

Министерство образования и молодежной политики Свердловской области

ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по учебной практике

РМП задание 4

Выполнил: Лежень Андрей Игоревич

Группа: ПР-31

Преподаватель: Мирошниченко Г.В.

2024

Содержание

[1. **Задание**](#_30j0zll) **Неявные интенты** [3](#_30j0zll)

[1.1](#_1fob9te) Описание задачи 4

[1.2](#_3znysh7) Структура проекта 4

[1.3 Описание разработанных функций 4](#_2et92p0)

[1.4 Алгоритм решения 5](#_tyjcwt)

[1.5 Используемые библиотеки 6](#_3dy6vkm)

[1.6 Тестовые случаи 6](#_1t3h5sf)

[1.7 Используемые инструменты 6](#_4d34og8)

[1.8 Описание пользовательского интерфейса 7](#_2s8eyo1)

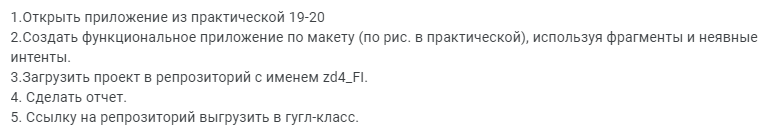
[1.9 Приложение (pr screen экранов) 11](#_17dp8vu)

# 1. **Задание Неявные интенты.**

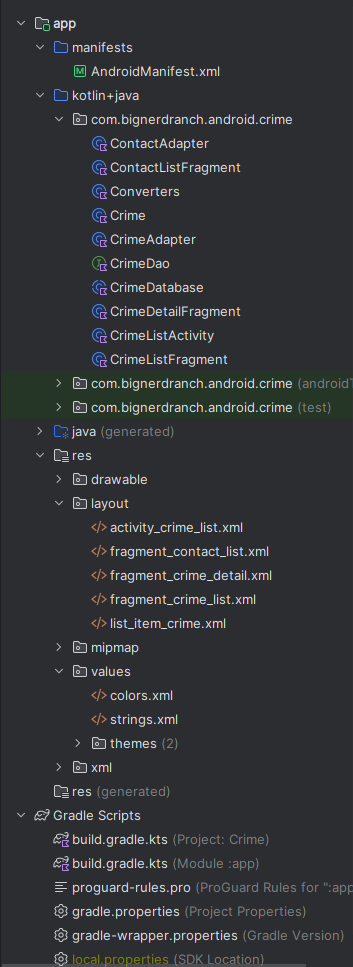
## Описание задачи

Доработать приложение из практики 19-20:  
Выборку «преступника»;

Отправку сообщения о «преступлении» «преступнику».



## Структура проекта



ContactAdapter – Адаптер для контакта.

ContactListFragment – класс фрагмента для вывода списка контактов.

Converter – конвертер для правильной конвертации значения даты из бд и в бд.

Crime – класс содержащий поля бд.

CrimeAdapter – адаптер для работы с «преступлениями».

CrimeDao – запросы для бд.

CrimeDatabase – бд.

CrimeDetailFragment – класс для работы с экраном регистрации «преступления».

CrimeListActivity – класс экрана списка «преступлений».

CrimeListFragment – класс фрагмента листа «преступлений».

Activity\_crime\_list – контейнер фрагментов.

fragment\_contact\_llist – фрагмент листа контактов.  
fragment\_crime\_detail – экран регистрации «преступления».

fragment\_crime\_list – экран с листом «преступлений».

list\_item\_crime – блок «преступления» с информацией о нём.

## Описание разработанных функций

**pickContact** - Регистрация Activity Result API для выбора контакта из телефонной книги.

**checkContactPermissionAndPickContact** - Проверяет разрешение на чтение контактов и открывает контактный интерфейс, если разрешение предоставлено.

**sendCrimeReport** - Создает и запускает Intent для отправки SMS сообщения с информацией о преступлении.

**getCrimeReport** - Формирует и возвращает текст отчета о преступлении, включая название, дату и подробности.

**updateUI** - Обновляет элементы пользовательского интерфейса данными текущего объекта Crime.

**updateDate** - Обновляет текст на кнопке даты, форматируя текущую дату.

**saveCrime** - Сохраняет текущее состояние преступления в базу данных, добавляя новое или обновляя существующее преступление.

**submitList** - Обновляет текущий список преступлений и уведомляет адаптер о необходимости обновления данных.

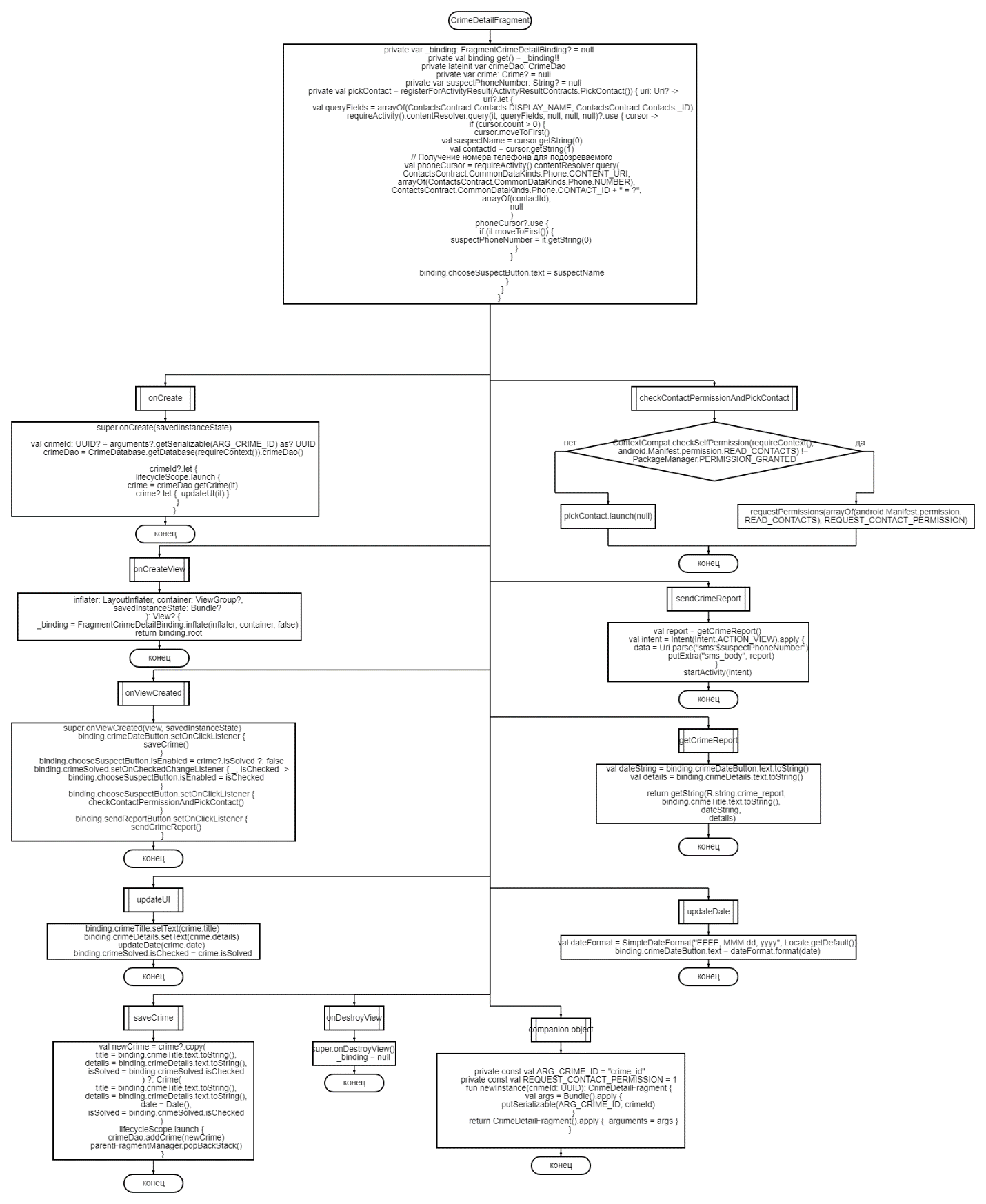
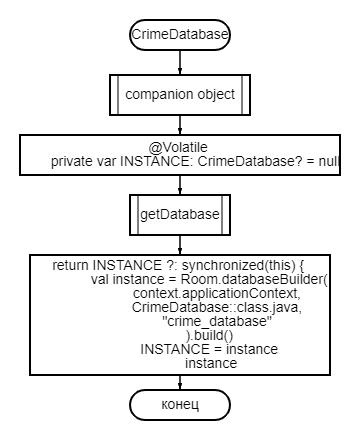
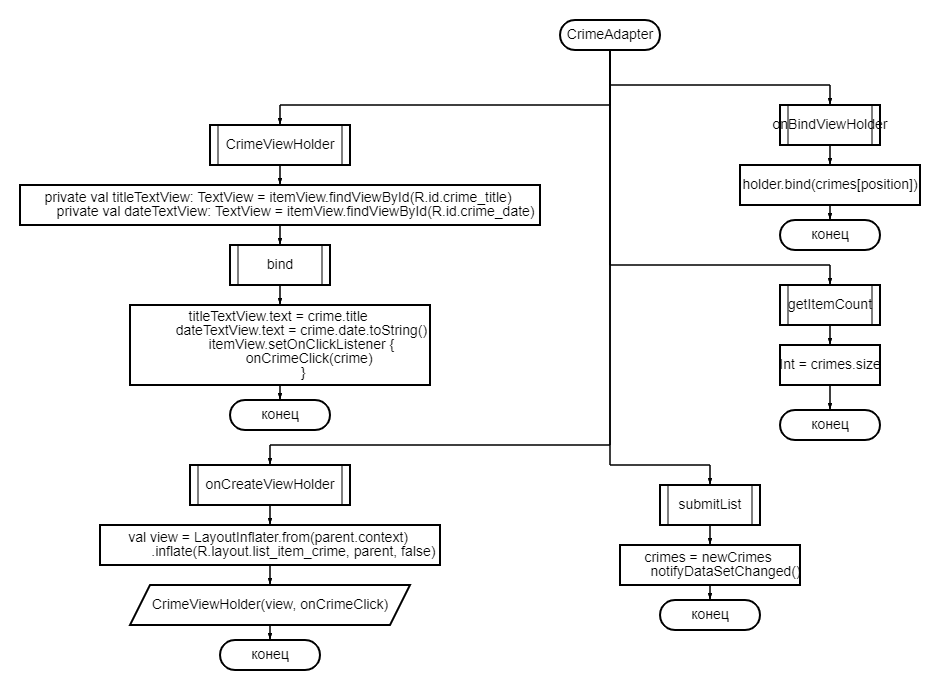
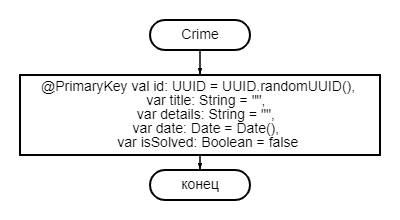
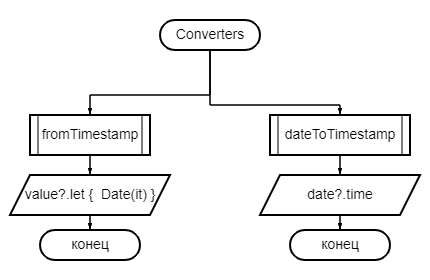
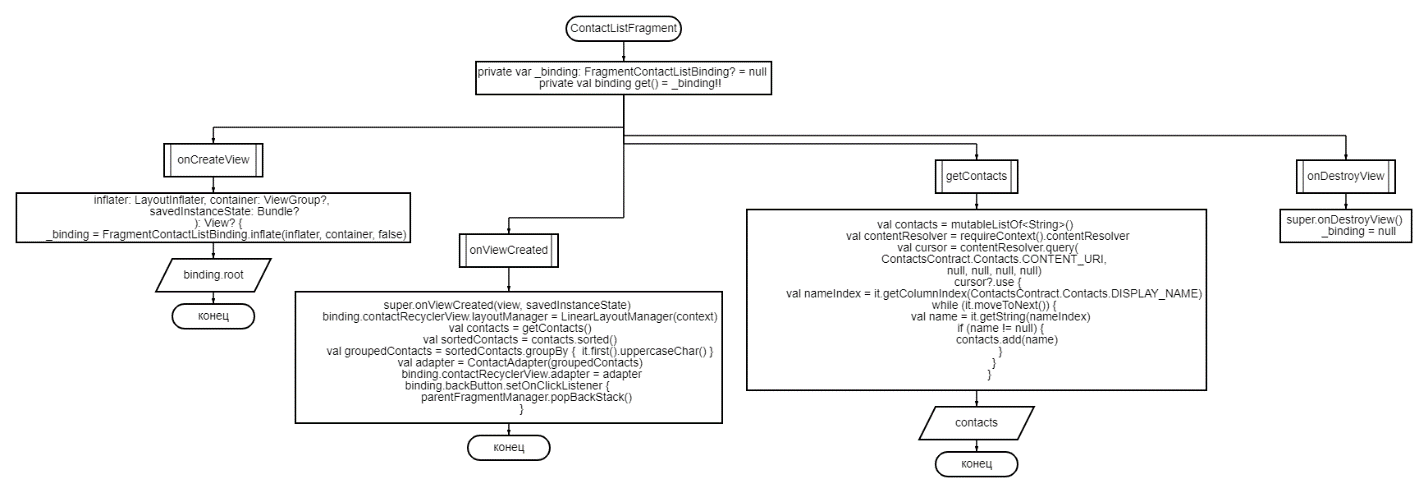
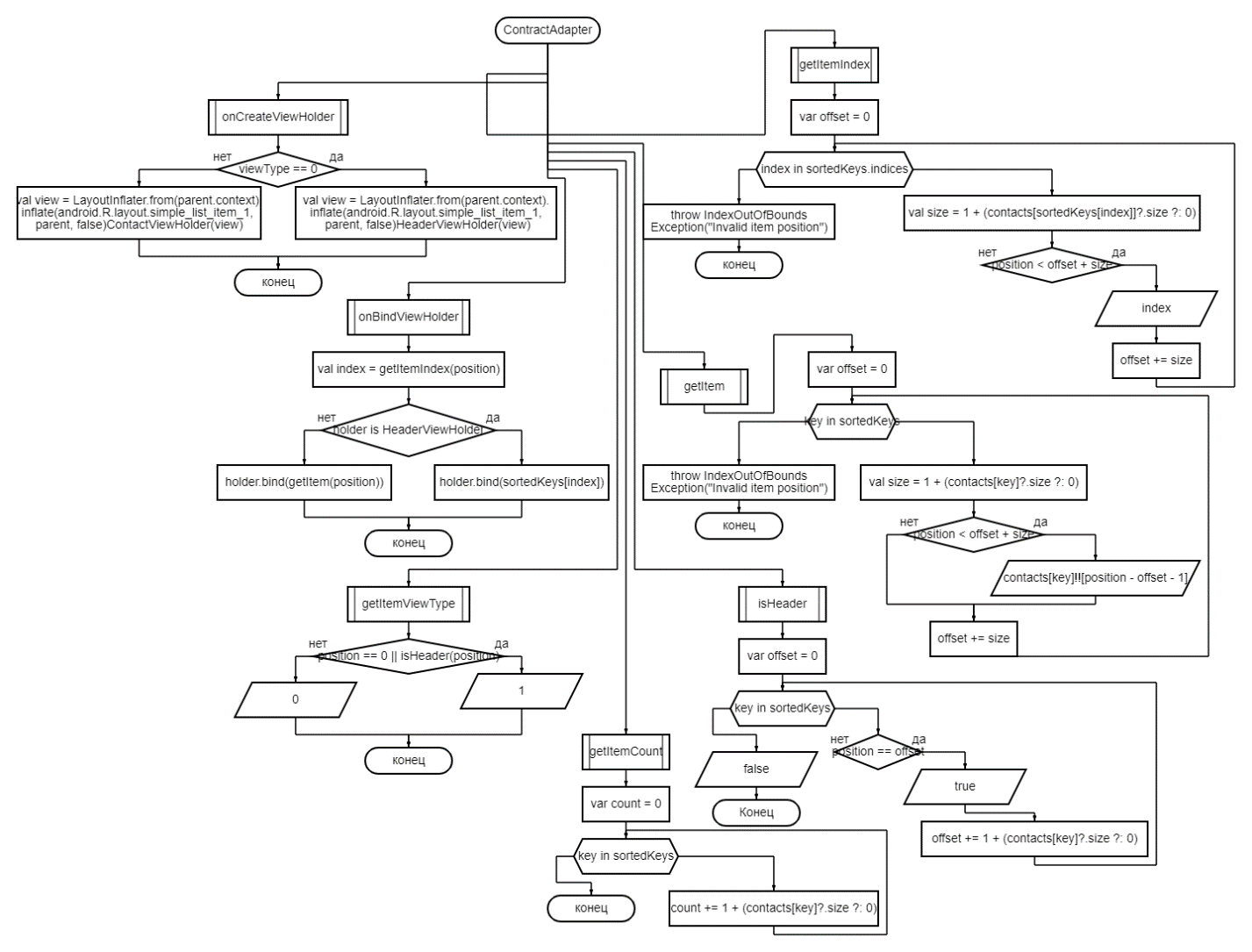
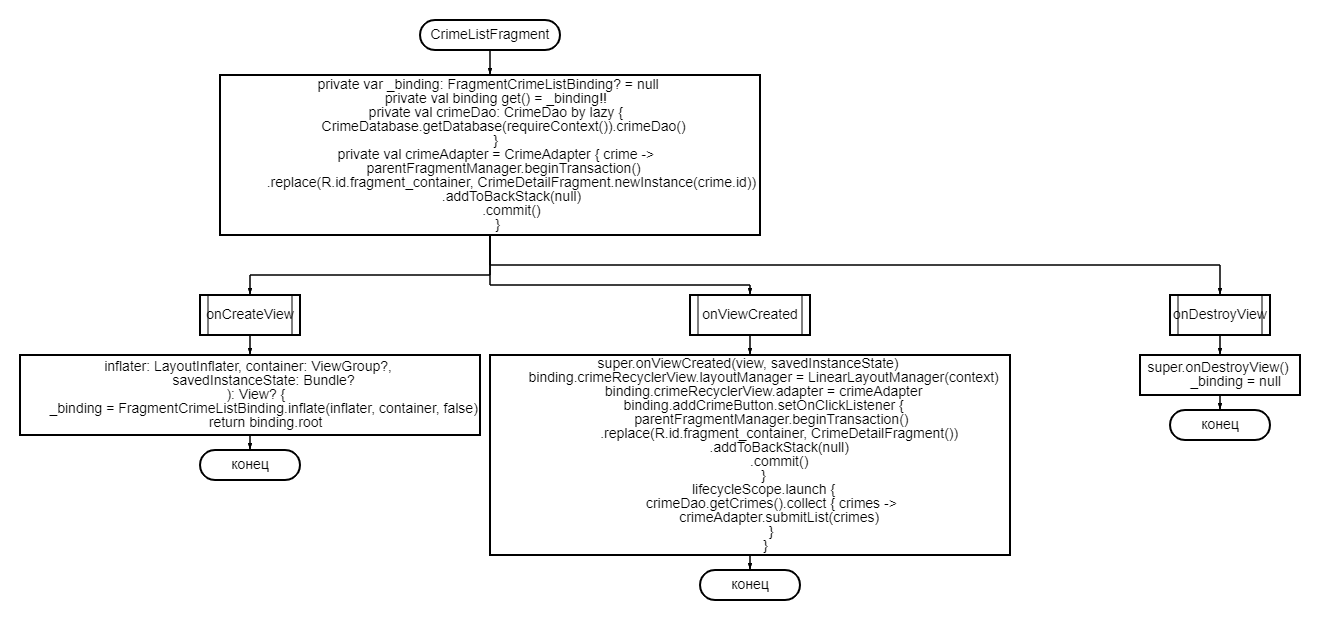
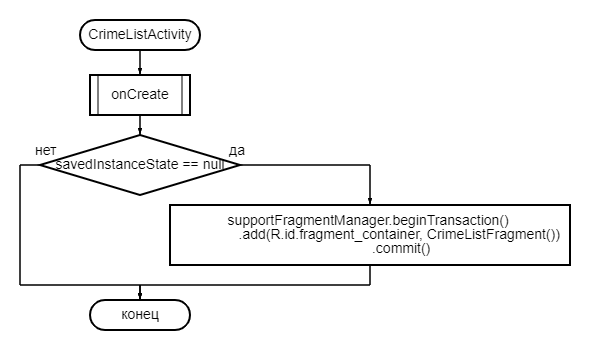
**getDatabase** - Возвращает единственный экземпляр базы данных CrimeDatabase, используя паттерн Singleton.

**getCrimes** - Возвращает поток списка всех преступлений из базы данных.

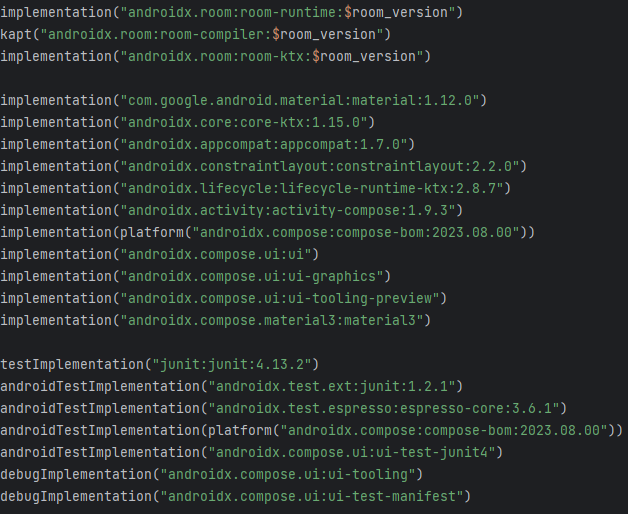
**getCrime** - Возвращает конкретное преступление по его уникальному идентификатору.

**addCrime** - Добавляет новое преступление в базу данных или обновляет его, если оно уже существует.

## 1.4 Алгоритм решения



## Используемые библиотеки



## 1.6 Тестовые случаи

**if (name != null) {**

**contacts.add(name)**

**}**

**if (viewType == 0) {**

**// Заголовок**

**} else {**

**// Контакт**

**}**

**if (holder is HeaderViewHolder) {**

**holder.bind(sortedKeys[index])**

**} else if (holder is ContactViewHolder) {**

**holder.bind(getItem(position))**

**}**

**if (position == 0 || isHeader(position)) {**

**return 0**

**} else {**

**return 1**

**}**

**if (position == offset) {**

**return true**

**}**

**if (cursor.count > 0) {**

**cursor.moveToFirst()**

**// Получение номера телефона для подозреваемого**

**if (it.moveToFirst()) {**

**suspectPhoneNumber = it.getString(0)**

**}**

**}**

**if (ContextCompat.checkSelfPermission(requireContext(), android.Manifest.permission.READ\_CONTACTS) != PackageManager.PERMISSION\_GRANTED) {**

**requestPermissions(arrayOf(android.Manifest.permission.READ\_CONTACTS), REQUEST\_CONTACT\_PERMISSION)**

**} else {**

**pickContact.launch(null)**

**}**

**if (crime?.isSolved ?: false) {**

**binding.chooseSuspectButton.isEnabled = true**

**} else {**

**binding.chooseSuspectButton.isEnabled = false**

**}**

**if (crimeId != null) {**

**lifecycleScope.launch {**

**crime = crimeDao.getCrime(crimeId)**

**if (crime != null) {**

**updateUI(crime!!)**

**}**

**}**

**}**

## 1.7 Используемые инструменты

Язык программирования Kotlin, программа Android Studio Hedgehog 2024.

Приложение проекта Empty Activity

## 1.8 Описание пользовательского интерфейса

При запуске программы загружается экран вида (рис 1).

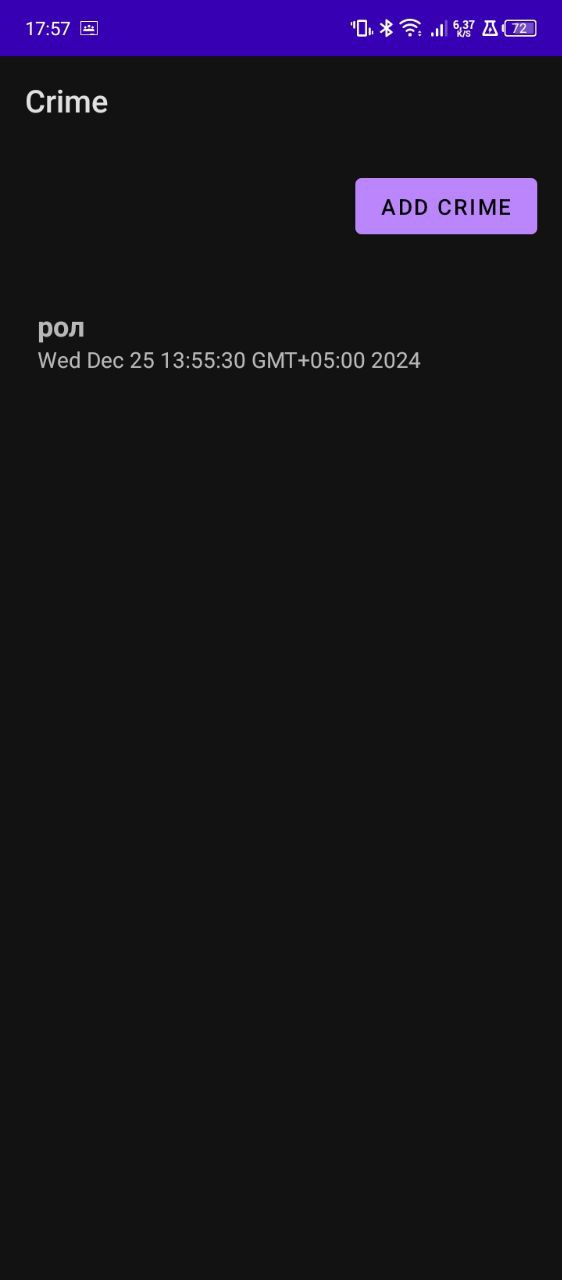


Рис. 1

После нажатия на кнопку “ADD CRIME” открывается экран регистрации преступления (рис. 2)

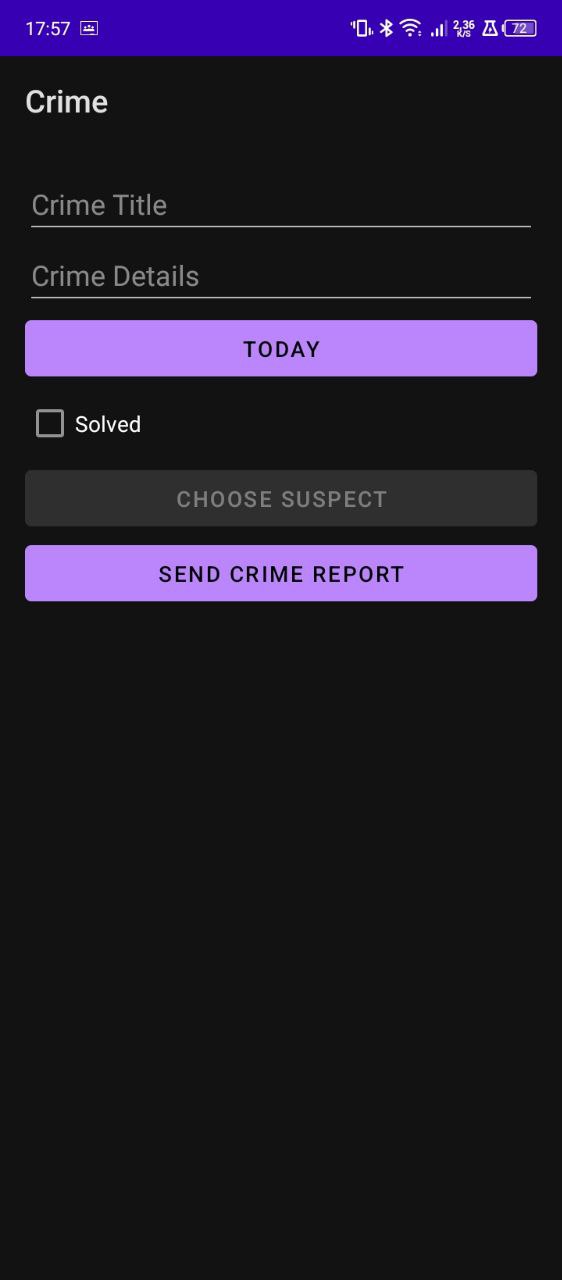


Рис. 2

После заполнения всех полей можно нажать на check box и откроется кнопка “CHOOSE SUSPECT” где можно выбрать контакт. (рис. 3)

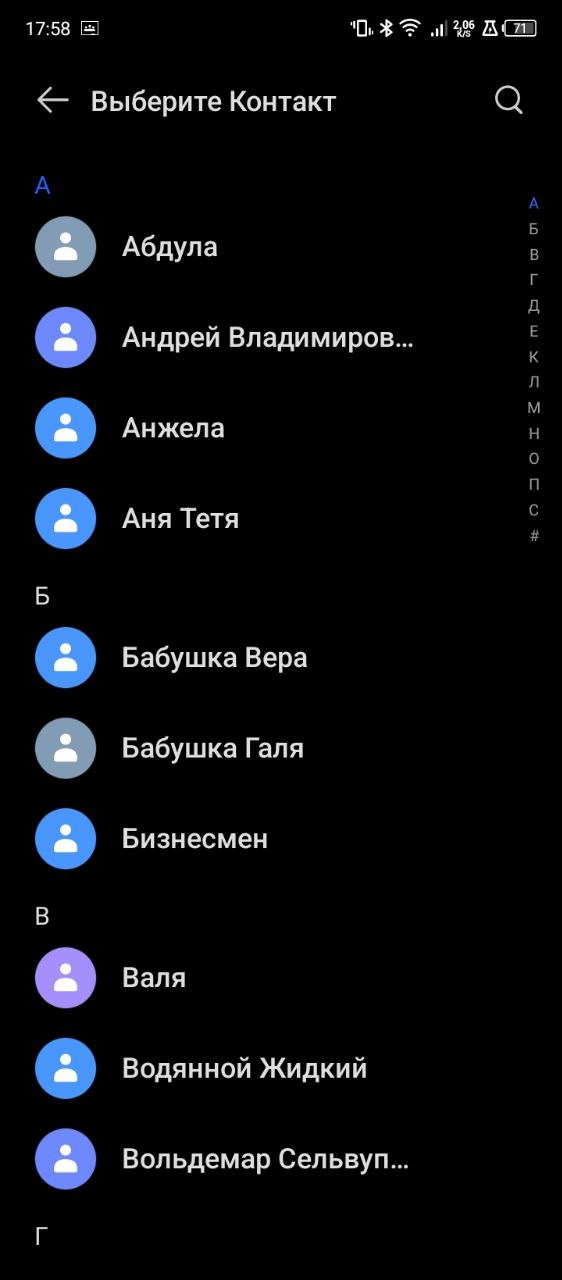
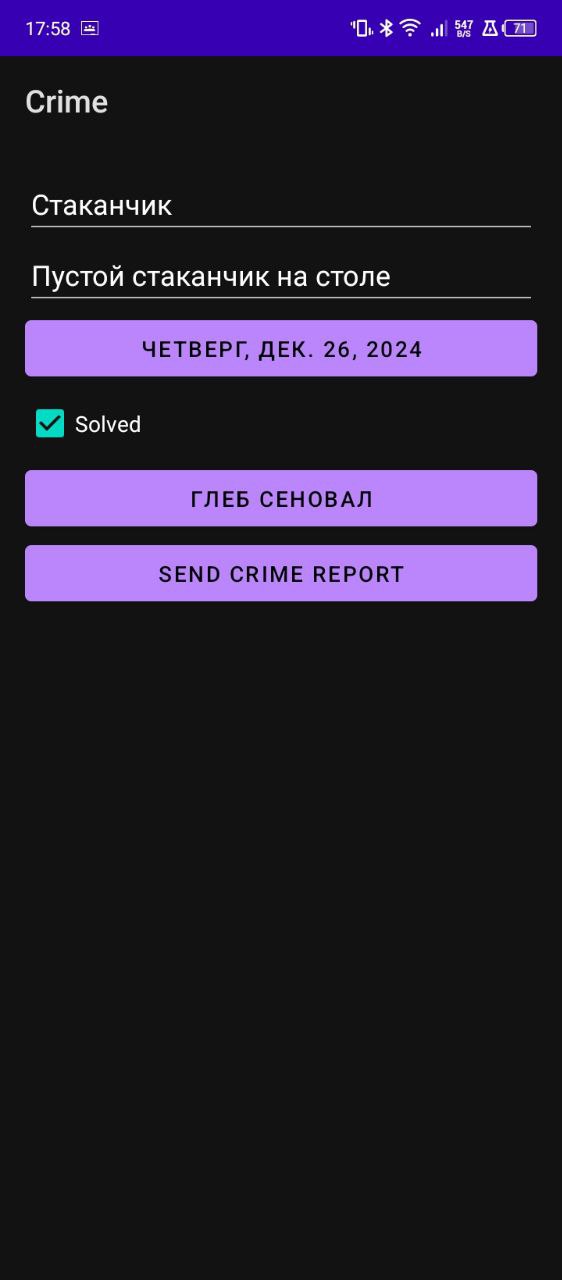


Рис. 3

После нажатия на кнопку с датой произойдёт сохранение преступления в базу данных. (рис. 4).

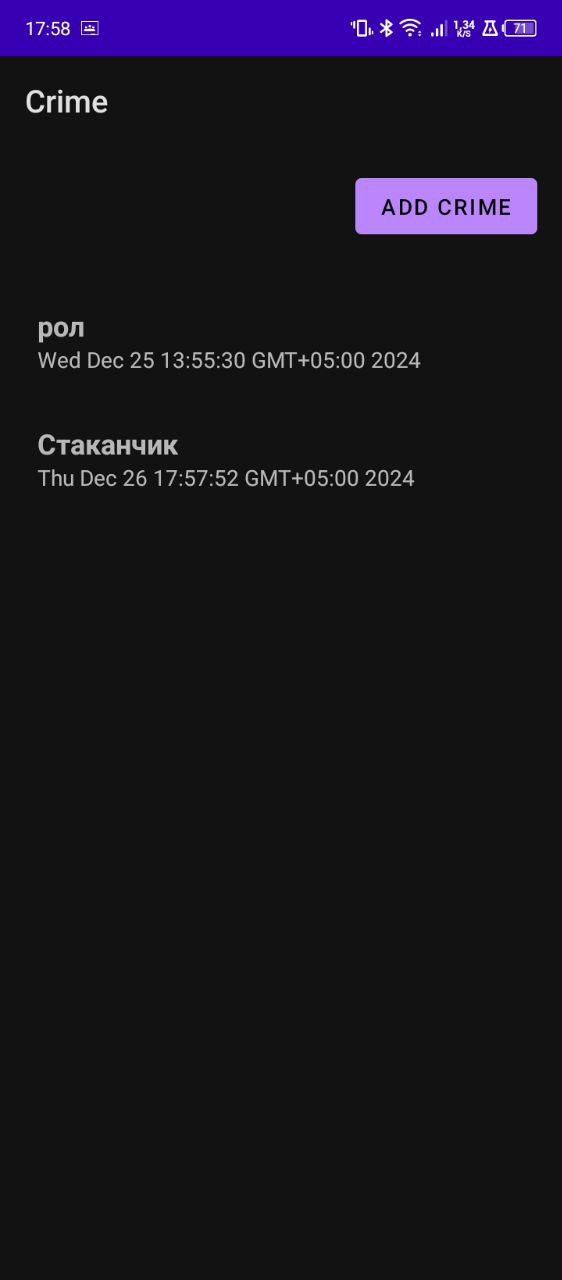


Рис. 4

После нажатия “SENC CRIME REPORT” откроется выбор где можно отправить сообщение нарушителю. (рис. 5)

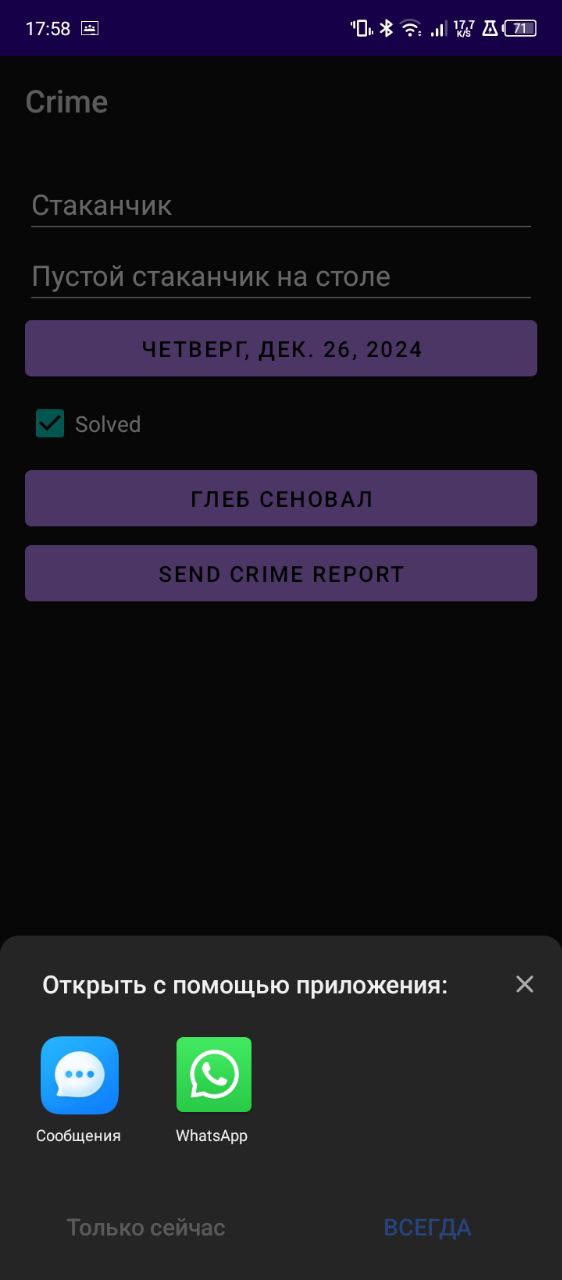


Рис. 5

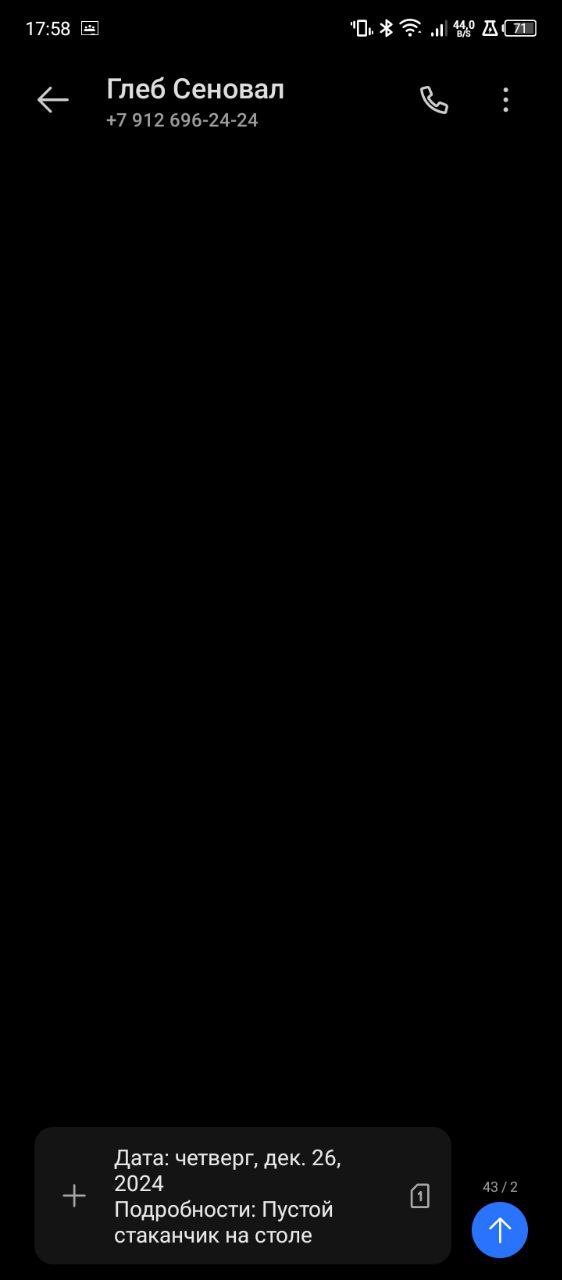
После выбора открывается выбранное приложение с тестом информации о преступлении (рис. 6)  


Рис. 6.

## 1.9 Приложение (pr screen экранов)

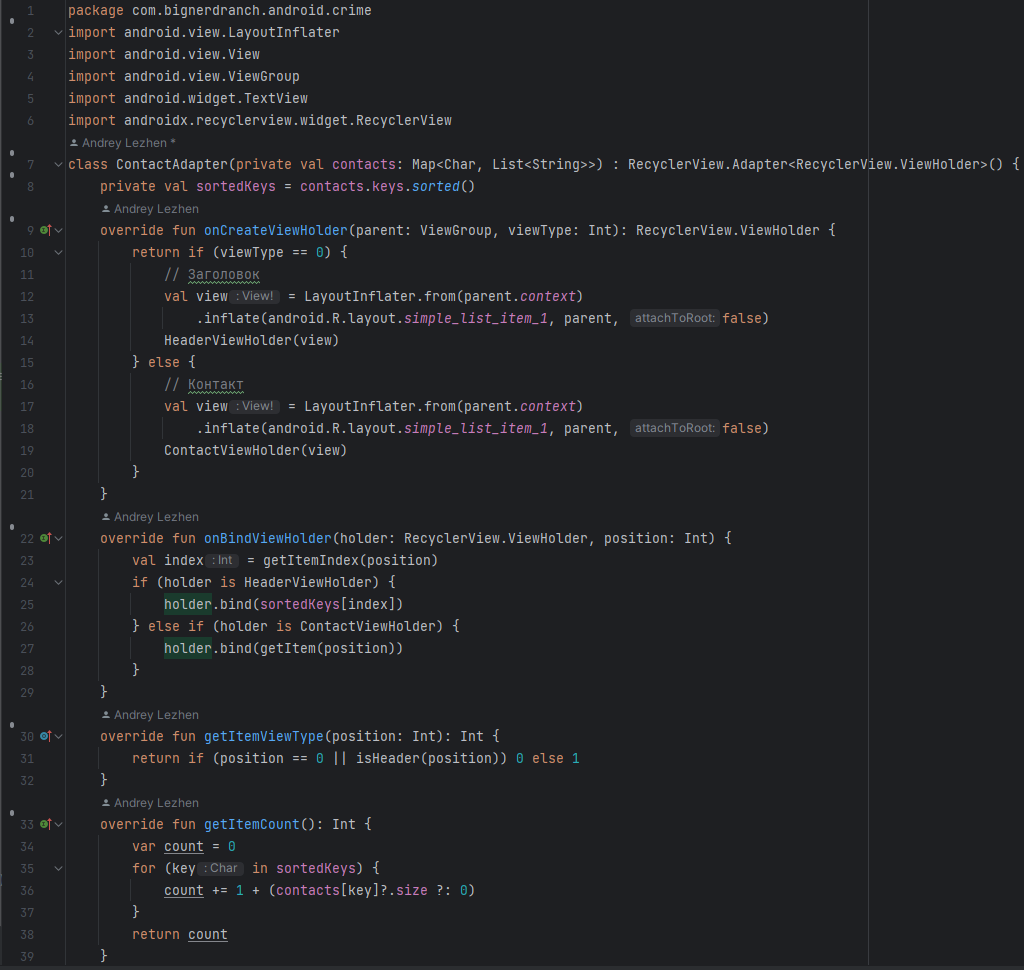
****

Рис. 7

****

Рис. 8

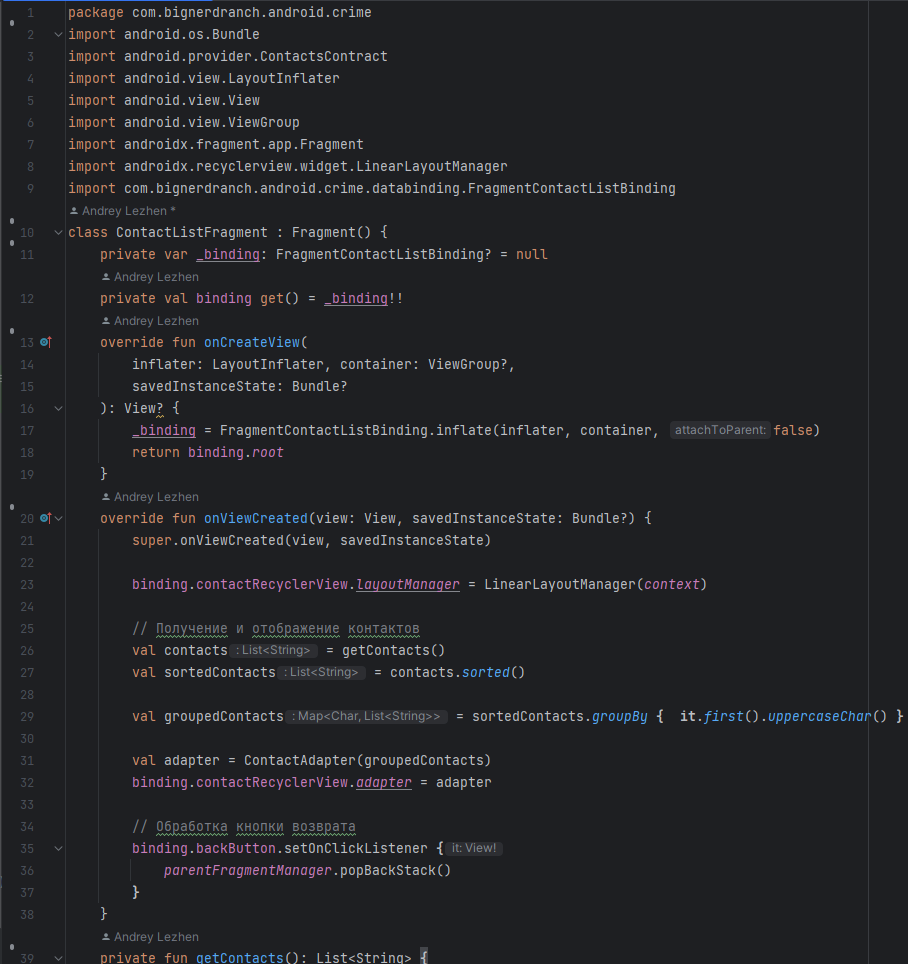


Рис. 9



Рис. 10

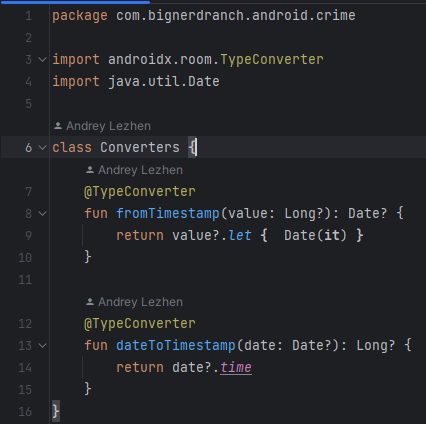


Рис. 11

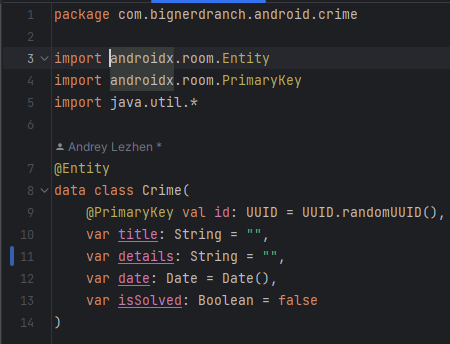


Рис.12

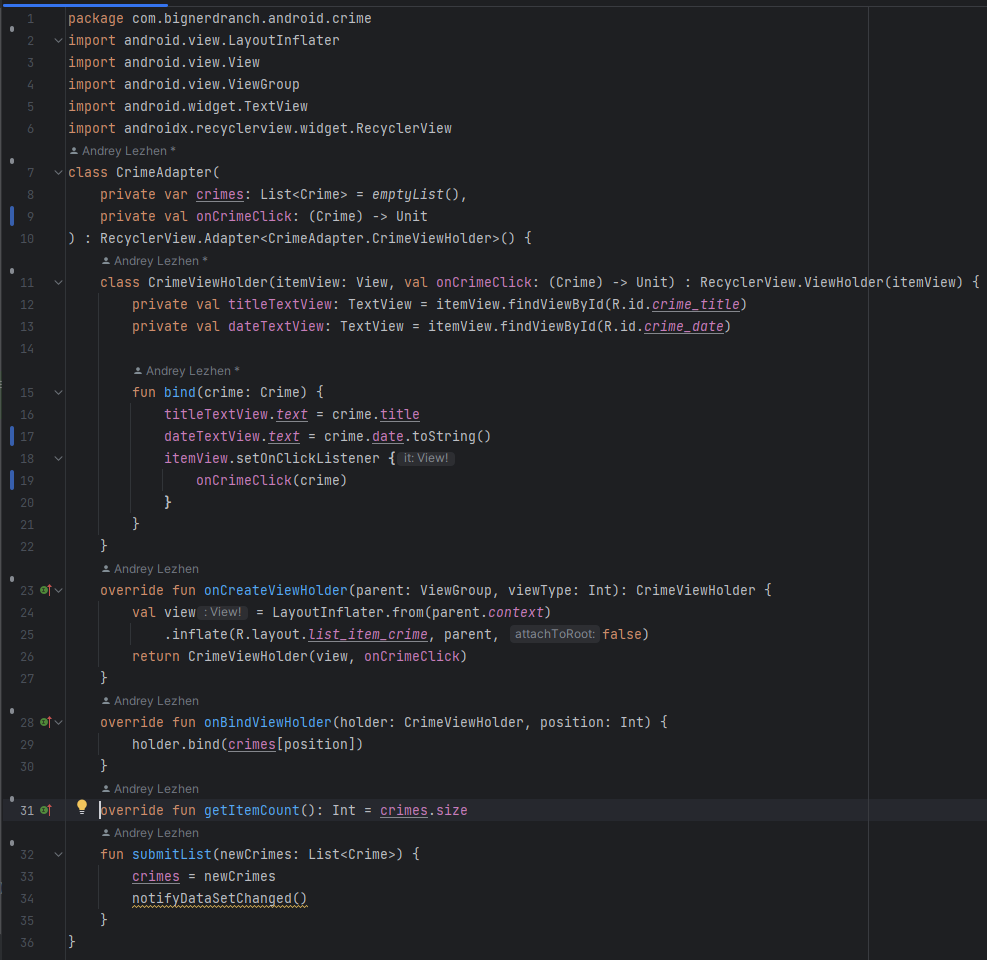


Рис. 13

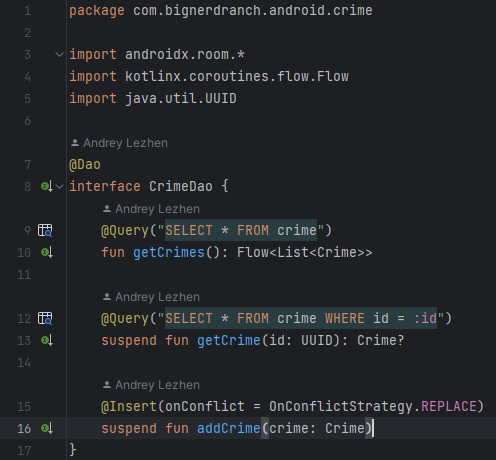


Рис. 14



Рис. 15

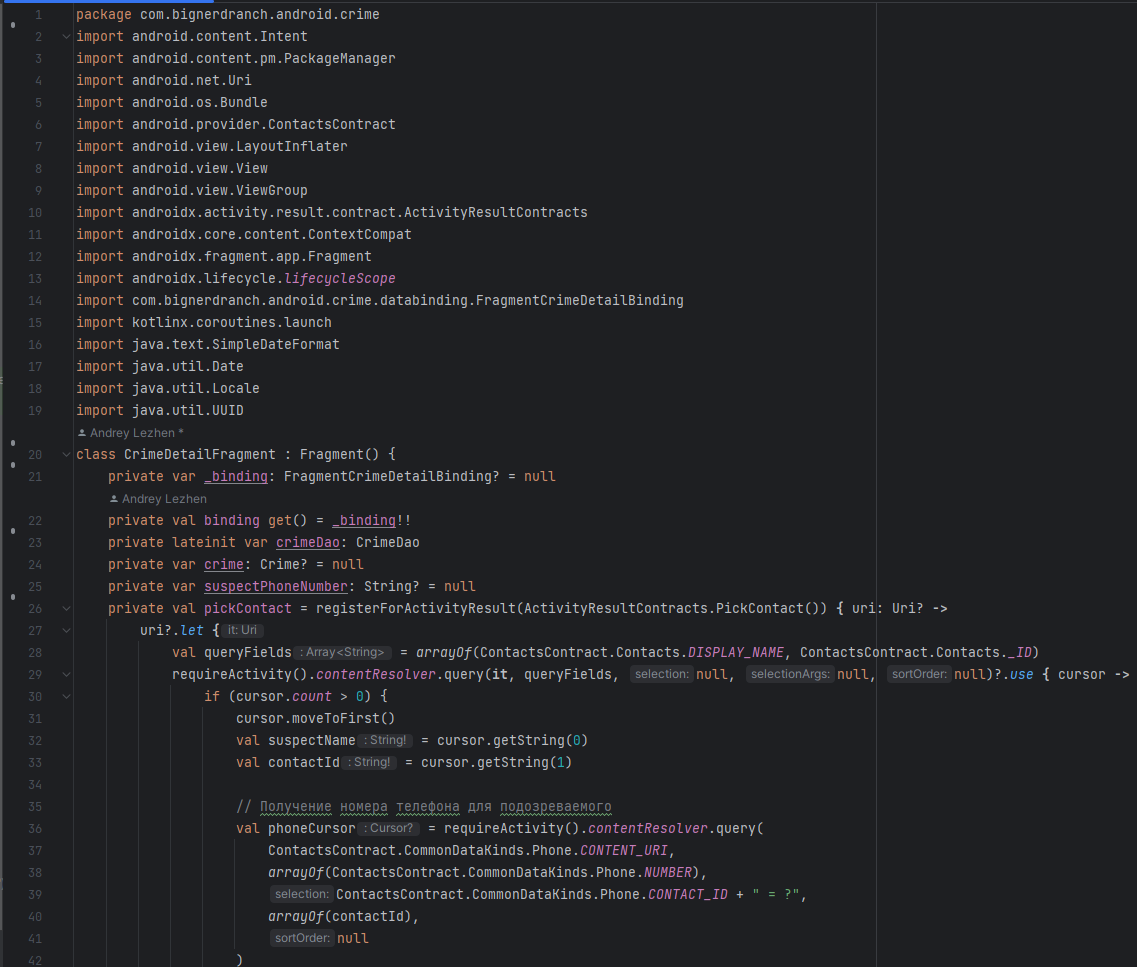


Рис. 16

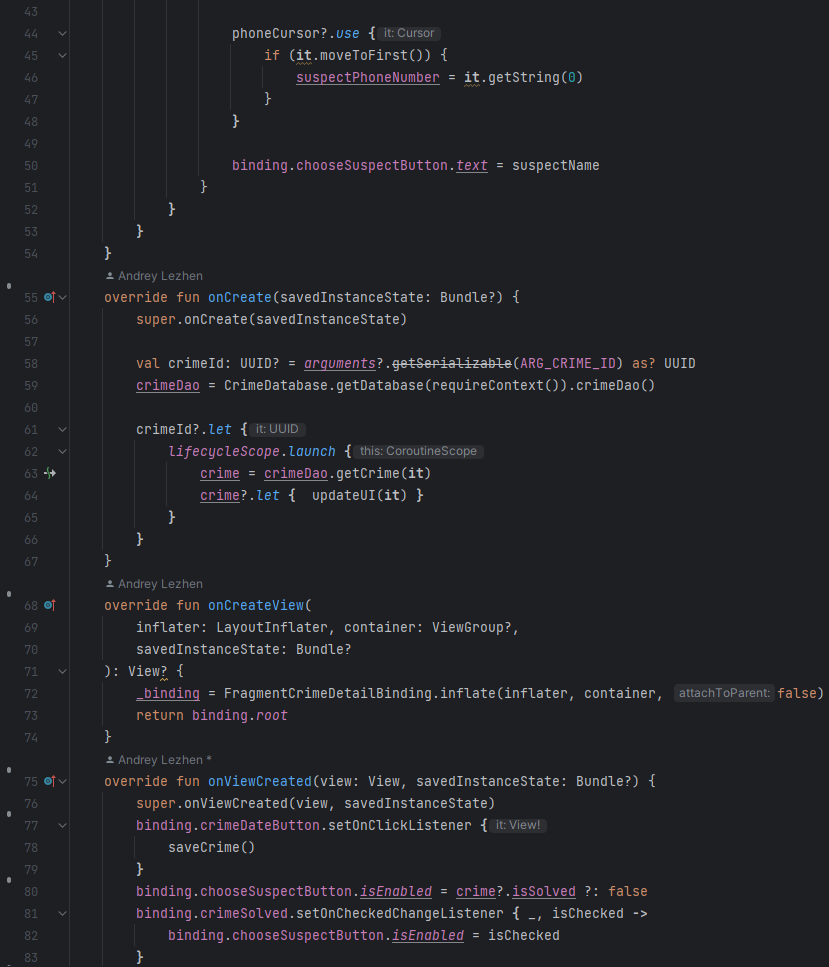


Рис. 17

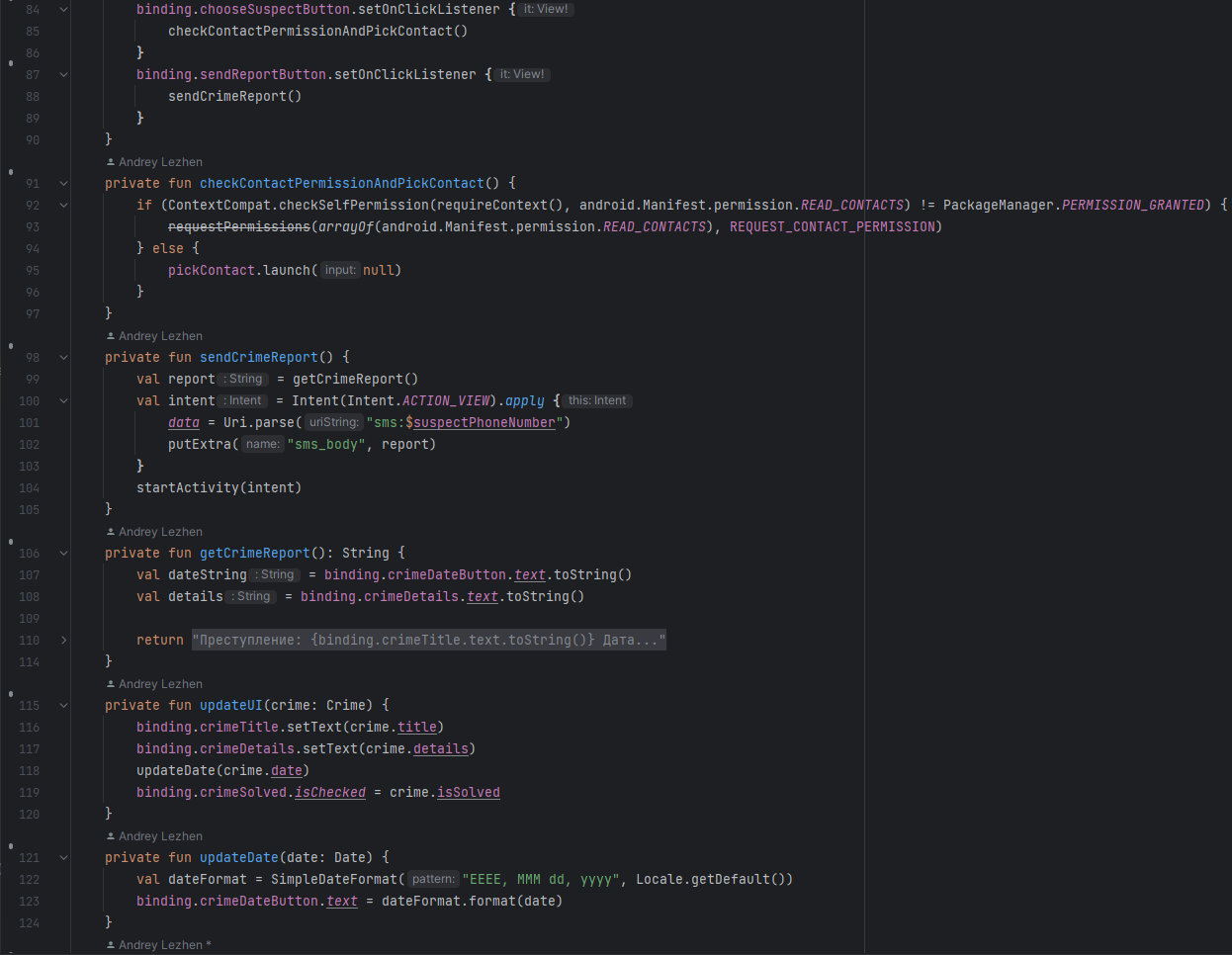


Рис.18



Рис. 19

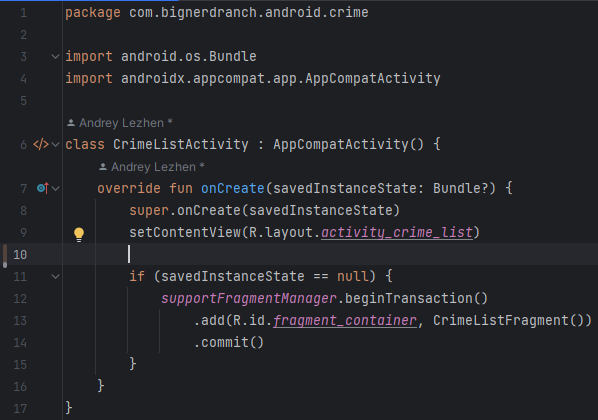


Рис. 20

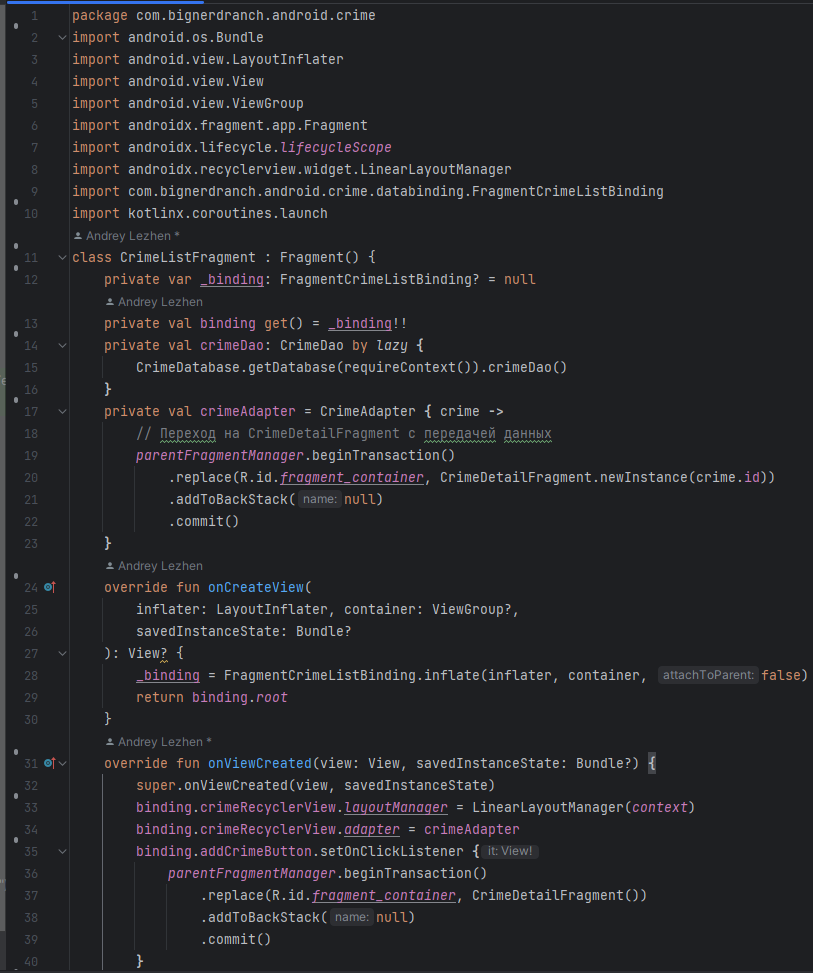


Рис. 21



Рис. 22