Universitatea tehnică din Cluj-Napoca

Facultatea de Electronică,Telecomunicații și Tehnologia Informației

Dezvoltarea de Aplicaţii pentru Telefoane Mobile

Password Manager

Specializarea:

Master: Telecomunicaţii

Profesor

SI.Dr.Ing. Adriana Stan

Student masterand

Ing. Robert Lascu

Cuprins

1. Introducere

Password Manager este o aplicaţie Android având ca scop stocarea datelor (username/parolă) diferitelor conturi folosite de utilizator. Aplicaţia constă în 3 activităţi: activitatea pentru login, o activitate pentru adăugarea conturilor şi o activitate pentru afişarea conturilor memorate.

Pentru partea de autentificare se folosesc Shared Preferences, cu scopul de a stoca parola master, introdusă la fiecare deschidere a aplicaţiei. Memorarea datelor se face cu ajutorul bazei de date de tip SQLite, disponibilă în Android.

Din punctul de vedere al securităţii datelor, aplicaţia stochează atât parola master, cât şi celelalte parole ale conturilor memorate, în mod criptat, folosind algoritmul simetric AES, pe 128 biţi.

2. Funcţionalitate

La deschiderea aplicaţiei, utilizatorul va fi întâmpinat de prima activitate, cea pentru login. Dacă este prima utilizare a aplicaţiei, va apărea un mesaj pentru introducerea unei parole master, care urmează a fi criptată şi stocată folosind Shared Preferences (Fig. 1).

În cazul în care utilizatorul a setat anterior o parolă master, Password Manager îl va întâmpina cu mesajul corespunzător, cerându-i introducerea parolei (Fig. 2). Aceasta va fi comparată cu cea existentă deja în Shared Preferences, pentru a putea valida autenticitatea utilizatorului. Compararea se va face între versiunile criptate ale parolelor.

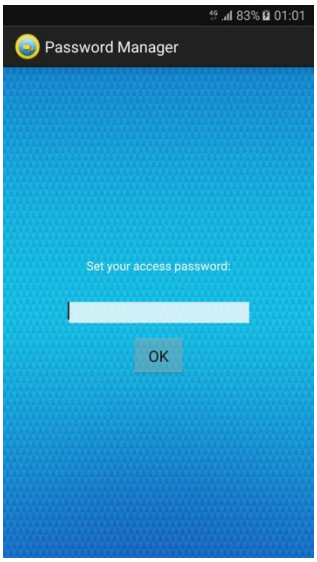

Fig. 1

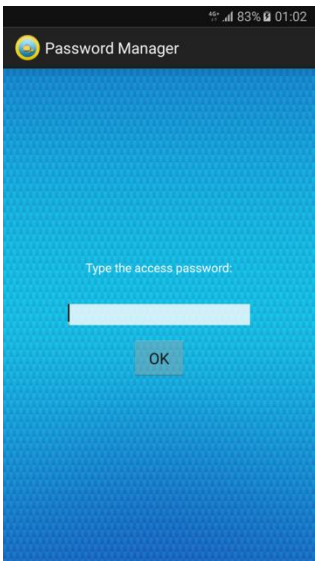

Fig. 2

Dupa introducerea parolei cu succes, utilizatorul va ajunge la cea de-a doua activitate (Fig. 3), unde va fi afişată lista conturilor memorate. Tot în acest screen, aplicaţia conţine 4 butoane pentru diferite operaţii:

- buton pentru adăugarea conturilor noi
- buton pentru ştergerea unui cont selectat, existent în listă
- buton pentru afişarea credenţialelor unui cont selectat din listă
- buton pentru resetarea parolei master

Pentru resetarea parolei, butonul Reset va duce utilizatorul la activitatea de login, unde i se va cere introducerea noii parole master (Fig. 1).

Credenţialele vor putea fi afişate prin selectarea unui cont din listă şi folosirea butonului Details (Fig. 4). Aplicaţia va returna username-ul şi versiunea necriptată a parolei contului.

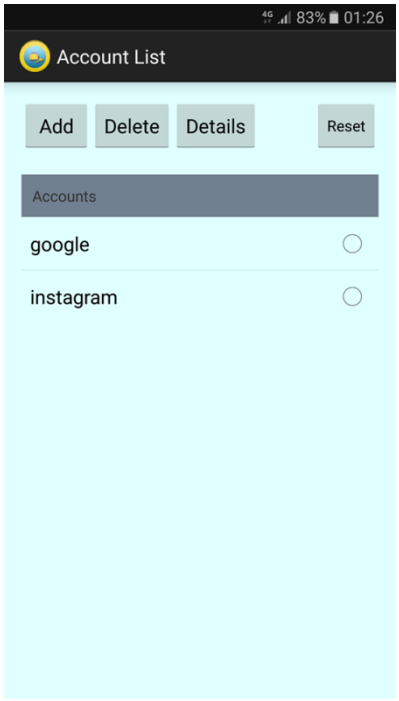

Fig. 3

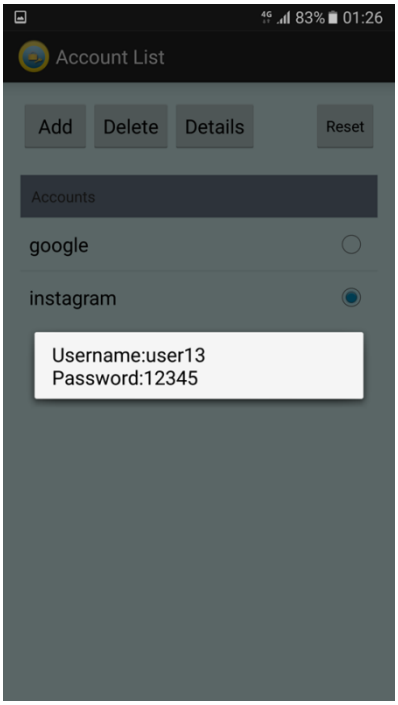

Fig. 4

Pentru a adăuga un cont nou, se va apăsa butonul Add, care va duce la următoarea activitate a aplicaţiei (Fig. 5). În această activitate, utilizatorul va trebui să introducă numele/domeniul contului (facebook, google, etc.), username-ul şi parola contului. În momentul în care se va apăsa butonul Done, contul va fi salvat, iar utilizatorul va fi întors în activitatea principală, unde va putea găsi, prin lista conturilor, şi noul cont creat.

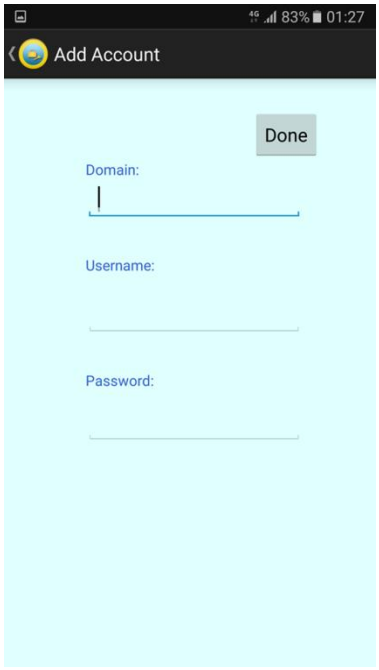

Fig. 5

3. Implementare

Pentru partea de memorare a datelor s-a construit un tabel într-o bază de date de tip SQLite, pusă la dispoziţie de Android Studio. Tabelul conţine 3 coloane: domain, username, password:

```
private static class Table {
    private static final String TAB_NAME = "account_table";
    private static final String COL_ID = "id";
    private static final String COL_DOMAIN = "domain";
    private static final String COL_USER = "username";
    private static final String COL_PASSWORD = "password";
}


public void onCreate(SQLiteDatabase db) {
    // creates the database
    db.execSQL(String.format(sql_create, Table.TAB_NAME, Table.COL_ID,
            Table.COL_DOMAIN, Table.COL_USER, Table.COL_PASSWORD));

}
```

Fiecare intrare din tabel, reprezintă o structură de date de tip WebAccount. În momentul afişării activităţii principale, ce afişează lista conturilor, se va face o interogare pe tabelul creat, intrările tabelului fiind stocate într-un ArrayList de tip WebAccount:

```
public WebAccount(String domain, String user, String password)
{
    super();
    this.domain = domain;
    this.user = user;
    this.password = password;
}

ArrayList<WebAccount> AccList = new ArrayList<WebAccount>();

model = new AccList();
//create our database helper
dbHelper = new ListDB(this);
//link model to database
model.addObserver(dbHelper);
//load the model from the database
model.setList(dbHelper.loadList());
```

Pentru partea de criptare a parolelor s-a folosit algoritmul simetric AES-128. Pentru procesul de criptare şi decriptare al parolelor, acest altgoritm are nevoie de un vector de iniţializare, construit pe baza unei chei simetrice, în cazul Password Manager, aceasta fiind hardcodată. Parola introdusă, împreună cu cheia simetrică, vor fi folosite în bytes pentru operaţiile succesive necesare criptării şi decriptării:

```
public class StringEncrypter {
    public static String encrypt(String seed, String cleartext) throws
Exception {
        byte[] rawKey = getRawKey(seed.getBytes());
        byte[] result = encrypt(rawKey, cleartext.getBytes());
        return toHex(result);
    }


private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception {
    SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
    byte[] encrypted = cipher.doFinal(clear);
    return encrypted;
}


encryptedpass = StringEncrypter.encrypt("passwd", pass);
```

4. Concluzii

În concluzie, aplicaţia Android, Password Manager, pune la dispoziţie posibilitatea de stocare în mod criptat a credenţialelor conturilor utilizatorului, cu ajutorul algoritmului de criptare simetric AES, folosit pe 128 de biţi, acesta fiind unul dintre cei mai siguri algoritmi la ora actuală.