

Министерство образования и молодежной политики Свердловской области

ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по учебной практике

РМП задание 2

Выполнил: Лежень Андрей Игоревич

Группа: ПР-31

Преподаватель: Мирошниченко Г.В.

2024

Содержание

[1. **Задание**](#_30j0zll) **Smart Watch** [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)

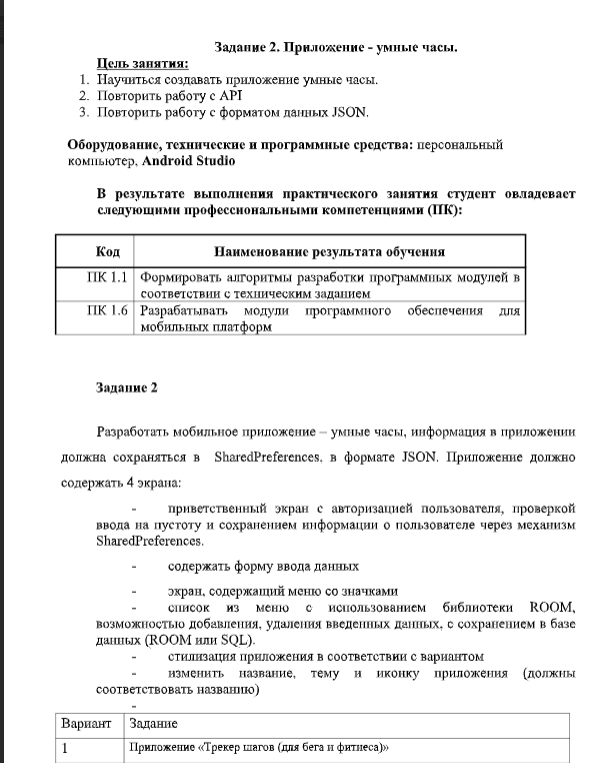
[1.7 Используемые инструменты 6](#_4d34og8)

[1.8 Описание пользовательского интерфейса 7](#_2s8eyo1)

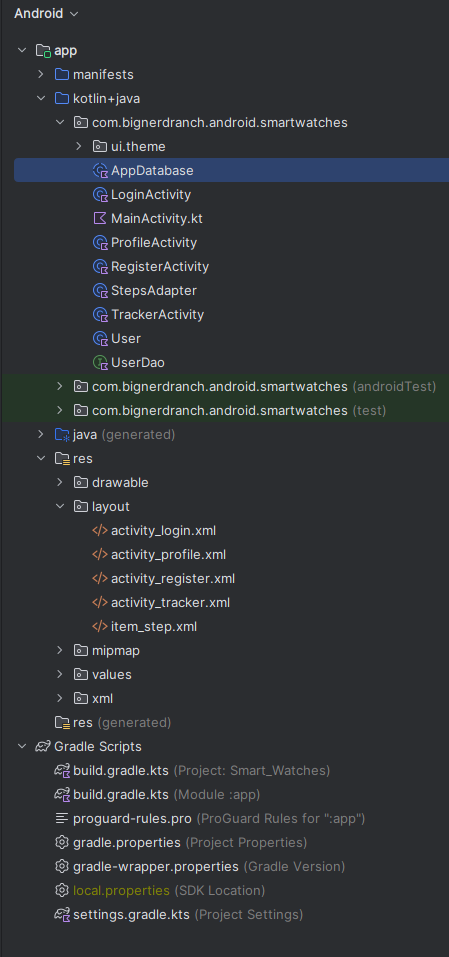
[1.9 Приложение (pr screen экранов) 11](#_17dp8vu)

# 1. **Задание Smart Watch.**

## Описание задачи



## Структура проекта



**ProfileActivity.kt** - Управляет профилем пользователя, включая отображение информации о пользователе и управление аватаром.

**RegisterActivity.kt** - Отвечает за регистрацию пользователя, захватывая данные пользователя и сохраняя их в базе данных.

**StepsAdapter.kt** - Адаптер для отображения списка данных о шагах в RecyclerView.

**TrackerActivity.kt** - Управляет отслеживанием шагов с использованием акселерометра устройства и отображает данные о шагах.

**User.kt** - Определяет сущность данных пользователя для базы данных Room.

**UserDao.kt** - Объект доступа к данным (DAO) для выполнения операций с базой данных на таблице **User**.

**AppDatabase.kt** - Настраивает базу данных Room и предоставляет экземпляр базы данных.

**LoginActivity.kt** - Обрабатывает вход пользователя, проверяет учетные данные и переходит на экран профиля.

**MainActivity.kt** - Основная точка входа для интерфейса Compose UI, отображает простое приветствие.

## Описание разработанных функций

## **loadAvatar(login: String)**- Загружает и устанавливает аватар пользователя.

## **saveAvatar(bitmap: Bitmap, login: String)**- Сохраняет выбранное изображение аватара во внутреннее хранилище.

## **showToast(message: String)**- Отображает всплывающее сообщение (toast).

## **loadStepsCount()**- Загружает количество шагов из базы данных для вошедшего пользователя.

## **loadStepsList() -** Загружает список данных о шагах из SharedPreferences.

## **updateStepsList() -** Обновляет количество шагов на текущую дату всписке.

## **loadStepsFromTextViewIntoDatabase()**- Сохраняет количество шагов в базу данных.

## **saveStepsList()**- Сохраняет список шагов в SharedPreferences.

## **onLogout()**- Очищает сохраненные шаги из SharedPreferences при выходе.

## **insert(user: User) -** Вставляет нового пользователя в базу данных.

## **findByLogin(login: String): User**? - Извлекает пользователя по логину.

## **updateUser(user: User**) - Обновляет информацию о существующем пользователе.

## **getStepsCountForUser(login: String): Int**- Извлекает количество шагов для конкретного пользователя.

## **getDatabase(context: Context): AppDatabase**- Предоставляет синглтон-экземпляр базы данных.

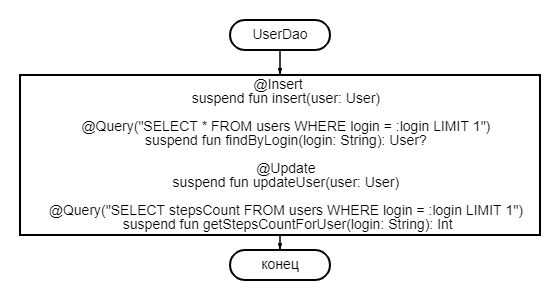
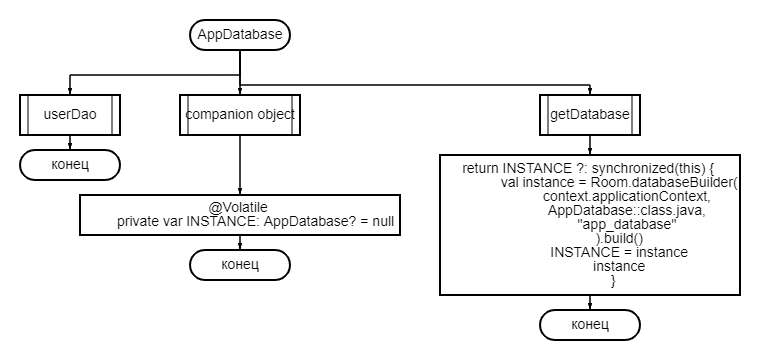
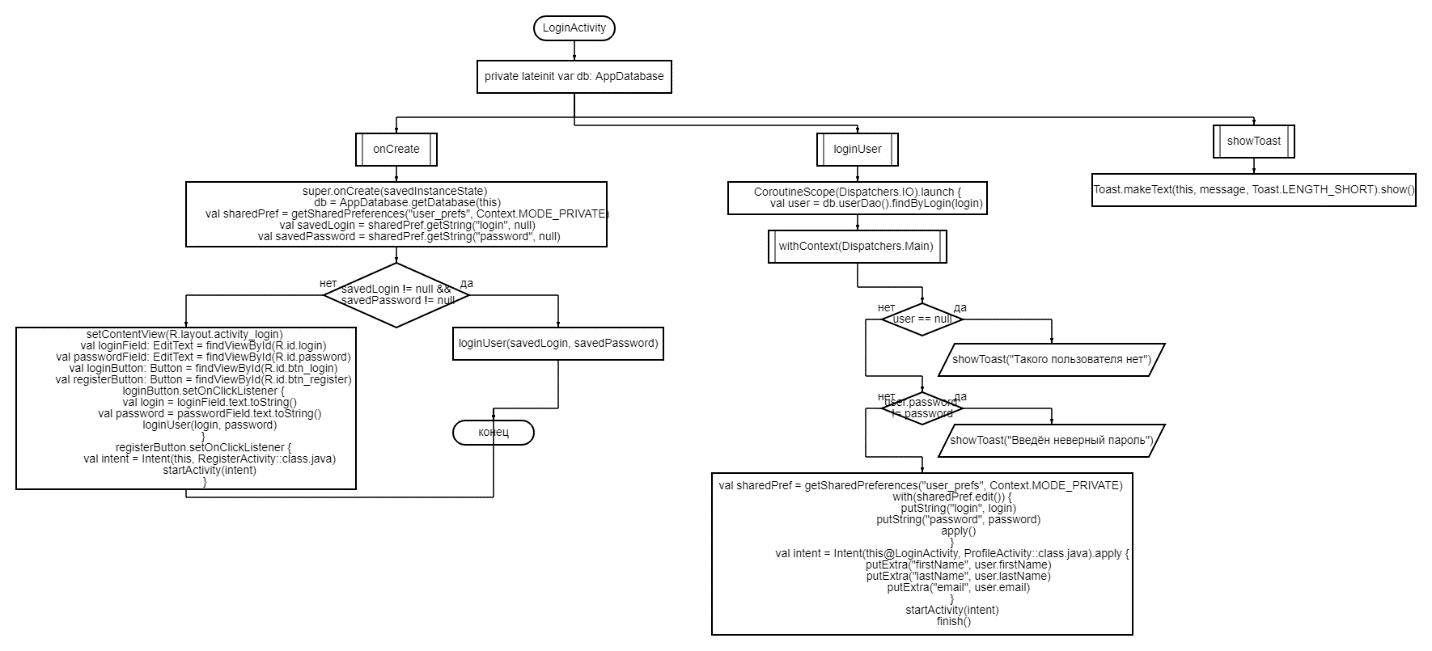
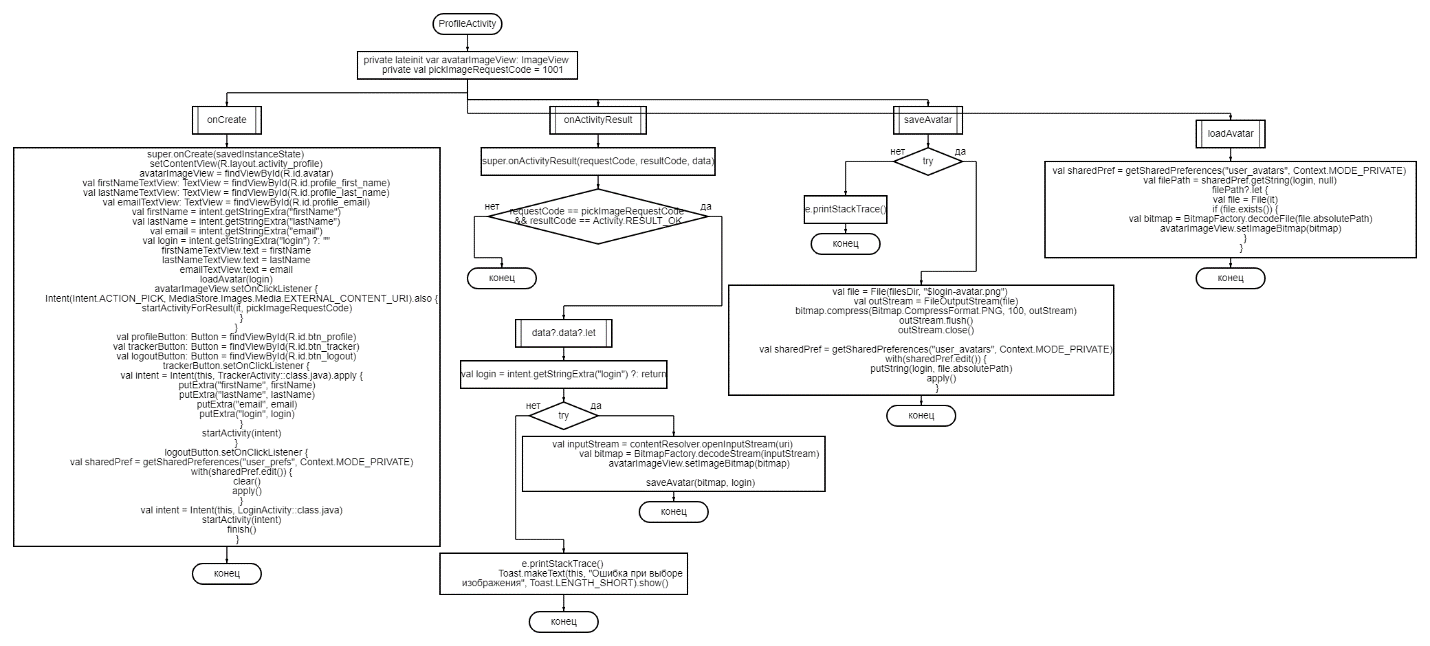
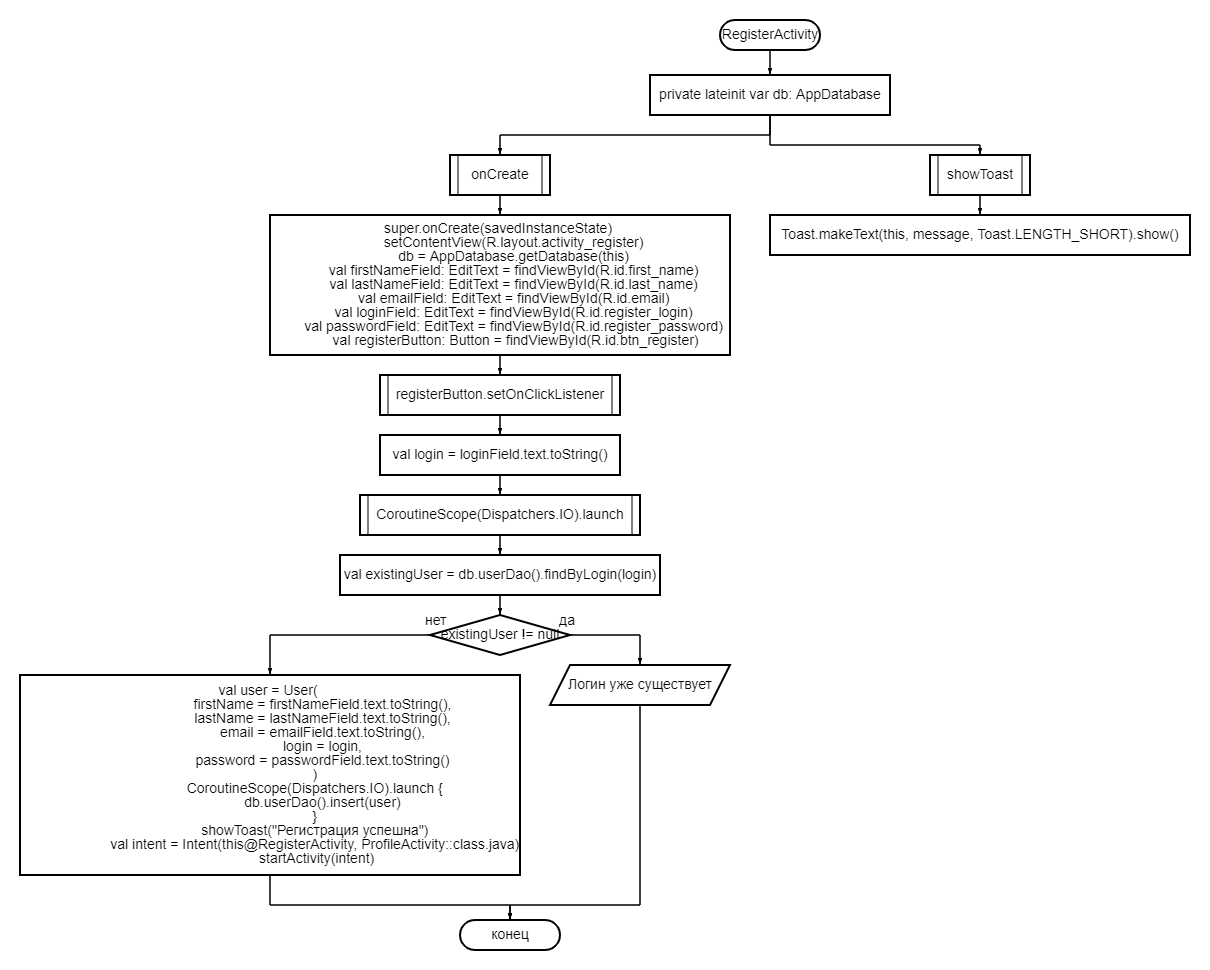
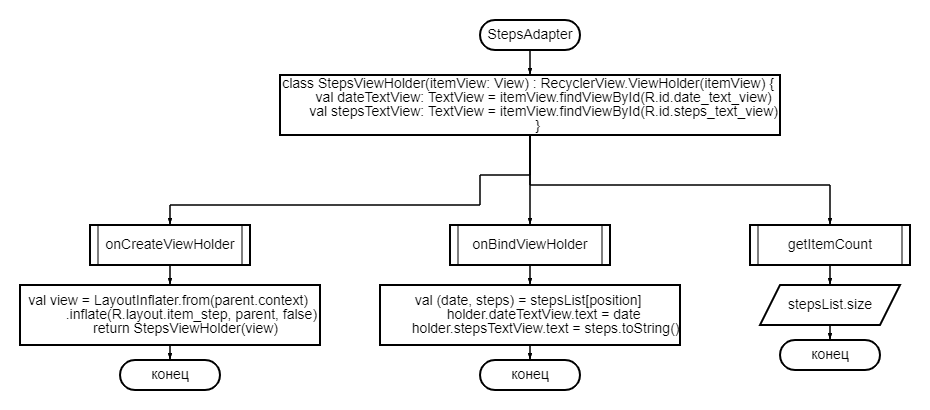
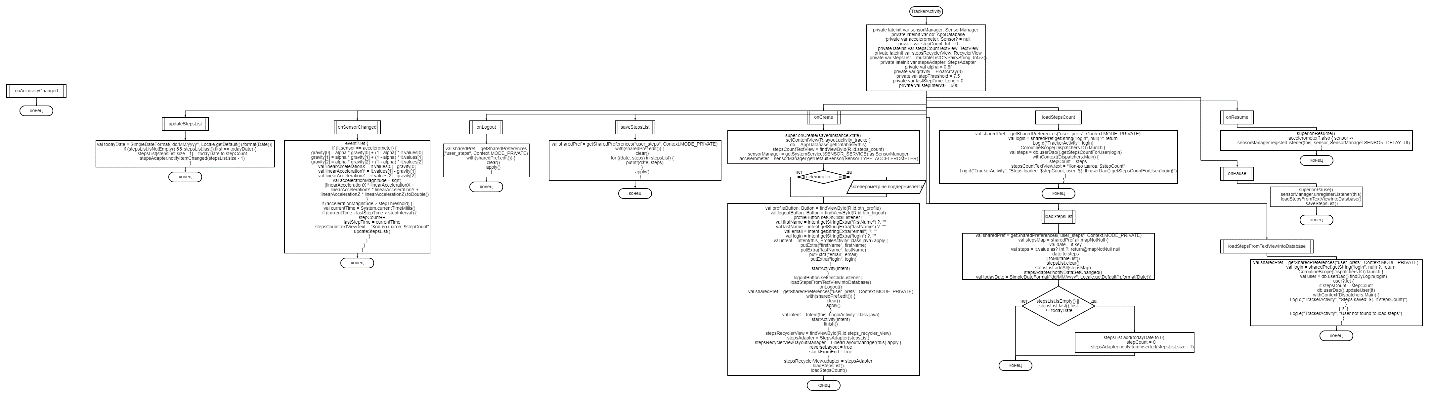
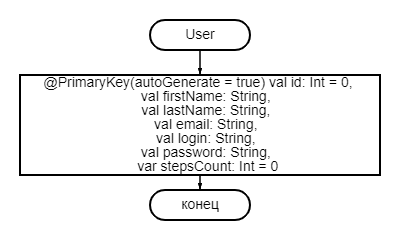
## **loginUser(login: String, password: String)**- Аутентифицирует пользователя и переходит в его профиль при успешной проверке.

## **showToast(message: String**) - Отображает всплывающее сообщение (toast).

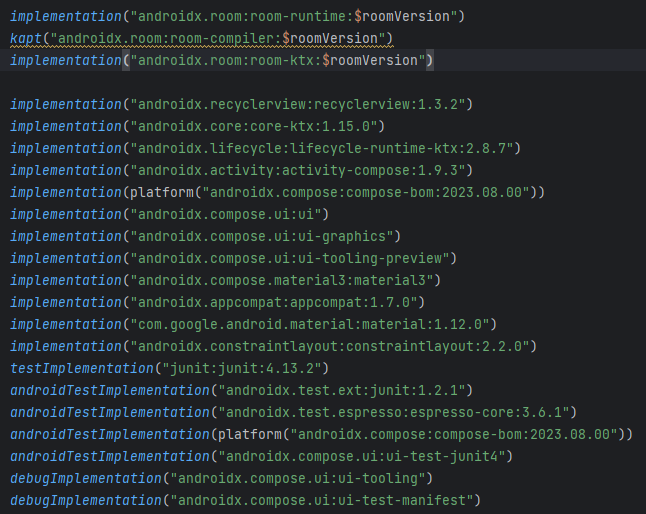
## **Greeting(name: String, modifier: Modifier)**- Компонент отображения приветственного сообщения.

## **GreetingPreview()**- Функция предварительного просмотра для компонента Greeting.

## 1.4 Алгоритм решения



## Используемые библиотеки



## 1.6 Тестовые случаи

**if (requestCode == pickImageRequestCode && resultCode == Activity.RESULT\_OK) { ... }**

**if (file.exists()) { ... }**

**RegisterActivity.kt**

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

**showToast("Логин уже существует")**

**} else { ... }**

**if (accelerometer == null) {**

**Toast.makeText(this, "Акселерометр не поддерживается", Toast.LENGTH\_SHORT).show()**

**}**

**if (stepsList.isEmpty() || stepsList.last().first != todayDate) { ... }**

**if (currentTime - lastStepTime > stepInterval) { ... }**

**if (stepsList.isNotEmpty() && stepsList.last().first == todayDate) { ... }**

**if (savedLogin != null && savedPassword != null) { ... }**

**if (user == null) {**

**showToast("Такого пользователя нет")**

**} else if (user.password != password) {**

**showToast("Введён неверный пароль")**

**} else { ... }**

## 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).

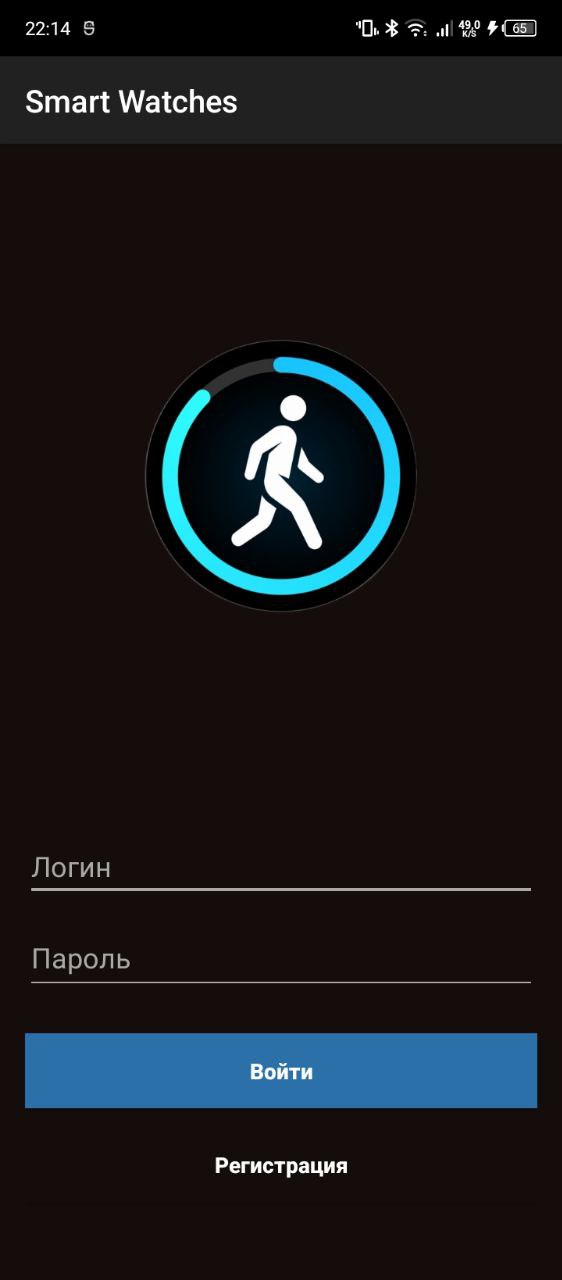


Рис. 1.

Вид экрана после нажатия на кнопку регистрации (рис 2).

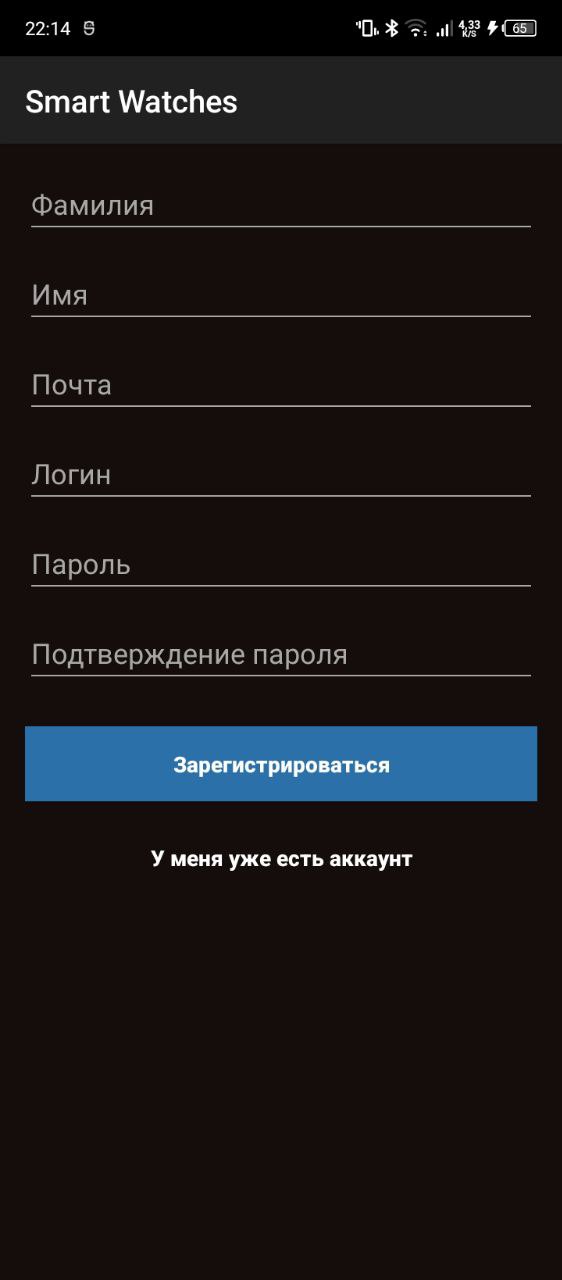


Рис. 2.

Вид экрана после ввода всех данных в поля (рис 3).

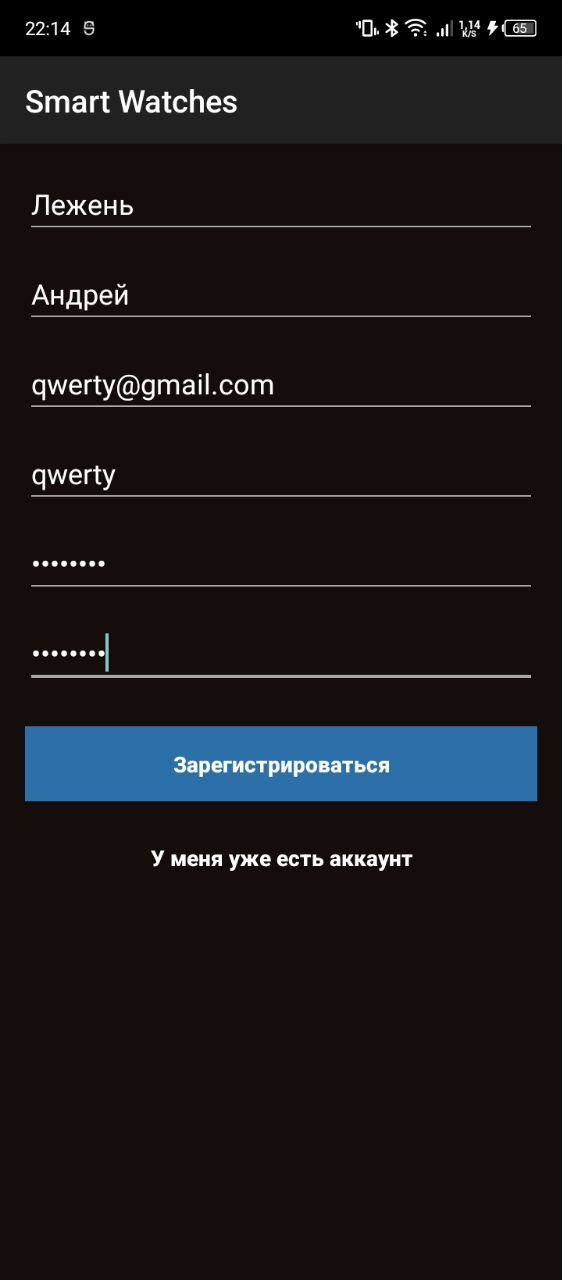


Рис. 3.

Экран профиля открывается сразу после регистрации или повторного входа в приложение (рис 4).



Рис. 4.

Экран трекера шагов (рис 5).

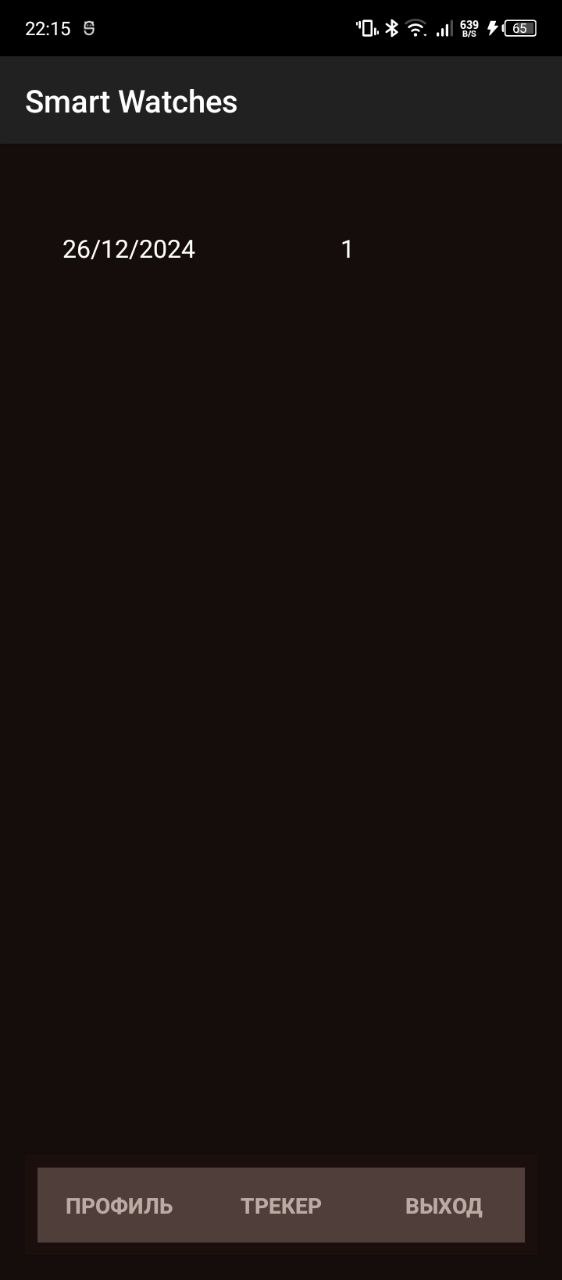


Рис. 5.

Экран трекера шагов после обнаружения «тряски» телефона (рис 6).



Рис. 6.

## 1.9 Приложение (pr screen экранов)

****

Рис. 7

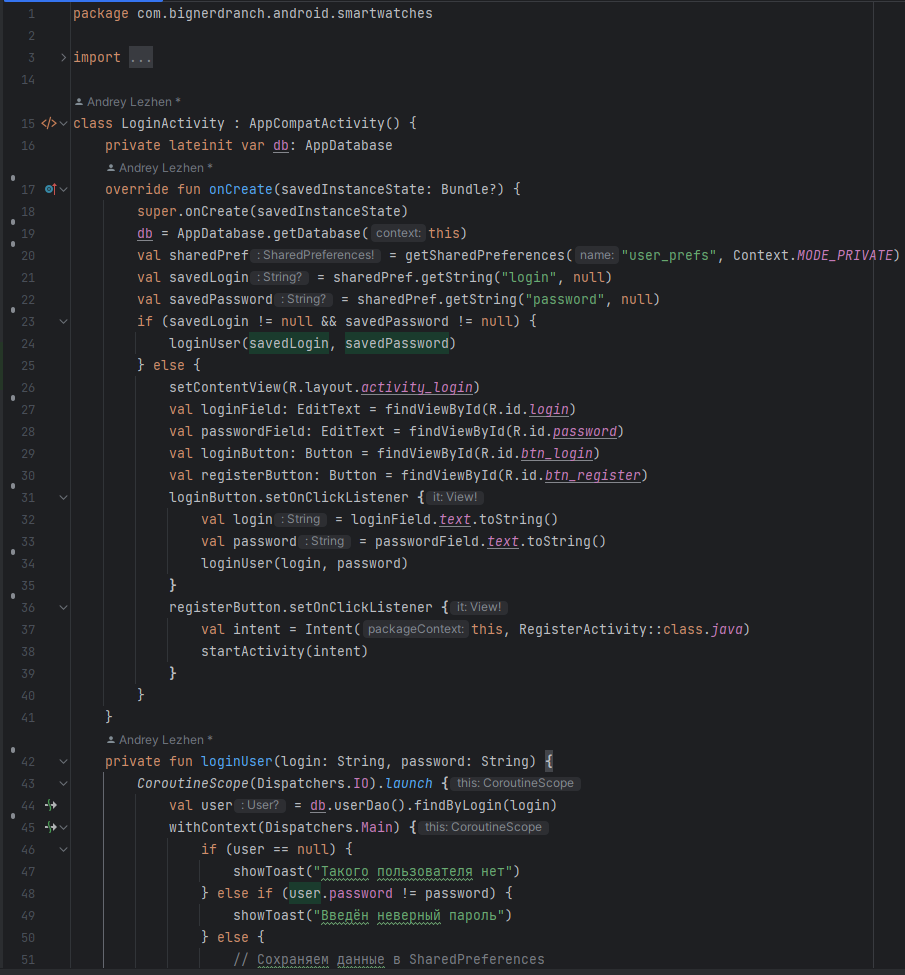


Рис. 8

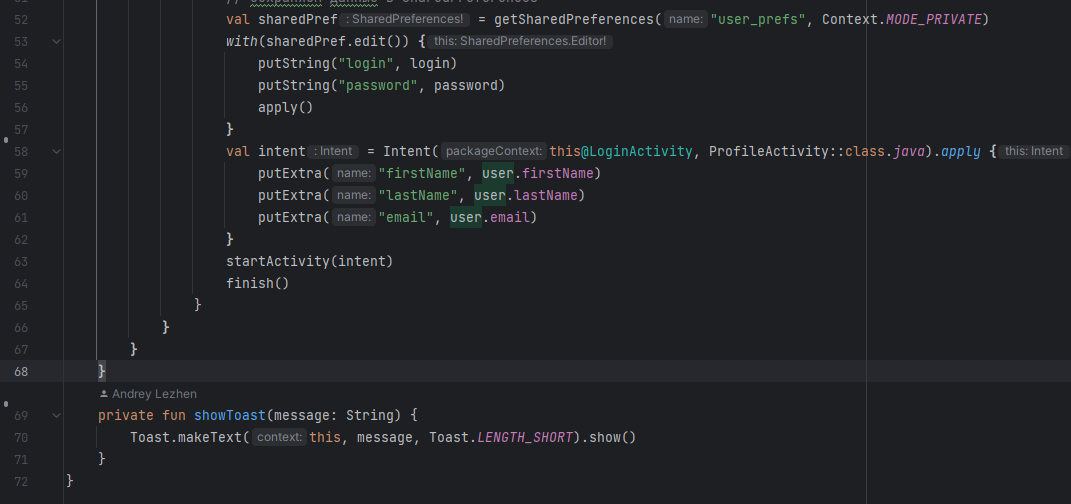


Рис. 9



Рис. 10

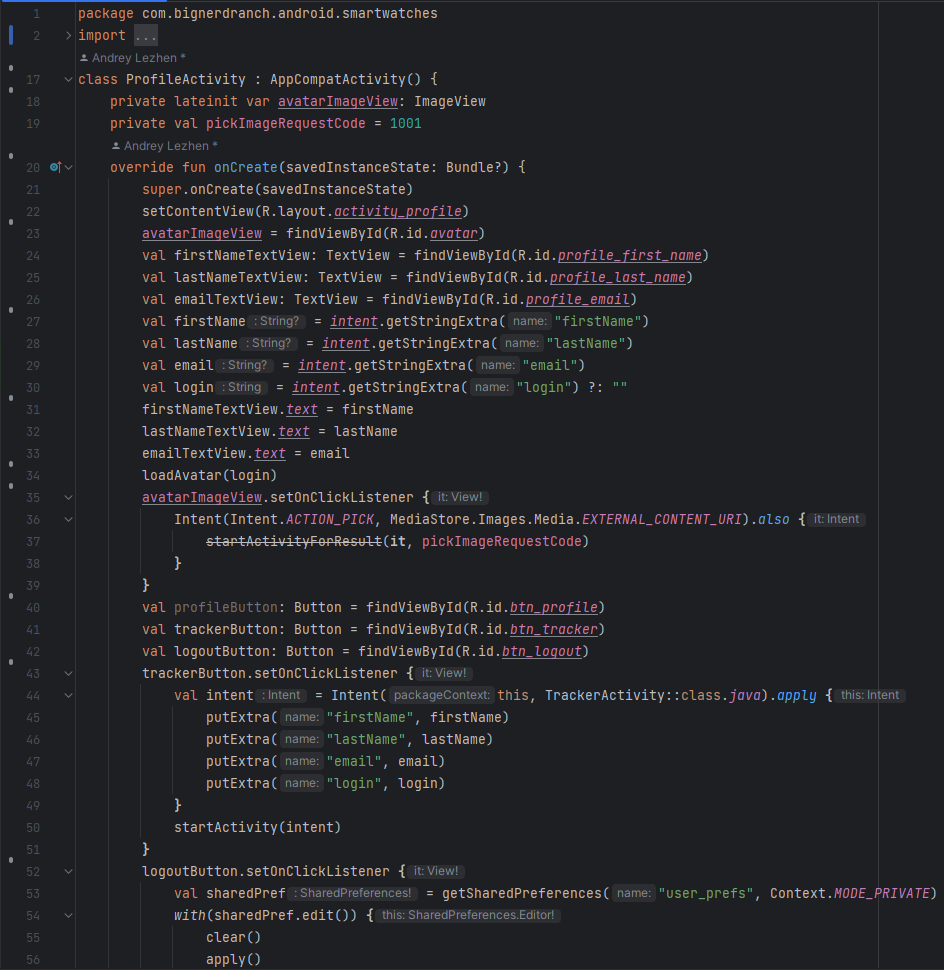


Рис. 11

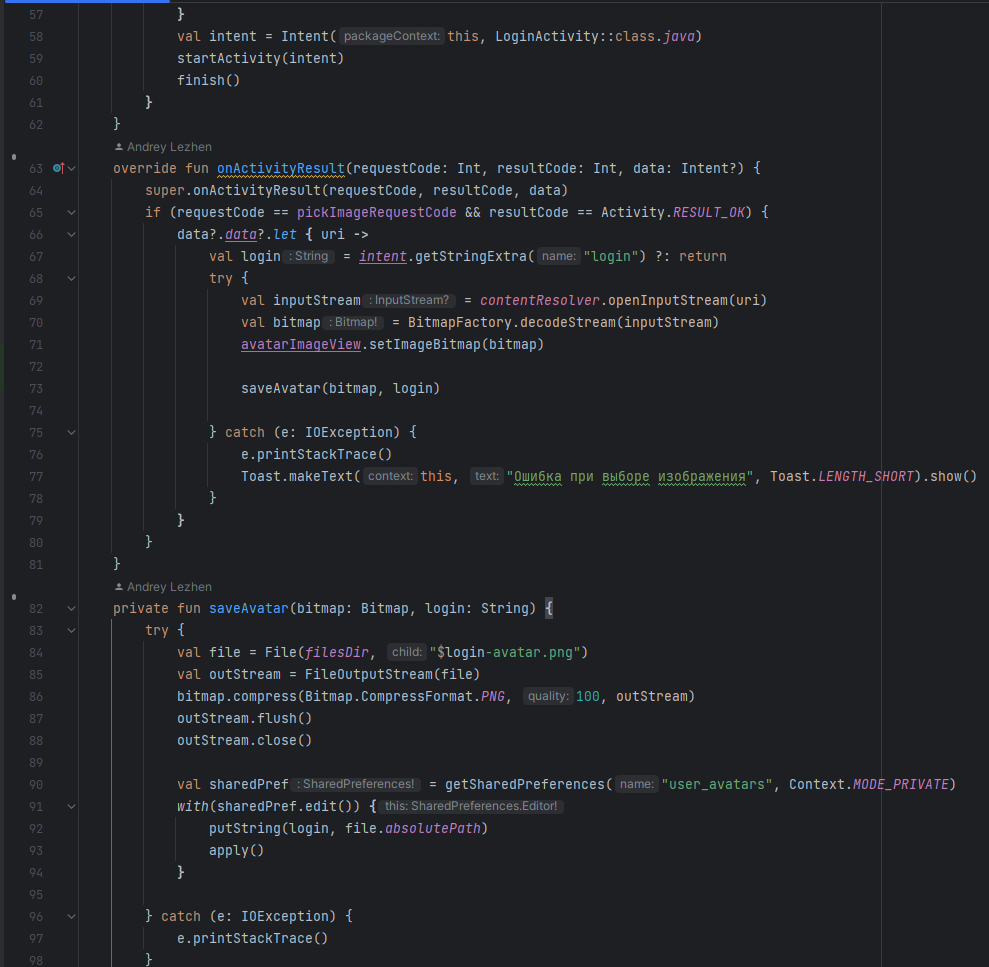


Рис.12

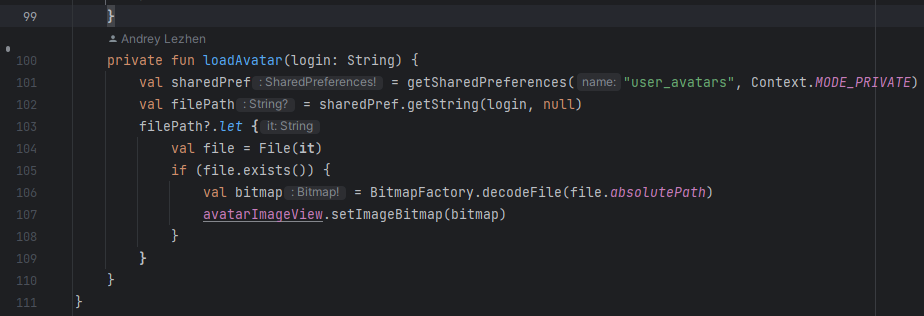


Рис. 13

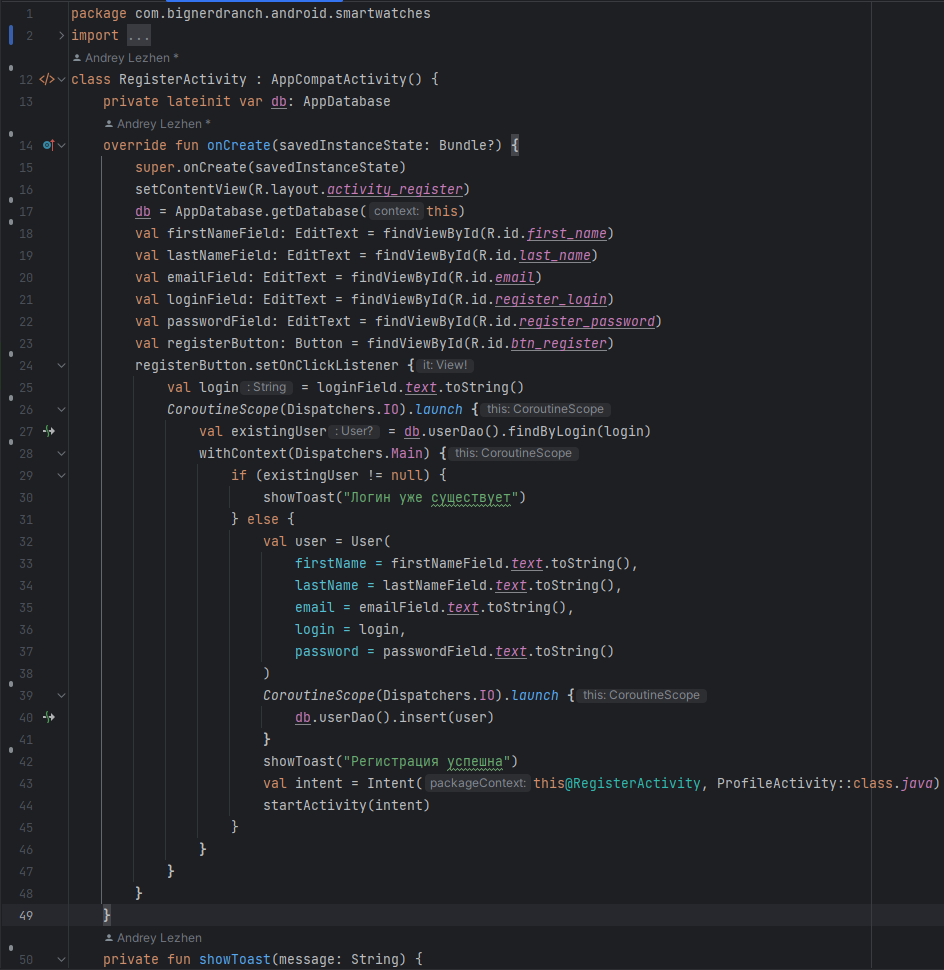


Рис. 14

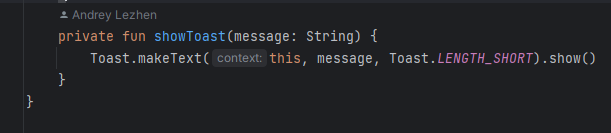


Рис. 15

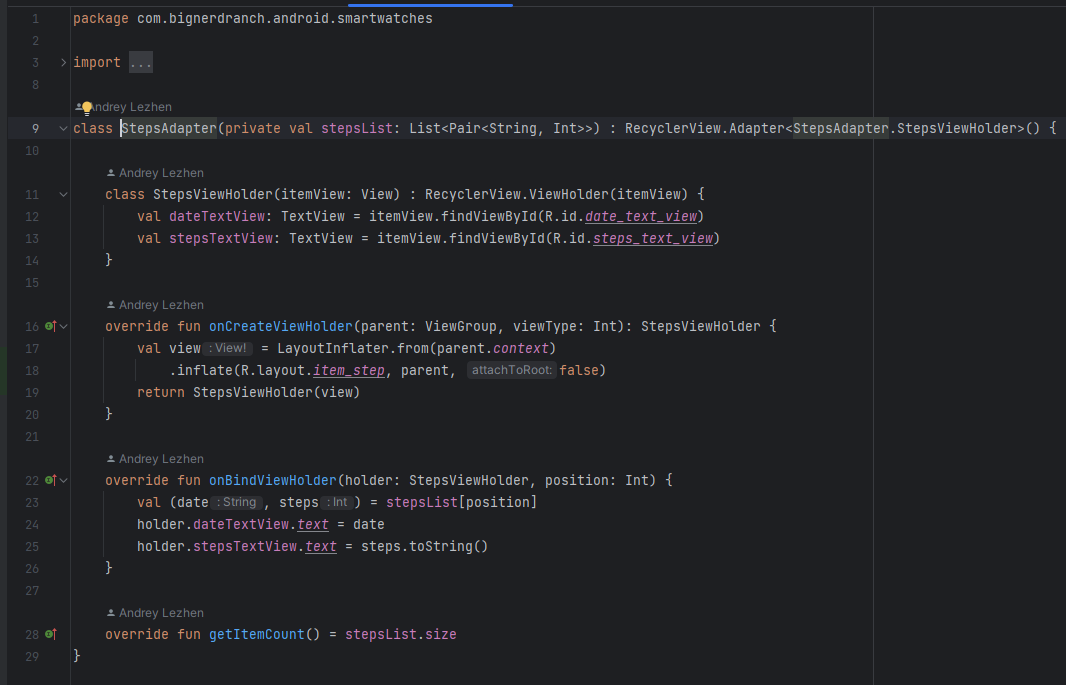


Рис. 16

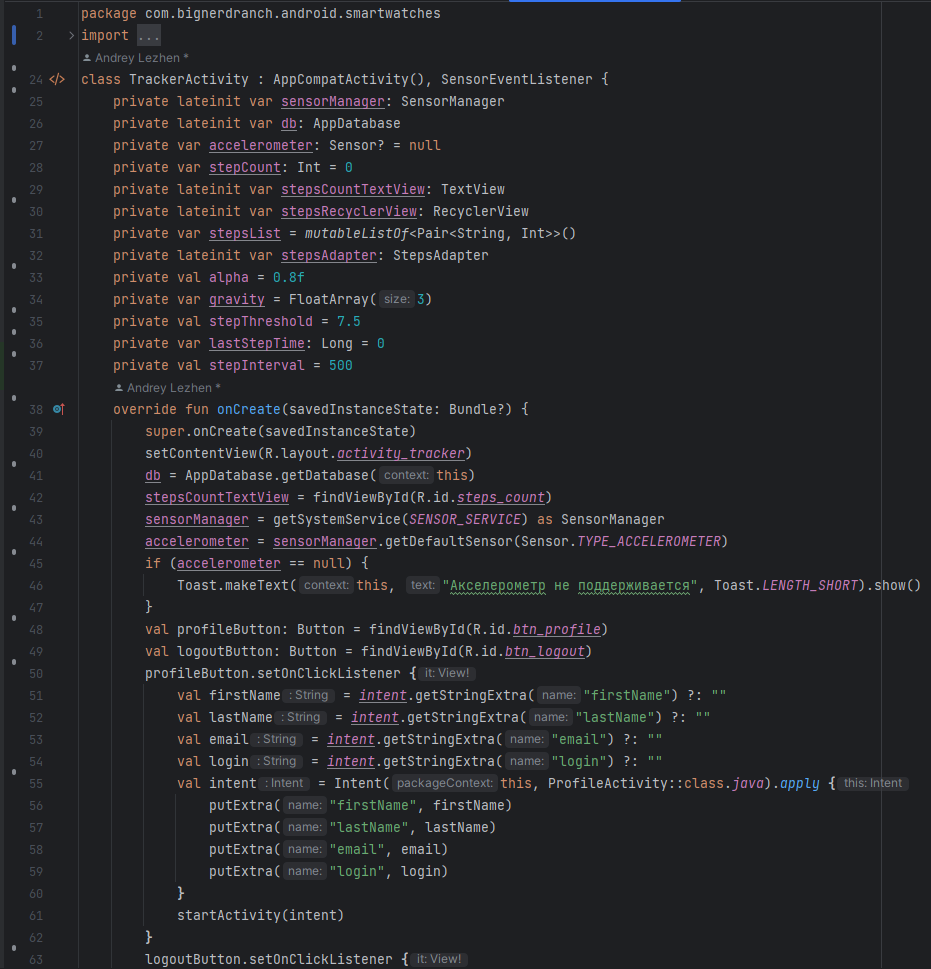


Рис. 17

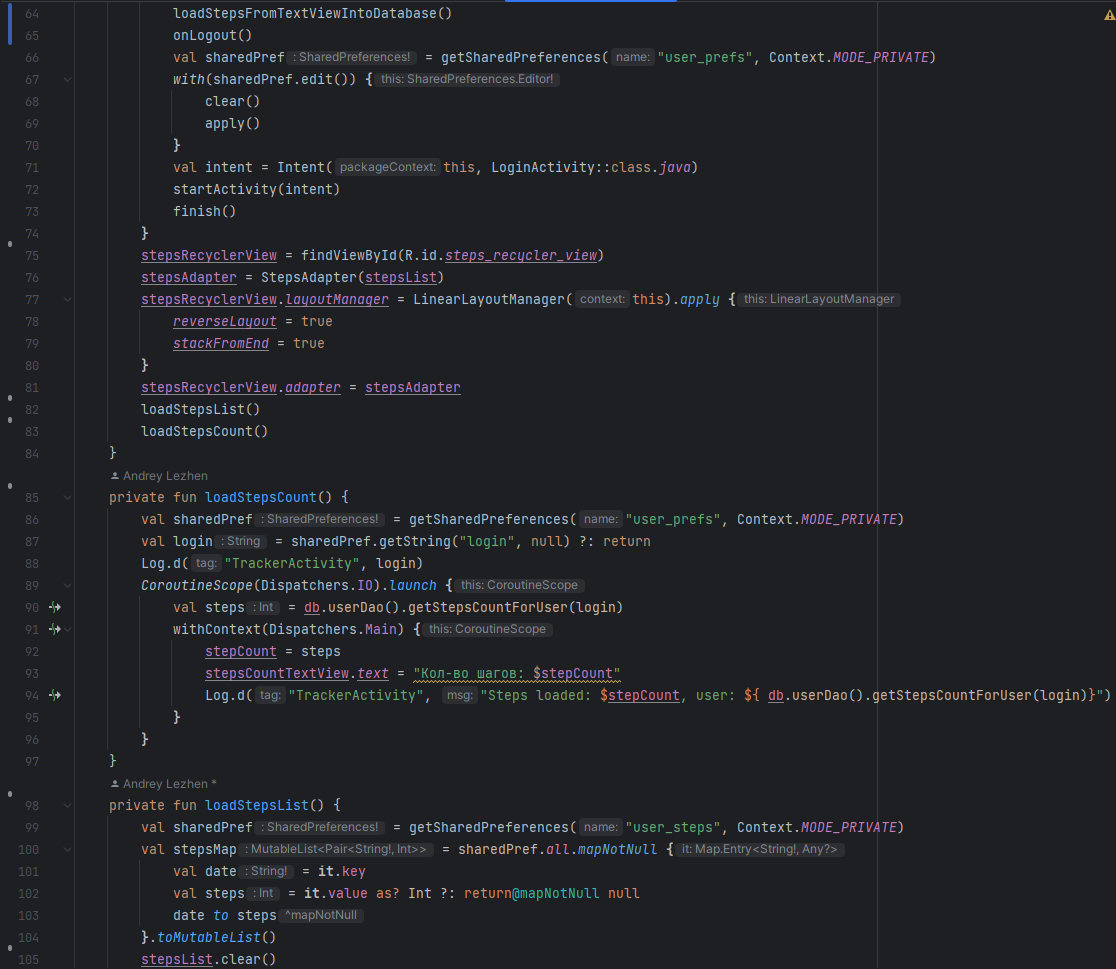


Рис.18

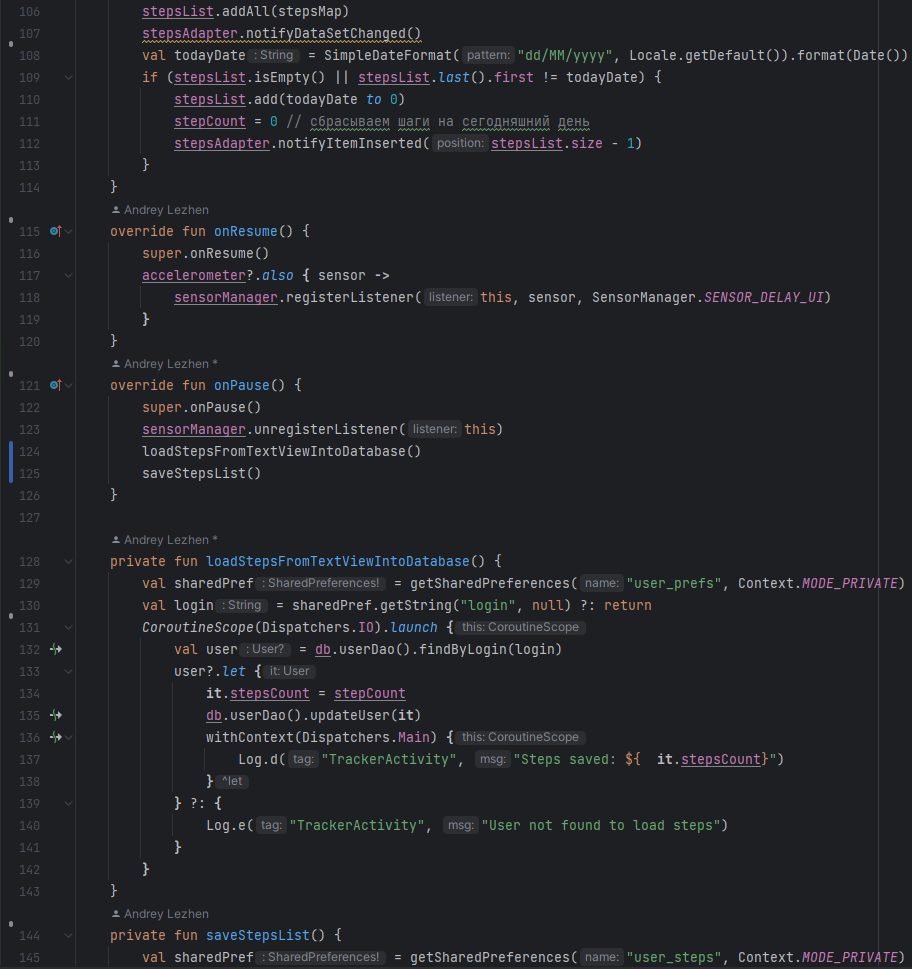


Рис. 19

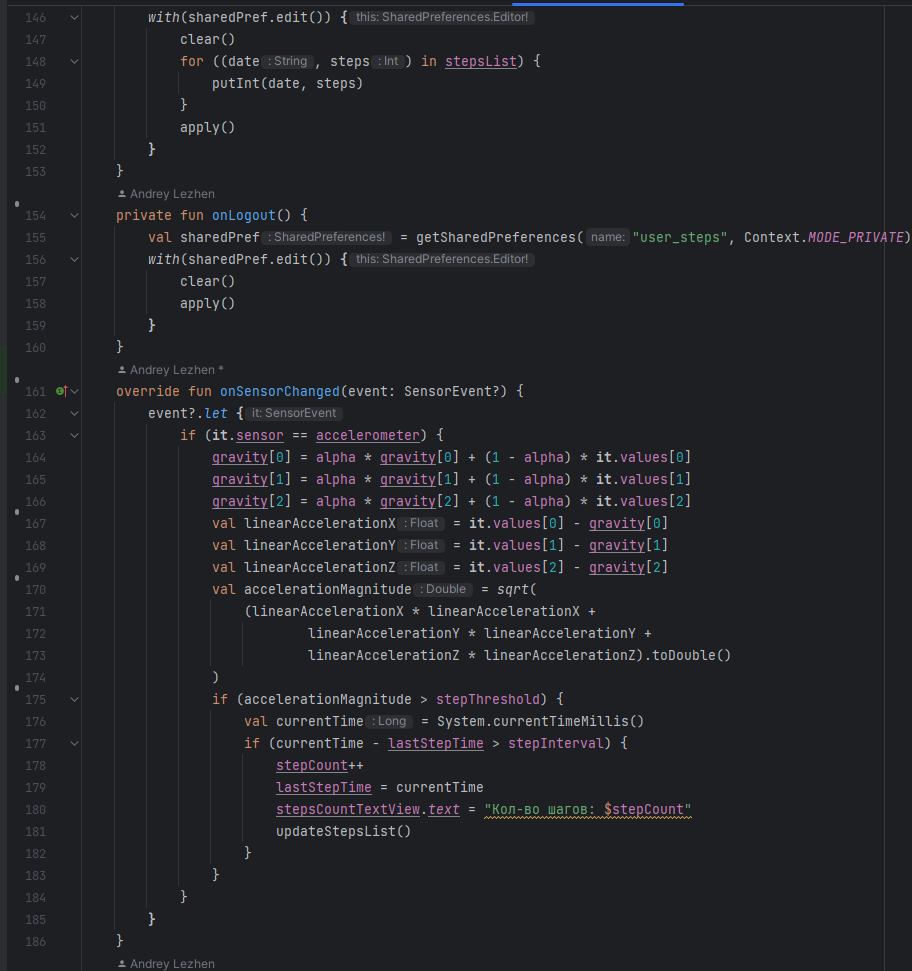


Рис. 20

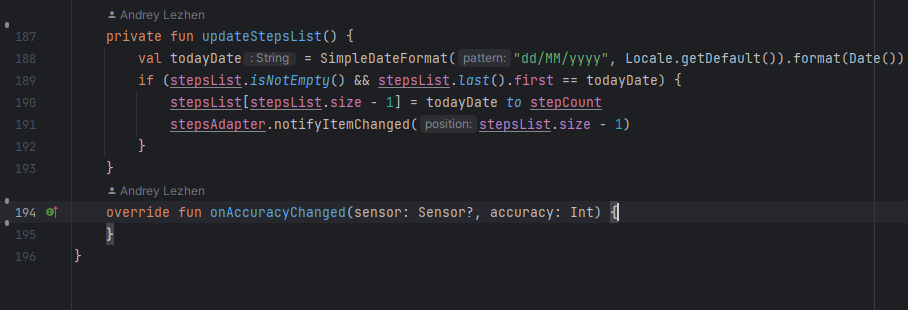


Рис. 21

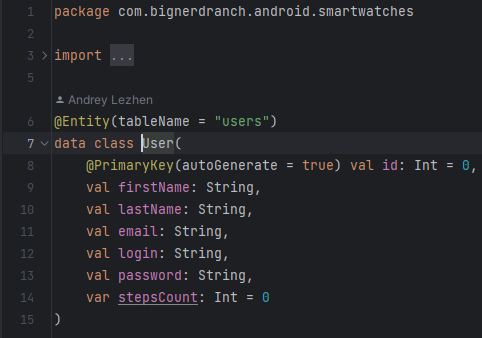


Рис. 22

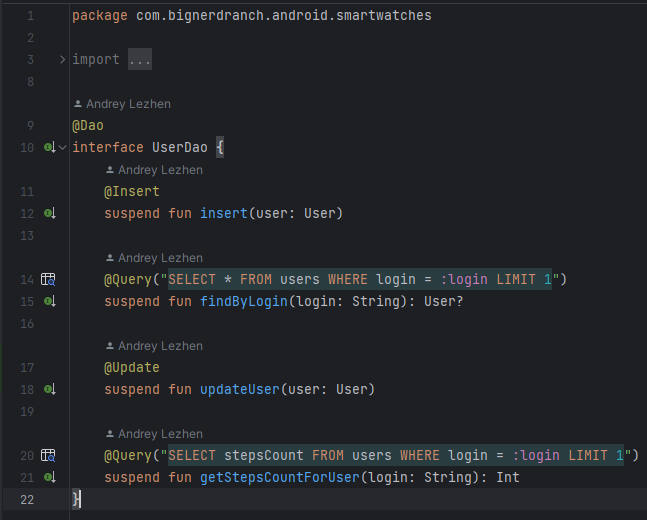


Рис. 23