Aplicació adaptada de gestió de contrasenyes per a mòbil

Jordi Adrià Campoy Callen

Resum– La protecció de dades s'ha tornat un tema molt important i discutit durant els últims anys degut a que, cada vegada més, s'utilitzen serveis a internet per gestionar aspectes de la nostra vida, com pot ser accés al nostre banc. Però al utilitzar tants serveis dels quals no es poden compartir les credencials sens presenta una disjuntiva, o utilitzar una sola contrasenya complicada o apuntar les credencials a algun lloc. És d'aquest problema que sorgeix la necessitat d'una aplicació que gestioni les credencials d'un usuari, però per descomptat ha de fer aquesta gestió de manera segura. El projecte explicat a continuació consisteix en la creació d'una aplicació per cobrir aquesta necessitat, afegint funcionalitats per millorar l'experiència d'usuari com reproduir àudio si es vol compartir una credencial específica, aquesta funcionalitat està orientada per a persones que no poden mirar actualment el telefón o tenen una discapacitat que s'ho impedeix.

Paraules clau– Aplicació mòbil, Android, Seguretat, Scrum, Android Studio, Java, SQLite, Room, Google Cloud, Cloud Function, TextToSpeech, JUnit, Criptografia.

Abstract– Data protection has become a very important and discussed topic in recent years because, increasingly, internet services are being used to manage aspects of our lives, such as access to our bank. But using that many services whose credentials cannot be shared with others presents a dilemma, using a single complicated password or writing down the credentials somewhere. It is from this problem that the need arises for an application that manages a user's credentials, but of course you have to do this management securely. The project explained below consists of creating an application to meet this need, adding features to improve the user experience having the option to play audio if you want to share a specific credential, this feature is designed for people who can not currently look at the phone or have a disability that prevents him to do so.

Keywords– Mobile Application, Android, Security, Scrum, Android Studio, Java, SQLite, Room, Google Cloud, Cloud Function, TextToSpeech, JUnit, Cryptography.

1 Introducció - Context del treball

En l'era en què ens trobem, identificar-nos a tota mena de pàgines web o aplicacions s'ha tornat una necessitat per a la majoria de persones. Per realitzar aquesta identificació, s'utilitzen contrasenyes que han de tenir un mínim de complicació per evitar que certs algoritmes les aconsegueixin, com podria ser l'algoritme de força bruta, que intenta provar contrasenyes per veure si encerta i aconsegueix accés. A més, no es recomana utilitzar una sola contrasenya complicada per totes les identificacions, per què això presenta un altre problema, aquest és que si una de les webs o aplicacions té una pèrdua de dades, la contrasenya pot acabar sent compartida a internet i llavors qualsevol pot autenticar-se per accedir a informació privilegiada amb la contrasenya publicada.

Per aquests dos problemes es pot concloure que no és bona idea utilitzar sempre la mateixa contrasenya ni simplificar-la, per tant, si es vol fer servir una contrasenya diferent per a cada identificació i aquesta no pot ser simple, és pràcticament impossible que una persona pugui recordar totes les seves contrasenyes.

En aquest document es descriuen les diferents fases d'un projecte amb l'objectiu d'alleujar aquest problema, dissenyant i proporcionant una app que guarda de manera segura les dades de les contrasenyes d'un usuari.

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

2 OBJECTIUS

Es vol crear una aplicació per a mòbil, que pugui guardar contrasenyes d'usuaris a una base de dades local. Per així no haver de dependre en una API externa per guardar les dades. L'objectiu en la realització d'aquesta app és arribar a desenvolupar una aplicació funcional passant per les fases de:

- Planificació
- Disseny
- Desenvolupament
- Test

Aplicant així coneixements i metodologies apresos durant la carrera. Per tant, s'hauran de planificar les tasques a realitzar durant el projecte, fer el disseny de les diferents parts de l'aplicació, per a que pugui guardar les contrasenyes xifrades de forma segura sota l'accés d'una contrasenya *master*. Caldrà també desenvolupar la interfície d'usuari per utilitzar l'aplicació i accedir a les dades o reproduir àudio apretant un botó que reprodueix les dades indicades consultant a una *Cloud Function* situada a *Google Cloud*. I finalment és necessari fer testing per assegurar la qualitat del resultat.

La intenció final és arribar a poder controlar l'accés a la pròpia aplicació, emmagatzemar i controlar les contrasenyes de l'usuari de manera segura i utilitzar serveis web per afegir funcionalitats a l'aplicació.

3 ESTAT DE L'ART

Per a suplir el problema esmenat anteriorment existeixen programes de gestió de contrasenyes, aquests es poden distingir de diferents maneres, segons on emmagatzemen les dades o segons que tipus d'accés utilitzen.

Aquests programes poden emmagatzemar les contrasenyes o en local o online a un *cloud*, n'hi ha que realitzen les dues coses com *1Password*,[1] que utilitza emmagatzematge local però fa còpies de seguretat i es sincronitza si estàs connectat al teu compte.

L'altra manera de distingir aquests programes és segons el tipus d'identificador que utilitzen, poden fer servir diferents *tokens* que poden ser hardware que t'identifica, un pin o contrasenya, o alguna cosa que et defineix, com l'empremta dactilar o la retina. Un exemple d'autenticació per hardware és la *YubiKey* de *Yubico*.[2]

Actualment no hi ha cap gestor de contrasenyes que reprodueixi o capturi àudio amb l'objectiu del tractament de dades.

4 METODOLOGIA

Per dur a terme el projecte s'ha decidit utilitzar una metodologia àgil degut a la seva alta adaptabilitat pel seu disseny iteratiu.[3] La metodologia escollida és *SCRUM* i s'han realitzat *sprints* de dues setmanes, el que significa que cada dues setmanes es farà una revisió dels objectius del *sprint* i dels canvis necessaris, si escau. Els *sprints* seran separats en diferents *releases* que indicaran diferents parts del projecte, com poden ser planificació, desenvolupament d'aplicació, implementació d'API d'àudio a partir de text i testing.[4]

S'utilitza **Trello** per organitzar les tasques de cada *sprint* fent un backlog on van totes les tasques a realitzar i les tasques del *sprint* actual van a un "TODO".[10]

Pel que fa al control de versions, s'utilitza **Github** per portar un control dels canvis realitzats durant el TFG i així tenir un historial amb tot el procés de desenvolupament i canvis.[11]

5 PLANIFICACIÓ

La planificació inicial del projecte ha constat de captar els requeriments i definir les tasques a fer per realitzar una aplicació que compleixi aquests requeriments.

5.1 Requeriments

Per a realitzar la presa de requeriments s'ha fet el següent diagrama:

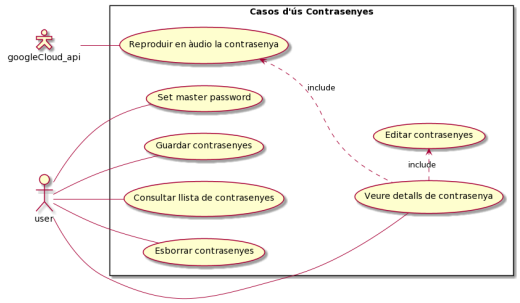

Fig. 1: Diagrama de casos d'ús fet amb **plantUML**[5]

Com es pot observar al diagrama de la *fig(1)*, l'aplicació creada al projecte interactua amb l'usuari i amb una *cloud function* allotjada a *Google* per passar la contrasenya a àudio. A més d'això, l'usuari pot controlar totalment les seves contrasenyes i canviar la seva autenticació a l'app.

5.2 Tasques

Per organitzar les tasques necessàries per a la realització d'aquest TFG s'ha utilitzat l'eina **Microsoft Project**, que organitza el temps i les dependències de les tasques. S'ha decidit deixar un temps després de les *releases* per si apareixia algun bloquejant.

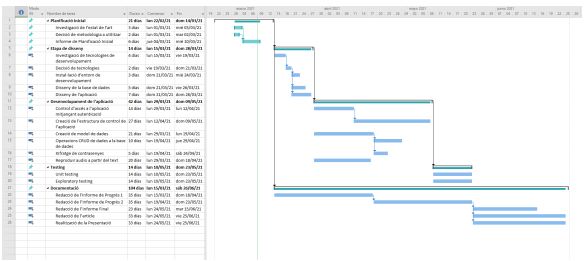

Fig. 2: Diagrama de *Gantt* fet amb **Microsoft Project**

Com es pot observar al diagrama de *Gantt* generat per l'eina es repartirà el temps en una etapa de disseny, una de desenvolupament, que inclou 3 *sprints*, i una de testing. Mentre que la documentació es realitzarà en paral·lel.

5.2.1 Planificació

La primera *release* ha consistit a fer un estudi de l'estat de l'art i planificar la metodologia a utilitzar en el projecte.

- **Investigació de l'estat de l'art**
  Un petit estudi de les aplicacions existents que tenen la mateixa intenció o un ús semblant.
- **Decisió de metodologia a utilitzar**
  Investigació de les possibles metodologies de treball i escollir la que millor s'adapti al projecte.
- **Informe de Planificació Inicial**
  Realització d'un informe de planificació inicial.

5.2.2 Etapa de disseny

Aquesta etapa dura un *sprint* i consisteix en la investigació i la presa de decisions sobre quines tecnologies utilitzar.

- **Investigació de diferents tecnologies de desenvolupament**
  Estudi de les diferents tecnologies de desenvolupament d'app per a mòbil, les tecnologies més disputades han sigut *Flutter*, *React Native* i *Android Studio*.
- **Decisió de tecnologies**
  Decisió de quina tecnologia a utilitzar basada en l'estudi realitzat anteriorment.
- **Instal·lació d'entorn de desenvolupament**
  Fer el SetUp de l'entorn de desenvolupament i comprovar el correcte funcionament tant amb un telèfon mòbil com simulant.
- **Disseny de la base de dades**
  Esquema relacional amb les dades necessàries pel correcte funcionament de l'aplicació, més informació sobre aquest a la secció 6.
- **Disseny de l'aplicació**
  Realització de presa de requeriments.

5.2.3 Desenvolupament de l'aplicació

Aquesta és l'etapa més llarga durant tres *sprints*. En aquest temps es pretén realitzar tota la implementació de l'aplicació seguint les dependències indicades al diagrama de *Gantt*, *fig(2)*.

- **Control d'accés a l'aplicació mitjançant autenticació**
  Crear control d'autenticació per a l'aplicació.
- **Creació de l'estructura de control de l'aplicació**
  Crear totes les pantalles necessàries per al correcte funcionament de l'aplicació.
- **Creació de model de dades**
  Crear el model de dades per emmagatzemar les dades necessàries.
- **Operacions *CRUD* de dades a la base de dades**
  Operacions de crear, llegir, actualitzar i esborrar les dades.
- **Xifratge de contrasenyes**
  Algoritme per xifrar i desxifrar les contrasenyes abans d'emmagatzemar-les a la base de dades.
- **Reproduir àudio a partir del text**
  Connexió amb l'API de *Google*.

5.2.4 Testing

L'etapa de testing existeix amb el propòsit d'assegurar el correcte funcionament de l'aplicació abans que arribi a mans de l'usuari final. Aquesta és l'última *release*. L'explicació dels tests i els seus resultats es troba a la secció 10 del document.

- **Unit testing**
  Tests de caixa blanca.
- **Exploratory testing**
  Tests del funcionament de l'aplicació.

5.3 Canvis durant el projecte

Durant la realització del projecte s'han afegit tasques per millorar la seva qualitat i han aparegut diferents problemes a la planificació que s'han necessitat abordar. En quant a les tasques afegides són la localització de l'app afegint control d'idiomes i la millora de l'*UI*. Pel que fa als problemes han sigut: dificultat a l'hora de fer peticions *http* al *Cloud* de *Google* i afegir tasques que semblaven de durada curta, però han presentat problemes a l'hora de fer la implementació, com pot ser la tasca de localitzar l'aplicació. La combinació d'aquests factors ha provocat que l'etapa consistent en fer la implementació durés més de l'esperat.

6 Disseny de la base de dades

La base de dades que utilitzen els sistemes *Android* és *SQLite*.[9] Aquesta és una base de dades relacional que s'ha utilitzat per emmagatzemar les dades de la manera que mostra l'esquema relacional de la següent *fig(3)*:

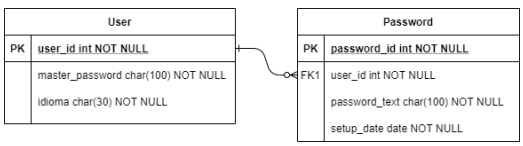

Fig. 3: Esquema relacional fet a **diagrams.net**[6]

La taula **User** conté la contrasenya *master* i l'idioma en què estarà la base de dades per a ell, mentre que la taula **Password** conté un conjunt de caràcters que correspondran a la contrasenya xifrada i la data en què es va guardar o modificar la mateixa.

Al disseny s'havia pres la decisió de guardar la clau màster utilitzada per xifrar a la pròpia base de dades, però al veure que estava utilitzant la mateixa base de dades per emmagatzemar les contrasenyes xifrades s'ha vist que si s'aconsegueix accés a aquesta base de dades s'obtenia les contrasenyes xifrades i la pròpia clau necessària per desxifrarles, fent que el procés de xifrar aquestes contrasenyes no tingui cap sentit. Per tant la clau màster ara és a la mateixa aplicació només accessible per qui fa el procés de xifrar.

També s'han afegit dos camps a la taula de contrasenya per identificar aquesta contrasenya per text i perquè s'ha determinat la necessitat de guardar també un usuari, per tal que a l'hora de mostrar una llista de les pròpies contrasenyes aquestes tinguin un identificador que un usuari pugui reconèixer fàcilment i l'usuari tingui també l'oportunitat de guardar diferents mails utilitzats per a diferents registres.

Nou diagrama sql:

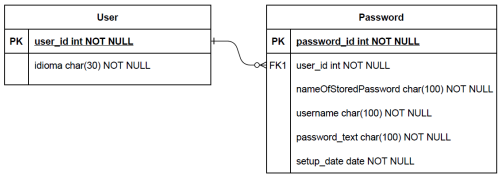

Fig. 4: Esquema relacional corregit, fet a **diagrams.net**

7 Eines

Algunes eines serveixen per ajudar de diferents maneres durant tot el procés de desenvolupament d'un projecte de software, a continuació s'expliquen les utilitzades per a dur a terme aquest TFG.

7.1 Trello

Trello és el software de gestió de projectes utilitzat. Totes les tasques han sigut dividides en subtasques més petites i col·locades a una columna anomenada *Backlog* on seran les tasques no començades, a part d'aquesta s'han creat 4 columnes més.

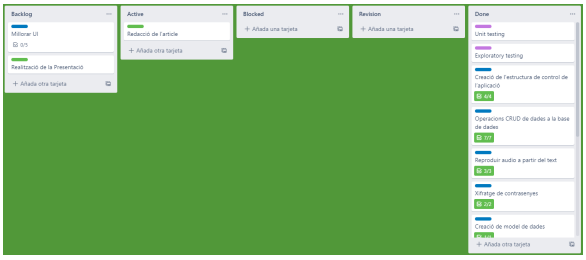

Fig. 5: Eina **Trello**

Com es pot veure a la *fig(5)* la següent columna és *Active*, on es col·loquen les tasques en les quals s'està treballant en el moment. A la columna *Blocked* es col·loquen les tasques que per algun motiu es troben bloquejades i no es poden continuar. A *Revision* hi ha les tasques acabades esperant per ser revisades i passen a *Done* quan han sigut revisades i acceptades.

7.2 Microsoft Projects

Microsoft Project és un software que permet organitzar projectes de manera simple. Aquest software ofereix moltes opcions d'organització i seguiment.

S'ha utilitzat l'eina per establir un *roadmap* que ajudi a tenir una base durant el desenvolupament, que indica com avança el projecte.[7]

7.3 PlantUML i diagrams.net

Els diagrames són una representació visual per proporcionar informació de manera senzilla, a aquest projecte s'han utilitzat dues eines per generar els diagrames del disseny de l'arquitectura. **PlantUML** per al diagrama de casos d'ús on es representa la recollida de requeriments i la seva relació amb els actors que interactuen amb l'aplicació, i la web **diagrams.net** per als esquemes relacionals que defineixen l'estructura de la base de dades i per al diagrama del flux de l'aplicació.

7.4 Android Studio

L'*IDE* **Android Studio** és un entorn de desenvolupament dedicat exclusivament al desenvolupament per a sistemes *Android*, a aquest *IDE* es pot desenvolupar utilitzant els idiomes *Kotlin* o *Java*.[8]

En el cas d'aquest projecte s'ha decidit utilitzar *Java* per coneixements previs del mateix. *Android Studio* ofereix a més molt bona integració amb el Software Development Kit (*SDK*) de *Android*.

7.5 Github

Github és on s'allotja el codi del projecte. Com s'ha comentat abans, el control de versions és amb *git* i utilitzar-lo ajuda a tenir un control sobre les versions de l'aplicació, això pot ser molt útil si en algun moment comença a fallar l'aplicació, perquè aquest motor permet revertir canvis fins a una versió que si funciones, a més que al allotjar el codi no es té la preocupació de perdre les dades que es tenen en local.

S'han creat diferents branques, la *master* s'ha deixat com a producció, és a dir, només es puja codi si ja està verificat que funciona perfectament. Les tasques es treballen a partir de branques que surten de master per més tard juntar els canvis.

7.6 Postman

L'eina **Postman** és una aplicació que permet fer i personalitzar peticions sense la necessitat de desenvolupar un client. S'ha utilitzat aquesta eina per comprovar el correcte funcionament de la *cloud function* i, més tard, per aprendre a realitzar la petició en *Java* amb l'entorn d'*Android*.[12]

8 Desenvolupament

Aquest apartat descriu com ha estat organitzada la implementació del projecte passant per tecnologies i el flux que segueix.

8.1 Flux

A **Android Studio** les vistes principals s'anomenen *Activities*, aquestes vistes contenen un flux propi per controlar les accions que fa quan es crea, destrueix o pausa. S'utilitzen *Activities* per definir com són les vistes utilitzades al projecte i quines accions realitzaran els objectes dins de la vista.

La *fig(6)* mostra un diagrama que indica com estan interconnectades entre si les classes de l'aplicació.

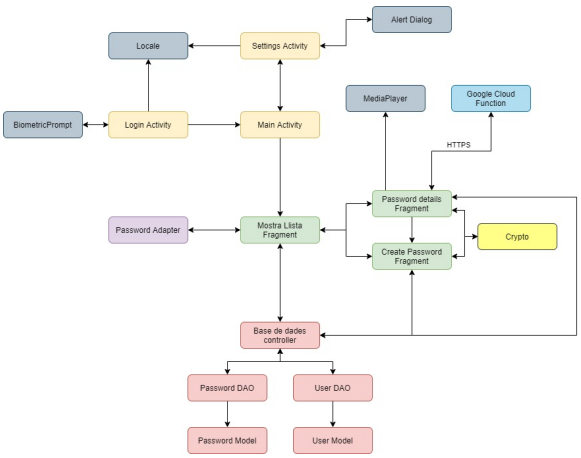

Fig. 6: Diagrama del fluxe de l'aplicació, fet a **diagrams.net**

Per començar, les classes en **gris** són objectes ja creats que donen un servei específic, durant el projecte s'han utilitzat més classes de *Java* d'aquesta manera, però les quatre indicades en el diagrama són les que donen una funcionalitat més clara en quant a *Android*. Aquestes quatre classes són:

- **BiometricPrompt**
  Proporciona accés al reconeixedor biomètric del telèfon mòbil.
- **MediaPlayer**
  Permet reproduir àudio en el mòbil.
- **AlertDialog**
  Obre una vista a sobre de la vista actual de l'aplicació, normalment per mostrar un formulari.
- **Locale**
  Permet definir l'idioma que utilitzarà l'aplicació.

Les *Activities* estan marcades en un **groc ataronjat**, són les classes que controlen el flux principal de l'aplicació. La primera és *LoginActivity* que s'encarrega del control d'accés. Després hi ha la *MainActivity*, que conté la navegació d'unes vistes anomenades *Fragments* a dins, aquests *Fragments* són els que porten el control de les contrasenyes. I per acabar hi ha la *SettingsActivity*, que és una vista feta per incloure totes les configuracions que tinguin a veure amb el control de l'aplicació.[13]

La classe marcada en **lila** és un *Adapter*, les classes tipus *Adapter* s'utilitzen per proporcionar accés a ítems. En el context del projecte s'ha creat un *adapter* per mostrar la llista de contrasenyes, l'*Adapter* s'encarrega de traduir les dades de cada una de les contrasenyes emmagatzemades a una llista a un ítem dins una vista per tal de poder fer scroll sobre tots els ítems i poder diferenciar cadascun d'ells o afegir-li's propietats.

8.2 Fragments

Els *Fragments* són classes marcades en **verd** al diagrama de la *fig(6)*. Aquests es situen a l'interior de les *Activities* i, per tant, no poden existir per si mateixos. Estenen de *FragmentActivity* i s'utilitzen per modular les aplicacions. Un *Fragment* creat es pot utilitzar de nou en altres *Activities* i es pot utilitzar més d'un a la mateixa *Activity*.

Al projecte s'han fet servir els *Fragments* per crear una navegació independent de l'*Activity*, fer això permet més control sobre les vistes de dins de l'*Activity* i, per tant, fer coses com animacions al fer un canvi de fragment mitjançant aquesta navegació.

8.2.1 Threads

Durant el procés de realització d'aquest treball s'han hagut d'utilitzar crides asíncrones, aquestes crides són peticions que es fan a un servei i és necessari esperar la seva resposta. Les funcions asíncrones han aparegut a l'hora de fer crides a la *Cloud Function* de *Google* que s'utilitza i al fer peticions a la base de dades local del telèfon. A l'haver d'esperar la seva resposta no es pot fer servir el flux principal de l'aplicació perquè això provocaria que es bloqueges mentre espera la resposta o fins i tot que es tanqués l'aplicació retornant un error. Per aquest motiu s'han utilitzat *threads*, al trobar problemes amb un tipus de *thread* s'han hagut de fer servir dos tipus diferents.

El primer tipus de *thread* utilitzat és la classe *Thread* normal de *Java*, aquest separa l'execució creant un fil independent del fil principal i executant o esperant la resposta allà.

Ens apareix un problema que és que al rebre la resposta de la petició que s'estava esperant probablement es vol realitzar un canvi o actualització a la vista, per tal que l'usuari pugui observar el resultat. Però en el cas que l'usuari hagi canviat ja de vista, el *thread* estarà intentant modificar una vista que ja no existeix, provocant un error i el tancament de l'aplicació. Per això mateix s'ha utilitzat *view.post()*, que crea un *thread* que es cancel·la si la vista és canviada per un altre en algun moment.

8.3 Criptografia

La classe que s'encarrega de la criptografia està marcada en **groc** a la *fig(6)* i és accedida pels *Fragments* que interactuen amb la dada de les contrasenyes. L'algoritme utilitzat per realitzar el xifratge de les contrasenyes ha sigut "AES"que és un algoritme de xifratge per blocs. S'ha escollit aquest algoritme per la facilitat que proporciona per xifrar i desxifrar, ja que utilitza una sola clau per fer aquest treball, es pot veure il·lustrat a la *fig(7)* que es pot encriptar i desencriptar amb una mateixa clau i això és exactament el que ens interessa a aquest projecte.[14]

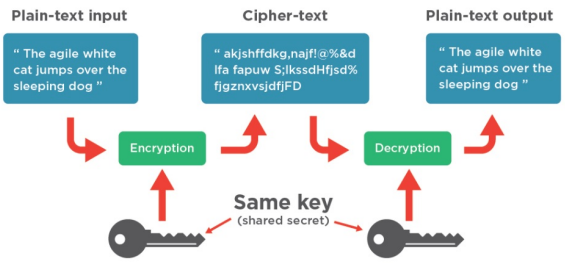

Fig. 7: Funcionament *AES*

8.3.1 Algoritme de xifratge

Les inicials *AES* signifiquen *Advanced Encryption Standard*, és un algoritme de xifratge que fa diferents operacions basades en una clau màster proporcionada, entre aquestes operacions es troben *shift* de bytes, mix de columnes i combinacions de bytes. Per tant, el resultat de les operacions retorna una combinació de bits completament diferent de les dades originals i només és possible tornar a trobar les dades originals fent les mateixes operacions però a l'inrevés amb la mateixa clau màster. En el projecte s'ha fet servir una mida de bloc de 128 bits per a l'algoritme, que és l'estàndard de *Java*.

8.4 Base de dades

Les classes de la base de dades estan destacades en **vermell** al diagrama de la *fig(6)*. La base de dades de l'aplicació és accedida pels tres *Fragments* que necessiten alguna dada de les contrasenyes, aquests *Fragments* fan alguna de les operacions *CRUD*, que vol dir crear, llegir, actualitzar o esborrar. El *Fragment* de **Mostrar Llista** només llegeix per mostrar les dades, el de **Create Password** fa les operacions de crear i actualitzar, i el de **Password Details** llegeix dades i esborra.

8.4.1 Room

Per a crear la base de dades en el sistema *Android* s'ha fet servir la llibreria de persistència *Room*.[15] Aquesta llibreria crea una capa d'abstracció per a utilitzar la base de dades *SQLite*. La capa funciona creant una interfície per accedir a les dades d'una manera simple.

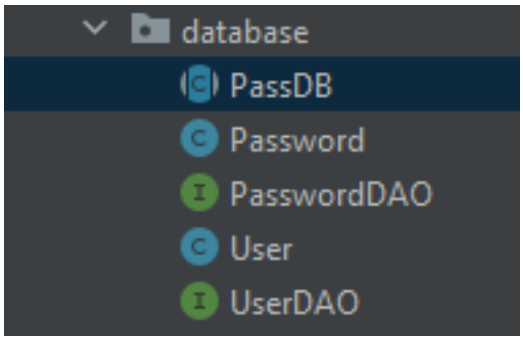

Fig. 8: Classes *Room*

Com es pot observar a la *fig(8)* s'han creat una classe per cada taula de la base de dades on aquesta classe té el seu nom i el que fa cada una d'aquestes classes és fer de model per crear la taula a la pròpia base de dades, aquest model s'utilitza després als *DAO*, que significa *Data Access Object*, per definir les funcions *CRUD* previament esmentades i finalment la classe **PassDB** que fa de controlador de la base de dades, és a través d'aquesta classe que es farà servir per accedir a la base de dades des d'altres parts del codi.

8.5 Google Cloud

Google ofereix molts serveis *cloud* com bases de dades, màquines virtuals i una gran varietat d'*APIs*, des d'Intel·ligència Artificial fins a mapejat amb *Google Maps*. En el cas d'aquest projecte s'ha aprofitat que *Google* proporciona una prova gratuïta de 90 dies o 300$ per crear una *Cloud Function*.

Les *Cloud Function* són funcions que permeten rebre peticions *http* i responen segons les necessitats que hagi codificat l'administrador de la mateixa. Aquestes *Cloud Function* tenen fàcil accés a les *APIs* de *Google* abans esmentades.

8.5.1 Cloud Function TextToSpeech

La classe creada a aquesta *Cloud Function* està marcada en **blau** al diagrama de la *fig(6)*, encara que no forma part d'*Android* si que hi ha una comunicació clara amb el sistema.

A la *Cloud Function* primerament s'han controlat les peticions, assegurant que es rep *content-type* i retornant un error *(415 Unsupported Media Type)* en cas que la petició no el contingui. Si la petició té *content-type* però no com es desitja, respon amb error *(400 Bad Request)*, és a dir, respon amb error *http* 400 o 415 en cas de no acceptar les dades. La funció accepta *content-type* json o "x-www-form-urlencoded".

Una vegada controlat l'accés i les dades agafa la string rebuda i la sintetitza amb un còdec MP3, més tard s'agafa l'àudio resultant, es passa a bytes i després es codifica amb *Base64* per enviar-ho com a resposta de la petició.[16]

8.5.2 Crides des de l'aplicació

Per a fer les peticions *https* des de l'aplicació s'ha utilitzat la llibreria *OkHttpClient* recomanada per **Postman**, aquesta tasca de realitzar la petició va endarrerir una mica el desenvolupament per una incompatibilitat entre *Android* i la llibreria *MediaType*, que també estava recomanada per **Postman** i serveix per encapsular les dades json al *content-type* de la petició, es pot veure un exemple de la petició a la fig(9).

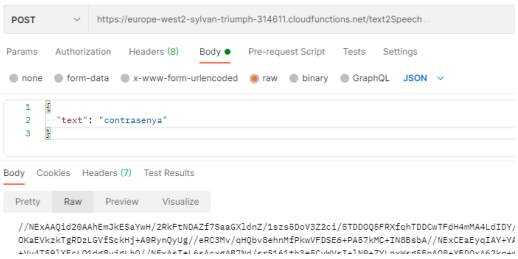

Fig. 9: Exemple de petició a la *Cloud Function*

La petició realitzada és *https*, això significa que utilitza ssl/tls a la capa de transport de les peticions, que és un mètode de xifratge. Això és un factor molt important a tenir en compte, ja que estem enviant una contrasenya.[17]

8.6 Gestió d'idiomes

L'aplicació suporta 3 idiomes: Angles, Català i Espanyol. Aquesta part dels idiomes es càrrega a l'iniciar l'aplicació i es carrega també al seleccionar un idioma. A l'inici s'ha intentat crear la pantalla de *Settings*, que és on es controlen els idiomes, com a *Fragment*, però com el seu accés es controla des de fora de la navegació d'aquests *Fragments* a resultat més complicat del previst.

9 User Interface

Una petita descripció de les pantalles que pot veure un usuari en fer ús de l'aplicació i el seu funcionament.

9.1 Login

A l'entrar a l'aplicació es carrega *LoginActivity* i es veu una pantalla on es demanda iniciar sessió de manera biomètrica.

Fig. 10: App Login

9.2 Control Page

Una vegada obtingut l'accés a l'app existeix una pantalla amb diferents *Fragments*, aquests *Fragments* tenen tots accés a *Settings* utilitzant els tres punts de a dalt a la dreta, que obren un menú d'opcions on apareix l'opció configuració.

9.2.1 Contrasenyes

A l'interior es pot veure una llista de les contrasenyes amb els noms d'identificació assignats per l'usuari i la seva data de creació o modificació.

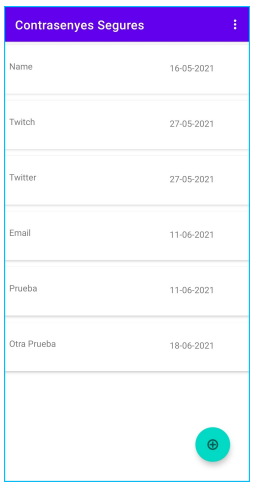

Fig. 11: App pàgina Main

El botó flotant ens porta al *Fragment* de creació de contrasenya i cada ítem de la llista obre els seus detalls al *Fragment* de detalls.

9.2.2 Detalls

A la pantalla on es mostra el *Fragment* de detalls hi han totes les dades relacionades amb cada contrasenya, es troba un botó de reproducció que, després d'informar a l'usuari de que la contrasenya es reproduirà en àudio i assegurar-se que realment és segur, fa la crida a la *Cloud Function* explicada anteriorment. A més d'un botó per copiar la contrasenya i emportar-se'la de manera simple per utilitzar-la on pertoqui.

Hi ha dos botons per edició i esborrar, el primer carrega la pantalla de Creació però per editar i la segona esborra la contrasenya carregada. Per últim, hi ha un botó flotant que retorna a la pàgina amb la llista de contrasenyes.

Fig. 12: App pàgina de detalls

9.2.3 Creació/Edició

A l'interior es pot veure el formulari de creació de la contrasenya amb un botó per guardar i un botó flotant que retorna a la pàgina amb la llista, els paràmetres de la Id i la contrasenya no poden ser buits, i en cas de la contrasenya ha de tenir una llargària mínima.

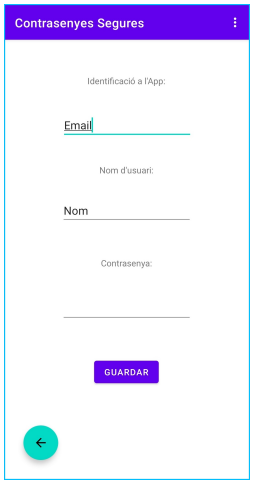

Fig. 13: App pàgina creació/edició

En cas que la navegació vingui de la pantalla de detalls de contrasenya s'omplen els camps directament i en comptes de crear un nou camp a la base de dades modifica el seleccionat.

9.3 Settings

La pantalla de *settings* és on es troben les opcions de configuració, a l'escollir un idioma s'actualitza reiniciant tots els texts. Conté un botó flotant que retorna a la pàgina de control amb la llista de contrasenyes oberta.

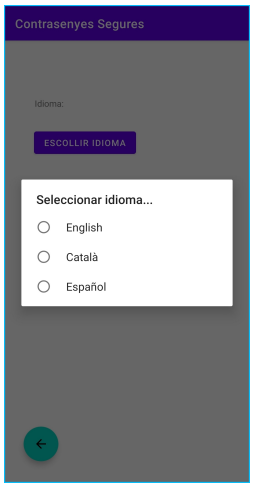

Fig. 14: App pàgina *settings*

10 TESTS I RESULTATS

S'han realitzat tres tipus diferents de tests per assegurar el correcte funcionament de l'aplicació i aconseguir minimitzar al màxim les probabilitats de trobar defectes.

10.1 Tests

Els tests s'han separat en *Unit Testing* per a tests de caixa blanca al propi codi, *Exploratory Testing* per a fer proves de la interfície d'usuari i *User Tests* on s'ha demanat a una persona sense coneixements previs que faci servir l'aplicació.

10.1.1 Unit Testing

Per a fer *Unit Testing* s'ha utilitzat *JUnit:4* només en les funcions que no fan servir funcionalitats d'*Android* perquè el motor de tests no carrega aquest subsistema. Com que totes les classes estan molt lligades amb el sistema *Android* només s'ha pogut fer *Unit Testing* de la classe de Criptografia.[18]

Durant l'etapa de creació dels tests ha sorgit la necessitat de crear **Mock Objects**, perquè la classe *AESCrypt*, que és l'encarregada de xifrar i desxifrar, utilitza el sistema *Android* per passar les dades a *Base64*. S'ha intentat utilitzar la llibreria *Mockito* per mockejar les dades, però no ha sigut possible per incompatibilitat de versions, per tant, s'ha decidit canviar l'enfocament i fer un *mock* de la classe *Base64* directament amb codi, aquest *mock* es troba al paquet *android.util*. Algunes de les proves dignes de menció que s'han realitzat són provar una *String* buida o caràcters no *ASCII*.

10.1.2 Exploratory Testing

L'*Exploratory Testing* s'ha documentat a un fitxer a part, apuntant els passos seguits a la prova i el resultat final.

Pantalla de creació de contrasenyes

Tasca	Estat
Accedir a Settings	Correcte
Insertar dades als tres camps	Correcte
Apretar botó de guardar guarda correctament les dades	Correcte
Apretar botó de tornar	Correcte
Insertar diferents dades als diferents apartats	Error: Es poden guardar els apartats identificador i contrasenya buits.

Fig. 15: Exploratory Testing de pantalla principal

Com es pot observar a la *fig(15)* s'han seguit totes les accions possibles i també fluxos si es necessari, com és el cas de canviar idiomes. I en cas de trobar un mal funcionament s'ha descrit aquest. Més endavant s'explica com ha sigut el flux de l'error per ser solucionat.

10.1.3 User Tests

Per als *User Tests* s'ha seleccionat una persona sense coneixements previs i se li ha descrit l'aplicació com una aplicació per guardar contrasenyes. A continuació se li ha donat l'aplicació per tal que la intenti utilitzar i s'ha observat el seu comportament.

S'ha pogut observar que la persona seleccionada mai ha entrat a la pantalla de *settings* i no ha utilitzat el porta-retalls per copiar i pegar les seves contrasenyes, pel que fa a la resta de funcionalitats les ha entès de manera bastant intuïtiva.

10.1.4 Gestió d'errors

Els errors trobats a l'aplicació s'han gestionat i documentat seguint el procediment mostrat a continuació.

Component	Bug	Passos per reproduir	Solucionat?
Creació de contrasenya	Els identificador i la contrasenya es poden emmagatzemar buits	Insertar strings buides al formulari de creació	SOLUCIONAT EN PROCÉS NO SOLUCIONAT NO ES UN BUG UI

Fig. 16: Gestio d'errors

A la *fig(16)* es veu com s'apunta el component que ha generat l'error, s'especifica quin ha sigut l'error també anomenat *Bug*, els passos necessaris per a la reproducció del mateix i el seu estat actual.

Una vegada s'avança en el procés de solucionar un *Bug* també s'escriuen petits comentaris a l'apartat de *Solucionat?*.

10.2 Resultats

En el cas del *User Test* s'ha pensat en canviar la icona dels tres puntets per un altre per fer-ho més intuïtiu.

Pel que fa a *Unit Testing*, com es pot veure a la *fig(17)* els resultats de la funció testejada han sigut satisfactoris.

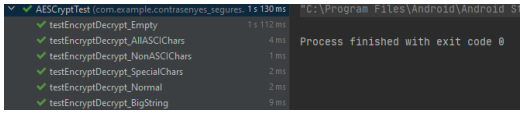

Fig. 17: Unit Testing resultats

L'*Exploratory Testing* i els *Bugs* han generat una taula que es troba en el següent estat.

Component	Bug	Passos per reproduir	Solucionat?
Creació de contrasenya	Els identificador i la contrasenya es poden emmagatzemar buits	Insertar strings buides al formulari de creació	*SOLUCIONAT* **S'ha posat un límit de 1 caracter per a l'identificador i 5 per a la contrasenya**
Detalls de contrasenya	El botó de pausa no para la reproducció	Al reproduir audio el botó de play es torna pausa. En aquest moment si s'apreta el botó de pausa l'audio es torna a reproduir.	*NO SOLUCIONAT*
Detalls de contrasenya	Els títols i les dades són difícils de distingir	Obrir la pantalla de detalls de contrasenya i observar les dades	*UI*
Pantalla de Login	La pantalla d'inici segueix en anglès	Canviar l'idioma a settings. Tancar l'app. Tornar a obrir l'app. Apretar botó de login.	*EN PROCÉS* **S'han canviat tots els strings menys el del interior del fingerprint que sembla utilitzar l'idioma del sistema**

Fig. 18: Estat actual dels bugs

La *fig(18)* mostra com s'han gestionat els *Bugs* i petits comentaris sobre com s'han solucionat. Si s'ha trobat algun problema en la solució del *Bug* i a causa d'aquest problema no s'ha pogut solucionar del tot, es deixa un comentari a l'estat.

11 Conclusions

Tenint en compte els objectius del projecte es podria dir que s'ha aconseguit assolir les finalitats de l'aplicació i de desenvolupament, es a dir, s'ha utilitzat una metodologia àgil per seguir totes les etapes que componen el desenvolupament d'un projecte arribant a obtenir un producte final que compleix els requisits funcionals definits. Aquest producte guarda les dades en una base de dades local evitant així possibles filtracions de dades d'un servidor, controla l'accés mitjançant una autorització biomètrica, xifra i desxifra dades amb una contrasenya *master*, i es comunica amb el *Cloud* de *Google* per proporcionar la funcionalitat de reproduir la contrasenya en àudio si es considera necessari, sempre informant l'usuari abans de reproduir l'àudio.

Pel que fa als canvis d'afegir la funcionalitat de controlar els idiomes en els quals està l'aplicació i els problemes a l'hora d'acabar algunes tasques com la de fer la petició al *Cloud* des d'*Android* aquests retards han fet que no es pogués acabar de millorar l'interficie d'usuari.

Com a possibles millores o passos a seguir es podria acabar de fer una interfície d'usuari més intuïtiva i visual, afegir control de volum a la pantalla on es troben els *settings* i comprovació de si la contrasenya és prou segura. Totes aquestes funcionalitats ajudarien a millorar l'experiència dels usuaris.

Agraïments

Agrair als meus pares per tot el suport que m'han donat durant temps difícils, a més, la meva mare, Luisa per fer de conillet d'índies per fer el test d'usuari. I a la meva germana, que encara que s'ha mudat seguim molt en contacte.

Referències

[1] Brian Turner. *Best password managers in 2021: Free and paid services to secure your passwords*. [online], 2021. Accedit 06 de març de 2021. URL: https://www.techradar.com/best/password-manager

[2] Yubico. *Proteja su mundo digital con YubiKey*. [online], 2021. Accedit 16 de juny de 2021. URL: https://www.yubico.com/?lang=es

[3] Wikipedia. *Agile software development*. [online], 2021. Accedit 07 de març de 2021. URL: https://en.wikipedia.org/wiki/Agile_software_development

[4] Wikipedia. *Scrum (software development)*. [online], 2021. Accedit 07 de març de 2021. URL: https://en.wikipedia.org/wiki/Scrum_(software_development)

[5] PlantUML. *PlantUML de un vistazo*. [online], 2021. Accedit 18 d'abril de 2021. URL: https://plantuml.com/es/

[6] diagrams.net. *Security-first diagramming for teams*. [online], 2021. Accedit 11 d'abril de 2021. URL: https://www.diagrams.net

[7] Microsoft. *El poder de simplificar la administración de proyectos*. [online], 2021. Accedit 03 de març de 2021. URL: https://www.microsoft.com/es-es/microsoft-365/project/project-management-software

[8] Documentation. *Qué es Android. La plataforma que está cambiando lo que pueden hacer los móviles*. [online], 2021. Accedit 14 de març de 2021. URL: https://www.android.com/intl/es_es/what-is-android/

[9] Documentation. *Cómo guardar datos con SQLite*. [online], 2020. Accedit 05 de abril de 2021. URL: https://developer.android.com/training/data-storage/sqlite#java

[10] Atlassian. *Trello siempre hace avanzar*. [online], 2021. Accedit 11 d'abril de 2021. URL: https://trello.com

[11] Github. *Github - Code Repository* [online], 2021. Accedit 24 de març de 2021. URL: https://github.com

[12] Postman, Inc. *The Collaboration Platform for API Development*. [online], 2021. Accedit 11 d'abril de 2021. URL: https://www.postman.com

[13] Documentation. *Documentación para desarrolladores de apps*. [online], 2021. Accedit 11 d'abril de 2021. URL: https://developer.android.com/docs

[14] Wikipedia. *Documentación para desarrolladores de apps*. [online], 2021. Accedit 29 d'abril de 2021. URL: https://es.wikipedia.org/wiki/Advanced_Encryption_Standard

[15] Documentation. *Cómo guardar contenido en una base de datos local con Room*. [online], 2021. Accedit 05 d'abril de 2021. URL: https://developer.android.com/training/data-storage/room#java

[16] Documentation. *Conceptos básicos de Cloud Text-to-Speech*. [online], 2021. Accedit 18 d'abril de 2021. URL: https://cloud.google.com/text-to-speech

[17] Wikipedia. *HTTPS*. [online], 2021. Accedit 18 de juny de 2021. URL: https://en.wikipedia.org/wiki/HTTPS

[18] Néstor Almeida Merino. *Testing con JUnit 4*. [online], 2019. Accedit 18 de juny de 2021. URL: https://www.nestoralmeida.com/testing-con-junit-4/