5ªA Informatica e Telecomunicazioni

Esame di Stato A.S. 2016/2017

Cassaforte 2.0

Di

Piscitelli Federico
Leggio Giuseppe
Perego Daniele
Rosselli Mattia

SOMMARIO

INTRODUZIONE

L' IDEA

L' idea di creare una cassaforte smart è nata dall' esigenza di creare un prodotto innovativo non ancora presente sul mercato. Un altro aspetto che ci ha spinto alla realizzazione di questo progetto è la possibilità di estendere il meccanismo di sicurezza ad altri ambiti, come ad esempio in quello della domotica.

QUALI FUNZIONI OFFRE

Oltre ad essere una normale cassaforte, il nostro prototipo offre diverse funzionalità che rappresentano sicuramente un approccio innovativo nel campo della sicurezza. Prime fra tutte:

- Applicazione Android
- Comunicazione Bluetooth
- Registrazione con successiva consultazione tramite app
- Sensore di riconoscimento biometrico

OBIETTIVI

La Cassaforte 2.0 si pone come obiettivo quello di creare un progetto sicuro, affidabile e intuitivo, con l'aggiunta di alcune funzioni "smart".

INNOVAZIONE

Questo progetto pone come primo obiettivo quello di trasformare un oggetto millenario in qualcosa di moderno, al passo con i tempi.

Il primo passo per la modernizzazione della cassaforte avviene tramite l'aggiunta di un sensore per il rilevamento dell'impronta digitale, diventato ormai un must della sicurezza.

Per rendere ancora più all'avanguardia questo progetto ci si è chiesto: "Quale oggetto oggi rappresenta l'innovazione?"
Subito è saltato all'occhio lo smartphone che al giorno d' oggi è parte integrante della nostra vita.

Il modo migliore per combinare questi due mondi è stato quello di sviluppare un'applicazione dedicata alla gestione della cassaforte.

Nonostante il progetto preveda l'utilizzo dello smartphone per alcune funzioni, la cassaforte può svolgere la sua normale funzione di custodia anche senza l'obbligo di utilizzo dell'applicazione, rendendo accessibile a tutti , grandi e piccoli, le sue funzioni, ampliando così il range di persone raggiungibili nel mercato.

SICUREZZA

Uno degli obiettivi fondamentali della Cassaforte 2.0 è quello di garantire una sicurezza costante ed efficiente.
Per fare ciò il prototipo propone una doppia chiave in base al caso d'uso:

1. Tastierino numerico
2. Sensore d' impronte digitali

Proprio grazie a questo doppio livello di sicurezza, anche scoperto il codice di accesso, risulato quasi impossibile l'apertura in maniera fraudolenta della cassaforte.

STRUMENTI

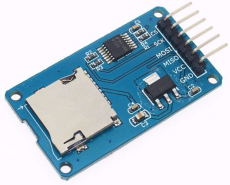
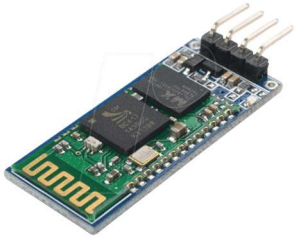
La Cassaforte 2.0 utilizza un Arduino che permette una completa adattabilità, data dalla possibilità di poter compilare il codice in base alle necessità del cliente, dando la possibilità di aggiungere o modificare la varia componentistica.

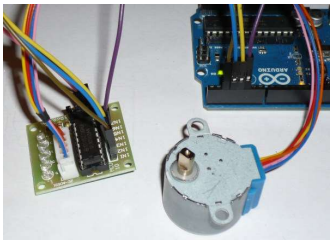
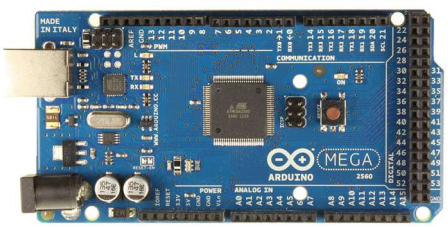
STRUMENTI UTILIZZATI

- Arduino Mega
- Modulo RTC (Real Time Clock, utile per tenere ora e giorno)
- Modulo Bluetooth HC-06
- Modulo per l'espansione della memoria
- Sensore d'impronte digitale
- Schermo LCD
- Motorino Step to Step
- Cassaforte in legno per simulare un reale caso d'uso.
- Android Studio e Arduino IDE come ambienti di sviluppo per la programmazione.
- Librerie fornite dai vari costruttori dei componenti

Durante la fase di progettazione si è deciso di utilizzare un Arduino Mega, a dispetto di un Arduino UNO, per la posssibilità di scalabilità che offriva il primo, data la quantità di pin disponibili.



METODOLOGIA E SVILUPPO

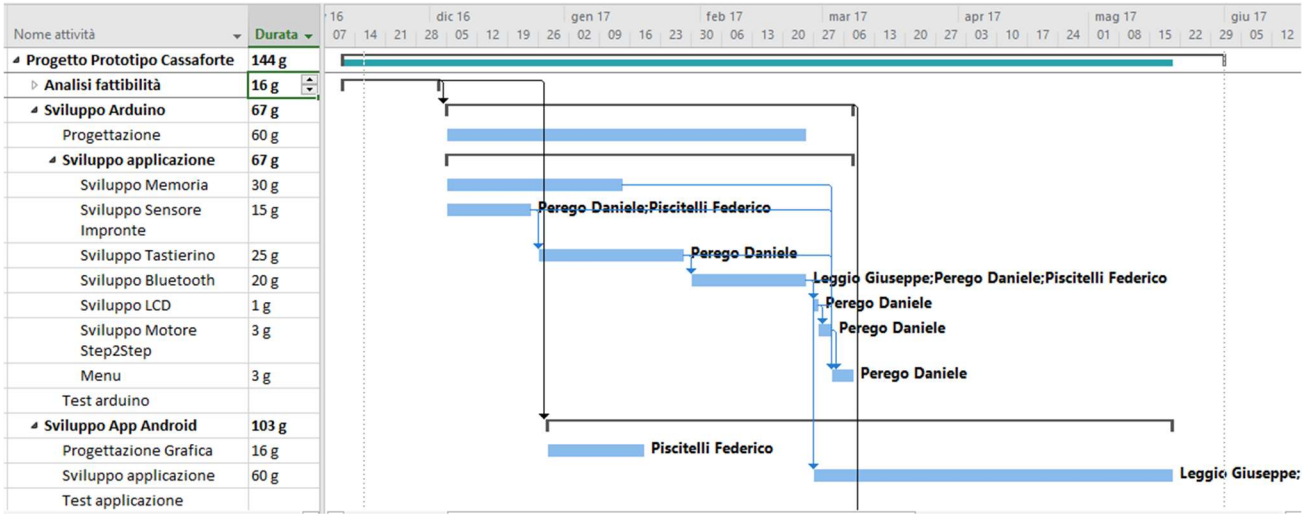
Essendoci due ambienti di sviluppo differenti si è optato per la divisione del team in due rami.

Il primo gruppo, composto da Leggio e Piscitelli, si è impegnato nella programmazione e nello sviluppo dell'applicazione Android.

Il secondo gruppo, composto da Perego e Rosselli, ha avuto come incarico il montaggio dei vari componenti hardware e la programmazione delle sue funzionalità.

TEMPISTICHE

Per organizzare al meglio i tempi e le risorse abbiamo utillizzato Microsoft Project, con il qule siamo riusciti a stilare una WBS che è stata rispettata da tutti i componenti del gruppo.



ANDROID

Lo sviluppo dell'applicazione Android si è divisa in tre fasi:

- Progettazione e realizzazione della grafica;
- Gestione file di log e ultimi accessi scaricati;
- Implementazione della comunicazione Bluetooth per le varie funzionalità.

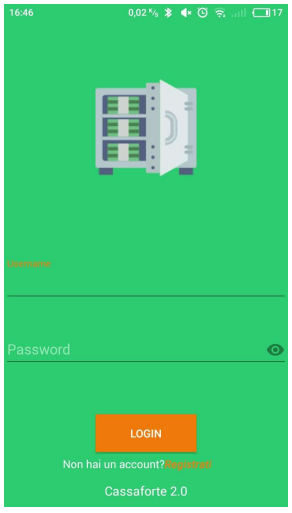
Durante la prima fase, si è realizzato un "bozzetto" in Android Studio di come sarebbe dovuta essere l'applicazione. Si è passati poi alla creazione dei colori per i bottoni e per lo sfondo. Successivamente sono state implementate tutte le animazioni nelle varie activity.

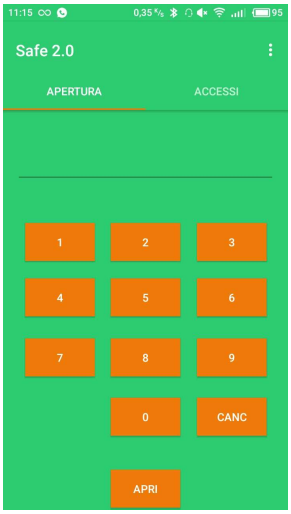
La parte più impegnativa a livello di design è arrivata quando si è dovuto implementare il Fragment, un elemento introdotto con Android 3, per la creazione di interfacce grafiche. È una sorta di sub activity. Infatti il suo funzionamento è legato ad un'activity principale.
La struttura di questo tipo di programmazione consiste nel creare diversi layout per ogni fragment (vista) che viene implementata, essi poi vanno collegati a una corrispondete classe, proprio come se fosse un'activity, con la differenza che poi queste diverse "viste" sono gestite da un'activity principale.
Questo ne rende più difficile la gestione in quanto tutti i metodi per richiamare dei componenti grafici dal file xml sono differenti. Ad esempio per richiamare un bottone in un'activity semplice si scrive questa linea di codice:

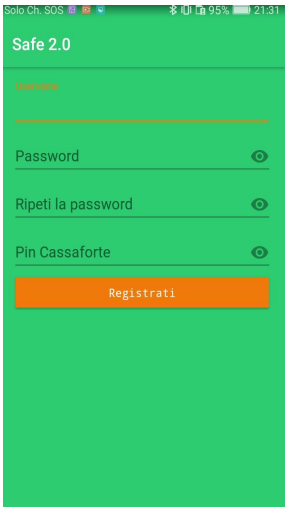
```
Button button = (Button)findViewById(R.id.button0);
```

Mentre in un Fragment:

```
View rootView = inflater.inflate(R.layout.fragment_layout,
container, false);
Button button =
(Button)rootView.findViewById(R.id.button0);
```







La fase seguente è stata l'implementazione della gestione dei file. L'implementazione si è resa necessaria perché si voleva garantire la possibilità di utilizzare l'applicazione in parte anche quando l'applicazione non è connessa alla Cassaforte.

I file di log che vengono registrati sono due:

- Il primo che memorizza i dati di accesso una volta che è stato eseguito con successo il login tramite connessione.
- Il secondo che memorizza gli ultimi accessi scaricati.

```
private Boolean leggiFileLog(String datiLogin)
    String line = "";//Riga letta
    BufferedReader br;
    boolean trovato = false;
    try {
        FileInputStream fin = openFileInput(filename);
        int c;//Indice linea del file corrente
        while ((c = fin.read()) != -1) {//legge un carattere alla volta
            if ((char) c != '%') //carattere che delimita login salvata
                line += Character.toString((char) c);//forma login
            else {//una volta trovato lo confronta con i
                if (line.equals(datiLogin)) {//dati inseriti dall'utente
                    trovato = true;
                    break;
                }
                line = "";
            }
        }
    } catch (Exception e) {}
    return trovato;//ritorna true o false in base se ha trovato o meno la
corrispondenza con i dati inseriti dall'utente
}
```

L'ultima fase, quella più lunga e in cui sono emerse notevoli difficoltà, è stata quella in cui è stata implementata la comunicazione Bluetooth.

Per comunicare con la Cassaforte utilizza tre elementi:

- **La classe "BluetoothComIno",** creata da Leggio. Il principio di questa classe è quello di istanziare una socket, che ha come IP il MAC del modulo HC-06, e come porta richiama questo metodo, `BluetoothDevice.createRfcommSocketToServiceRecord(uuid)`, il quale permette di aprire una connessione p2p.

- **Un thread per la ricezione dei dati**. Esso viene utilizzato poiché, oltre a un motivo di strettamente computazionale, nel sistema Android, questo tipo di operazioni non possono essere gestite dal *main-thread,* bisogna utilizzare dei *worker-thread*.

- **Un handler.** Poiché i dati devono poi essere elaborati dal main thread, diventa necessario l'utilizzo di questo oggetto, perché permette la comunicazione tra *main-thread e worker-thread.*
  In sintesi quindi, questo strumento fa da intermediario tra worker thread e main thread utilizzando un oggetto di tipo Message, il

quale viene aggiornato dal worker thread e passato tramite bundle al main thread in cui poi viene elaborato.

```
private boolean connetti() {//Metodo che tenta la connessione alla cassaforte

    if (bluetoothComIno.connectToIno()) {//se la connessione va a buon fine
        Toast.makeText(getActivity(), "Connesso", Toast.LENGTH_SHORT).show();
        connesso = true;
    } else {
        Toast.makeText(getActivity(), "Connessione non effettuata: "
                + bluetoothComIno.getErrorMessage(), Toast.LENGTH_SHORT).show();
    }
    return connesso;
}
```

```
h = (Handler) handleMessage(msg) → {
        switch (msg.what) {
            case RECIEVE_MESSAGE: //se il thread ha mandato dei dati
                byte[] readBuf = (byte[]) msg.obj;//prelevo i dati
                String strIncom = new String(readBuf, 0, msg.arg1);// li converto in stringa
                sb.append(strIncom);//li aggiungo a uno stringBuilder
                msgControlloRicevuto = sb.toString();//li assegno all' attributo di calsse
        }
};
```

```
public void run() {

    byte[] buffer = new byte[256];  // buffer
    int bytes;
    Log.d(TAG, "...Sto ricevendo...");
    while (true) {
        try {
            // Leggo dall'InputStream
            bytes = mmInStream.read(buffer);
            h.obtainMessage(RECIEVE_MESSAGE, bytes, -1, buffer).sendToTarget();//li mando all'Handler
        } catch (IOException e) {
            break;
        }
    }
}
```

Durante la programmazione della comunicazione Bluetooth sono stati riscontrati diversi problemi, soprattutto riguardanti la ricezione. Molte volte infatti venivano persi o duplicati caratteri.
Il problema derivava principalmente da due fattori:

- l'Arduino venivano utilizzato spesso in modalità "Debug" e ciò creava degli scompensi nella comunicazione.
- Un'alimentazione non sufficiente. Questo è stato riscontrato nei casi in cui è stato usato un net-book come alimentatore, dal quale esce corrente in modo poco stabile. Se l'Arduino è collegato invece ad una fonte di alimentazione esterna, che non sia una porta USB del pc, questo problema non persiste.

Altri problemi sono emersi nuovamente per via dell'utilizzo del via del menu Fragment, per via del suo ciclo di vita differente rispetto alla normale activity.
Essendo collegati entrambi alla stessa activity, essi vengono creati allo

stesso momento. Di conseguenza è stato necessario anche sincronizzare quale dei due si doveva connettere per primo, e le varie fasi di disconnessione e riconnessione.

ARDUINO

Anche lo sviluppo di Arduino e dei componenti è stato diviso in due momenti:

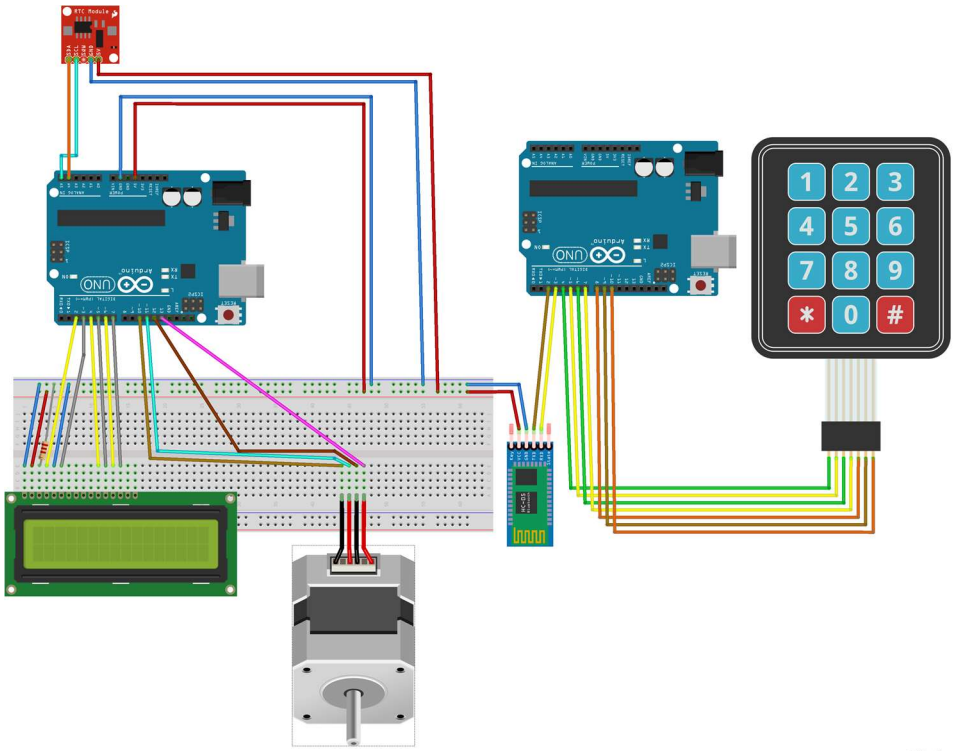
- Programmazione e test dei singoli componenti
- Assemblaggio e test complessivo

Per la parte fisica abbiamo subito riscontrato delle problematiche, siccome abbiamo dovuto attendere più di un mese dall'acquisto dei componenti affinché questi arrivassero. Una volta arrivata la componentistica, il team hardware si è diviso in due e ha cominciato a testare e a programmare uno ad uno ogni modulo.

I primi problemi sono stati riscontrati con il tastierino numerico che non corrispondeva allo schema elettrico fornitoci dal venditore. Quindi il gruppo hardware si è dovuto occupare della stesura dello schema elettrico corretto, aprendo il tastierino per controllarne il reale funzionamento dei pin.

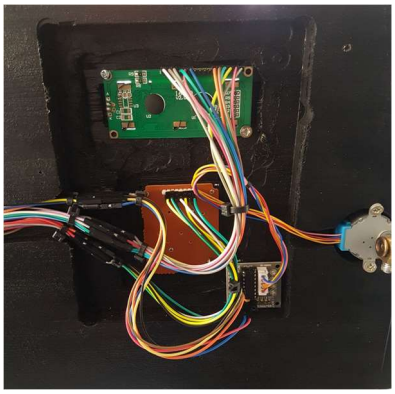
Arrivati al test del sensore d'impronte digitali abbiamo avuto altri problemi, dato che la libreria fornitaci dal costruttore non era in grado di rilevare il sensore. Per ovviare a questo inconveniente abbiamo dovuto ricercare una libreria che si adattasse al nostro sensore. Fortunatamente abbiamo trovato la libreria "AdaFruit-Fingerprint" che pur non essendo quella apposita per il modello del sensore, funzionava perfettamente.

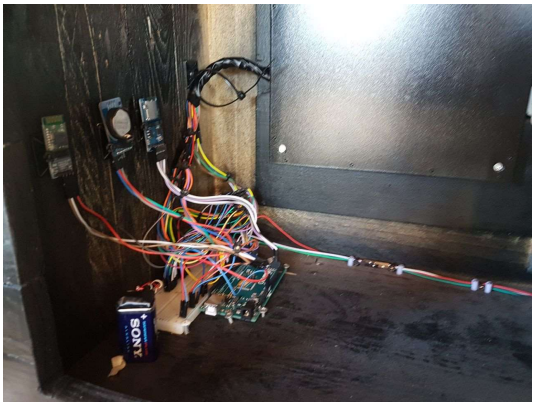
Dopo aver testato e programmato ogni componente singolarmente il team preposto per lo sviluppo di Arduino ha montato su una breadboard tutti i componenti e ha cominciato lo sviluppo del "Menu", ossia il programma che gestisce le funzionalità della cassaforte sincronizzando il lavoro di tutti i componenti.



Durante questa fase ci si è accorti di un problema di alimentazione. Tutti i componenti infatti non riuscivano ad essere alimentati dall'Arduino. Questo inconveniente è stato risolto facilmente inserendo una batteria e collegandola alla breadboard in modo da dare un supplemento di energia a tutti i componenti collegati.

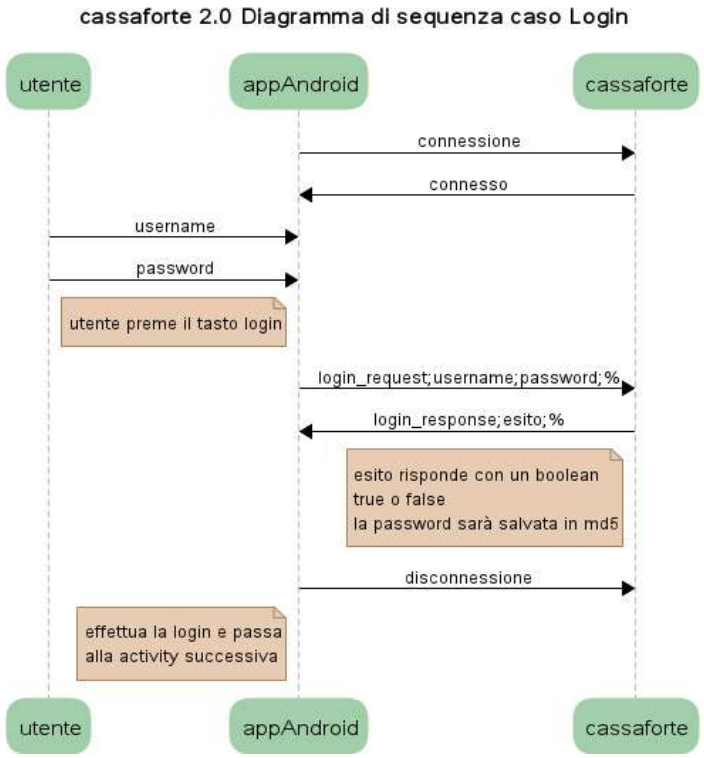
Successivamente ci si è occupati del montaggio di tutti i componenti all' interno della cassaforte e il risultato finale è il seguente:

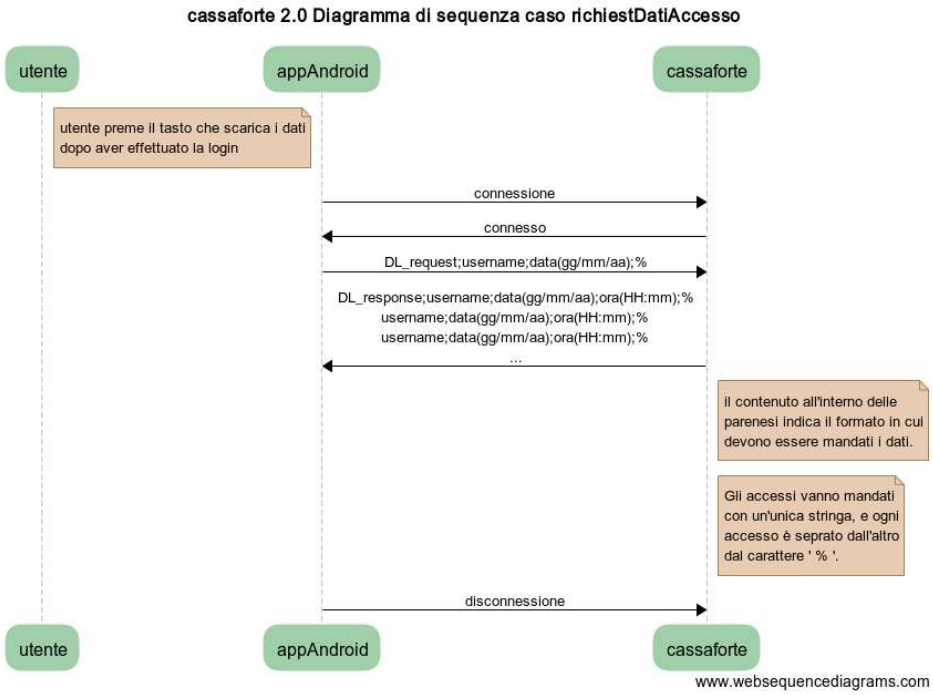


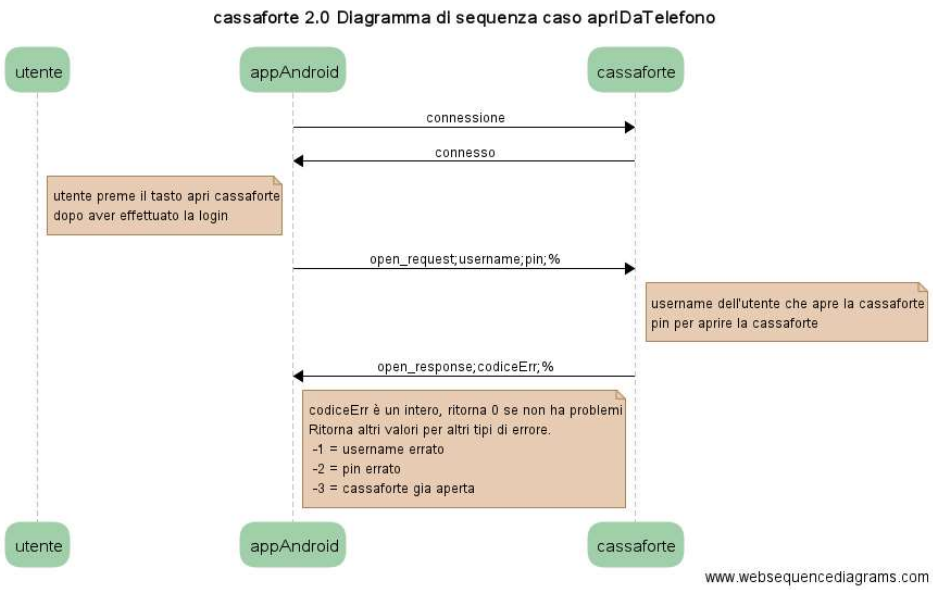


PROTOCOLLI DI COMUNICAZIONE

Per gestire in maniera ottimale la comunicazione sono stati creati dei protocolli di comunicazione grazie al sito “websequencediagrams.com”. I principali sono riportati di seguito:






cassaforte 2.0 Diagramma di sequenza caso aprIDaTelefono
utente
appAndroid
cassaforte
connessione
connesso
utente preme il tasto apri cassaforte dopo aver effettuato la login
open_request;username;pin;%
username dell'utente che apre la cassaforte
pin per aprire la cassaforte
open_response;codiceErr;%
codiceErr è un intero, ritorna 0 se non ha problemi
Ritorna altri valori per altri tipi di errore.
-1 = username errato
-2 = pin errato
-3 = cassaforte gia aperta
utente
appAndroid
cassaforte
www.websequencediagrams.com

BETA TEST

Durante il beta test, il gruppo si è riunito e ha provato ogni funzionalità della Cassaforte 2.0, risolvendo eventuali errori nella comunicazione e aumentando le prestazioni in termini di velocità.

Principalmente sono stati corretti:

- Ritardi nella comunicazione
- Errori nella comunicazione
- Bug dell'applicazione
- Errori nella registrazione degli utenti su Arduino
- Errori nell'invio degli accessi in base alla data

POSSIBILI MIGLIORIE

Durante la fase del beta test, il team ha anche pensato a dei possibili miglioramenti che potrebbero rendere ancora più efficiente il progetto:

- Aggiungere una fotocamera per scattare delle foto a chi effettua gli accessi
- Usare Raspberry al posto di Arduino Mega per gestire al meglio le richieste dell'utente e per potersi connettere alla rete e salvare gli accessi su un database, contattabile anche da applicazione tramite un web service.
- Utilizzare un modulo wifi al posto del modulo Bluetooth per velocizzare la comunicazione e per renderla più stabile.

FUNZIONAMENTO

PRIMO AVVIO SENZA APPLICAZIONE

Se si vuole utilizzare la cassaforte senza applicazione al primo avvio bisognerà salvare l'impronta digitale dell'utente digitando il codice *5 e premendo successivamente # per inviare. Verrà richiesto il pin della cassaforte (fornito dagli sviluppatori) e successivamente partirà la sequenza per salvare l'impronta. Il sensore d'impronte (in basso a sinistrà) comincerà a lampeggiare. Bisognerà appoggiare il dito e tenerlo fermo per 5 secondi. La luce smetterà di lampeggiare. Ripetere la sequenza fino a quando sullo schermo non comparirà la scritta "Impronta registrata".

La cassaforte è pronta ora per l' utilizzo.

Ogni qualvolta che si vorrà accedere alla cassaforte bisognerà digitare il pin e premere #, inserire l'impronta appena il sensore comincerà a lampeggiare e attendere l'apertura della Cassaforte. Una volta finito l' utilizzo, si dovrà chiudere manualmente lo sportello e si dovrà premere # per avviare la chiusura della cassaforte.

PRIMO AVVIO CON APPLICAZIONE

Se si vuole utilizzare la cassaforte con l'applicazione bisognerà, come prima cosa, creare un nuovo utente. Quindi una volta aperta l'applicazione l'utente dovrà andare nell'apposita activity e inserire uno username (minimo 6 caratteri), una password (minimo 5 caratteri) e il pin della cassaforte. Inviando tutto alla cassaforte verrà richiesto di salvare l'impronta dell'utente. Il sensore comincerà a lampeggiare e la procedura da seguire è la medesima descritta sopra. Una volta che l'utente ha l'account registrato, può usufruire dei servizi dell'applicazione effettuando la login. Ad esempio potrà aprire la cassaforte da telefono o potrà controllare gli accessi che sono stati effettuati in un determinato giorno.

RINGRAZIAMENTI

Il progetto ci ha portato via molto tempo negli ultimi mesi e nonostante i nostri sforzi continui ci sono persone esterne al nostro gruppo che ci hanno sostenuto ed aiutato nella realizzazione di questo prototipo.
In particolare vogliamo ringraziare Perego Attilio per il tempo dedicatoci per la realizzazione della cassaforte in legno e i prof per la disponibilità nel fornire consigli e aiuti quando necessario.

BIBLIOGRAFIA E SITOGRAFIA

www.stackoverflow.com

www.arduino.com

www.github.com

developer.android.com

www.androidworld.it