Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных

Центр программной инженерии

ОТЧЕТ

к лабораторной работе №7 по дисциплине:

«Объектно-ориентированное программирование»

«Разработка Android-приложения»

наименование темы

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: |  | ИСТб-23-1 |  |  |  | Иржицкая А.А. |
|  |  | шифр группы |  | подпись |  | Фамилия ИО |
| Проверил: |  | Доцент |  |  |  | Аршинский В.Л. |
|  |  | должность |  | подпись |  | Фамилия ИО |

Иркутск 2025 г.

**Оглавление**

[1 Постановка задачи 3](#_Toc198728295)

[2 Проектирование графического интерфейса 3](#_Toc198728296)

[3 Проектирование классов 5](#_Toc198728297)

[4 Проектирование тестов 6](#_Toc198728298)

[5 Код программы 7](#_Toc198728299)

[Заключение 16](#_Toc198728300)

[Список использованных источников 17](#_Toc198728301)

**1 Постановка задачи**

**Цель работы:** Разработка Android-приложения для хранения PIN-кодов банковских карт с защитой паролем.

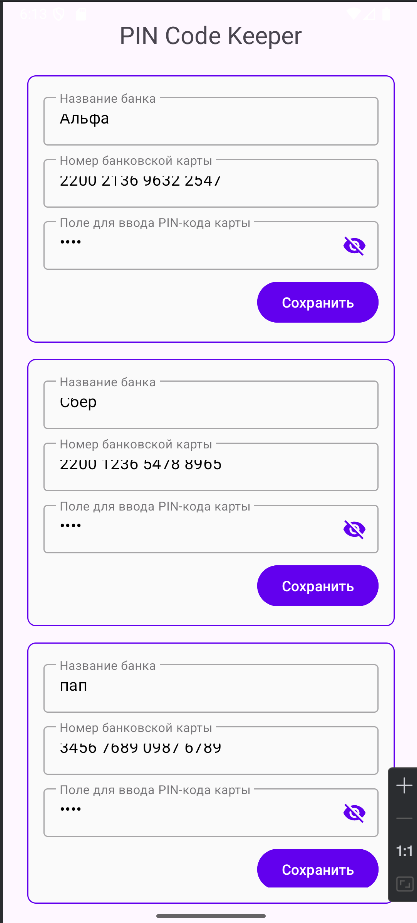
**Задачи:**

1. Реализовать безопасное хранение PIN-кодов карт.
2. Обеспечить защиту данных с помощью пароля.
3. Разработать удобный интерфейс для ввода и отображения информации о картах.
4. Реализовать форматирование номера карты для удобства ввода.

**Индивидуальное задание:**

"Вторая память". Приложение для хранения пинкодов от банковских карт (для 3 карт). Хранение данных может осуществляться в SharedPreferences, вход в приложение по паролю.

**2 Проектирование графического интерфейса**

****

**Рисунок 1 – Главный экран приложения**

Описание элементов интерфейса:

1. Поля для ввода данных:

* Название банка.
* Номер карты (с автоматическим форматированием).
* PIN-код (с возможностью скрытия/отображения).

1. Кнопка "Сохранить":

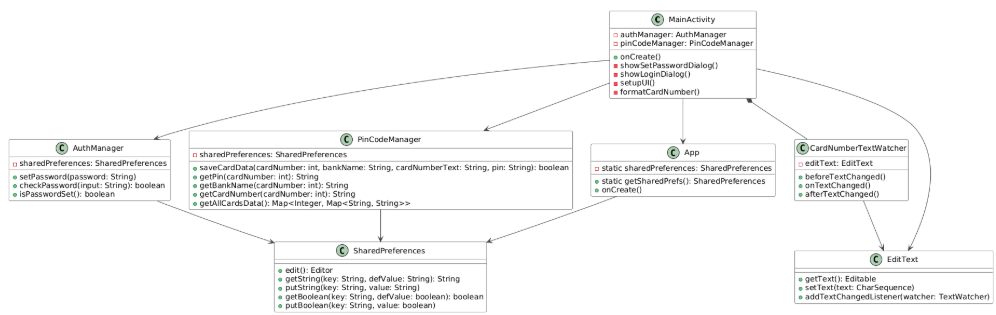
* Сохраняет введенные данные карты.

1. Система аутентификации:

* Диалоговые окна для установки и ввода пароля.

**3 Проектирование классов**

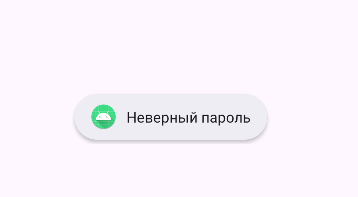
На рисунке 2 представлена UML диаграмма классов.

****

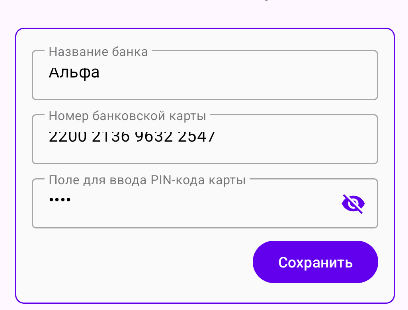
**Рисунок 2 -** UML диаграмма классов

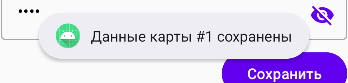
**4 Проектирование тестов**

|  |  |  |
| --- | --- | --- |
| **№** | **Действие** | **Ожидаемый результат** |
| **1** | Ввод неверного пароля | Сообщение об ошибке |
| **2** | Ввод данных карты и сохранение | Данные сохраняются и отображаются корректно |
| **3** | Проверка форматирования номера карты | Номер карты отображается в формате XXXX XXXX XXXX XXXX |

****

**Рисунок 3 – Результат теста №1**

****

****

**Рисунок 4,5 – Результаты теста №2**

****

**Рисунок 6 – Результат теста №3**

**5 Код программы**

**Класс AuthManager**

package com.example.a7laba.managers;  
  
import android.content.SharedPreferences;  
  
public class AuthManager {  
 private static final String *PASSWORD\_KEY* = "app\_password";  
 private final SharedPreferences sharedPreferences;  
 /\* public AuthManager() {  
 this.sharedPreferences = App.getSharedPreferences();  
 }\*/  
 public AuthManager(SharedPreferences sharedPreferences) {  
 this.sharedPreferences = sharedPreferences;  
 }  
  
 //установка пароля  
 public void setPassword(String password) {  
 sharedPreferences.edit().putString(*PASSWORD\_KEY*, password).apply();  
 }  
 //проверка пароля  
 public boolean checkPassword(String input) {  
 String savedPassword = sharedPreferences.getString(*PASSWORD\_KEY*, null);  
 return input.equals(savedPassword);  
 }  
 //проверка наличия парола  
 public boolean isPasswordSet() {  
 return sharedPreferences.contains(*PASSWORD\_KEY*);  
 }  
}

**Класс PinCodeManager**

package com.example.a7laba.managers;  
  
import android.content.SharedPreferences;  
import java.util.HashMap;  
import java.util.Map;  
  
public class PinCodeManager {  
 private static final String *CARD\_PREFIX* = "card\_";  
 private static final String *BANK\_PREFIX* = "bank\_";  
 private static final String *NUMBER\_PREFIX* = "number\_";  
 private static final int *MAX\_CARDS* = 3;  
 private final SharedPreferences sharedPreferences;  
  
 public PinCodeManager(SharedPreferences sharedPreferences) {  
 this.sharedPreferences = sharedPreferences;  
 }  
  
 public boolean saveCardData(int cardNumber, String bankName, String cardNumberText, String pin) {  
 if (cardNumber < 1 || cardNumber > *MAX\_CARDS*) return false;  
  
 SharedPreferences.Editor editor = sharedPreferences.edit();  
 editor.putString(*BANK\_PREFIX* + cardNumber, bankName);  
 editor.putString(*NUMBER\_PREFIX* + cardNumber, cardNumberText);  
 editor.putString(*CARD\_PREFIX* + cardNumber, pin);  
 editor.apply();  
 return true;  
 }  
  
 public String getPin(int cardNumber) {  
 if (cardNumber < 1 || cardNumber > *MAX\_CARDS*) return null;  
 return sharedPreferences.getString(*CARD\_PREFIX* + cardNumber, null);  
 }  
  
 public String getBankName(int cardNumber) {  
 if (cardNumber < 1 || cardNumber > *MAX\_CARDS*) return null;  
 return sharedPreferences.getString(*BANK\_PREFIX* + cardNumber, null);  
 }  
  
 public String getCardNumber(int cardNumber) {  
 if (cardNumber < 1 || cardNumber > *MAX\_CARDS*) return null;  
 return sharedPreferences.getString(*NUMBER\_PREFIX* + cardNumber, null);  
 }  
  
 public Map<Integer, Map<String, String>> getAllCardsData() {  
 Map<Integer, Map<String, String>> cardsData = new HashMap<>();  
 for (int i = 1; i <= *MAX\_CARDS*; i++) {  
 Map<String, String> cardData = new HashMap<>();  
 cardData.put("bank", getBankName(i));  
 cardData.put("number", getCardNumber(i));  
 cardData.put("pin", getPin(i));  
  
 if (cardData.get("bank") != null || cardData.get("number") != null || cardData.get("pin") != null) {  
 cardsData.put(i, cardData);  
 }  
 }  
 return cardsData;  
 }  
}

Класс **App**

package com.example.a7laba;  
  
import android.app.Application;  
import android.content.Context;  
import android.content.SharedPreferences;  
  
public class App extends Application {  
 private static SharedPreferences *sharedPreferences*;  
  
 public static SharedPreferences getSharedPrefs() {  
 return *sharedPreferences*;  
 }  
  
 @Override  
 public void onCreate() {  
 super.onCreate();  
 *sharedPreferences* = getSharedPreferences("PinCodeAppPrefs", Context.*MODE\_PRIVATE*);  
  
 }  
}

**Класс MainActivity**

package com.example.a7laba;  
  
  
import android.os.Bundle;  
import android.text.Editable;  
import android.text.TextWatcher;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.LinearLayout;  
import android.widget.Toast;  
import androidx.appcompat.app.AlertDialog;  
import androidx.appcompat.app.AppCompatActivity;  
import com.example.a7laba.managers.AuthManager;  
import com.example.a7laba.managers.PinCodeManager;  
import com.google.android.material.textfield.TextInputEditText;  
  
public class MainActivity extends AppCompatActivity {  
 private AuthManager authManager;  
 private PinCodeManager pinCodeManager;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 authManager = new AuthManager(App.*getSharedPrefs*());  
 pinCodeManager = new PinCodeManager(App.*getSharedPrefs*());  
  
 if (!authManager.isPasswordSet()) {  
 showSetPasswordDialog();  
 } else {  
 showLoginDialog();  
 }  
 }  
  
 private void showSetPasswordDialog() {  
 AlertDialog.Builder builder = new AlertDialog.Builder(this);  
 builder.setTitle("Установите пароль");  
  
 final EditText input = new EditText(this);  
 input.setHint("Введите пароль");  
 builder.setView(input);  
  
 builder.setPositiveButton("Установить", (dialog, which) -> {  
 String password = input.getText().toString();  
 if (!password.isEmpty()) {  
 authManager.setPassword(password);  
 setupUI();  
 }  
 });  
 builder.setNegativeButton("Отмена", (dialog, which) -> dialog.cancel());  
 builder.show();  
 }  
  
 private void showLoginDialog() {  
 AlertDialog.Builder builder = new AlertDialog.Builder(this);  
 builder.setTitle("Введите пароль");  
  
 final EditText input = new EditText(this);  
 input.setHint("Введите пароль");  
 builder.setView(input);  
  
 builder.setPositiveButton("Войти", (dialog, which) -> {  
 String password = input.getText().toString();  
 if (authManager.checkPassword(password)) {  
 setupUI();  
 } else {  
 Toast.*makeText*(MainActivity.this, "Неверный пароль", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 builder.setNegativeButton("Отмена", (dialog, which) -> finish());  
 builder.show();  
 }  
  
 private void setupUI() {  
 setContentView(R.layout.*activity\_main*);  
 LinearLayout cardsLayout = findViewById(R.id.*cards\_layout*);  
  
 for (int i = 1; i <= 3; i++) {  
 View cardView = getLayoutInflater().inflate(R.layout.*card\_item*, cardsLayout, false);  
  
 TextInputEditText bankNameEdit = cardView.findViewById(R.id.*bank\_name*);  
 TextInputEditText cardNumberEdit = cardView.findViewById(R.id.*card\_number*);  
 TextInputEditText pinCodeEdit = cardView.findViewById(R.id.*pin\_code*);  
 Button saveButton = cardView.findViewById(R.id.*save\_button*);  
  
 final int cardIndex = i;  
  
 // Устанавливаем TextWatcher для форматирования номера карты  
 cardNumberEdit.addTextChangedListener(new CardNumberTextWatcher(cardNumberEdit));  
  
 // Загружаем сохраненные данные  
 String savedBankName = pinCodeManager.getBankName(cardIndex);  
 String savedCardNumber = pinCodeManager.getCardNumber(cardIndex);  
 String savedPin = pinCodeManager.getPin(cardIndex);  
  
 if (savedBankName != null) bankNameEdit.setText(savedBankName);  
 if (savedCardNumber != null) {  
 cardNumberEdit.setText(formatCardNumber(savedCardNumber));  
 }  
 if (savedPin != null) pinCodeEdit.setText(savedPin);  
  
 saveButton.setOnClickListener(v -> {  
 String bankName = bankNameEdit.getText().toString().trim();  
 String cardNumber = cardNumberEdit.getText().toString().replaceAll(" ", "");  
 String pin = pinCodeEdit.getText().toString().trim();  
  
 if (bankName.isEmpty() || cardNumber.isEmpty() || pin.isEmpty()) {  
 Toast.*makeText*(MainActivity.this, "Заполните все поля", Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
  
 if (cardNumber.length() != 16) {  
 Toast.*makeText*(MainActivity.this, "Номер карты должен содержать 16 цифр", Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
  
 if (pin.length() != 4) {  
 Toast.*makeText*(MainActivity.this, "PIN-код должен содержать 4 цифры", Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
  
 pinCodeManager.saveCardData(cardIndex, bankName, cardNumber, pin);  
 Toast.*makeText*(MainActivity.this, "Данные карты #" + cardIndex + " сохранены", Toast.*LENGTH\_SHORT*).show();  
 });  
  
 cardsLayout.addView(cardView);  
 }  
 }  
  
 private class CardNumberTextWatcher implements TextWatcher {  
 private final EditText editText;  
 private boolean isFormatting;  
 private int previousLength;  
 private int addedSpaces;  
  
 public CardNumberTextWatcher(EditText editText) {  
 this.editText = editText;  
 }  
  
 @Override  
 public void beforeTextChanged(CharSequence s, int start, int count, int after) {  
 previousLength = s.length();  
 }  
  
 @Override  
 public void onTextChanged(CharSequence s, int start, int before, int count) {  
 addedSpaces = count > before ? count - before : 0;  
 }  
  
 @Override  
 public void afterTextChanged(Editable s) {  
 if (isFormatting) return;  
 isFormatting = true;  
  
 // Удаляем все пробелы и нецифровые символы  
 String original = s.toString().replaceAll("[^0-9]", "");  
 StringBuilder formatted = new StringBuilder();  
  
 // Форматируем с пробелами через каждые 4 цифры (максимум 16 цифр)  
 for (int i = 0; i < original.length() && i < 16; i++) {  
 if (i > 0 && i % 4 == 0) {  
 formatted.append(" ");  
 }  
 formatted.append(original.charAt(i));  
 }  
  
 // Обновляем текст, если он изменился  
 if (!s.toString().equals(formatted.toString())) {  
 s.replace(0, s.length(), formatted.toString());  
  
 // Корректируем позицию курсора  
 int cursorPos = editText.getSelectionEnd();  
 if (addedSpaces > 0) {  
 // При добавлении символа  
 if (cursorPos % 5 == 0 && cursorPos < formatted.length()) {  
 cursorPos++; // Перескакиваем через пробел  
 }  
 }  
 cursorPos = Math.*min*(cursorPos, formatted.length());  
 editText.setSelection(cursorPos);  
 }  
  
 isFormatting = false;  
 }  
 }  
  
 private String formatCardNumber(String cardNumber) {  
 if (cardNumber == null || cardNumber.isEmpty()) {  
 return "";  
 }  
  
 // Удаляем все пробелы и нецифровые символы  
 String digits = cardNumber.replaceAll("[^0-9]", "");  
 StringBuilder formatted = new StringBuilder();  
  
 // Форматируем с пробелами через каждые 4 цифры (максимум 16 цифр)  
 for (int i = 0; i < digits.length() && i < 16; i++) {  
 if (i > 0 && i % 4 == 0) {  
 formatted.append(" ");  
 }  
 formatted.append(digits.charAt(i));  
 }  
  
 return formatted.toString();  
 }  
}

****Заключение****

В ходе лабораторной работы я освоила базовые принципы разработки Android-приложений, включая проектирование интерфейса и реализацию логики взаимодействия с пользователем. Был получен практический опыт создания макетов экранов в Android Studio, использования контейнеров и вью-элементов, а также работы с SharedPreferences для сохранения данных между сеансами. Разработанное приложение «PIN Code Keeper» позволяет безопасно хранить данные банковских карт (название банка, номер карты и PIN-код) с защитой паролем.

****Список использованных источников****

1. Объектно-ориентированное программирование. Метод. указания по выполнению лабораторных работ / сост.: В.Л. Аршинский. – Иркутск: Изд-во ИРНИТУ, 2017. – 24 c