YourPass: Una aplicació mòbil per a la gestió de contrasenyes online

Gerard Noguer Pagès

Resum—Vivim en un món informatitzat on tota la nostre informació es protegida a través de contrasenyes. Les plataformes i serveis que utilitzem comporten certs riscos i existeixen nombrosos casos de filtratges de dades en empreses que tots considerarìem segures a primer vista. A més, la gran majoria, utilitzem contrasenyes simples o les reutilitzem en múltiples serveis, cosa que posa encara més en risc la nostra informació. Però, la realitat es que memoritzar una única, i complexa contrasenya per a cada servei que utilitzem és una tasca impossible d'assolir en la pràctica. Aquest projecte es tracta de plantejar una solució pròpia a aquest problema amb una aplicació mòbil per Android capaç de gestionar les nostres contrasenyes i altre informació confidencial de forma més segura.

Paraules clau—App, Mòbil, Servidor, Gestor, Seguretat, Criptografia, Contrasenya, Notes.

Abstract—We live in a computerized world where all our information is protected through passwords. The platforms and services we use carry certain risks and there are numerous cases of data filtering in companyies that we would all consider secure at first glance. In addition, the vast majority of us use simple passwords or reuse them in multiple services, which puts our information even more at risk. But in reality, memorizing a single, and complex password for each service we use is an impossible task to accomplish in practice. This project is to come up with a solution to this problem with a Mobile apllication for Android capable of managing our passowrds and other confidential information in a more secure way.

Index Terms—Android, App, Phone, Server, Manager, Security, Cryptography, Password, Notes.

1 Introducció - Context del treball

Avui en dia, a internet, cada clic, visita o registre s'arxiva en alguna part. Depenent del punt de vista que adoptis això pot espantar o resultar increïblement interessant. Amb tota aquesta informació sobre nosaltres a diferents llocs es molt important tenir en compte la teva seguretat, i com a usuaris el millor que podem fer és tenir contrasenyes segures i úniques als serveis on ens donem d'alta.

És per això, que sempre s'han donat certes recomanacions als usuaris. Evitar paraules comuns, realitzar combinacions alfanumèriques i longituds mínimes. Però la realitat és molt diferent i empreses de seguretat que publiquen les contrasenyes més populars cada any ens ho demostren, donant "123456" com a la contrasenya més utilitzada per davant d'altres com "qwerty" o "password" com es mostra a l'article [1].

De fet, en la pràctica, complir aquests requisits és complicat, ja que hem de tenir en compte la gran quantitat de serveis amb els que comptem avui en dia i que utilitzem de forma regular. És per això, que la gran majoria de persones acaben utilitzant contrasenyes simples o per altre banda, utilitzen una clau única per accedir a totes les plataformes. Els usuaris no som conscients dels riscos reals que comporta exposar els nostres comptes que contenen informació privada. Tot i que podem creure que les nostres dades no interessen a ningú existeixen nombrosos algoritmes i tècniques per a capturar aquesta informació.

És aquí on entren en joc els gestors de contrasenyes, una eina que ens permet guardar les nostres claus d'una forma fàcil i ens permeten millorar la seguretat de les nostres dades.

2 Objectius

La finalitat d'aquest treball és l'utilització de tots els coneixements obtinguts en el grau per al desenvolupament d'una aplicació mòbil Android destinada principalment a la gestió de contrasenyes de forma segura.

Els usuaris podran accedir a l'aplicació i crear un compte amb una contrasenya mestra a partir de la qual podran obtenir els diferents usuaris i contrasenyes enregistrats prèviament a més de notes, targetes de pagament i informació de comptes bancàries. També podran generar contrasenyes per guardar a l'aplicació i comprovar que aquestes no apareguin en llistes filtrades de contrasenyes vulnerades i, en el cas de trobar una coincidència avisar a l'usuari. Tot això assegurant sempre la correcte seguretat del sistema i de les connexions que es realitzin. No només el funcionament haurà de ser correcte sinó que també es vol aconseguir una interfície amigable i atractiva on la facilitat d'ús per als usuaris sigui primordial.

- *E-mail de contacte: Gerard.noguer@e-campus.uab.cat*
- *Menció realitzada: Enginyeria del Software*
- *Treball tutoritzat per: Jordi Serra Ruiz (Dep. Ciències de la Computació)*
- *Curs 2020/21*

3 ESTAT DEL ART

Dins de les aplicacions utilitzades per a la gestió de contrasenyes es podrien distingir dos tipus.
Els gestors de contrasenyes offline comporten poc risc, la principal causa de una vulneració de dades és a través de malware al dispositiu. Tot i així, sense un gestor de contrasenyes aquest malware podria també capturar les contrasenyes que vas utilitzant.
Un exemple d'aquest tipus de gestor seria l'aplicació KeePass, una opció gratuïta de codi lliure que guarda totes les contrasenyes en una base de dades encriptada local al dispositiu [17].

Per altre banda tenim els gestors de contrasenyes online, aquests tenen l'avantatge de poder accedir a aquestes des de més d'un dispositiu, però també comporten un risc addicional, ja que la base de dades on s'emmagatzemen totes les dades podria ser vulnerada. D'aquest perfil, LastPass és de les apps més conegudes [18].

4 METODOLOGIA

Per al compliment de tots els objectius és necessària una metodologia adequada per a estructurar, planificar y controlar el procés de desenvolupament de l'aplicació.
La duració d'aquest projecte és de 21 setmanes. Es per això que és important idear un pla de projecte en el qual tinguem en compte el temps disponible i recursos. Aquest pla ha de ser el més realista possible per així poder-lo seguir tenint en compte el treball a fer. El projecte es dividirà en 5 fases, utilitzant la metodologia de desenvolupament d'una aplicació mòbil detallada al article [2].

Inicialment es va començar amb una fase de **planificació i anàlisi**, on s'inicia el projecte, es defineix el seu abast i les principals tasques ha realitzar durant el desenvolupament.
La segona fase serà la de **disseny**, on es va tractar de plasmar l'idea de l'aplicació mitjançant diagrames i esquemes de l'interfície, els quals es van anar revisant posteriorment per entendre el funcionament que s'havia plantejat en un inici i per realitzar les modificacions necessàries si es decidien canvis durant l'implementació.
Una vegada completada la fase de disseny, es passa a una **fase d'implementació**, on s'elaboren, afegeixen i proven els elements prèviament contemplats en el disseny de l'aplicació.
Finalment, s'ha realitzat una **fase de test**, amb l'objectiu de verificar el correcte funcionament de l'aplicació en els diferents escenaris i condicions i poder realitzar proves amb usuaris i poder així rebre feedback d'aquests.
La fase final serià el **tancament del projecte**, on es preparen tots els documents a entregar i es presenta el treball realitzat durant el transcurs del desenvolupament de l'aplicació.

5 PLANIFICACIÓ

5.1 Planificació inicial

Amb una duració de 3 setmanes, es tenien com a objectius:
-Concretar l'idea de l'aplicació i els seus objectius
-Definir una planificació per el projecte tenint en compte temps i recursos.
-Definir les eines de treball i la metodologia que utilitzarem

5.2 Disseny

Amb una duració de 5 setmanes es tenien com a objectius:
-Creació de diagrames i documentació que ens ajudin a entendre com serà el funcionament de l'aplicació.
-Recollir i especificar tots els requisits de l'aplicació.
-Creació d'un diagrama de casos d'ús.
-Dissenyar la interfície de l'aplicació.
-Disseny de la base de dades

5.3 Implementació

Amb una duració de 8 setmanes, es tenien com a objectius:
-Creació de la BD.
-Creació de la UI.
-Implementació i test de les funcionalitats de l'aplicació.
-Documentar codi.

5.4 Test

Amb una duració de 5 setmanes, es tenien com a objectius:
-Proves de l'aplicació final.
-Proves amb usuaris reals.

5.5 Tancament del projecte

Amb una duració de 3 setmanes, es tenien com a objectius:
-Acabar documentació.
-Finalitzar el projecte.
-Presentar el treball realitzat en públic.

La suma de setmanes es superior al total ja que en certes fases es treballa en paral·lel.
Tot i així, un projecte comporta riscos i dificultats que poden retardar el seu desenvolupament, per tant, hem d'estar preparats per modificar la planificació si és necessari per al seu correcte compliment. Durant el projecte es va utilitzar un diagrama de Gantt amb la llista de tasques que apareixen a la planificació juntament amb les entregues de seguiment com a milestones. Aquest es va anar modificant a mesura que certes tasques requerien de més o menys temps.

6 REQUISITS DE L'APLICACIÓ

Per al correcte funcionament de l'aplicació s'han definit una sèrie de funcionalitats i requisits que els usuaris han de ser capaços de realitzar.
Els usuaris hauran de poder crear un compte d'usuari utilitzant un email i indicant una contrasenya mestra, informació amb la qual podran fer login posteriorment.

Una vegada dins, aquests hauran de ser capaços de consultar i d'administrar les seves contrasenyes, notes, targetes de pagament i comptes bancàries, es a dir, tant afegirne de noves, editar la seva informació o eliminar les ja existents. Per a poder visualitzar i trobar més ràpidament la informació que es busca, l'aplicació ha de tenir un cercador, que mostrarà el item corresponent segons el seu nom. L'aplicació també permet a l'usuari generar noves contrasenyes segures, amb certes opcions de parametrització que l'usuari pot escollir i indicarà la força de les diferents claus que l'usuari escolleixi. A més, també podrà seleccionar si vol que la contrasenya tingui un temps d'expiració i, una vegada sobrepassat rebre una notificació indicant quina contrasenya ha expirat. L'usuari també ha de tenir l'opció de copiar la contrasenya sense la necessitat de visualitzar aquesta i mostrar-la per pantalla.
En les opcions de l'aplicació, l'usuari podrà activar i desactivar la comprovació de les seves contrasenyes contra contrasenyes que apareguin en bases de dades filtrades, rebent un avís a través d'una notificació en cas de que una d'elles hagi sigut vulnerada, i podrà indicar l'hora del dia en que es realitzi aquesta acció, que només es realitzarà en cas de que el mòbil estigui carregant-se o disposi de més d'un percentatge alt de bateria.
Per a la documentació de tots aquest requisits és va realitzar un document d'especificació amb el següent diagrama de casos d'ús de la *fig 1* i tots aquests requisits detallats de forma clara.

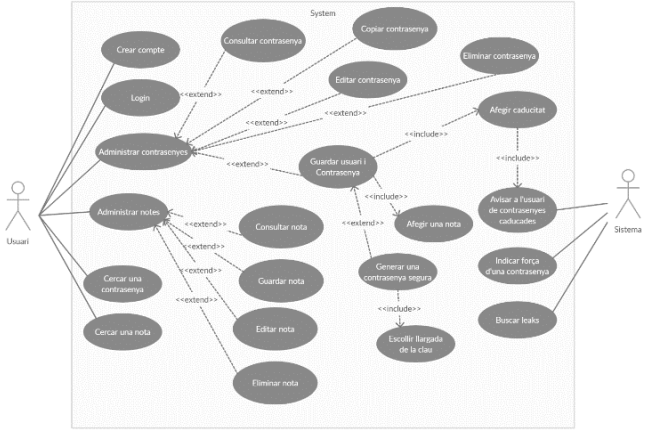

Fig. 1.Diagrama de casos d'ús

7 EINES DE TREBALL I TECNOLOGIES

7.1 Android Studio

Android Studio és l'entorn de desenvolupament integrat oficial per la plataforma Android basat en el software IntelliJ IDEA de JetBrains, disponible de forma gratuïta en una gran varietat de sistemes operatius. A més, disposa d'un dispositiu virtual que és de gran utilitat per a executar i provar aplicacions i eines que ens ajuden a medir el rendiment de la nostra aplicació. Suporta varis llenguatges de programació com Java, C++ i Kotlin, per a aquest app s'ha decidit utilitzar **Java** com a llenguatge.

7.2 Github

Tot i que el treball ha sigut realitzat en solitari, el control de versions en un projecte de software segueix sent vital per al seguiment dels canvis en els fitxer i el manteniment de versions anteriors. Per aquest projecte s'ha utilitzat Github, que utilitza el sistema de control de versions Git per allotjar tot el codi.

7.3 XAMPP

Per a crear un servidor web de forma local i implementar un entorn de desenvolupament per a realitzar les proves de l'aplicació s'ha utilitzat XAMPP [9], una distribució de Apache completament gratuïta i fàcil d'instal·lar que conté MySQL, PHP i Perl. També conté un mòdul útil com **PhpMyAdmin**, la cual ha sigut l'eina utilitzada per a gestionar d'una forma senzilla la base de dades.

7.4 Volley

Per a realitzar les peticions al servidor des de l'aplicació s'utilitza la llibreria Volley, una llibreria HTTP que ens ajuda a fer les peticions en aplicacions Android desenvolupada per Google. Ens ofereix beneficis com programació automàtica de les peticions i facilitat per a popular l'interfície amb dades obtingudes de forma asíncrona.

7.5 JSON

Tot l'intercanvi de dades entre el servidor i l'aplicació es transmet amb format JSON, un format lleuger d'intercanvi de dades. El format JSON és fàcil per els humans de llegir i escriure i fàcil per les maquines de analitzar i generar.

7.6 Postman

Postman es un popular client API que fa més fàcil el crear, testejar i documentar APIs. Aquest permet crear i guardar peticions http i llegir les respostes en diferents formats de forma senzilla.

8 DISSENY

8.1 Arquitectura

Al tractar d'un gestor de contrasenyes online, aquestes seran guardades en una base de dades allotjada en un servidor web, el qual rebrà peticions http.
Els principis bàsics per entendre el funcionament de l'aplicació són els següents:

- El servidor només rep informació ja encriptada.
- La clau per encriptar les dades es genera i guarda en el dispositiu, no s'envia mai al servidor.
- L'encriptació i desencriptació de les dades es realitza en el dispositiu, el servidor només les emmagatzema.

Aquests es veuen representats en el següent diagrama de la *fig 2*.

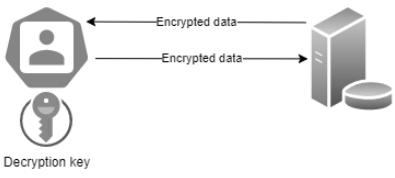

Fig. 2. Diagrama arquitectura client-servidor.

8.2 Disseny de l'interfície

Per al disseny de l'interfície de l'aplicació busquem que aquesta sigui fàcil i ràpida d'utilitzar, fàcil d'aprendre, fàcil de recordar i que sobretot sigui satisfactòria d'utilitzar i que el usuari no tingui dificultats en fer el que desitja.

Aquests primers dissenys de baixa fidelitat es van realitzar en paper i a mà per a poder realitzar canvis de forma ràpida. Una vegada els dissenys eren definitius es va passar directament a la implementació de l'interfície sense funcionalitat, on realitzar canvis seria molt més costós.

8.3 Disseny de la base de dades

Per a poder emmagatzemar totes les dades i l'informació dels usuaris i permetre que aquests puguin accedir a aquestes des de diferents dispositius requerirem d'una base de dades. Aquesta consta de les taules de la següent *fig 3*.

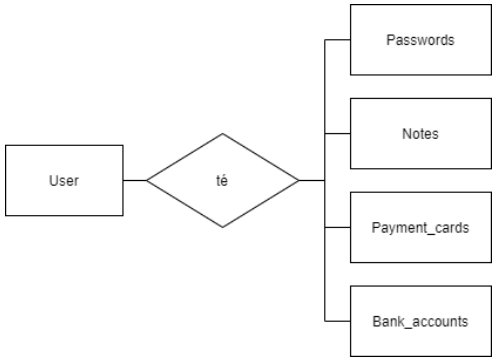

Fig. 3. Diagrama entitat-relació.

Taula users: Conté la id, el email la seva contrasenya mestra encriptada i la data de registre.

Taula services: Conté un codi únic, la id del usuari al que pertany, el nom del servei, el nom d'usuari, la contrasenya, una nota i la data d'expiració.

Taula payment cards: Conté un codi únic, la id del usuari al que pertany, el nom per identificar la targeta, el nom de la persona a la que pertany, el número, el codi de seguretat i la data d'expiració d'aquesta.

Taula notes: Conté un codi únic, la id del usuari al que pertany, el nom de la nota i la nota.

Taula bank accounts: Conté un codi únic, la id del usuari el nom per identificar la compte, el iban i el pin.

8.4 Seguretat

Per a la generació de la clau que s'utilitzarà per encriptar i desencriptar tota l'informació en el dispositiu utilitzarem PBKDF2 (Password-Based Key Derivation Function 2) [15].

En criptografia, una KDF (Key derivation function) és una funció que deriva una o més claus utilitzant una funció pseudoaleatòria.

En el nostre cas, aplicarem aquesta funció a la contrasenya mestra amb una sal i repetirem aquest procés 1000 vegades per a obtenir una clau de 256 bits. Degut a que necessitem generar la mateixa clau a partir de la mateixa contrasenya mestra per a poder generar aquesta en els diferents dispositius que utilitzem, haurem d'utilitzar una sal fixe com poden ser els bytes extrets de l'email de l'usuari. El seu funcionament queda representat a la *fig 4*.

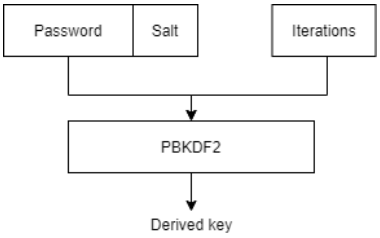

Fig. 4. Diagrama funció PBKDF2.

Per al xifratge de les dades s'utilitza el xifrat AES [10], un esquema de xifrat per blocs que es el estàndard en quant a seguretat de dades i suporta llargades de clau de 128, 192 i de 256 bits com és en el nostre cas. Al ser un xifrat simètric ens permetrà encriptar i desencriptar l'informació a partir de la nostra única clau secreta que haguem generat. A la següent *fig 5* es representa el funcionament.

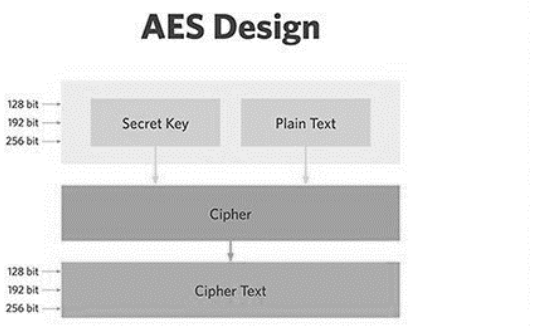

Fig. 5. Diagrama disseny algoritme AES.

9 IMPLEMENTACIÓ

9.1 Web service

La nostre aplicació no pot accedir al nostre servidor directament, és per això que fa falta un intermediari entre el servidor i l'aplicació per a que aquesta pugui accedir a l'informació de la base de dades, per això utilitzem APIs.

Dins la carpeta htdocs del nostre servidor s'ha creat una nova carpeta de projecte que contindrà tots els fitxers php per el nostre web service, **DbConnect.php** amb el codi necessari per realitzar la connexió a la base de dades i per altre banda dins del fitxer **Api.php** controlarem totes les peticions que ens arribin cridant a la funció necessària en cada cas.

Les funcionalitats necessàries per al correcte funcionament de l'aplicació són poder llegir, escriure, editar i eliminar l'informació de les diferents taules on l'usuari guarda la seva informació (CRUD) a més del registre i el login.

9.2 Implementació de l'interfície

Android proporciona un vocabulari XML que ens permet construir els nostres layouts. Gràcies al Layout Editor d'Android Studio podem construir la nostre interfície de forma més ràpida movent i arrossegant els elements a un

editor visual en comptes d'escriure tot el layout XML a mà, tot i que això acaba sent necessari per a certes accions més específiques.
Per a l'implementació d'una interfície responsiva i que es pugui adaptar als diferents tamanys de dispositius s'han utilitzat ConstraintLayouts [16]. Això ens permetrà col·locar i posar restriccions entre les diferents vistes i poder crear l'interfície d'una manera més visual com al exemple de la *fig 6*.

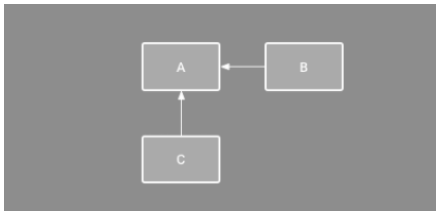

Fig. 6. Vista A està restringida horitzontalment per B i verticalment

9.3 Gestió de peticions al servidor

A la majoria d'activitats de l'aplicació, es realitzen peticions al servidor per a obtenir informació, com poden ser la llista de serveis o notes de l'usuari.
Per a la gestió de totes aquestes peticions s'ha creat una classe **VolleySingleton.java**, on una sola instancia de la cua de peticions de Volley controla totes les peticions.
Per a una millor organització d'aquestes també tenim una classe **URLs.java** que conté les diferents URLs amb les peticions necessàries que volem realitzar al Web Service.

9.4 Encrypted Shared Preferences

Les shared preferences de l'aplicació ens permetrà guardar informació de l'usuari en un fitxer al dispositiu per a poder utilitzar. Un objecte SharedPreferences apunta a un fitxer que conté valors guardats en parelles clau-valor que ens permet llegir i escriure de forma simple.
D'un usuari, guardarem la seva id, per a poder fer les peticions i obtenir les seves dades, el seu email i la seva clau generada per encriptar la diferent informació emmagatzemada a l'aplicació. Òbviament, no volem la nostra clau generada visible en un fitxer del dispositiu. Tot i que les Shared Preferences es guarden en el directori de la app amb permisos que indiquen que només la app en qüestió pot accedir-hi, qualsevol amb nivell root d'accés en el dispositiu serà capaç de veure'ls.
És per això, que utilitzarem la llibreria EncryptedSharedPreferences, que utilitza **Android KeyStore** per encriptar les nostres preferències que guardem.
El Android keystore System ens permet guardar les nostres claus criptogràfiques de manera que siguin més difícils d'extreure del dispositiu, per a més informació del seu funcionament consultar la referència [11].
Per controlar totes aquestes funcions s'ha creat la classe **SharedPrefManager.java.** Aquesta conté les funcions per a escriure, llegir i eliminar les dades que vulguem guardar.

9.5 Crypter

Per a totes les funcionalitats relacionades amb l'encriptació i desencriptació de les dades s'ha fet la classe **Crypter.java**. Aquesta consta de tres funcions principals, **generateKey**, **encrypt** i **decrypt**. La primera funció generateKey, li passem un String i un array de Bytes que actuarà com a sal per a generar la clau.
Les funcions encrypt i decrypt reben com a paràmetre un String i l'encripten o desencripten utilitzant la clau que existeix-hi guardada a les Shared Preferences.
Per a realitzar totes les funcions criptogràfiques s'utilitza el paquet javax.crypto. D'aquest, s'utilitza la classe SecretKeyFactoy per a la generació de la clau i la classe Cipher que proporciona les funcionalitats per a l'encriptació i desencriptació de les dades. Per a més informació del seu funcionament consultar la referència [14].

9.6 Password generator

De la generació de noves contrasenyes segures a l'aplicació s'encarrega la classe **PasswordGenerator.java**. Aquesta conté una funció a la qual se li passen com a paràmetres la mida de la contrasenya i 4 booleans indicant si aquesta ha de contenir majúscules, minúscules, dígits i caràcters especials.
El seu funcionament es basa en generar un String aleatori assegurant un mínim de 2 caràcters per a cada tipus de caràcter que s'hagi passat com a True als paràmetres i generar la resta per a completar la mida indicada de forma aleatòria amb el tipus de caràcters permesos per finalment barrejar el String.

9.7 Alarm receiver

Tant per a enviar les notificacions al dispositiu tant quan una contrasenya ha arribat a la seva data d'expiració o per la comprovació periòdica d'aquestes amb una llista de contrasenyes filtrades s'utilitza la classe AlarmManager d'Android. Aquesta ens permet l'accés als serveis d'alarmes del sistema. El que ens permet això és programar que la nostre aplicació es pugui executar en algun moment futur. Quan una alarma s'executa, el Intent registrat s'emet al sistema, automàticament executant l'aplicació en qüestió si no ho està.
Per a gestionar i rebre aquestes alarmes s'ha fet la classe **AlarmReceiver.java,** que hereta de la classe d'android BroadcastReceiver [12]. En aquesta, es rebrà el Intent en qüestió, que contindrà un codi indicant si l'alarma correspon a l'expiració d'una contrasenya o a una comprovació de contrasenyes filtrades. Es pot veure a la *fig 7*.

```
@Override
public void onReceive(Context context, Intent intent) {

    int reqCode = intent.getExtras().getInt( key: "code");
    if(reqCode == 0){ //Checking for leaks
        int batteryPct = getBatteryPct(context);
        if(batteryPct > 70 || isCharging(context)){
            checkLeaks(context);
        }
    }else if(reqCode == 1){ //Passwords expiration
        int id = intent.getExtras().getInt( key: "notificationId");
        sendExpirationNotification(context, intent, id);
    }
}
```

Fig. 7. Codi al rebre una alarma a la classe AlarmReceiver.java.

En el cas d'una expiració d'una contrasenya es cridà a una funció que crea una notificació indicant quina contrasenya ha expirat, avisant a l'usuari de que l'hauria de can-

viar. En el cas d'una comprovació de contrasenyes filtrades, primer es comprovarà si la bateria es superior a un 70% o si el dispositiu s'està carregant. Si això es compleix, es realitzarà una petició al servidor que retornarà un JSON amb una llista de contrasenyes i es comprovaran amb les que l'usuari tingui guardades. Si es troba alguna coincidència es crea una notificació indicant a l'usuari quina de les seves contrasenyes ha sigut filtrada i indicant que s'hauria de canviar.

9.8 Gestió d'usuaris

De la gestió d'usuaris a l'aplicació s'encarreguen les classes **LoginActivity.java** i **SignupActivity.java**.
Per una banda, la classe SignupActivity es l'encarregada de donar d'alta a nous usuaris.
Com es veu a la *fig 8,* es demanarà a l'usuari que introdueixi un email i una contrasenya mestra de mínim 8 caràcters que contingui tant dígits com minúscules com majúscules. Si tots els inputs són correctes és generarà la clau secreta i la guardarem a les Shared Preferences. Utilitzarem aquesta mateixa clau per encriptar la contrasenya abans d'enviar-la al servidor i registrar l'usuari a la base de dades.

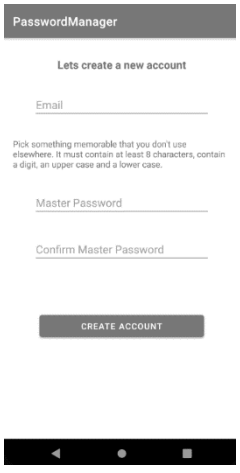

Fig. 8. Pantalla de registre d'un nou usuari a l'aplicació.

Per altre banda, la classe LoginActivity s'encarrega del login de l'usuari de la *fig 9*.

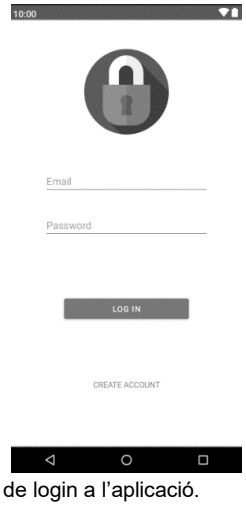

Fig. 9. Pantalla de login a l'aplicació.

Una vegada l'usuari introdueix-hi l'informació i es confirmi que aquest ja ha estat enregistrat, comprovem que a les Shared Preferences tinguem la clau secreta guardada. En cas de que no hi estigui, significarà que estem accedint des de un altre dispositiu i l'haurem de generar i guardar altre vegada.

9.9 Main Vault

Una vegada fet login a l'aplicació, accediríem a el que seria la pantalla principal de la app. Des de aquí podrem accedir a les diferents "Vaults" on tindrem els diferents tipus de dades com es pot veure a la *fig 10*.

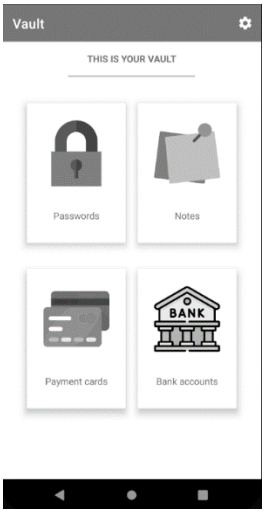

Fig. 10. Pantalla principal de l'aplicació.

9.10 Password Vault

Per a la vault de contrasenyes tenim la classe **PasswordsVaultActivity.java**. En aquesta activitat l'usuari podrà veure totes les contrasenyes que té guardades, gestionar-les i afegir-ne de noves. Ho podem veure a la *fig 11*.

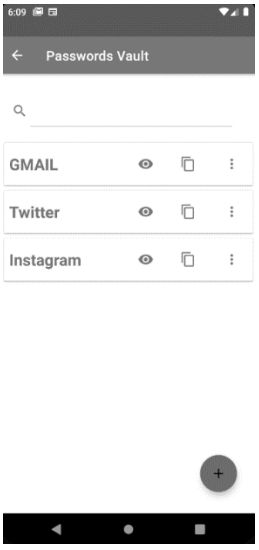

Fig. 11. Vault de contrasenyes, pantalla on es llisten totes les contrasenyes enregistrades.

Una vegada s'inicia l'activitat PasswordsVaultActivity, es realitza una petició al servidor per a obtenir l'informació dels serveis que el usuari té guardats. Una vegada aquestes es reben, es desencripten les contrasenyes i es crea una llista de serveis que passarem a el adapter de la Recycler View per a que es mostrin els items de la llista.
Cada item de la llista representa un servei, sobre els quals podrem realitzar diverses accions.
Primerament tenim un boto amb forma d'ull, el qual ens mostra el Diàleg de la *fig 12*, implementat a la classe **PasswordsDialog.java** amb el nom d'usuari del servei i la contrasenya.

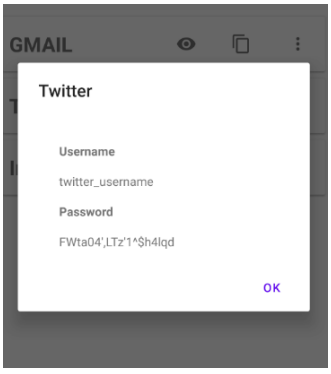

Fig. 12. Diàleg amb l'informació d'una contrasenya.

El segon botó que apareix realitza l'acció de copiar la contrasenya al clipboard del dispositiu, per així poder-la enganxar sense la necessitat d'escriure-la o tan sols mostrar-la en qualsevol moment.
Finalment, hi ha un botó d'opcions el qual desplegarà un menú donant les opcions d'editar i eliminar el servei en qüestió.

9.11 Agregar contrasenya

Prement el botó flotant de la *fig 11* accedirem a la pantalla per agregar una nova contrasenya a la Vault. D'aquesta activitat s'encarrega la classe **AddPasswordActivity.java** de la següent *fig 13*.

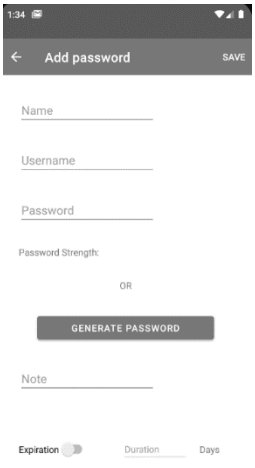

Fig. 13. Pantalla per afegir una nova contrasenya.

És aquí on l'usuari ha d'introduir totes les dades necessàries per a guardar una nova contrasenya. Serà obligatori que s'indiqui un nom, pel qual poder mostrar la contrasenya a la llista, i la contrasenya. Tot i així es dona l'opció a guardar també un nom d'usuari i una nota associada a aquesta. A més, hi ha l'opció de generar una contrasenya i de especificar una expiració per aquesta, que una vegada complerta, es rebrà una notificació indicant-ho. Una vegada guardem, la contrasenya s'encripta amb la clau secreta de l'usuari i es realitza la petició per guardar aquesta a la base de dades. Per a no haver de realitzar una nova petició al tornar a l'activitat anterior de la *fig 10*, l'informació de la contrasenya guardada s'envia a l'activitat anterior i es torna a generar la llista.
En el cas de que el Switch indicant que es vol determinar una expiració per a aquesta contrasenya, estigués activat, es crea una alarma utilitzant la classe d'Android AlamManager que rebrà en un futur la nostre classe AlarmReceiver.java.

9.12 Generar contrasenya

Si al afegir una nova contrasenya premem el boto per a generar-ne una, accedirem a la nova activitat de la *fig 14* **PasswordGeneratorActivity.java**.

Fig. 14. Pantalla de generació de contrasenyes.

En aquesta podrem generar una nova contrasenya a partir de certs paràmetres que l'usuari pot modificar. Per defecte, se'ns mostrarà una contrasenya d'una llargada de 8 caràcters amb totes les opcions activades. Cada vegada que es prem el botó per a generar es mostrarà una nova contrasenya generada per la classe **PasswordGenerator.java,** amb les opcions que hi hagin activades en aquell moment. Una vegada guardem, retornarem a l'activitat anterior *fig 13*, la contrasenya apareixerà omplerta en el camp corresponent.

9.13 Comprovació de contrasenyes

A la pantalla principal de l'aplicació, disposem d'un icona d'ajustos, aquest ens porta a la següent pantalla de la *fig 15* on podrem tant activar com desactivar si volem que aquest tipus de comprovacions es realitzin.

Fig. 15. Pantalla d'ajustos.

Si el Switch es troba activat, es fa visible un DatePicker en el que podem seleccionar una hora i minuts concrets. Una vegada guardem, es crearà una alarma repetitiva amb AlarmManager que es repetirà a l'hora indicada cada dia. Aquest serà el temps en el que es realitzaran les comprovacions.

9.14 Altres Vaults

En quant als altres items que l'aplicació pot guardar, la funcionalitat és molt similar a les contrasenyes. Les activitats **NotesVaultActivity.java**, **PaymentCardsVaultActivity.java** i **BankAccsVaultActivity.java** contenen cada una una RecyclerView igual a la Vault de contrasenyes. Tot i així cada una d'aquestes té el seu propi adapter personalitzat per així mostrar els items a la llista segons les necessitats i en un futur poder mostrar diferents funcionalitats segons el tipus d'item. Per exemple, no podem copiar al clipboard una targeta de pagament, per tant s'elimina el boto del item de la llista com es veu a la *fig 16*.

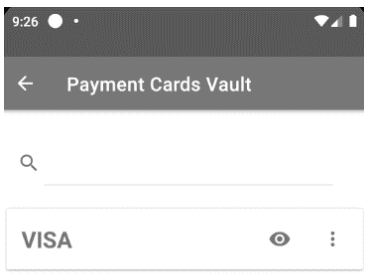

Fig. 16. Item a la llista de targetes de pagament.

Cada Vault té també els seus diàlegs **NotesDialog.java**, **PaymentCardsDialog.java** i **BankAccDialog.java** personalitzat per a mostrar l'informació indicada de forma comprensible en cada cas i les seves pròpies pantalles per afegir un nou item com el de la *fig 17*.

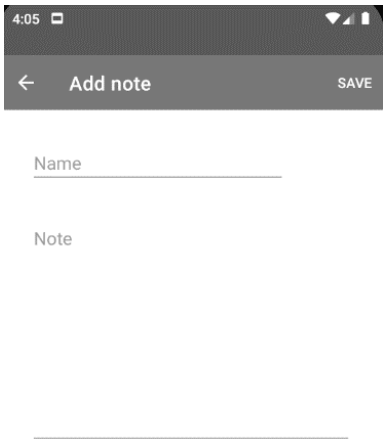

Fig. 17. Pantalla per afegir una nova nota.

9.15 Edició d'items

En cas de que vulguem editar l'informació d'un item que haguem guardat prèviament, s'haurà de seleccionar l'opció al menú del item com es veu a la següent *fig 18.*

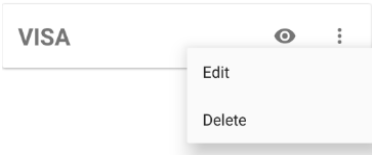

Fig. 18. Menú d'opcions d'un item.

Per a l'edició d'un item s'utilitza la mateixa activitat AddPasswordActivity.java de la *fig 13*. Al iniciar l'activitat, passem un codi, indicant si aquesta activitat s'està iniciant per a afegir una nova contrasenya o per a editar-ne una d'existent. Per aquest segon cas, el Intent contindrà l'informació del item en que s'hagi seleccionat i apareixeran els camps ja omplerts i llestos per modificar. Una vegada premem el botó per guardar, es realitzarà l'acció corresponent d'editar o guardar segons el codi com es veu a la següent *fig 19*.

```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if (item.getItemId() == R.id.save) {
        if(requestCode == ADD){
            try {
                savePassword();
            } catch (GeneralSecurityException | IOException e) {
                e.printStackTrace();
            }
        }
        if(requestCode == EDIT){
            try {
                editPassword();
            } catch (GeneralSecurityException | IOException e) {
                e.printStackTrace();
            }
        }
    }
    return super.onOptionsItemSelected(item);
}
```

Fig. 19. Codi que s'executa al clicar el botó save a la toolbar per guardar o editar un item.

9.16 Eliminar items

Per eliminar un item de la llista haurem de seleccionar l'opció "Delete" del menú mostrat a la *fig 18*. Per a realitzar aquesta acció s'envia una petició al servidor amb el codi únic del item i aquest s'elimina de la base de dades. Una vegada s'obtingui la resposta sense cap error, es notifica al adapter que s'ha eliminat un item de la llista per a que la llista s'actualitzi sense haver de inicialitzar altre vegada l'activitat.
Al eliminar una contrasenya, haurem de tenir en compte si aquesta tenia expiració, i cancel·lar l'alarma que havíem programat per a crear la notificació.

10 PROVES

Tot i anar realitzant proves a les diferents funcionalitats que s'anaven implementant durant tot el desenvolupament del projecte, amb una versió ja avançada de l'aplicació s'han realitzat certes proves per assegurar el correcte funcionament.
Principalment s'han realitzat proves a partir de casos d'ús relacionats amb les contrasenyes, on cada cas d'ús es descomposa en una sèrie de fluxos i es comproven les entrades i sortides com a la següent *fig 20* .

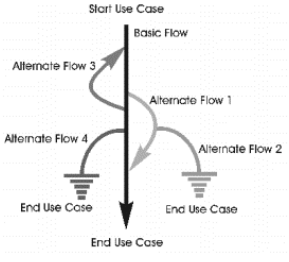

Fig. 20. Respresentació de fluxos alternatius en un cas d'ús.

Per exemple, per el escenari on afegim una contrasenya correctament, seguim els passos d'entrar a la vault de contrasenyes, clicar el botó flotant per afegir-ne una de nova, introduïm les dades i cliquem el botó per guardar i comprovem que aquesta s'hagi guardat correctament. Tenim aquest exemple a la *fig 21*.

Test case ID	TC_Afegir_Contrasenya
Description	Guardem una nova contrasenya a l'aplicació

Sl. NO	Step Description	Expected Results	Actual Results
1	Fem Login a l'aplicació.	Apareix la pàgina principal de l'aplicació.	--
2	Premem a la vault de contrasenyes.	Apareix una llista amb les contrasenyes que tenim guardades.	--
3	Premem el botó per afegir una nova contrasenya.	Apareix la pantalla per afegir una nova contrasenya	--
4	Introduïm les dades per afegir una nova contrasenya i guardem.	Tornem al llistat de contrasenyes i la nova contrasenya s'ha guardat correctament.	--

Test Data Sets			
Data Type	**Data Set 1**	**Data set 2**	**Data Set 3**
Name	Twitter	Twitter	Twitter
Username	Username1		Username1
Password	Password1	Password1	Password1
Note	Note1	Note1	
Test Case Result	ERROR		

Fig. 21. Test del escenari afegir contrasenya. El resultat del test es error ja que no s'ha trobat cap defecte.

Un escenari alternatiu seria per exemple afegir una contrasenya però aquesta vegada amb dades incorrectes, deixant l'espai en blanc i comprovar que apareguin els corresponents missatges d'error o afegir una contrasenya utilitzant el generador de claus.
Degut a la situació actual, les úniques persones amb les que he pogut comptar per a realitzar proves amb usuaris deixant que aquests provin l'aplicació han sigut la meva família més propera. Tot i així gràcies al feedback proporcionat es van realitzar certes correccions i millores a l'aplicació.

11 RESULTATS

El resultat final d'aquest projecte és una aplicació android capaç de gestionar contrasenyes, notes, targetes de pagament i comptes bancaries amb una encriptació a nivell de dispositiu com s'havia plantejat en un inici.
L'aplicació final és capaç de registrar i logejar nous usuaris generant una clau secreta que es guarda en un fitxer encriptada al dispositiu amb la qual es realitza tot el xifratge i no s'envia ni mostra mai al servidor. Aquesta també genera de forma simple i intuïtiva per a l'usuari noves contrasenyes que pot utilitzar amb certesa de la seva seguretat i permet indicar una caducitat per aquestes, rebent un avís amb la notificació de la següent fig *22*.

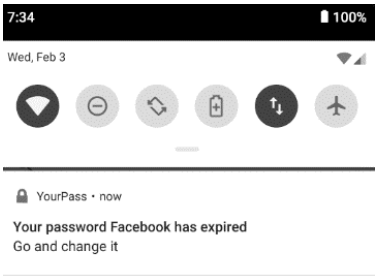

Fig. 22. Notificació d'avís d'una contrasenya expirada.

Per a les connexions i les peticions que es realitzen, tota l'informació confidencial és encriptada abans de ser enviada al servidor, per tant, tot i tenir accés a la base de dades som incapaços de saber quina es l'informació que s'hi guarda sense la clau secreta.
En quant a la comprovació de contrasenyes filtrades, es realitza a partir de paraules aleatòries en un fitxer de text situat en el servidor, però demostra el correcte comportament que s'havia proposat i l'usuari rep una notificació indicant la contrasenya que ha de canviar com es veu a la següent *fig 23*.

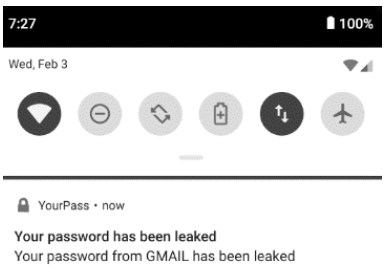

Fig. 23. Notificació d'avís d'una contrasenya que ha sigut filtrada.

En quant a l'interfície de l'aplicació, pot ser millorada per a que quedi un producte final més estètic de cara a l'usuari. Tot i així, la versió actual compleix els objectius inicials, on es buscava principalment la facilitat d'ús i la similitud amb altres tipus d'aplicacions per tenir un fàcil aprenentatge. Ja amb l'aplicació final es va deixar provar l'aplicació a unes poques persones i s'ha demanat que omplin un qüestionari sobre certs aspectes i visuals demanant la seva opinió i idees per millores.

12 Conclusió

S'han complert els objectius inicials que s'havien plantejat durant l'inici del projecte i l'aplicació desenvolupada ofereix les funcionalitats necessàries per a ser utilitzada com a un gestor de contrasenyes.

En quant a la comprovació de contrasenyes filtrades, s'hauria de realitzar a partir de bases de dades de contrasenyes filtrades reals de forma que aquesta funcionalitat pugui tenir una aplicació real. El desenvolupament del projecte s'ha plantejat amb un servidor en local, per lo que es podria planejar una migració a serveis cloud que puguin allotjar el nostre servidor web i així que hi puguin accedir usuaris reals.

Com a línies de continuació del treball tindríem l'accés via web a l'informació, que tot i incrementar la possibilitat dels atacs oferirà més opcions per als usuaris i també l'incorporació de nous items que es pugin guardar a l'aplicació.

Agraïments

A la meva família, amb la que degut a la situació actual compartim més temps que mai, i als meus amics de tota la vida tot i que ja no ens puguem veure tant.

Bibliografia

[1] TeamsID. TeamPassword.com, worst passwords of 2019 Disponible a https://www.teampassword.com/blog/top-50-worst-passwords-of-2019, Desembre 2020.

[2] Universidad del Magdalena. Metodología para el desarrollo de aplicacions móviles, Colombia. Disponible a https://www.researchgate.net/publication/276780521_Metodolo-gia_para_el_desarrollo_de_aplicaciones_moviles/fulltext/55f7a95a08ae07629dca967c/Metodologia-para-el-desarrollo-de-aplicaciones-moviles.pdf, Desembre 2020.

[3] Creately. Disponible a: https://creately.com/diagram-type/use-case, Desembre 2020.

[4] Android Developers. Disponible a https://developer.android.com/topic/security/data, Desembre 2020.

[5] Android Developers. Create a dynamic list with RecyclerView. Disponible a https://developer.android.com/guide/topics/ui/layout/recyclerview, Desembre 2020.

[6] Android Developers. Distribution dashboard. Disponible a https://developer.android.com/about/dashboards, Gener 2021.

[7] Android Developers. Android.app AlarmManager reference. Disponible a https://developer.android.com/reference/android/app/AlarmManager, Gener 2021.

[8] Android Developers. Notification builder class reference disponible a https://developer.android.com/reference/android/app/AlarmManager, Gener 2021.

[9] XAMPP. ¿Que és XAMPP?. Disponible a https://www.apachefriends.org/es/index.html, Desembre 2020.

[10] TechTarget- Adcanced Encryption Standard (AES), Disponible a https://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard, Gener 2021.

[11] Android Developers. Android Keystore System. Disponible a https://developer.android.com/training/articles/keystore?hl=es-419, Gener 2021.

[12] Android Developers. Android BroadcastReceiver. Disponible a https://developer.android.com/guide/components/broadcasts, Gener 2021.

[13] Postman. The Collaboration Platform for API Development. Disponible a https://developer.android.com/guide/components/broadcasts, Gener 2021.

[14] Docs.Oracle. Package javax.crypto summary. Disponible a https://docs.oracle.com/javase/7/docs/api/javax/crypto/package-summary.html, Gener 2021.

[15] Practical Cryptography for developers. PBKDF2. Disponible a https://cryptobook.nakov.com/mac-and-key-derivation/pbkdf2, Gener 2021.

[16] Android Developers. Respomsive UI woth ConstraintLayout. Disponible a https://developer.android.com/training/constraint-layout, Gener 2021.

[17] KeePass. KeePass Password Safe. Disponible a https://keepass.info/, Gener 2021.

[18] LastPass. Disponible a https://www.lastpass.com/es/, Gener 2021.

[19] Icones utilitzats disponibles a https://www.flaticon.com/, Desembre 2020.