_{30} & k_{31} & \cdots & k_{3,N_k-1} \end{pmatrix}$$

Se realizan dos operaciones. La primera es una expansión hasta completar una matriz W de tamaño 4×(r+1)Nb añadiendo consecutivamente columnas a la matriz K.
Posteriormente se toman como matrices Ki, i = 0, . . . , r, las sucesivas r + 1 cajas de tamaño 4 × Nb de la matriz W = (K0, . . . ,Kr).
Denotamos por W(i), (0 _ i < (r + 1)Nb), las columnas de la matriz W. Recordemos que las Nk primeras coinciden con las de la matriz K. El resto se calculan de acuerdo al siguiente algoritmo:

Algoritmo [Generación de claves]

Entrada: Las columnas *W(0), . . . ,W(N_k − 1).*

Para $i = N_k, \ldots, (r+1)N_b$ hacer

Si $i \equiv 0 \bmod N_k$ entonces

t := ByteSub $(\sigma(W(i-1)) \bigoplus (x^{i-1}, 0, 0, 0)$

$W(i) := W(i - N_k) \bigoplus t$

En otro caso $W(i) := W(i-1) \oplus W(i-N_k)$.

La operación ð indica una permutación circular a la izquierda.

En el caso en que $N_b > 6$ el algoritmo anterior sufre algunas modificaciones. Se puede encontrar toda la información detallada, en particular el algoritmo completo para la generación de las distintas claves de vuelta, en las páginas oficiales de Rijndael: y de NIST

Escenario 1: Gestión de usuarios Host-Proof Hosting

Host-Proof Hosting ¿Qué es? El sistema en el que he basado mi aplicación es un modelo de seguridad pública que permite para albergar los datos sin que la aplicación sea capaz de acceder a ellos. La información que pasa a través de Keyper está cifrada y no es rastreable, nadie puede ver los datos, ni siquiera Keyper.

Al alojar los datos confidenciales de forma cifrada, sólo el cliente del usuario puede acceder y manipularlos. El cliente en nuestro caso es un navegador de Internet con JavaScript habilitado.

Una vez que el usuario elige una clave de encriptado (la clave de cifrado utilizada para cifrar /descifrar sus datos), esta clave no es transmitida nunca al servidor. El servidor se limita al almacenamiento de los datos cifrados que el navegador le envía. Todo el cifrado y descifrado se realiza dentro del propio navegador.

Este método se compone de dos módulos principalmente. En el desarrollo del mismo he separado los procesos de autentificación y de encriptación en dos módulos totalmente diferentes. Esta idea ha sido tomada de la aplicación Passpack. En una primera instancia el usuario introduce su USER ID y su contraseña para loguearse en su cuenta. Al crear un nuevo usuario también se le indica la fortaleza de la contraseña que ha elegido para registrarse en nuestro servicio.

¿Cómo se mide la fortaleza de la contraseña? Mediante la entropía de la contraseña.

En seguridad informática, la entropía de una contraseña es la cantidad de aleatoriedad para hacer a la misma difícil de adivinar. Este término se expresa en términos de bits, y define una medida en el número de intentos que debe realizar un programa de computación para adivinar la misma. Una contraseña con entropía de n bits puede ser encontrada en 2n intentos. Así por ejemplo una contraseña con entropía de 1 bit debería poderse adivinar en 2 1intentos que es igual a 2 intentos. El matemático Claude Shannon fue el primero en introducir el término de entropía en la teoría de la información.

¿Cómo calcular la entropía de una contraseña?

La entropía de una contraseña cualquiera se determina por la longitud de la cadena por la entropía de cada carácter. Para determinar la entropía de cada carácter se determina por el logaritmo en base 2 por el tamaño del conjunto al que pertenece el carácter. La fórmula es la siguiente:

entropia_x_caracter= log2(n)

entropia_password= longitud * entropia_x_caracter

Donde n es el tamaño del conjunto al que pertenece el carácter y longitud es la longitud de la contraseña. Partiendo de esto podemos calcular la entropía de un conjunto de caracteres como nuestro alfabeto (a-z) como el valor de log2 (26) = 4.7 bits. A continuación mostramos los valores de entropía de bits para cada conjunto de caracteres.

- Números (0-9): Log2(10)= 3.32
- Letras minúsculas (a-z): Log2(26)= 4.7
- Letras mayúsculas, minúsculas y números (A-Z, a-z, 0-9): Log2(62)=5.95
- Todos los caracteres del teclado standard (94): Log2(94): 6.55

Mientras más alto es el valor de la entropía más segura es la contraseña, así podemos definir baremos para especificar la fortaleza de la misma por ejemplo:

- Entropía < 28 bits: Muy débil.
- Entropía <36 bits: Débil.
- Entropía <60 bits: Razonable.
- Entropía < 128bits: Segura.
- Entropía > 128 bits: Muy Segura.

Así por ejemplo una contraseña de longitud 10 de solo números tendrá una entropía de 33.2 Bits que se considera débil. [1]

[1] http://en.wikipedia.org/wiki/Password_strength

Si el proceso de autentificación se completa con éxito el sistema envía al usuario sus datos de forma encriptada al usuario.

Escenario 2: Zona de usuario y encriptación de los datos

Una vez el sistema ha detectado que el usuario ha completado el proceso de autentificación correctamente. El sistema automáticamente nos redirigirirá a la zona personal de cada usuario. Nada más entrar se nos volverá a solicitar la clave de desencriptado, esto como se ha mencionado antes nos permitirá desencriptar los datos recibidos desde el servidor. Desde esta página el usuario será capaz de consultar, crear, editar y eliminar la información que ha ido almacenando en el servicio. El sistema te permite almacenar los siguientes datos:

- Service, nombre del servicio o aplicación web de la cual queremos almacenar usuario y contraseña
- User, identificador de usuario
- Password, contraseña del servicio
- Comments, campo para almacenar valores adicionales

A través de la intuitiva barra de menú el usuario puede realizar todas las acciones mencionadas.

Requisitos no funcionales

A continuación se especifican detalladamente el conjunto de requisitos no funcionales que deberá cumplir la aplicación.

Al tratarse de una aplicación web y que accederemos a través de un navegador de internet se considera fundamental el acceso desde cualquier de los navegadores más utilizados como son: INTERNET EXPLORER, GOOGLE CHROME y FIREFOX. Aun así consideramos recomendable utilizar GOOGLE CHROME para una mejor experiencia.

Rapidez

Tanto el acceso como la navegación por la aplicación deben ser rápidos permitiendo al usuario realizar el trabajo eficientemente. Ya que la aplicación almacena contraseñas que probablemente se

Escalabilidad

La aplicación está pensada para grupos de desarrollo pequeños por lo que el número de usuarios que accederán concurrentemente no será elevado. Aun así la aplicación está diseñada para que puedan acceder un elevado número de usuarios por lo que permite una fácil escalabilidad apoyándose en una arquitectura de 3 capas.

Facilidad de uso

La interfaz de usuario debe ser simple y permitir acceder a las diferentes opciones con la menor navegación posible.

DIAGRAMA CASOS DE USO

A continuación se mostrará el conjunto de diagramas de casos de uso junto con una tabla descriptiva para cada uno de los casos de uso que intervienen.

CU01 Inicio de sesión

CU01	INICIO DE SESION	
PRECONDICION	El sistema deberá permitir a todos los usuarios al iniciar la aplicación realizar la validación de acceso con sus credenciales.	
POSTCONDICION	El usuario se ha validado correctamente dentro del sistema con sus credenciales.	
SECUENCIA	PASO	ACCION
	1	Al acceder a la aplicación se solicitarán las

		credenciales a través de un formulario de acceso.
	2	El usuario introducirá su usuario y contraseña y confirmará la acción.
	3	El sistema comprobará que los datos introducidos por el usuario son correctos.
	4	Si los datos son correctos, el sistema permite el acceso y asigna los permisos correspondientes habilitando las funcionalidades disponibles para el usuario.
EXCEPCIONES	**PASO**	**ACCION**
	3	Si los datos no son correctos, el sistema advertirá del error de acceso y volverá al paso 2.

CU02 Alta de usuario

CU02	ALTA DE USUARIO	
PRECONDICION	El sistema deberá permitir añadir clientes en el sistema.	
POSTCONDICION	Se han añadido un cliente en el sistema.	
SECUENCIA	PASO	ACCION
	1	Se introduce el usuario y la contraseña deseada en los dos campos destinados a ella
	2	El sistema comprobará que los datos introducidos por el usuario son correctos.
	3	Si los datos son correctos, el sistema crea un perfil nuevo
EXCEPCIONES	PASO	ACCION
	3	Si los datos no son correctos, el sistema advertirá del error de acceso y volverá al paso 1

CU03 Baja de usuario

CU02	ALTA DE USUARIO	
PRECONDICION	El sistema deberá permitir añadir clientes en el sistema.	
POSTCONDICION	Se han añadido un cliente en el sistema.	
SECUENCIA	PASO	ACCION
	1	Se introduce el usuario y la contraseña deseada en los dos campos destinados a ella
	2	El usuario pulsa el botón destinado a desactivar la cuenta
	3	El sistema pide confirmación
	4	El usuario confirma y el sistema borra la cuenta
EXCEPCIONES	PASO	ACCION
	3	El usuario cancela el procesa y la cuenta no es borrada

CU03 Alta de datos

CU03	ALTA DE CONTRASEÑA	
PRECONDICION	El sistema deberá permitir añadir datos en el sistema	
POSTCONDICION	Se han añadido datos en el sistema.	
SECUENCIA	PASO	ACCION
	1	Se introducen los datos a través del formulario destinado a ello
	2	El sistema comprobará que los datos introducidos cumplen las reglas de validación.
EXCEPCIONES	PASO	ACCION
	3	Si los datos no son correctos, el sistema advertirá del error de acceso y volverá al paso 1

CU04 Listado de datos

CU04	LISTADO DE DATOS
PRECONDICION	El sistema deberá permitir obtener un

	listado con los datos principales del usuario	
POSTCONDICION	Se han añadido datos en el sistema.	
SECUENCIA	**PASO**	**ACCION**
	1	El sistema realiza una búsqueda a partir del usuario logueado
	2	Devuelve el los datos correspondientes.
EXCEPCIONES	**PASO**	**ACCION**

CU05 Borrado de datos

CU05	BORRADO DE DATOS	
PRECONDICION	El sistema deberá permitir borrar los datos principales del usuario	
POSTCONDICION	Se han añadido datos en el sistema.	
SECUENCIA	PASO	ACCION
	1	El usuario selecciona el registro que se desea borrar
	2	El sistema solicita confirmación
	3	Los datos seleccionados se borran
EXCEPCIONES	PASO	ACCION

CU06 Actualización de datos

CU06	ACTUALIZACION DE DATOS
PRECONDICION	El sistema deberá permitir obtener un listado con los datos principales del usuario
POSTCONDICION	Se han añadido datos en el sistema.

SECUENCIA	PASO	ACCION
	1	El usuario selecciona el registro que se desea actualizar
	2	Se muestra el formulario de actualización
	3	El usuario actualiza los campos deseados y confirma la actualización
	4	Los datos son actualizados en la bd
EXCEPCIONES	**PASO**	**ACCION**

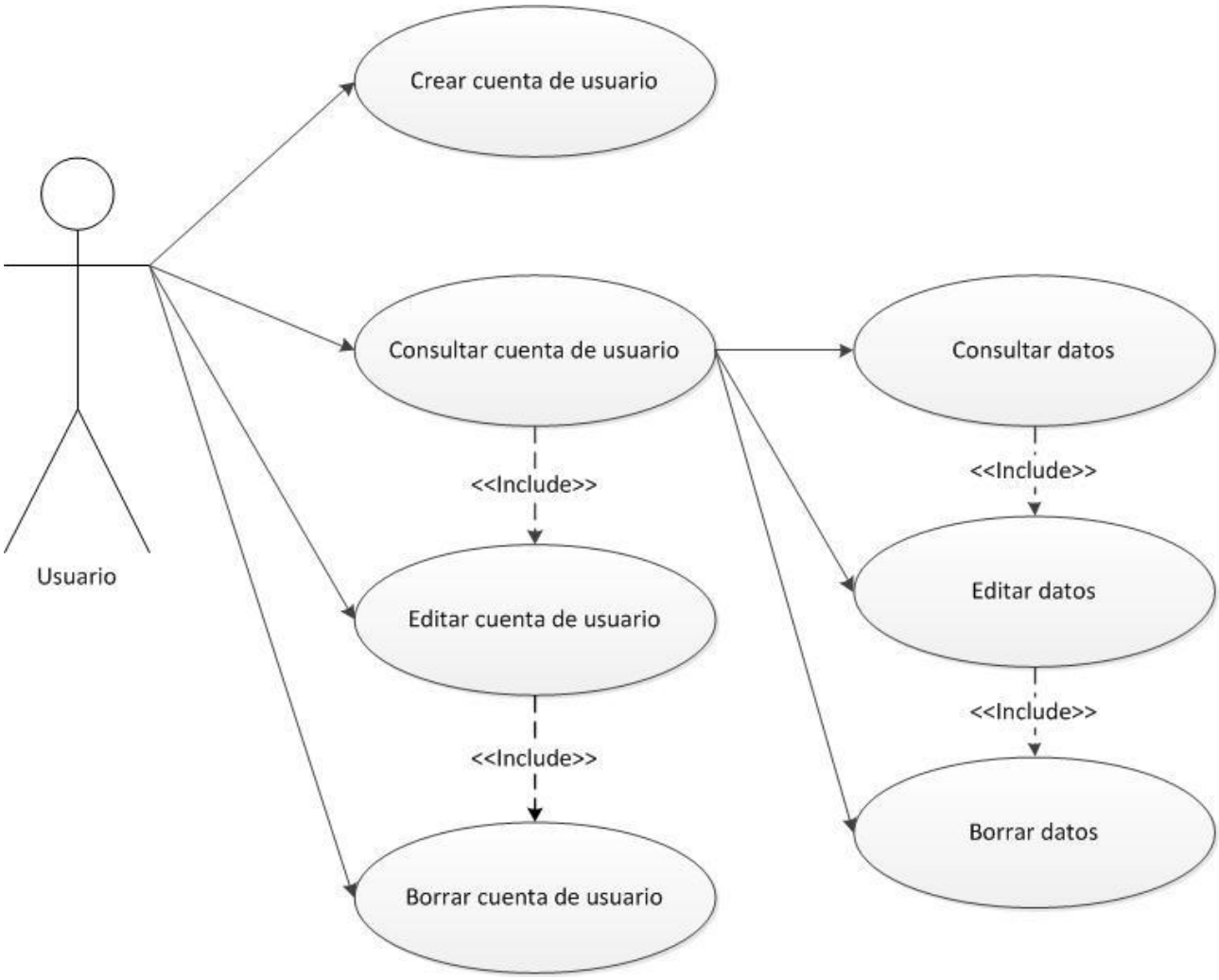
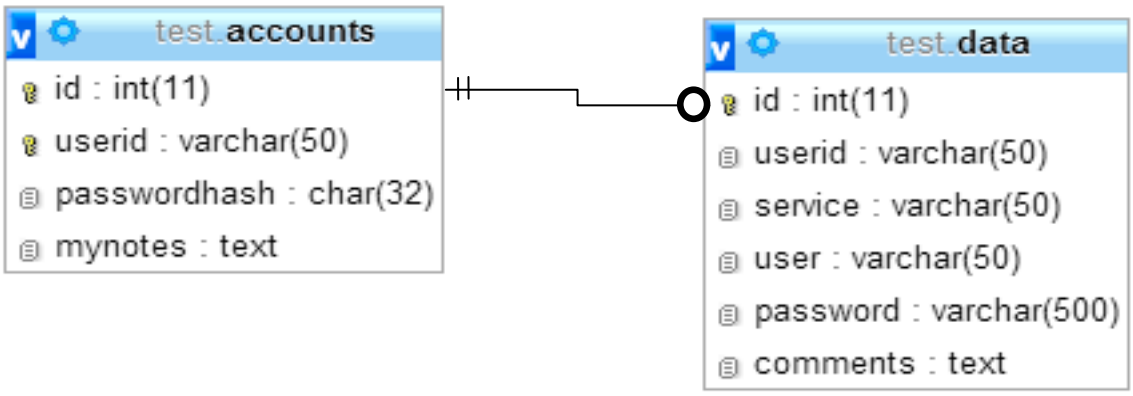


Fig. 3 Diagrama de casos de uso

DIAGRAMA DE BASE DE DATOS E-R



En este diagrama de base de datos podemos observar el conjunto de tablas y relaciones existentes que utilizara la aplicación para mantener toda la información

A continuación detallaremos todos los campos y la función principal de cada una de las tablas permitiendo una mejor comprensión

Tabla Accounts

Esta tabla contiene la información de autentificación de los usuarios de la aplicación en ella almacenamos el nombre de usuario, el hash de la contraseña de usuario y el campo my notes actualmente se usa para el proceso de login

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id	int(11)			No	None	auto_increment	
☐	userid	varchar(50)	latin1_swedish_ci		No	None		
☐	passwordhash	char(32)	latin1_swedish_ci		No	None		
☐	mynotes	text	latin1_swedish_ci		No	None		

Tabla Users

En esta tabla almacenaremos los datos relativos a cada usuario. Se compone de>

- Un índice sin encriptar que nos permitirá realizar las operaciones de borrado y actualización
- El nombre del servicio del cual queremos almacenar la pareja de usuario y contraseña
- El usuario de ese servicio
- La contraseña
- Un campo de comentarios para introducir valores diversos.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id	int(11)			No	*None*	auto_increment	
☐	userid	varchar(50)	latin1_swedish_ci		No	*None*		
☐	service	varchar(500)	latin1_swedish_ci		Sí	*NULL*		
☐	user	text	latin1_swedish_ci		Sí	*NULL*		
☐	password	text	latin1_swedish_ci		Sí	*NULL*		
☐	comments	text	latin1_swedish_ci		No	*None*		

IMPLEMENTACIÓN

Tanto en la introducción como en el diseño se han citado las tecnologías y requisitos que requerirá el proyecto. A continuación detallaremos todos los aspectos que han englobado la fase de implementación ampliando la información sobre los elementos citados

Software utilizado

El conjunto de herramientas utilizadas para el desarrollo de todos los componentes del proyecto es el siguiente:

- IDE de desarrollo. Aptana
- Servidor de plataforma. XAMPP para MAC OS X
- Microsoft Word 2010
- Microsoft Visio 2010

Frameworks y librerías

Los frameworks utilizados para el desarrollo de la interfaz:

- JQuery
- JQuery EasyUI

Las librerías utilizadas para la encriptación de los datos:

- Stanford JavaScript Crypto Library
- Passpack

Aplicaciones Cliente

Los usuarios de la aplicación tan solo requerirán de un PC con un navegador de internet. En diseño inicial se contempló el acceso con cualquiera de los principales navegadores. Tal como se detallará en el apartado de **Trabajo futuro** a pesar de ser posible acceder a través de cualquier navegador, se recomienda utilizar Google Chrome, ya que es el que ha arrojado mejores resultados durante el desarrollo de la aplicación.

Instalación y despliegue

A continuación se detalla paso a paso la instalación de la aplicación para su correcto funcionamiento.

La instalación se divide en dos partes:

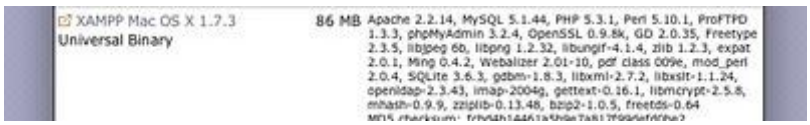
- Instalación de XAMPP
- Instalación de la aplicación en el servidor

Se incluye también el detalle de todos los aspectos de configuración a tener en cuenta como son las cadenas de conexión, directorios de ficheros, seguridad y permisos.

El primer paso de todos, es conseguir el software XAMPP. Para ello, nos dirigimos a la web de apache, que promueve este proyecto:

Descarga e instrucciones de XAMPP para mac

Descargamos el binario universal del programa (por lo que los usuarios de powerpc también pueden seguir estos pasos)



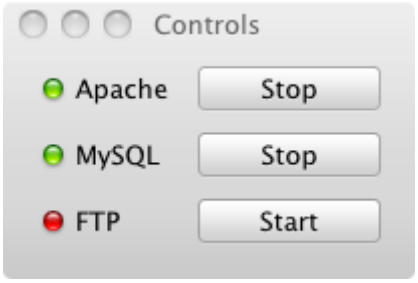
Una vez descargado y montado de forma automática la imagen (si no aparece un icono similar al de un USB con el nombre de "XAMPP for Mac OS X 1.7.3", ve a la carpeta descargas y haz doble click sobre el archivo recientemente descargado) Nos aparecerá la siguiente imagen:

Como muchas de las aplicaciones para este sistema operativo, se instala arrastrando el archivo a la carpeta aplicaciones.

Una vez copiado, si abrimos la carpeta, podemos ver que tiene varios alias. No nos interesan, solo queremos usar el icono llamado "XAMPP Control".

En Getting started, nos informa de información básica, como que, para acceder a la web, nos basta con poner, en un navegador, http://localhost (aún no funciona, tenemos que levantar los servidores), y que, por defecto, el usuario root de MySQL no tiene contraseña.

Ahora, procedemos a dar el botón "Start" de Apache (servidor web) y de MySQL. Es posible que nos pida la contraseña de nuestro usuario de Mac OS. Esto es así porque ejecuta unos scripts para poder hacer funcionar todo el sistema de BBDD. Si todo ha ido bien, las luces pertenecientes a los grupos que nos interesan estarán en verde, tal que así:

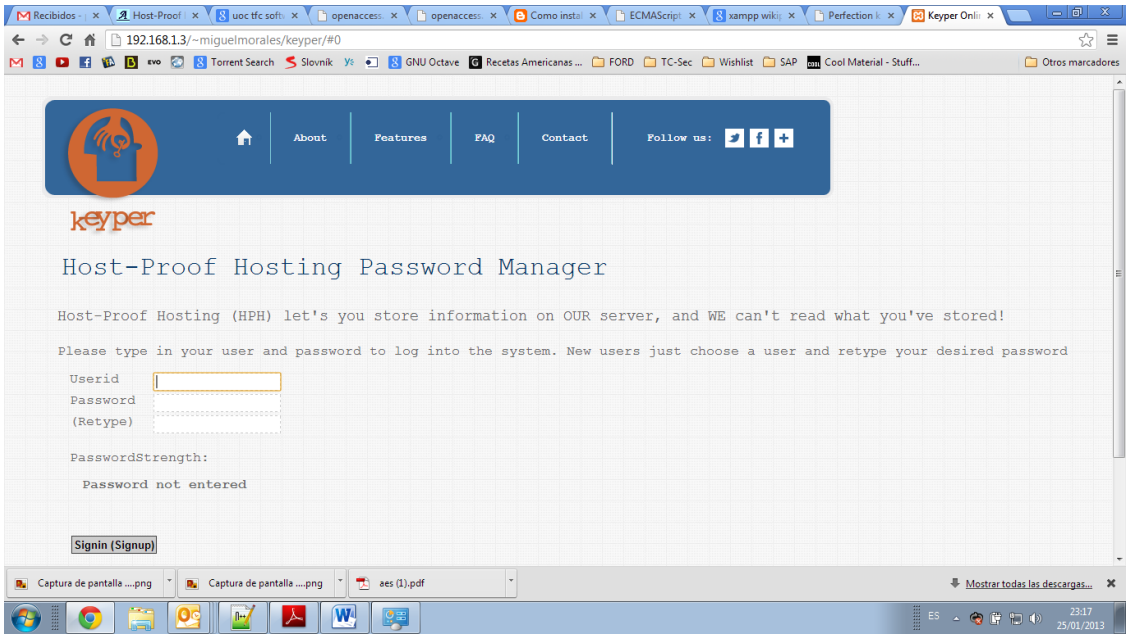


Una vez tengamos los dos servicios corriendo sólo nos quedará configurar la base de datos y subir la aplicación al espacio que hayamos definido en nuestro servidor.
Para configurar la BD nos dirigiremos a localhost/phpmyadmin.

Seleccionaremos la opción importar, seleccionaremos el archivo Keyper_db.sql y el script se encargará de configurar toda la base de datos por nosotros

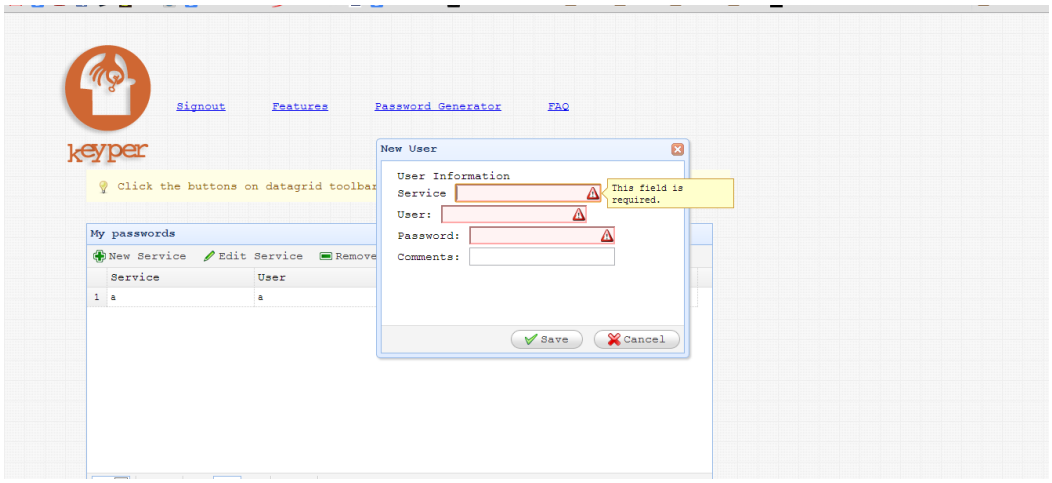
Manual

Login / Creación de un nuevo usuario



Para loguearnos en el sistema únicamente tendremos que introducir nuestras credenciales en el sistema. Para la creación de un nuevo usuario únicamente tendremos que elegir un usuario y una contraseña que tendremos que escribir de nuevo. El sistema nos indicará la fortaleza de la misma a través del mensaje Password Strength. En el caso que el usuario ya esté registrado el sistema nos mostrará un error.

Introducir datos nuevos



Una vez nos encontremos en el area personal únicamente tendremos que apretar el botón new service y un nuevo formulario se nos mostrará. Desde el introduciremos los datos en el sistema.

Editar/ Eliminar datos

Para editar los datos seleccionaremos la fila deseada que se nos marcará en amarillo y presionaremos el botón deseado. El sistema nos solicitará una confirmación de la acción deseada antes de realizar la acción.

TRABAJO FUTURO

A pesar de haber cumplido con los objetivos principales, hay una serie de puntos mejorables o ampliables que se detallan a continuación:

- Compartir contraseñas entre diferentes usuarios.
- One-click login de las aplicaciones de las cuales almacenamos las contraseñas
- Versión offline para uso doméstico
- Bugfixing

CONCLUSIONES

El trabajo final de carrera ha consistido en la creación de una aplicación web que nos permite almacenar información totalmente anónima y segura en un servidor web

La aplicación ha sido bautizada: Keyper.

En este proyecto se emplearon varios lenguajes de programación y técnicas de diseño y codificación lo cual me ha permitido aplicar muchos de los conocimientos adquiridos durante mi carrera universitaria y etapa profesional y al mismo tiempo aprender muchísimo de todo el entorno de la aplicación.

Considero que todos los objetivos se han cumplido, aunque el proyecto haya sufrido alguna desviación temporal

Una meta alcanzada, ha sido también el logro de crear una aplicación de fácil manejo para el usuario, ya que es vital para que un programa de esta magnitud tenga éxito. No obstante, se pueden añadir más funcionalidades a Keyper, siempre orientándose al usuario final, y permaneciendo alineados con los objetivos marcados.