Relatore

DedicaDedica

SommarioSommario
*Sommario

Ringraziamentiringraziamenti

*Ringraziamenti

,

tableofcontents

lof

lot

Lista degli elementi di codicelol
*
* [1]

Contesto aziendale

Il presente capitolo introduce l'azienda presso cui ho svolto l'attività di *stage*, **SyncLab S.r.l**, fornendo la base nec

Presentazione azienda L'azienda ospitante, *SyncLab S.r.l*, è stata fondata nel 2002 a Napoli e si è affermata fin da s

[!ht] [width=0.9]../assets/sediSynclab.png Sedi operative dell'azienda - Fonte: synclab.it

Inizialmente nata come *software house*, l'azienda si è dedicata allo sviluppo di soluzioni *software* innovative, progett

Nel tempo, *SyncLab S.r.l* ha ampliato il proprio raggio d'azione, assumendo un ruolo rilevante come *system integrator*.
La duplice identità coniuga la creatività e la proattività di una *software house* con l'approccio orientato all'efficienza e a
SyncLab S.r.l promuove attivamente la collaborazione interna, incoraggiando l'interazione non solo tra i membri della st

Rami aziendali e progetti *SyncLab S.r.l* collabora con un ampio numero di clienti appartenenti a molteplici settori i
Alcuni dei *software* che l'azienda ha prodotto sono:
Sobereye (ambito *web*): un'applicazione innovativa progettata per monitorare il rischio di deterioramento neuro-cognit
SynClinic (ambito sanitario): un sistema integrato che supporta la gestione completa dei processi clinici e amministrat
DPS 4.0 (ambito *web* e *privacy*): una piattaforma *web* che supporta i titolari, responsabili, *data protection officer* (DP
Fast Ride (ambito trasporti): una soluzione per la gestione di servizi di trasporto pubblico a chiamata, in contesti urba

Way of working In questa sezione tratterò di alcune tecnologie di cui ho avuto esperienza diretta per lo sviluppo de
Dart: linguaggio di programmazione orientato agli oggetti sviluppato da Google, noto per la sua versatilità. Offre nume
lo sviluppo di un applicativo *cross-platform* che consente di creare applicazioni native per diversi sistemi da un'unica ba
compilazione *Ahead-Of-Time (AOT)* che accelerano lo sviluppo.
supporto alla programmazione asincrona che riduce la gestione dei processi in *background*.
Flutter: un *framework* basato su Dart per lo sviluppo di applicazioni multi-piattaforma, che offre vantaggi come:
Hot-reload: permette di visualizzare immediatamente le modifiche al codice durante lo sviluppo senza dover aspettare
Compatibilità con *material*: mette a disposizione un ricco arsenale di *widget* per creare interfacce utente moderne e
Firebase: un *database* sviluppato da Google e ben integrato con Flutter che offre una serie di funzionalità di *backend* p
Authentication: servizio che offre i servizi di *backend* e librerie pronte all'uso per autenticare gli utenti nell'applicazion
Firestore: *database NoSQL* orientato ai documenti, che permette di archiviare i documenti in raccolte, le quali fungono

Per garantire continuità operativa e una efficace gestione dei processi collaborativi, è necessario disporre di strumen

[!ht] [width=0.8]../assets/sviluppoSupporto.png Strumenti di supporto.

Git: un sistema di controllo versione distribuito che permette di tracciare in modo efficiente le modifiche ai *file* e di coo
Android Studio: è un ambiente di sviluppo integrato *(IDE)* gratuito, progettato per lo sviluppo di applicazioni **Andr**
Xcode: è un ambiente di sviluppo integrato *(IDE)*, sviluppato e mantenuto da Apple che contiene una *suite* di strumen
UMLet: è uno strumento gratuito e *open source* che permette di creare diagrammi UML[1]. In particolare diagrammi de

Smart working e strumenti di comunicazione L'azienda adotta un modello di lavoro prevalentemente da remoto: la
Nel mio caso, trattandosi di un progetto di *scouting* tecnologico, gli incontri con il responsabile designato sono stati

Per garantire una gestione efficace delle attività e tracciare correttamente l'evoluzione del progetto, ho utilizzato diversi
Google Calendar: un calendario condiviso che permette la creazione e la modifica di eventi, specificandone durata e lu
GitHub Projects: una sezione di GitHub dedicata alla gestione dei *ticket*, preferita in alternativa a **Trello** normalmen
Per la comunicazione interna durante il lavoro da remoto:
Discord: utilizzato dai dipendenti per lo scambio di informazioni tramite *chat* testuali e vocali. La piattaforma funge a
[!ht] [width=0.8]../assets/sviluppoComunicazione.png Strumenti di comunicazione

Spirito di innovazione aziendale

L'azienda *SyncLab* si caratterizza per una forte propensione all'innovazione, elemento che guida in modo significativ
Miglioramento delle dinamiche interne: l'azienda investe costantemente nell'ottimizzazione dei processi di gestione
Proposizione di soluzioni d'avanguardia: L'impegno innovativo non si limita all'organizzazione interna, ma si esten
In questo contesto, orientato alla ricerca dell'avanguardia si è collocata la mia esperienza di *stage*. La vocazione aperta a

[!ht] [width=0.75]../assets/progettoStage.png Progetto di *stage* come intersezione tra le aree di competenze consolid

Il progetto si è inserito coerentemente nella strategia aziendale, contribuendo su due fronti principali:

Scouting tecnologico: L'utilizzo di Flutter per la realizzazione di un'applicazione, caratterizzata da requisiti in ambit

Adattamento di competenze consolidate: sebbene *SyncLab* possieda già un panorama della sicurezza e dello svilup

L'approccio fondato su posizioni all'avanguardia di *SyncLab* ha rappresentato quindi un motore concreto che ha res

Visione aziendale Offerte aziendali Per *SyncLab*, i programmi di *stage* rappresentano uno strumento strategico per t

[!ht] [width=]../assets/collab.png Collaborazioni dell'azienda - Fonte: synclab.it

I programmi di *stage* si articolano principalmente in tre aree:
Integrazione: gli stagisti contribuiscono al perfezionamento di *software* già in uso, intervenendo su funzionalità specifi
Analisi e **ottimizzazione**: questa area prevede una valutazione approfondita delle soluzioni *software* esistenti, con l'ob
Innovazione: in questo ambito gli stagisti svolgono analisi teoriche e sperimentazioni su tecnologie emergenti, con *focu*
[!ht] [width=0.8]../assets/stage.png Contributo dei programmi di *stage* in SyncLab.

Stage in azienda Gli *stage* presso *SyncLab* rappresentano un investimento strategico bidirezionale, che riflette la du
Formazione e inserimento di risorse qualificate: Il programma di *stage* funge da canale primario per identificare e
Acquisizione di nuove competenze: Il programma di *stage* funge da banco di prova per l'innovazione continua, ridu
[!ht] [width=0.8]../assets/cap2Stage.png Investimento strategico degli *stage* per l'azienda
Ruolo del *tutor* aziendale Durante il percorso ogni stagista è affidato a un *tutor*. Il *tutor* è una figura, professionale
Guida: il *tutor* propone linee guida su come organizzare il lavoro e fornisce allo stagista materiali utili che possono aiut
Supporto: il *tutor* offre una serie di suggerimenti o soluzioni possibili ai problemi incontrati, questo per scongiurare l'e
Supervisore: il *tutor* offre *feedback* allo stagista nella valutazione del lavoro svolto per verificarne la qualità e supporta
Nel mio caso i dialoghi con il *tutor* assegnato sono stati molto importanti in quanto senza il suo confronto, e i suoi *feedb*
Motivazione dello *stage* L'obiettivo dello *stage* è stato la valutazione di fattibilità e il potenziale impiego di nuove te
[!ht] [width=0.8]../assets/codebaseUnica.png Confronto tra lo sviluppo nativo e l'approccio *cross-platform* con *Flut*
Progetto proposto Nel contesto attuale, i dispositivi mobili rappresentano uno strumento di comunicazione ampiam
Per tale motivo, l'azienda ha proposto la progettazione e l'implementazione di un'applicazione in grado si generare e ges
La realizzazione del progetto ha comportato l'identificazione e la risoluzione di alcune sfide progettuali e tecnologiche qu
*Gestione delle chiavi crittografiche Il recupero della chiave privata usata per decifrare un messaggio rappresenta un
[!ht] [width=0.9]../assets/cap2crittografia.png Esempio di comunicazione criptata
*Integrazione con il lettore NFC L'applicazione deve supportare l'utilizzo del lettore NFC presente sui dispositivi m
Obiettivi e vincoli Obiettivi Il progetto proposto è stato guidato da una serie di obiettivi volti a garantirne il succes

Studio di fattibilità: effettuare uno studio dettagliato e un'analisi approfondita sulla fattibilità tecnica dell'utilizzo di
l'integrazione con il *server Firebase*;
il funzionamento del lettore NFC;
la generazione di una coppia di chiavi (pubblica e privata) crittografiche;
Questo per permettere di valutare le potenzialità e i limiti degli strumenti, in modo da individuare i migliori da utilizzar

Sviluppo di una applicazione: per verificare che i componenti applicativi funzionino bene tra di loro, e che permetta
Vincoli tecnologici:
Utilizzo obbligatorio del *framework Flutter*:
L'intera applicazione deve essere sviluppata interamente tramite *Flutter*, con il conseguente utilizzo del linguaggio *Dart*
Utilizzo del piano gratuito di *Firebase*:
L'uso di *Firebase* presenta alcune limitazioni derivanti dal piano gratuito, che esclude funzionalità specifiche presenti sol
Vincoli architettura di sicurezza:
Protezione della chiave privata:
L'architettura deve impedire la trasmissione *online* della chiave privata, garantendo però un meccanismo sicuro che cons
Associazione delle chiavi ai documenti:
Ogni coppia di chiavi deve essere associata a un documento, e non a *tag NFC* generici.
Unicità del *wallet* per documento e *account*:
Non deve essere consentita la creazione di più *wallet* associati allo stesso documento per un medesimo *account*. Preserva
[!ht] [width=0.8]../assets/dualitàChiavi.png Schema che rappresenta l'architettura di sistema, la chiave blu corrispo
Scelta dello *stage* Motivazioni della scelta Ho conosciuto *Synclab* nel mese di aprile attraverso una comunicazione te
Parallelamente, durante il periodo estivo avevo ricevuto ulteriori offerte di stage da altre aziende e stavo valutando
Alla fine ho scelto di intraprendere il mio percorso in *Synclab* per i seguenti motivi:
Possibilità di lavorare da remoto, condizione per me fondamentale poiché, dovendomi spostare frequentemente e no
Elevato livello di autonomia offerto dal progetto, trattandosi di uno *scouting* tecnologico finalizzato alla valutazio
Obiettivi personali prefissati Nell'attuale contesto tecnologico, l'uso dei dispositivi mobili è divenuto parte integrant
L'introduzione della proposta di legge europea relativa al controllo delle *chat* e alla possibilità di analizzare i dati d
Da qui è nata la domanda che ha guidato parte della mia ricerca: esiste un modo per conservare le chiavi private in

Nello sviluppo dell'applicazione mi sono quindi posto i seguenti obiettivi:
Progettare un'applicazione capace di gestire i dati privati degli utenti in modo distinto rispetto a quelli pubblici, increm
Apprendere lo sviluppo di applicazioni per dispositivi mobili attraverso l'utilizzo del *framework* Flutter;
Acquisire i principi di crittografia necessari a consentire una comunicazione sicura tra due dispositivi mobili.
Per quanto qui permesso, lo stage presso *Synclab* costituiva per me un'importante occasione per approfondire i conc

	Attività	**Settimane** 1 2 3 4 5 6 7 8	**Ore**	
	Ripasso costrutti di Java	X	5	
	Studio di Dart	XX	30	
	Studio di Flutter	XX	40	
	Studio algoritmi di criptazione	X X X	30	
	Analisi del problema	X	10	
[H]	Progettazione della piattaforma	X	25	Pianificazione del lavoro durante le 8 settimane
	Sviluppo maschera di *login*	X	5	
	Sviluppo di un prototipo che genera chiavi	X	30	
	Sviluppo applicazione finale	XXX	100	
	Stesura finale della specifica tecnica	XX	20	
	Live demo e presentazione finale	X	5	
	totale ore	300		

Durante le prime settimane di stage, è stata applicata una metodologia a cascata, essenziale per stabilire le fondam

Una volta completata la prototipazione, il progetto è transitato verso una metodologia agile, in particolare durante la re

L'adozione quindi dell'approccio ibrido ha garantito la copertura formativa e analitica iniziale, consentendo al contempo
Analisi dei requisiti La fase di analisi svolta durante le prime settimane del percorso di *stage*, sono state cruciali per

Requisiti funzionali I requisiti funzionali definiscono le funzionalità e i servizi specifici che l'applicazione deve fornir

	Codice	Descrizione	
[H]	**RF1**	Generazione di una coppia di chiavi pubbliche e private	Requisiti funzionali
	RF2	Generazione ed eliminazione di *wallet* contenenti le coppie di chiavi	
	RF3	Implementazione della parte di accesso e registrazione tramite appoggio di *database*	
	RF4	Implementazione di un meccanismo di storage sicuro che permetta agli utenti di salvare le chiavi private sul *Keystore/Keychain* del dispositivo	
	RF5	Riuscire a criptare un messaggio inserito nell'applicazione tramite la chiave pubblica di un altro utente	
	RF6	Riuscire a decifrare un messaggio tramite la propria chiave privata personale	
	RF7	Implementazione di un meccanismo di recupero *wallet*	
	RF8	Implementazione di un meccanismo di lettura di *tag NFC* per associare un documento a un *wallet*	

Requisiti di qualità I requisiti di qualità stabiliscono degli *standard* secondo cui tali funzionalità del prodotto devon

	Codice	Descrizione	
[H]	**RQ1**	Assicurare un'accuratezza nella generazione e crittografia delle chiavi superiore al 95%	Requisiti di qualità
	RQ2	Mantenere il tempo di risposta per la generazione del *wallet* al di sotto dei 2 secondi	
	RQ3	Raggiungere una copertura di test automatici del 70% per le principali funzionalità	

Ognuno di questi requisiti è stato verificato attraverso lo sviluppo di test automatici.
Requisiti di vincolo I requisiti di vincolo definiscono le limitazioni operative e tecnologiche cui il progetto ha dovut

	Codice	Descrizione	
[H]	**RV1**	L'applicazione finale deve essere sviluppata tramite il *framework* Flutter e il linguaggio Dart	Requisiti di vincolo
	RV2	L'applicazione finale deve appoggiarsi a Firebase come appoggio per una base di dati	
	RV3	Le chiavi private generate non devono essere trasmesse *online* ma devono essere cutodite in spazi sicuri del dispositivo come il *Keychain* e *Keystore*	
	RV4	Ogni documento deve essere associato a unico wallet	

Implementazione Il linguaggio Dart Lo studio di questo linguaggio di programmazione ha rappresentato la prima sf

Dart è un linguaggio di programmazione orientato agli oggetti sviluppato da Google, è stato concepito come alternativa
[H] Esempio di *type-safe* dart void main() int temp = 10; temp = 20; // OK //temp = "ciao"; // ILLEGALE: str
print(temp); // stampa 20
dynamic temp1 = 30; print(temp1); // stampa 30 temp1 = "ciao"; print(temp1); // stampa ciao
**null-safe* Il linguaggio integra un sistema di *null safety*: il valore `null` può essere associato a una variabile solo se
[H] Esempio di *null safety* dart void main() String nome = "Mario"; print(nome); // stampa: Mario //nome = nu
String? cognome = "Rossi"; cognome = null; print(cognome); //Nessun problema stampa null
*Interfacce e classi astratte In Dart sia le interfacce esplicite che le classi astratte sono dichiarate usando la *keywor*
Un'interfaccia definisce un contratto: qualsiasi classe che la derivi deve utilizzare la *keyword* `implements`, impegnar
[H] Esempio di interfaccia e classe astratta dart abstract class ContrattoDiConnessione void connetti(); void discon
class B void bMethod()
abstract class BaseScreen void logicaComune()
class GestoreServizio implements ContrattoDiConnessione, B @override void connetti() /* implementazione */ @
class HomePage extends BaseScreen void init() super.logicaComune(); In Dart, ogni classe definita genera impl

Il *framework* Flutter
La seconda sfida è stata lo studio del *framework* Flutter. Flutter non utilizza componenti di interfaccia nativi del si

Per lavorare bene in Flutter è fondamentale comprendere che ogni elemento dell'interfaccia è rappresentato da un *widge*
[H] *Stateless Widget* dart class Titolo extends StatelessWidget final String testo;
const Titolo(super.key, required this.testo);
@override Widget build(BuildContext context) return Text(testo);
**Stateful widget* Uno *stateful widget* è un *widget* mutabile, in grado di mantenere e aggiornare un suo stato interno.
[H] *Stateful Widget* dart class Contatore extends StatefulWidget const Contatore(super.key);
@override State¡Contatore¿ createState() =¿ $_{C}ontatoreState()$;
class $_{C}ontatoreStateextendsState < Contatore > intvalore = 0$;
@override Widget build(BuildContext context) return Column(children: [Text('Valore: $valore'), ElevatedButton$
Architettura a tre livelli L'applicazione sviluppata adotta il modello di architettura a tre livelli, una struttura che s

Il primo livello di presentazione rappresenta un punto di interazione tra l'utente e il sistema. È composto da tutto ciò cl

Il secondo livello rappresenta la logica di *business*. Coordina le operazioni tra il livello di presentazione e il livello dei da

Il terzo livello è responsabile della persistenza e recupero dei dati. Definisce come queste vengono archiviate e rese dispo
Permette la separazione delle responsabilità (SoC[4]) eliminando la necessità di propagare le informazioni tramite la gera
Aumenta le *performance* consentendoci di costruire i *widget* strettamente necessari quando lo stato cambia, evitando i *r*
Fornisce un meccanismo per accedere e aggiornare lo stato rendendo il codice più facile da mantenere, facilitando quindi
la tecnologia NFC La tecnologia NFC permette a due dispositivi di connettersi, scambiarsi informazioni o attivare f
Lettura dati nei *chip* RFID Uno dei requisiti era l'implementazione di un meccanismo di riconoscimento dei docume

Per realizzare questo sono state analizzate due librerie Flutter che permettono la scansione e la scrittura attivando il ser
Grazie alla documentazione fornita dagli sviluppatori eseguire un prototipo che permetta la scansione dei documenti è s
[H] [width=0.5]../assets/PrototipoNFC.png Applicazione prototipale per dimostrare la fattibilità della scansione NF
La libreria ci permette di recuperare i campi di ID, lo *standard* ISO del *chip* RFID (ad esempio la carta d'identità e

Testando l'applicazione con una varietà di documenti e *tag* NFC, ho riscontrato differenze significative nella struttura de
Nel caso della CIE ho rilevato un comportamento peculiare: il campo identificativo (`id`) letto tramite NFC non rappres

Il *chip* della Carta d'Identità Digitale (CIE) è infatti progettato per memorizzare dati personali sensibili, proteggendole
L'identificatore temporaneo e non persistente rientra nelle misure previste per ridurre i rischi legati al tracciamento e all

Questo rappresenta per il progetto un bel problema in quanto senza un identificativo del documento recuperabile tramit

Una prima soluzione valutata è stata quella di consultare la piattaforma dedicata agli sviluppatori che realizzano i softw

A seguito di riunioni interne con il mio responsabile, si è scelto di non procedere su questa strada, sia per ragioni di tem

I documenti quali la CIE, la tessera sanitaria o le carte di credito adottano lo standard ISO/IEC 14443-4, che prevedon
[H] Funzione che permette la scansione del tag nfc [fontsize=]dart Future¡NFCTag?¿ fetchNfcData() async try if (
Future¡void¿ $_{s}canNfcTag()asyncif(_{i}sScanning)return; setState(() =>_{i} sScanning = true); tryfinalnfcService =$

Studio degli algoritmi di crittografia In questa sezione si espongono e si analizzano i principali algoritmi crittografic

Per analizzare la comunicazione digitale è importante garantire quattro aspetti:
Confidenzialità: assicurare che nessun osservatore non autorizzato sia in grado di leggere il messaggio durante la trasm
Integrità: garantire che il messaggio non sia alterato da nessuno.
Autenticazione: verificare che il mittente sia chi dichiara di essere.
Non ripudio: impedire che il mittente possa negare di aver inviato il messaggio.

Algoritmi a chiave simmetrica Gli algoritmi di crittografia simmetrica utilizzano un'unica chiave segreta condivisa t
Il funzionamento è descritto come segue:
Alice e Bob concordano o si scambiano in modo sicuro una chiave condivisa k;
Alice prende il messaggio in chiaro P e applica un algoritmo di cifratura simmetrica S, ottenendo il messaggio cifrato C
Quando Bob riceve C, applica l'algoritmo di decifratura D, utilizzando k, ricostruendo il messaggio originale P.
Pertanto per adottare questo algoritmo Alice e Bob devono conoscere e adoperare la stessa chiave utilizzando un canale
Se la chiave venisse rubata, l'attaccante può fingersi il mittente originale.
Non è possibile distribuire in modo sicuro la chiave privata a un partner remoto senza utilizzare un altro sistema di sicu

Algoritmi a chiave asimmetrica

Gli algoritmi di crittografia asimmetrica superano le limitazioni logistiche degli algoritmi di crittografia simmetrici.
Il funzionamento è descritto come segue.
Bob genera una coppia di chiavi (k^B_{pub}, K^B_{priv}) e rende k^B pubblico a tutti;
Alice ricerca la chiave pubblica di Bob e cifra il messaggio in chiaro P con la chiave pubblica di Bob k^B_{pub}, ottenendo un
Bob riceve il messaggio e lo decifra con utilizzando la propria chiave privata k^B_{priv} ottenendo P.
L'uso di questi algoritmi risolve il problema dello scambio della chiave presente negli algoritmi di crittografia simmetrica

Realizzazione di una *chat end-to-end* Uno degli obbiettivi del progetto riguarda la generazione di coppie di chiavi p

[H] Generatore di numeri casuali dart SecureRandom getSecureRandom() final secureRandom = FortunaRandom(
Questo generatore viene passato al metodo `generateRSAkeyPair()` responsabile della creazione delle chiavi RSA. I
[H] Generazione della coppia di chiavi dart AsymmetricKeyPair¡PublicKey, PrivateKey¿ generateRSAkeyPair(Secur
Per rispettare i requisiti e permettere sia allo *stakeholder* di visualizzare il risultato ottenuto le chiavi sono mostrate
[H] [width=0.9]../assets/KeyPairVisualized.png Risultato del processo di creazione delle chiavi

Codifica e decodifica del messaggio Una volta generate le coppie di chiavi RSA è possibile utilizzarle per la realizzaz
La cifratura è gestita dal metodo `rsaEncryptBase64()` che elabora il messaggio in chiaro tramite il motore crittografico
[H] Codifica di un messaggio mediante l'uso delle chiave pubblica dart Future¡String?¿ rsaEncryptBase64(String pla
try final encrypted = engine.process(Uint8List.fromList(utf8.encode(plainText))); return base64Encode(encrypted)
Il messaggio cifrato restituito viene poi convertito in un formato *Base64* in modo tale che sia memorizzato e trasme
[H] [width=]../assets/Base64Decoder.png decodifica del testo cifrato in *base64* – Strumento: base64decode.org
La decifratura è affidata a `rsaDecryptBase64()`, che esegue l'operazione inversa decodificando il testo cifrato trami
[H] Decodifica di un messaggio mediante l'uso della chiave privata dart Future¡String?¿ rsaDecryptBase64(String ci
try final decrypted = engine.process(base64Decode(cipherText)); return utf8.decode(decrypted); catch (e) return
Problema di questo algoritmo L'implementazione dell'algoritmo mostrato presenta un problema specifico per la real
È possibile procedere in due modi diversi per la risoluzione del problema:
la prima soluzione consiste nel salvare localmente sul dispositivo del mittente una copia del messaggio in chiaro prima cl
la seconda soluzione adottata nel progetto consiste nel inviare due copie dello stesso messaggio al *database*: la prima cifr
Tuttavia questa scelta espone un rischio aggiuntivo in quanto un attaccante potrebbe individuare nel *database* due mess
[H] Modifica agli algoritmi di codifica e decodifica aggiungendo OAEP dart Future¡String?¿ rsaEncryptBase64(Strin
Future¡String?¿ rsaDecryptBase64(String cipherText,RSAPrivateKey privateKey) async final engine = OAEPEnco
Gestione sicura delle chiavi pubbliche e private La protezione delle chiavi crittografiche generate sul dispositivo dell
A questo punto avviene le separazione delle due chiavi ricordando che la chiave privata deve essere salvata localmen

// Inserire Immagine

La chiave pubblica viene inviata al *database* Firestore per essere associata al *wallet* dell'utente insieme ai dati di creazio
DocumentReference docRef = await $_{f}irestore.collection('wallets').add(walletDataForFirestore)$;
Per la chiave privata invece viene chiamata la classe di `SecureStorage` che utilizza la libreria di `flutter_secure_s`
[H] Salvataggio della chiave privata sul dispositivo dart await $_{s}ecureStorage.writeSecureData(tempWallet.localK$
La classe `SecureStorage` La classe `SecureStorage` è responsabile del salvataggio, lettura ed eliminazione della chia
[H] La classe `SecureStorage` dart
class SecureStorage implements ISecureStorage final FlutterSecureStorage storage;
SecureStorage(FlutterSecureStorage? storage) : storage = storage ?? const FlutterSecureStorage();
@override Future¡void¿ writeSecureData(String key, String value) async await storage.write(key: key, value: value)
@override Future¡String?¿ readSecureData(String key) async final value = await storage.read(key: key); return val
@override Future¡void¿ deleteSecureData(String key) async await storage.delete(key: key);
Per accedere al dato salvato localmente è quindi necessario che alla creazione della coppia di chiavi venga anche ge
[H] Creazione della chiave identificativa dart var uuid = const Uuid(); final newLocalKeyIdentifier = uuid.v4();
Tramite il pacchetto `uuid` è possibile generare una chiave unica da associare alla chiave privata per permetterne il r
Risultati Raggiunti (in questa sezione parlo dei risultati e raggiunti, delle funzioni implementate nell'applicazione c

Bibliografia
[heading=subbibliography,title=Siti web consultati,type=online]