Ebenezer Isaac

2020178014

Mobile Application Development – Exercise 5 revised with firebase

MainActivity.java

```
package com.mycrolinks.passwdmgr;

import android.content.Intent;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import android.view.View;
import android.widget.Toast;

import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.android.gms.auth.api.signin.GoogleSignInClient;
import com.google.android.gms.auth.api.signin.GoogleSignInOptions;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.GoogleAuthProvider;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.Map;
import java.util.Objects;

public class MainActivity extends AppCompatActivity {
    GoogleSignInClient mGoogleSignInClient;
    private static final int RC_SIGN_IN = 1000;
    private FirebaseAuth mAuth;
    FirebaseUser currentUser;
    FirebaseFirestore db;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mAuth = FirebaseAuth.getInstance();
        // Configure Google Sign In
        GoogleSignInOptions gso = new
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
                .requestIdToken(getString(R.string.default_web_client_id))
                .requestEmail()
                .build();

        mGoogleSignInClient = GoogleSignIn.getClient(this, gso);
        db = FirebaseFirestore.getInstance();
    }
```

```
    @SuppressWarnings("deprecation")
    public void sign(View v) {
        Toast.makeText(this, "Signing you in", Toast.LENGTH_SHORT).show();
        Intent signInIntent = mGoogleSignInClient.getSignInIntent();
        startActivityForResult(signInIntent, RC_SIGN_IN);
    }

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent
data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == RC_SIGN_IN) {
            Task<GoogleSignInAccount> task =
GoogleSignIn.getSignedInAccountFromIntent(data);
            try {
                GoogleSignInAccount account =
task.getResult(ApiException.class);
                firebaseAuthWithGoogle(account.getIdToken());
            } catch (ApiException e) {
                Toast.makeText(this, e.getMessage(),
Toast.LENGTH_SHORT).show();
            }
        }
    }

//    @Override
//    public void onStart() {
//        super.onStart();
//        currentUser = mAuth.getCurrentUser();
//        if (currentUser != null) {
//            vaultRedirect();
//        }
//    }

    private void firebaseAuthWithGoogle(String idToken) {
        AuthCredential credential = GoogleAuthProvider.getCredential(idToken,
null);
        mAuth.signInWithCredential(credential)
                .addOnCompleteListener(this, task -> {
                    if (task.isSuccessful()) {
                        System.out.println("Logged in");
                        currentUser = mAuth.getCurrentUser();
                        vaultRedirect();
                    } else {
                        Toast.makeText(MainActivity.this, "Sorry,
Authentication Failed", Toast.LENGTH_SHORT).show();
                    }
                });
    }

    private void vaultRedirect() {

db.collection("root").document(Objects.requireNonNull(currentUser.getEmail())
)
                .get()
                .addOnCompleteListener(task -> {
                    if (task.isSuccessful()) {
```

```
                        System.out.println("Got db");
                        Map<String, Object> map_data =
Objects.requireNonNull(task.getResult()).getData();
                        try {
                            String password = null;
                            for (Map.Entry<String, Object> entry :
Objects.requireNonNull(map_data).entrySet()) {
                                password = (String) entry.getValue();
                            }
                            Intent intent = new
Intent(getApplicationContext(), vault_pin.class);
                            intent.putExtra("data",password);
                            startActivity(intent);
                        } catch (Exception e) {
                            Intent intent = new
Intent(getApplicationContext(), create_vault.class);
                            startActivity(intent);
                        }
                    } else {
                        System.out.println("Database Failed");
                    }
                });
    }
}
```

create_vault.java

```
package com.mycrolinks.passwdmgr;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;


import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.firebase.firestore.FirebaseFirestore;
import androidx.appcompat.app.AppCompatActivity;
import java.io.IOException;
import java.security.GeneralSecurityException;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;
import java.util.Objects;

public class create_vault extends AppCompatActivity {
    EditText pass_new, pass_conf;
```

```
    Button create;
    FirebaseFirestore db;
    GoogleSignInAccount signInAccount;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_vault);
        pass_new = findViewById(R.id.pass_new);
        pass_conf = findViewById(R.id.pass_conf);
        create = findViewById(R.id.create);
        create.setOnClickListener(view -> {
            if
(pass_new.getText().toString().equals(pass_conf.getText().toString())) {
                String password =
Encryptor.hash_wrapper(pass_new.getText().toString());
                Map<String, Object> passwd = new HashMap<>();
                passwd.put("password", password);
                signInAccount = GoogleSignIn.getLastSignedInAccount(this);
                db = FirebaseFirestore.getInstance();


db.collection("root").document(Objects.requireNonNull(signInAccount.getEmail(
))).set(passwd)
                        .addOnSuccessListener(aVoid -> {
                                    Toast.makeText(this, "Vault created
successfully", Toast.LENGTH_SHORT).show();
                                    try {
                                        ArrayList<PasswordItem> arrayList =
new ArrayList<>();
                                        Date today =
Calendar.getInstance().getTime();
                                        @SuppressLint("SimpleDateFormat")
DateFormat df = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
                                        PasswordItem dummy= new
PasswordItem("Dummy", "example password", df.format(today));
                                        arrayList.add(dummy);

Encryptor.save(Objects.requireNonNull(signInAccount.getId()),
Encryptor.passwordListToByteArray(arrayList), getApplicationContext());
                                        Intent intent = new
Intent(getApplicationContext(), vault.class);
                                        startActivity(intent);
                                    } catch (GeneralSecurityException |
IOException e) {
                                        e.printStackTrace();
                                    }

                                }
                        )
                        .addOnFailureListener(e -> System.out.println("Error
writing document" + e.getMessage()));


            } else {
                Toast.makeText(this, "Passwords Didn't Match",
Toast.LENGTH_SHORT).show();
```

```
            }
        });

    }
}
```

CustomAdapter.java

```
package com.mycrolinks.passwdmgr;

import android.annotation.SuppressLint;
import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.Context;
import android.content.Intent;
import android.database.DataSetObserver;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.EditText;
import android.widget.ListAdapter;
import android.widget.TextView;
import android.widget.Toast;


import androidx.appcompat.app.AlertDialog;

import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;

class CustomAdapter implements ListAdapter {
    ArrayList<PasswordItem> arrayList;
    Context context;
    GoogleSignInAccount signInAccount;
    DateFormat df;
    @SuppressLint("SimpleDateFormat")
    public CustomAdapter(Context context, ArrayList<PasswordItem> arrayList)
{
        this.arrayList=arrayList;
        this.context=context;
        signInAccount = GoogleSignIn.getLastSignedInAccount(context);
        df = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
    }
    @Override
    public boolean areAllItemsEnabled() {
        return false;
    }
    @Override
```

```
    public boolean isEnabled(int position) {
        return true;
    }
    @Override
    public void registerDataSetObserver(DataSetObserver observer) {
    }
    @Override
    public void unregisterDataSetObserver(DataSetObserver observer) {
    }
    @Override
    public int getCount() {
        return arrayList.size();
    }
    @Override
    public Object getItem(int position) {
        return position;
    }
    @Override
    public long getItemId(int position) {
        return position;
    }
    @Override
    public boolean hasStableIds() {
        return false;
    }
    @SuppressLint({"InflateParams", "SetTextI18n"})
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        PasswordItem pwdItem=arrayList.get(position);
        if(convertView==null) {
            LayoutInflater layoutInflater = LayoutInflater.from(context);
            convertView=layoutInflater.inflate(R.layout.list_row, null);
            TextView title=convertView.findViewById(R.id.list_title);
            title.setText(pwdItem.title);
            convertView.setOnClickListener(v -> {
                final View alertBox =
layoutInflater.inflate(R.layout.password_item_dialog, null);
                AlertDialog.Builder alertDialog = new
AlertDialog.Builder(context);
                alertDialog.setTitle("Credentials");
                alertDialog.setView(alertBox);
                EditText itemTitle = (EditText)
alertBox.findViewById(R.id.alert_title);
                itemTitle.setText(pwdItem.title);
                EditText itemPwd = (EditText)
alertBox.findViewById(R.id.alert_password);
                itemPwd.setText(pwdItem.password);
                TextView itemTime = (TextView)
alertBox.findViewById(R.id.alert_time);
                itemTime.setText("Last Modified : " + pwdItem.timestamp);
                alertDialog.setIcon(R.drawable.lock_key);
                alertDialog.setPositiveButton("Save",
                        (dialog, which) -> {
                            Date today = Calendar.getInstance().getTime();
                            PasswordItem new_pwdItem = new
PasswordItem(itemTitle.getText().toString(),itemPwd.getText().toString(),df.f
ormat(today));
```

```
                                PasswordItemWrapper.update(pwdItem, new_pwdItem,
signInAccount.getId(), context);
                                Toast.makeText(context, "Credentials Saved",
Toast.LENGTH_SHORT).show();
                                Intent intent = new Intent(context, vault.class);
                                context.startActivity(intent);
                            });
                    alertDialog.setNeutralButton("Copy",
                            (dialog, which) -> {
                                ClipboardManager clipboard = (ClipboardManager)
context.getSystemService(Context.CLIPBOARD_SERVICE);
                                ClipData clip =
ClipData.newPlainText(pwdItem.title, pwdItem.password);
                                clipboard.setPrimaryClip(clip);
                                dialog.cancel();
                                Toast.makeText(context, "Password Copied to
Clipboard", Toast.LENGTH_SHORT).show();
                            });
                    alertDialog.setNegativeButton("Delete",
                            (dialog, which) -> {
                                PasswordItemWrapper.delete(pwdItem,
signInAccount.getId(), context);
                                Toast.makeText(context, "Credentials Deleted",
Toast.LENGTH_SHORT).show();
                                Intent intent = new Intent(context, vault.class);
                                context.startActivity(intent);
                            });
                    alertDialog.show();
                });

        }
        return convertView;
    }
    @Override
    public int getItemViewType(int position) {
        return position;
    }
    @Override
    public int getViewTypeCount() {
        return arrayList.size();
    }
    @Override
    public boolean isEmpty() {
        return false;
    }
}
```

Encryptor.java

```
package com.mycrolinks.passwdmgr;

import android.content.Context;

import com.google.crypto.tink.Aead;
```

```
import com.google.crypto.tink.CleartextKeysetHandle;
import com.google.crypto.tink.JsonKeysetReader;
import com.google.crypto.tink.JsonKeysetWriter;
import com.google.crypto.tink.KeyTemplates;
import com.google.crypto.tink.KeysetHandle;
import com.google.crypto.tink.aead.AeadConfig;

import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.security.GeneralSecurityException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;


public class Encryptor {

    private static String bytesToHexString(byte[] bytes) {
        StringBuilder sb = new StringBuilder();
        for (byte aByte : bytes) {
            String hex = Integer.toHexString(0xFF & aByte);
            if (hex.length() == 1) {
                sb.append('0');
            }
            sb.append(hex);
        }
        return sb.toString();
    }

    private static String hash(String password, String salt) {
        MessageDigest digest;
        String hash;
        try {
            digest = MessageDigest.getInstance("SHA-256");
            digest.update((password + salt).getBytes());
            hash = bytesToHexString(digest.digest());
            return hash;
        } catch (NoSuchAlgorithmException e1) {
            e1.printStackTrace();
        }
        return null;
    }

    public static String hash_wrapper(String password) {
        String hash = password;
        for (int x = 0; x < password.length(); x++) {
            hash = hash(hash, password.charAt(x) + "");
        }
        return hash;
    }
```

```
    public static byte[] read(String id, Context applicationContext) throws
GeneralSecurityException, IOException {
        AeadConfig.register();
        KeysetHandle keyHandle =
CleartextKeysetHandle.read(JsonKeysetReader.withInputStream(applicationContex
t.openFileInput(id + ".json")));
        Aead aead = keyHandle.getPrimitive(Aead.class);
        byte[] ciphertext = new byte[0];
        try {
            FileInputStream fis=  applicationContext.openFileInput(id +
".bin");
            ObjectInputStream in = new ObjectInputStream(fis);
            ciphertext = (byte[]) in.readObject();
            in.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
            e.printStackTrace();
        }
        return aead.decrypt(ciphertext, id.getBytes());
    }

    public static void save(String id, byte[] data, Context
applicationContext) throws GeneralSecurityException, IOException {
        AeadConfig.register();
        KeysetHandle keyHandler;
        keyHandler =
KeysetHandle.generateNew(KeyTemplates.get("AES128_GCM"));
        File dir = applicationContext.getFilesDir();
        File file = new File(dir, id + ".json");
        System.out.println(file.delete());
        file = new File(dir, id + ".bin");
        System.out.println(file.delete());
        CleartextKeysetHandle.write(keyHandler,
JsonKeysetWriter.withOutputStream(applicationContext.openFileOutput(id +
".json", Context.MODE_PRIVATE)));
        Aead aead = keyHandler.getPrimitive(Aead.class);

        byte[] ciphertext = aead.encrypt(data, id.getBytes());
        try {
            FileOutputStream fileOut = applicationContext.openFileOutput(id +
".bin", Context.MODE_PRIVATE);
            ObjectOutputStream out = new ObjectOutputStream(fileOut);
            out.writeObject(ciphertext);
            out.flush();
            out.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    public static byte[] passwordListToByteArray(ArrayList<PasswordItem>
data) throws IOException {
        ByteArrayOutputStream byteOut = new ByteArrayOutputStream();
        ObjectOutputStream out = new ObjectOutputStream(byteOut);
        out.writeObject(data);
        return byteOut.toByteArray();
```

```
    }

    public static ArrayList<PasswordItem> byteArrayToPasswordList(byte[]
data) throws IOException, ClassNotFoundException {
        ByteArrayInputStream byteIn = new ByteArrayInputStream(data);
        ObjectInputStream in = new ObjectInputStream(byteIn);
        return  (ArrayList<PasswordItem>) in.readObject();
    }
}
```

PasswordItem.java

```
package com.mycrolinks.passwdmgr;

import java.io.Serializable;

public class PasswordItem implements Serializable {
    String title;
    String password;
    String timestamp;
    public PasswordItem(String title, String password, String timestamp) {
        this.title = title;
        this.password = password;
        this.timestamp = timestamp;
    }

    public void display() {
        System.out.println("title : "+title);
        System.out.println("password : "+password);
        System.out.println("timestamp : "+timestamp);

    }
}
```

PasswordItemWrapper.java

```
package com.mycrolinks.passwdmgr;

import android.content.Context;

import java.io.IOException;
import java.security.GeneralSecurityException;
import java.util.ArrayList;

public class PasswordItemWrapper {

    public static void delete(PasswordItem obj,String id, Context context){
        try {
            ArrayList<PasswordItem> arrayList =
Encryptor.byteArrayToPasswordList(Encryptor.read(id, context));
            for(int index =0; index<arrayList.size();index++){
                PasswordItem dummy = arrayList.get(index);
```

```
                if(dummy.title.equals(obj.title)){
                    arrayList.remove(index);
                    break;
                }
            }
            Encryptor.save(id,
Encryptor.passwordListToByteArray(arrayList),context);
        } catch (IOException | ClassNotFoundException |
GeneralSecurityException e) {
            e.printStackTrace();
        }
    }

    public static void add(PasswordItem obj,String id, Context context){
        System.out.println("add");
        try {
            ArrayList<PasswordItem> arrayList =
Encryptor.byteArrayToPasswordList(Encryptor.read(id, context));
            System.out.println(arrayList.size());
            arrayList.add(obj);
            System.out.println(arrayList.size());
            Encryptor.save(id,
Encryptor.passwordListToByteArray(arrayList),context);
        } catch (IOException | ClassNotFoundException |
GeneralSecurityException e) {
            e.printStackTrace();
        }
        System.out.println("add over");
    }

    public static void update(PasswordItem old_obj, PasswordItem
new_obj,String id, Context context){
        try {
            ArrayList<PasswordItem> arrayList =
Encryptor.byteArrayToPasswordList(Encryptor.read(id, context));
            for(int index =0; index<arrayList.size();index++){
                PasswordItem dummy = arrayList.get(index);
                if(dummy.title.equals(old_obj.title)){
                    arrayList.remove(index);
                    arrayList.add(index,new_obj);
                    break;
                }
            }
            Encryptor.save(id,
Encryptor.passwordListToByteArray(arrayList),context);
        } catch (IOException | ClassNotFoundException |
GeneralSecurityException e) {
            e.printStackTrace();
        }
    }
}
```

vault.java

```java
package com.mycrolinks.passwdmgr;

import android.annotation.SuppressLint;
import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.FirebaseFirestore;

import java.io.File;
import java.io.IOException;
import java.security.GeneralSecurityException;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.Objects;

public class vault extends AppCompatActivity {

    GoogleSignInAccount signInAccount;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        //System.out.println("hello");
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_vault);
        signInAccount = GoogleSignIn.getLastSignedInAccount(this);
        final ListView list = findViewById(R.id.list);
        try {
            ArrayList<PasswordItem> arrayList =
Encryptor.byteArrayToPasswordList(Encryptor.read(signInAccount.getId(),
getApplicationContext()));
            for (int i = 0; i < arrayList.size(); i++) {
                PasswordItem temp = arrayList.get(i);
                temp.display();
            }
            CustomAdapter customAdapter = new CustomAdapter(this, arrayList);
            list.setAdapter(customAdapter);
        } catch (IOException | ClassNotFoundException |
GeneralSecurityException e) {
            e.printStackTrace();
```

```
        }
        Button exit = (Button) findViewById(R.id.exit);
        exit.setOnClickListener(v -> {
            FirebaseAuth.getInstance().signOut();
            Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
            startActivity(intent);
        });
        Button add = (Button) findViewById(R.id.add);
        add.setOnClickListener(v -> {
            LayoutInflater layoutInflater = LayoutInflater.from(this);
            final View alertBox =
layoutInflater.inflate(R.layout.password item dialog, null);
            AlertDialog.Builder alertDialog = new AlertDialog.Builder(this);
            alertDialog.setTitle("Credentials");
            alertDialog.setView(alertBox);
            EditText itemTitle = (EditText)
alertBox.findViewById(R.id.alert_title);
            EditText itemPwd = (EditText)
alertBox.findViewById(R.id.alert password);
            alertDialog.setIcon(R.drawable.lock_key);
            alertDialog.setPositiveButton("Save",
                    (dialog, which) -> {
                        Date today = Calendar.getInstance().getTime();
                        @SuppressLint("SimpleDateFormat") DateFormat df = new
SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
                        PasswordItem pwdItem = new
PasswordItem(itemTitle.getText().toString(), itemPwd.getText().toString(),
df.format(today));
                        PasswordItemWrapper.add(pwdItem,
signInAccount.getId(), this);
                        Toast.makeText(this, "Credentials Saved",
Toast.LENGTH_SHORT).show();
                        Intent intent = new Intent(this, vault.class);
                        this.startActivity(intent);
                    });
            alertDialog.setNeutralButton("Copy",
                    (dialog, which) -> {

                        ClipboardManager clipboard = (ClipboardManager)
this.getSystemService(Context.CLIPBOARD_SERVICE);
                        ClipData clip =
ClipData.newPlainText(itemTitle.getText().toString(),
itemPwd.getText().toString());
                        clipboard.setPrimaryClip(clip);
                        dialog.cancel();
                        Toast.makeText(this, "Password Copied to Clipboard",
Toast.LENGTH_SHORT).show();
                    });
            alertDialog.show();

        });
        Button delete = (Button) findViewById(R.id.del_but);
        delete.setOnClickListener(view -> {
            String id = signInAccount.getId();
            File dir = this.getFilesDir();
            File file = new File(dir, id + ".json");
```

```
            file.delete();
            file = new File(dir, id + ".bin");
            file.delete();
            FirebaseFirestore db = FirebaseFirestore.getInstance();

db.collection("root").document(Objects.requireNonNull(signInAccount.getEmail(
)))
                    .delete()
                    .addOnSuccessListener(aVoid -> {
                        Toast.makeText(vault.this, "Vault successfully
deleted!", Toast.LENGTH_SHORT).show();
                        exit.performClick();
                    })
                    .addOnFailureListener(e -> {
                        Toast.makeText(vault.this, "Error deleting vault",
Toast.LENGTH_SHORT).show();
                        System.out.println(e.getMessage());
                    });

        });
    }
}
```

vault_pin.java

```
package com.mycrolinks.passwdmgr;

//import android.Manifest;
import android.annotation.SuppressLint;
import android.content.Intent;
import android.location.Location;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.android.gms.location.LocationListener;
import com.google.firebase.auth.FirebaseAuth;

//import pub.devrel.easypermissions.AfterPermissionGranted;
//import pub.devrel.easypermissions.EasyPermissions;

public class vault_pin extends AppCompatActivity implements LocationListener
{
    //private final int REQUEST_LOCATION_PERMISSION = 1;
    GoogleSignInAccount signInAccount;
    String[] loc;
    String password_db;
    Button unlock, logout, delete;
```

```
    @SuppressLint("MissingPermission")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_vault_pin);
        unlock = findViewById(R.id.signin);
        logout = findViewById(R.id.log_but);
        delete = findViewById(R.id.del_but);
        signInAccount = GoogleSignIn.getLastSignedInAccount(this);
        password_db = getIntent().getStringExtra("data");
        logout.setOnClickListener(view -> {
            FirebaseAuth.getInstance().signOut();
            Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
            startActivity(intent);
        });
        unlock.setOnClickListener(view -> {
            EditText passwordEditText = findViewById(R.id.password_editText);
            String password = passwordEditText.getText().toString();
            if (Encryptor.hash_wrapper(password).equals(password_db)) {
                Intent intent = new Intent(getApplicationContext(),
vault.class);
                startActivity(intent);
            } else {
                Toast.makeText(this, "Wrong Password",
Toast.LENGTH_SHORT).show();
            }
        });

        //requestLocationPermission();
        //LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);

//locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0,
this);

    }

//    @Override
//    public void onRequestPermissionsResult(int requestCode, @NonNull
String[] permissions, @NonNull int[] grantResults) {
//        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
//
//        // Forward results to EasyPermissions
//        EasyPermissions.onRequestPermissionsResult(requestCode,
permissions, grantResults, this);
//    }
//
//    @AfterPermissionGranted(REQUEST_LOCATION_PERMISSION)
//    public void requestLocationPermission() {
//        String[] perms = {Manifest.permission.ACCESS_FINE_LOCATION};
//        if (EasyPermissions.hasPermissions(this, perms)) {
//            Toast.makeText(this, "Permission already granted",
Toast.LENGTH_SHORT).show();
//        } else {
```

```
//            EasyPermissions.requestPermissions(this, "Please grant the
location permission", REQUEST_LOCATION_PERMISSION, perms);
//        }
//    }

    @Override
    public void onLocationChanged(@NonNull Location location) {
        loc[0] = String.valueOf(location.getLatitude());
        loc[1] = String.valueOf(location.getLongitude());
        System.out.println(loc[0]);
        System.out.println(loc[1]);

    }
}
```

activity_create_vault.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".create_vault"
    android:layout_margin="10dp">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="right"
        android:orientation="horizontal">

        <Switch
            android:id="@+id/switch1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Location"
            tools:ignore="UseSwitchCompatOrMaterialXml"
            android:layout_margin="10dp"/>
        <Button
            android:id="@+id/create"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Create Vault"
            android:layout_margin="10dp"/>
    </LinearLayout>
    <Space
        android:layout_width="wrap_content"
        android:layout_height="20dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_gravity="center"
        android:text="Create Vault"
        android:textSize="20sp" />

    <Space
        android:layout width="wrap content"
        android:layout_height="20dp" />

    <ImageView
        android:layout width="150dp"
        android:layout_height="150dp"
        android:layout_gravity="center"
        android:src="@drawable/lock key" />

    <Space
        android:layout_width="wrap_content"
        android:layout_height="20dp" />
    <EditText
        android:id="@+id/pass_new"
        android:layout width="325dp"
        android:layout_height="wrap_content"
        android:hint="New Password"
        android:layout_gravity="center"
        android:inputType="textPassword"
        />
    <Space
        android:layout width="wrap content"
        android:layout_height="20dp" />
    <EditText
        android:id="@+id/pass_conf"
        android:layout_width="325dp"
        android:layout_height="wrap_content"
        android:hint="Confirm Password"
        android:layout_gravity="center"
        android:inputType="textPassword" />
</LinearLayout>
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout height="match parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="514dp"
```

```
        android:gravity="center"
        android:onClick="sign"
        android:background="@color/colorPrimary"
        android:text="Sign In with Google"
        />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        app:srcCompat="@drawable/fui_ic_googleg_color_24dp" />

</RelativeLayout>
```

Activity_vault.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingLeft="16dp"
    android:paddingTop="16dp"
    android:paddingRight="16dp"
    android:paddingBottom="16dp"
    tools:context=".vault">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="right"
        android:orientation="horizontal">

        <Button
            android:id="@+id/del_but"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:layout_gravity="right"
            android:text="Delete Vault" />

        <Button
            android:id="@+id/add"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="right"
            android:layout_margin="10dp"
            android:text="Add" />

        <Button
```

```
            android:id="@+id/exit"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="right"
            android:layout_margin="10dp"
            android:text="Exit" />

    </LinearLayout>

    <ListView
        android:id="@+id/list"
        android:layout_width="wrap_content"
        android:layout_height="600dp"
        android:divider="#000"
        android:dividerHeight="1dp"
        android:footerDividersEnabled="false"
        android:headerDividersEnabled="false"
        android:isScrollContainer="true"
        android:scrollbarAlwaysDrawVerticalTrack="true" />


</LinearLayout>
```

Activity_vault_pin.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".vault_pin">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:gravity="right"
        android:orientation="horizontal">

        <Button
            android:id="@+id/log_but"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:text="Logout" />

        <Button
            android:id="@+id/signin"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:text="Unlock" />
```

```
    </LinearLayout>

    <Space
        android:layout_width="wrap_content"
        android:layout_height="20dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Master Password"
        android:textSize="20sp" />

    <Space
        android:layout_width="wrap_content"
        android:layout_height="20dp" />

    <ImageView
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_gravity="center"
        android:src="@drawable/lock_key" />

    <Space
        android:layout_width="wrap_content"
        android:layout_height="20dp" />

    <EditText
        android:id="@+id/password_editText"
        android:layout_width="325dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:hint="Enter Master Password"
        android:inputType="textPassword" />

</LinearLayout>
```

List_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="5dip">
        <TextView
            android:id="@+id/list_title"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textSize="25sp"
            android:gravity="center"
             />
</LinearLayout>
```

Password_item_dialog.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <EditText android:inputType="textAutoComplete"
        android:textSize="20sp"
        android:id="@+id/alert_title"
        android:scrollHorizontally="true"
        android:layout_height="70dp"
        android:hint="Title / Username"
        android:layout_width="250dp"
        android:layout_gravity="center"
        android:autofillHints="username" />
    <EditText android:inputType="textVisiblePassword"
        android:textSize="20sp"
        android:id="@+id/alert_password"
        android:scrollHorizontally="true"
        android:layout_height="70dp"
        android:hint="Password"
        android:layout_width="250dp"
        android:layout_gravity="center"
        android:autofillHints="password" />
    <TextView
        android:textSize="20sp"
        android:id="@+id/alert_time"
        android:scrollHorizontally="true"
        android:layout_height="70dp"
        android:layout_width="250dp"
        android:layout_gravity="center"
        android:autofillHints="password" />

</LinearLayout>
```

Project Gradle

```
// Top-level build file where you can add configuration options common to all
sub-projects/modules.
buildscript {
    repositories {
        google()
        mavenCentral()
    }
    dependencies {
        classpath "com.android.tools.build:gradle:4.2.2"
        classpath 'com.google.gms:google-services:4.3.8'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
```

```
allprojects {
    repositories {
        google()
        mavenCentral()
            jcenter() // Warning: this repository is going to shut down soon
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Module Gradle

```
plugins {
    id 'com.android.application'
    id 'com.google.gms.google-services'
}

android {
    compileSdkVersion 30
    buildToolsVersion "30.0.3"

    defaultConfig {
        applicationId "com.mycrolinks.passwdmgr"
        minSdkVersion 28
        targetSdkVersion 30
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.3.0'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
```

```
    implementation platform('com.google.firebase:firebase-bom:28.2.1')
    implementation 'com.google.firebase:firebase-auth'
    implementation 'com.google.android.gms:play-services-auth:19.0.0'
    implementation 'com.firebaseui:firebase-ui-auth:7.2.0'
    implementation 'com.google.firebase:firebase-firestore'
    implementation 'com.google.crypto.tink:tink-android:1.6.0'
    implementation 'com.google.android.gms:play-services-location:18.0.0'
    implementation 'com.google.android.gms:play-services-safetynet:17.0.1'
    implementation 'pub.devrel:easypermissions:2.0.1'
}
```

Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.mycrolinks.passwdmgr">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS COARSE LOCATION"
/>
    <uses-permission
android:name="android.permission.ACCESS BACKGROUND LOCATION" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.PasswordManager">
        <activity android:name=".vault"></activity>
        <activity android:name=".create vault" />
        <activity android:name=".vault_pin" />
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```