

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности  
Кафедра компьютерных интеллектуальных технологий проектирования

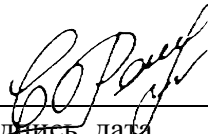
КУРСОВОЙ ПРОЕКТ

по дисциплине: «Разработка приложений для мобильных устройств»

Тема: «Создание системы учета потребления электроэнергии и воды»

Расчетно-пояснительная записка

Разработал студент  
гр. 6ПО-221

  
Подпись, дата

С.А. Ремнева

Инициалы, фамилия

Проверил

Подпись, дата

В.В. Сокольников

Инициалы, фамилия

Нормоконтролер

Подпись, дата

В.В. Сокольников

Инициалы, фамилия

Защищена \_\_\_\_\_  
дата

Оценка \_\_\_\_\_

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВГТУ», ВГТУ)

Кафедра компьютерных интеллектуальных технологий проектирования

ЗАДАНИЕ  
на курсовой проект

по дисциплине: «Разработка приложений для мобильных устройств»

Тема: «Создание системы учета потребления электроэнергии и воды»

Вариант работы: 34

Студент группы бПО-221 Ремнева Софья Алексеевна

Группа

Фамилия, имя, отчество

Технические условия: среда разработки Android Studio 25.1.04 , язык Kotlin, ОС  
Windows 11, 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz 2.69 GHz

Содержание и объем проекта (графические работы, расчеты и прочее):

46 страниц, 22 рисунка , 7 таблиц, 2 приложения

Сроки выполнения этапов: рассмотрение теоретических сведений (сентябрь – ок-  
тябрь 2025), выполнение практических заданий (ноябрь – декабрь 2025), оформле-  
ние пояснительной записки (декабрь 2025)

Срок защиты курсового проекта: декабрь 2025

Руководитель: В.В. Сокольников

Подпись, дата

Инициалы, фамилия

Задание принял студент: С.А. Ремнева

Подпись, дата

Инициалы, фамилия

Замечания руководителя

## СОДЕРЖАНИЕ

Задание на курсовой проект .....	2
Замечания руководителя.....	3
Введение .....	5
1 Сценарий использования .....	7
2 Пользовательский интерфейс.....	12
2.1 Макет интерфейса с графом переходов.....	12
2.2 Целевые устройства, обоснование требований и максимально подробные характеристики .....	14
3 Разработанное приложение .....	16
3.1 Краткое описание.....	16
3.2 Схема архитектуры.....	17
3.3 Используемые технологии .....	17
3.4 Используемые модули и системные библиотеки .....	18
4 Последовательность действий для осуществления сценариев использования.....	21
4.1 Измерение последовательности действий для осуществления сценариев использования.....	21
4.2 Пути для сокращения последовательности.....	22
Заключение.....	25
Список литературы.....	26
Приложение А. Инструкция для пользователя .....	27
Приложение Б. Снимки экрана приложения .....	37

## ВВЕДЕНИЕ

В условиях растущей осознанности в вопросах ресурсосбережения и экономии, а также в связи с постоянным увеличением тарифов на коммунальные услуги, актуальной задачей становится необходимость в эффективном и удобном инструменте для контроля потребления электроэнергии и воды. Существующие решения часто представляют собой сложные веб-порталы или устаревшие десктопные программы, которые не предоставляют пользователю оперативного доступа к данным и не обладают достаточной гибкостью для ежедневного использования. Это создает пробел между накоплением информации от счетчиков и ее практическим применением для анализа и оптимизации расходов, что в свою очередь ведет к неэффективному использованию ресурсов и неоправданным финансовым затратам.

В данной работе ставится задача разработки современной, интуитивно понятной и функциональной системы учета, которая позволит пользователям вносить показания счетчиков, отслеживать динамику потребления в наглядной форме, анализировать статистику и прогнозировать будущие расходы. Ключевыми требованиями являются обеспечение надежного хранения данных, возможность формирования детализированных отчетов за выбранные периоды и реализация персонализированных уведомлений для своевременного внесения показаний. Такая система призвана стать персональным ассистентом, переводящим рутинный учет в плоскость осознанного управления ресурсами.

Предлагаемое решение — это нативное мобильное приложение для операционной системы Android, разработанное с использованием языка Kotlin и современных компонентов фреймворка Jetpack Compose. Выбор именно этого технологического стека обусловлен стремлением создать производительный, отзывчивый и современный пользовательский интерфейс, соответствующий принципам Material Design, а также обеспечить устойчивость архитектуры приложения за счет использования рекомендованных Google паттернов, таких как ViewModel и Room для работы с данными. Приложение будет иметь модульную структуру, что облегчит дальнейшую поддержку и расширение функционала, и будет ориентировано на

безопасное локальное хранение информации с возможностью последующей интеграции с облачными сервисами.

Необходимость реализации именно в формате мобильного приложения диктуется самим характером решаемой задачи. Процесс снятия показаний счетчиков физически привязан к месту их установки, и возможность мгновенно внести данные через смартфон, который всегда под рукой, является ключевым преимуществом перед стационарными системами. Кроме того, мобильная платформа позволяет использовать push-уведомления для напоминаний, обеспечивает доступ к истории расходов в любой момент и предоставляет возможности для визуализации данных через диаграммы и графики непосредственно на экране устройства. Таким образом, мобильное приложение не просто дублирует функционал десктопной версии, а создает принципиально новый, более прямой и эффективный канал взаимодействия пользователя с данными о потреблении ресурсов, способствуя формированию ответственных привычек.

## 1 Сценарий использования

Основным сценарием использования приложения «Система учета потребления электроэнергии и воды» является регулярная фиксация показаний счетчиков и анализ потребления ресурсов пользователем. Типичный пользователь – это владелец жилья или арендатор квартиры, стремящийся к точному учету коммунальных расходов.

Сценарий использования (Use Case) демонстрирует варианты взаимодействия пользователя с системой в различных ситуациях. Использование данного компонента на этапе проектирования позволяет структурировать функциональные требования, наглядно представить последовательность действий пользователя и ответы системы, а также заложить основу для тестирования и разработки. В рамках данного проекта были выделены и формализованы следующие ключевые сценарии.

- UC1 Добавление показаний через сканирование (см. таблицу 1);
- UC2 Добавление показаний вручную (см. таблицу 2);
- UC3 Просмотр истории показаний с фильтрацией и поиском (см. таблицу 3);
- UC4 Просмотр статистики и анализ потребления (см. таблицу 4);
- UC5 Управление настройками приложения (см. таблицу 5);
- UC6 Просмотр справки и помощи (см. таблицу 6).

Таблица 1 – UC1 (Добавление показаний через сканирование)

Элемент	Описание
Описание	Пользователь использует камеру для распознавания показаний счётчика, система автоматически заполняет значение.
Акторы	Пользователь, Система, Камера, ML Kit (распознавание текста).
Начальное состояние	Пользователь находится на экране добавления показаний (AddMeterScreen).
Триггер	Пользователь нажимает кнопку «Сканировать» или значок камеры.

## Продолжение таблицы 1

Элемент	Описание
Успешный сценарий	<ol style="list-style-type: none"> <li>1. Пользователь открывает экран сканирования (CameraScanScreen).</li> <li>2. Система запрашивает разрешение на камеру (если не выдано).</li> <li>3. Пользователь наводит камеру на счётчик в область зелёной рамки.</li> <li>4. Пользователь нажимает кнопку съёмки.</li> <li>5. Система делает фото и обрезает область сканирования.</li> <li>6. ML Kit распознаёт цифры в области сканирования.</li> <li>7. Система извлекает числа и фильтрует показания (4–8 цифр).</li> <li>8. Система отображает распознанное значение в поле ввода.</li> <li>9. Пользователь подтверждает или редактирует значение.</li> <li>10. Система сохраняет фото и значение, возвращает на экран добавления.</li> </ol>
Альтернативный сценарий	<ol style="list-style-type: none"> <li>5а. Камера недоступна — система показывает ошибку и предлагает ввести вручную.</li> <li>7а. Цифры не распознаны — система показывает сообщение «Значимые цифры не найдены» и предлагает ввести вручную или повторить.</li> <li>7б. Распознано несколько чисел — система предлагает выбрать из списка или ввести вручную.</li> </ol>

Таблица 2 – UC2 (Добавление показаний вручную)

Элемент	Описание
Описание	Пользователь вручную вводит показания счётчика, выбирает тип и добавляет заметку.
Актёры	Пользователь, Система.
Начальное состояние	Пользователь находится на экране добавления показаний (AddMeterScreen).
Триггер	Пользователь заполняет форму вручную.

## Продолжение таблицы 2

Элемент	Описание
Успешный сценарий	<p>1.Пользователь выбирает тип счётчика из выпадающего списка (электричество, холодная вода, горячая вода).</p> <p>2.Система показывает последнее показание для этого типа.</p> <p>3. Пользователь вводит значение в поле «Показания».</p> <p>4.Пользователь добавляет заметку (опционально).</p> <p>5.Пользователь нажимает «Сохранить показания».</p> <p>6. Система проверяет, что значение больше предыдущего.</p> <p>7.Система сохраняет запись в БД.</p> <p>8. Система показывает уведомление об успешном сохранении и возвращает на главный экран.</p>
Альтернативный сценарий	<p>ба. Значение меньше предыдущего — система показывает ошибку «Новое показание должно быть больше предыдущего».</p> <p>За. Введено некорректное число — система подсвечивает поле и показывает сообщение об ошибке.</p>

Таблица 3 – UC3 (Просмотр историй показаний с фильтрациями)

Элемент	Описание
Описание	Пользователь просматривает историю показаний с возможностью фильтрации, поиска, сортировки и группировки.
Актёры	Пользователь, Система.
Начальное состояние	Пользователь находится на главном экране или в боковом меню.
Триггер	Пользователь нажимает «История» в боковом меню или кнопку истории на главном экране.
Успешный сценарий	<p>1.Система открывает экран истории (HistoryScreen).</p> <p>2.Система загружает все показания из БД.</p> <p>3. Пользователь использует поисковую строку для поиска по заметкам.</p> <p>4.Пользователь открывает панель фильтров.</p> <p>5.Пользователь выбирает период (сегодня/неделя/месяц/год/всё время).</p> <p>6. Пользователь выбирает тип счётчика для фильтрации.</p> <p>7. Пользователь выбирает сортировку (по дате/значению).</p> <p>8. Пользователь выбирает группировку (по дате/типу/без группировки).</p> <p>9.Система применяет фильтры и обновляет список.</p> <p>10.Пользователь просматривает отфильтрованный список сгруппированных записей.</p>

### Продолжение таблицы 3

Элемент	Описание
Альтернативный сценарий	2а. Нет показаний — система показывает экран «Нет показаний» с предложением добавить первое. 9а. Нет данных по выбранным фильтрам — система показывает сообщение «Показания не найдены» и предлагает изменить фильтры.

Таблица 4 — UC4 — Просмотр статистики и анализа

Элемент	Описание
Описание	Пользователь анализирует потребление ресурсов через графики, KPI и сравнение периодов.
Акторы	Пользователь, Система.
Начальное состояние	Пользователь находится в боковом меню или на главном экране.
Триггер	Пользователь нажимает «Статистика» в боковом меню.
Успешный сценарий	1. Система открывает экран статистики (StatisticsScreen) 2. Пользователь выбирает период анализа (день/неделя/месяц/год). 3. Пользователь фильтрует данные по типу счётчика (опционально). 4. Система отображает KPI: всего, в день, максимум 5. Система строит линейный график динамики потребления. 6. Система показывает сравнение с предыдущим периодом 7. Система показывает распределение по типам (если фильтр не выбран) 8. Система отображает последние показания..
Альтернативный сценарий	2а. Данных недостаточно — система показывает сообщение «Недостаточно данных для анализа». 4а. Нет данных — система показывает «Нет данных» в карточках KPI.

Таблица 5 — UC5 — Управление настройками

Элемент	Описание
Описание	Пользователь настраивает параметры приложения: уведомления, тему, резервное копирование.
Акторы	Пользователь, Система.
Начальное состояние	Пользователь находится в боковом меню.

## Продолжение таблицы 5

Элемент	Описание
Триггер	Пользователь нажимает «Настройки» в боковом меню.
Успешный сценарий	1. Система открывает экран настроек (SettingsScreen). 2. Пользователь включает/выключает уведомления.
Альтернативный сценарий	3а. Ошибка при создании резервной копии — система показывает сообщение об ошибке.

Таблица 6 — UC6 — Просмотр справки и помощи

Элемент	Описание
Описание	Пользователь получает ответы на частые вопросы, инструкции и контакты поддержки.
Акторы	Пользователь, Система.
Начальное состояние	Пользователь находится в боковом меню.
Триггер	Пользователь нажимает «Помощь» в боковом меню.
Успешный сценарий	1. Система открывает экран помощи (HelpScreen). 2. Пользователь просматривает FAQ, раскрывая интересные вопросы. 3. Пользователь читает инструкции по использованию функций. 4. Пользователь переходит к контактам поддержки (email, телефон, чат). 5. Пользователь скачивает руководство пользователя (если доступно). 6. Пользователь получает необходимую информацию и закрывает экран.
Альтернативный сценарий	4а. Контакты недоступны — система показывает сообщение «Служба поддержки временно недоступна».

## 2 Пользовательский интерфейс

### 2.1 Макет интерфейса с графом переходов

Архитектура пользовательского интерфейса приложения построена на основе единого навигационного графа, реализованного с помощью компонента Jetpack Navigation Compose. Граф определен в MainActivity.kt внутри функции AppNavigation, где NavHost является контейнером, управляющим переходами между экранами (composable-назначениями). Начальной точкой входа (startDestination) является главный экран (Routes.MAIN\_SCREEN).

Граф переходов включает 9 ключевых экранов, каждый из которых представляет отдельный функциональный модуль системы:

- главный экран (MainScreen) — дашборд с краткими сводками, последними показаниями и быстрым доступом к основным функциям.
- экран добавления показаний (AddMeterScreen) — форма для ручного ввода данных или перехода к сканированию;
- экран истории (HistoryScreen) — список всех показаний с расширенными возможностями фильтрации, поиска и сортировки;
- экран статистики (StatisticsScreen) — визуализация потребления через KPI-карточки, графики и сравнение периодов;
- экран настроек (SettingsScreen) — управление параметрами приложения, резервным копированием и данными;
- экран помощи (HelpScreen) — FAQ, инструкции и контакты поддержки.
- экран "О приложении" (AboutScreen) — информация о версии, разработчиках и функциях;
- экран сканирования камерой (CameraScanScreen) — полноэкранный интерфейс для захвата и распознавания показаний счетчика;
- экран предпросмотра фото (PhotoPreviewScreen) — просмотр и подтверждение сделанного снимка.

Логика переходов организована по принципу стека. Большинство экранов имеют навигационную панель (AppBar) с кнопкой "Назад" (`popBackStack()`), обеспечивая линейную навигацию "вглубь". Основные переходы инициируются:

- из бокового меню (NavigationDrawerContent на MainScreen) — для перехода к разделам "История", "Статистика", "Настройки", "Помощь", "О приложении".
- с главного экрана — через FAB (кнопка с плюсом) на экран добавления показаний и через карточку "Вся история" на экран истории.
- с экрана добавления показаний — через кнопку "Сканировать" на экран камеры.
- из экрана сканирования — возврат с результатом на экран добавления через механизм `savedStateHandle` или переход на экран предпросмотра фото при успешном распознавании.

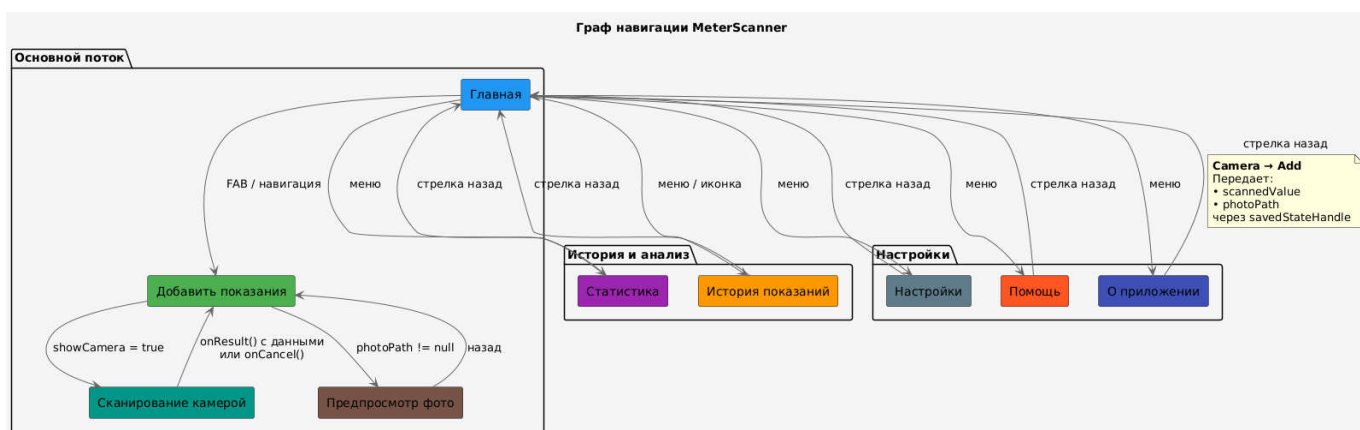


Рисунок 2.1 – Граф перехода в PlantUML

Особенностью навигации является использование параметризованных маршрутов (например, `Routes.PHOTO_PREVIEW_SCREEN`) для передачи данных между экранами (путь к фото, распознанное значение).

Модальные окна (диалоги редактирования, удаления, деталей) реализованы не как отдельные экраны навигационного графа, а как компоненты поверх основного интерфейса с использованием `AlertDialog`, что соответствует принципам Material Design для контекстных действий.

Структура интерфейса построена на компонентах Scaffold и Material 3, что обеспечивает единообразие: на большинстве экранов присутствует TopAppBar, FloatingActionButton (где уместно) и содержимое, организованное в LazyColumn или Column.

Боковое меню (ModalNavigationDrawer) доступно только с главного экрана, предоставляя доступ ко всем ключевым разделам приложения, что делает навигацию интуитивно понятной и соответствующей современным стандартам UX для Android.

## 2.2 Целевые устройства, обоснование требований и максимально подробные характеристики

Разрабатываемое приложение ориентировано на пользователей мобильных устройств под управлением операционной системы Android. Данная платформа выбрана в силу ее широкого распространения на российском рынке, что обеспечивает максимальный потенциальный охват целевой аудитории. В контексте задачи учета коммунальных ресурсов критически важно, чтобы приложение было доступно на устройствах разного ценового сегмента, включая бюджетные модели, так как необходимость контроля расходов часто не зависит от уровня дохода пользователя. Следовательно, ключевым требованием является обеспечение стабильной работы и приемлемой производительности на широком спектре аппаратного обеспечения.

Минимальные системные требования определены с балансом между доступностью и необходимостью поддержки современных библиотек разработки. Для целевой операционной системы установлена минимальная версия Android 8.0 (API level 26, Oreo). Выбор данной версии обусловлен несколькими факторами. Во-первых, она предоставляет достаточную базу для использования современных компонентов Jetpack, включая Jetpack Compose, который хотя и имеет лучшую поддержку с API 21, но стабильно работает с API 26, обеспечивая доступ к ключевым функциям. Во-вторых, согласно актуальной статистике распространения платформ, совокупная доля устройств на Android 8.0 и выше покрывает подавляющее большинство активно-

го парка, что делает разработку под более ранние версии нецелесообразной с точки зрения трудозатрат. Для полноценной работы всех функций приложения устройство должно поддерживать Google Play Services, что является стандартом для большинства пользовательских устройств вне специфических сегментов.

Аппаратные требования детализированы для каждой ключевой функции. Для основного функционала ввода и отображения данных достаточно 1.5 ГБ оперативной памяти (RAM) и одноядерного процессора с частотой около 1.2 ГГц. Однако функции, связанные с обработкой изображений, предъявляют повышенные требования. Режим «Сканирование» с использованием камеры в реальном времени и машинного обучения для распознавания цифр потребует более производительного процессора (рекомендуется двухъядерный от 1.8 ГГц) и не менее 2 ГБ RAM для эффективной работы ML-моделей.

Требования к дисплею определены как поддержка плотности пикселей `hdpi` (около 240 dpi) и выше, что гарантирует читаемость текста и элементов интерфейса, особенно важную при работе с цифрами показаний. Приложение спроектировано с использованием Responsive и Adaptive Design принципов в Compose. Это означает, что интерфейс корректно отображается и функционирует на смартфонах с различными соотношениями сторон (от 18:9 до 21:9) и размерами экранов (от 4.5 до 6.7 дюймов по диагонали). Для планшетов или устройств с foldable-экранами будет применена более сложная адаптивная логика размещения компонентов (например, использование Navigation Rail вместо BottomNavigation и двухпанельных макетов), однако данная адаптация выходит за рамки текущей версии и зарезервирована для будущих обновлений.

### 3 Разработанное приложение

#### 3.1 Краткое описание

Разрабатываемое приложение представляет собой мобильную систему учета потребления коммунальных ресурсов (электроэнергии и воды) для платформы Android. Его основное назначение — предоставить пользователю удобный, точный и централизованный инструмент для замены традиционных бумажных журналов учета или неудобных таблиц в электронных заметках. Ядром системы является возможность надежной фиксации показаний счетчиков с минимальной вероятностью ошибки. Для этого, помимо ручного ввода, реализованы продвинутые механизмы получения данных: сканирование табло счетчика в реальном времени с помощью камеры смартфона и машинного обучения, а также загрузка и анализ заранее сделанных фотографий. Все данные, включая прикрепленные изображения для архивного подтверждения, сохраняются в структурированной локальной базе данных на устройстве пользователя.

Вторым ключевым функциональным блоком является модуль аналитики и визуализации. Приложение преобразует накопленные числовые данные в наглядные интерактивные отчеты: столбчатые диаграммы для отслеживания динамики потребления по месяцам и круговые диаграммы для анализа структуры расходов. Это позволяет пользователю не просто хранить историю, а активно анализировать свои паттерны потребления.

Важной отличительной чертой приложения является его сфокусированность именно на учете, а не на оплате. Оно сознательно исключает любой функционал, связанный с проведением платежей, работой с банковскими картами или онлайн-оплатой квитанций. Такой подход позволяет сосредоточить усилия на точности, удобстве ввода данных и глубине анализа, обеспечивая пользователю цифровой, надежный и персональный архив его потребления ресурсов. Приложение работает преимущественно в оффлайн-режиме, гарантируя доступ к истории и базовым функциям в любой момент, а для загрузки сервисной информации и работы с картами использует сетевые запросы по требованию.

## 3.2 Схема архитектуры

Данное приложение построено по принципу чистой архитектуры с разделением на три основных слоя. Пользовательский интерфейс состоит из экранов для отображения главной страницы, добавления показаний счетчиков, сканирования через камеру и просмотра истории. Все экраны взаимодействуют с ViewModel-слоем, который содержит бизнес-логику и управление состоянием приложения.

Уровень данных включает репозиторий для работы с информацией о счетчиках и локальную базу данных для постоянного хранения. ViewModel-слой делегирует операции с данными репозиторию, который в свою очередь взаимодействует с базой данных. Это обеспечивает абстракцию и разделение ответственности между компонентами.

Для работы со сканированием используется отдельный CameraViewModel, который координирует работу с внешними сервисами — CameraX для захвата изображений и ML Kit для распознавания текста. Полученные данные передаются в основной MeterViewModel для дальнейшей обработки и сохранения, создавая сквозной поток данных от сканирования до хранения информации.

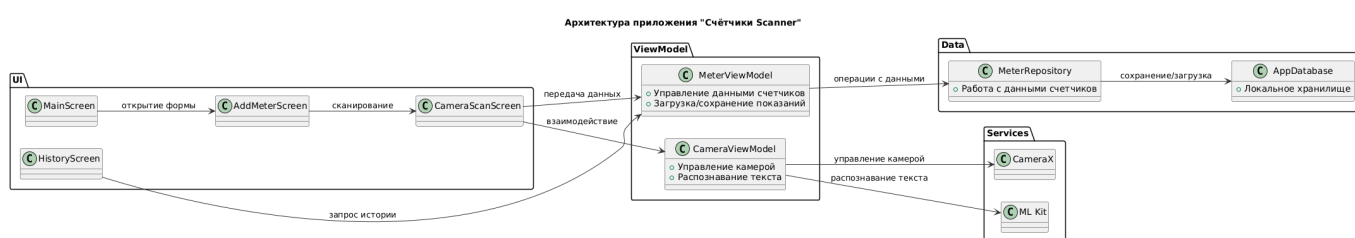


Рисунок 3.1 – Схема архитектуры

## 3.3 Используемые технологии

В рамках выполнения курсовой работы были использованы современные технологии и библиотеки, соответствующие требованиям к производительности, безопасности и удобству разработки. Приложение построено на базе платформы Android

с использованием языка Kotlin и фреймворка Jetpack Compose для создания декларативного пользовательского интерфейса. Данная технология позволяет разрабатывать адаптивные и отзывчивые интерфейсы с минимальным количеством шаблонного кода, что существенно ускоряет процесс разработки и упрощает поддержку кода. В качестве системы навигации использован компонент Navigation Compose, который обеспечивает удобную и безопасную навигацию между экранами приложения с поддержкой передачи параметров и сохранения состояния.

Для работы с данными приложение использует архитектурный паттерн Model-View-ViewModel (MVVM) в сочетании с компонентами Android Jetpack, такими как LiveData и StateFlow, для реактивного управления состоянием интерфейса. Это позволяет эффективно разделять бизнес-логику и представление, обеспечивая высокую тестируемость и поддерживаемость кода. Локальное хранение данных реализовано с помощью Room Persistence Library, которая предоставляет абстракцию над SQLite и позволяет безопасно и производительно работать с базой данных. Для сканирования показаний счетчиков с использованием камеры устройства задействована библиотека CameraX, обеспечивающая единый API для работы с камерой на различных версиях Android, а также ML Kit от Google для распознавания текста на изображениях. Эти технологии позволяют реализовать функцию автоматического распознавания показаний с высокой точностью и минимальными требованиями к вычислительным ресурсам.

### 3.4 Используемые модули и системные библиотеки

В процессе разработки были использованы следующие модули и системные библиотеки:

- Jetpack Compose с компонентами Material Design 3 – для создания современного адаптивного интерфейса, соответствующего гайдлайнам Material You;
- Navigation Compose – для реализации навигации между экранами с поддержкой передачи аргументов и сохранения состояния;

- Kotlin Coroutines – для асинхронного выполнения операций работы с базой данных, обработки изображений и сетевых запросов без блокировки основного потока;
- Room Persistence Library – для локального хранения структурированных данных с использованием SQLite;
- CameraX – для унифицированного доступа к камере устройства и съёмки изображений;
- ML Kit (Text Recognition) – для распознавания цифровых показаний со счетчиков на фотографиях;
- Android KTX и Jetpack Lifecycle – для упрощённой работы с компонентами жизненного цикла и расширениями Kotlin;
- Timber – для удобного логирования в различных конфигурациях сборки;
- JUnit и Espresso – для модульного и интерфейсного тестирования соответственно;
- Gradle – для управления зависимостями, сборки и конфигурации проекта.

Основу приложения составляет набор компонентов Jetpack Compose, включающий Material Design 3 для создания современного интерфейса, соответствующих гайдлайнам Material You. Для асинхронной работы и управления фоновыми задачами применяется библиотека Kotlin Coroutines, которая позволяет выполнять длительные операции, такие как работа с базой данных или обработка изображений, без блокировки основного потока. Для обработки изображений и работы с графикой используются системные библиотеки Android, такие как `android.graphics` и `androidx.camera`, а также утилиты для декодирования и кодирования изображений в различных форматах. Сетевые возможности, хотя и не являются основными в данном приложении, могут быть расширены за счет библиотек Retrofit и OkHttp для взаимодействия с внешними API, например, для синхронизации данных с облачным хранилищем.

Для обеспечения безопасности данных применяются стандартные механизмы Android, такие как разрешения для доступа к камере и файловой системе, а также шифрование локальной базы данных. Логирование и отладка осуществляются с помощью библиотеки Timber, которая упрощает мониторинг работы приложения в различных режимах сборки. Тестирование проводится с использованием фреймворков JUnit и Espresso для модульного и UI-тестирования соответственно, что гарантирует стабильность и надежность приложения. Все зависимости управляются через систему Gradle, что обеспечивает удобное обновление библиотек и контроль за версионностью.

## 4 Последовательность действий для осуществления сценариев использования

### 4.1 Измерение последовательности действий для осуществления сценариев использования

Оценка эффективности пользовательского взаимодействия с приложением проводилась через анализ критических сценариев использования (Use Case), определённых в разделе 1. Основным метрическим показателем выступило количество пользовательских действий (тапов и свайпов), необходимое для достижения целевого результата. Для каждого сценария был проведён замер в условиях стандартного использования на эталонном устройстве (Google Pixel 4a, Android 13). Результаты представлены в Таблице 7.

Таблица 7 – Количество шагов для ключевых сценариев

Сценарий (Use Case)	Краткое описание	Минимальное число шагов	Стандартное число шагов (с учётом проверки данных)	Сценарий (Use Case)
UC1 – Добавление через сканирование	От главного экрана до сохранения rozpoznанных показаний	5	7	UC1 – Добавление через сканирование
UC2 – Ручной ввод	От выбора типа счетчика до сохранения данных	4	6	UC2 – Ручной ввод
UC3 – Просмотр истории с фильтрами	Открытие истории, применение двух фильтров и сортировки	3 (без фильтров)	7 (с фильтрацией и сортировкой)	UC3 – Просмотр истории с фильтрами

Наиболее оптимальным с точки зрения количества действий является сценарий ручного ввода (UC2), требующий всего 4 обязательных шага: открытие экрана добавления, выбор типа счетчика, ввод значения и подтверждение сохранения. Это обеспечивается продуманной компоновкой формы на экране AddMeterScreen, где

фокус автоматически переходит между ключевыми полями, а кнопка сохранения активируется только при валидных данных.

Сценарий сканирования (UC1), несмотря на технологическую сложность, также оптимизирован: после запуска камеры процесс сведения к минимуму рутинных операций. Основные шаги пользователя сводятся к наведению камеры и однократному тапу для съёмки. Последующая обработка изображения, его обрезка до области сканирования и распознавание текста выполняются автоматически, без участия пользователя. Однако общее количество шагов увеличивается за счёт необходимости подтверждения или корректировки распознанного значения, что является важным этапом для обеспечения точности данных. Таким образом, баланс между автоматизацией и контролем со стороны пользователя сохранён.

Сценарии работы с историей (UC3) и статистикой (UC4) демонстрируют нелинейную зависимость числа шагов от глубины анализа. Базовый просмотр требует минимальных действий, однако расширенные возможности (многоуровневая фильтрация, сравнение периодов) закономерно увеличивают путь взаимодействия. Важным инженерным решением здесь стало реализация компактных и контекстных панелей фильтров, которые появляются по требованию, не перегружая основной интерфейс постоянными элементами управления.

## 4.2 Пути для сокращения последовательности

На основе проведённого анализа были выявлены потенциальные точки оптимизации пользовательских сценариев, направленные на дальнейшее сокращение числа необходимых действий без ущерба для функциональности и контроля над данными.

1. Контекстные действия и прогнозируемый ввод. Для сценария UC2 (ручной ввод) в качестве основной оптимизации предлагается внедрение системы интеллектуальных подсказок. При выборе типа счетчика система, основываясь на истории предыдущих внесений, может мгновенно предложить наиболее вероятное следующее значение, рассчитанное по среднему приросту за последние периоды. Пользова-

телю останется лишь подтвердить предложенный вариант одним тапом. Алгоритм может быть реализован на уровне MeterViewModel с использованием простой линейной регрессии по последним 3-5 показаниям. Это сократит стандартный сценарий ручного ввода с 6 до 3-4 шагов.

2. Пакетная обработка и съёмка в реальном времени для UC1. Текущая реализация сценария сканирования предполагает статичную съёмку. Эффективность можно повысить, реализовав непрерывный режим предпросмотра с живым распознаванием. В этом режиме ML Kit будет анализировать видеопоток с камеры в реальном времени, а не единичный снимок. Как только система с заданной долей уверенности обнаружит и распознает набор цифр, соответствующий шаблону показаний счетчика (4-8 цифр, возможна точка), она автоматически зафиксирует результат и предложит его пользователю. Это позволит сократить фазу наведения и съёмки, объединив её с этапом распознавания, и потенциально уменьшить количество шагов до 4-5.

3. Персонализированные быстрые фильтры и сохранённые пресеты для UC3 и UC4. Анализ поведения пользователя может выявить наиболее часто используемые комбинации фильтров (например, «электричество за текущий месяц» или «вода за прошлый год»). Интерфейс можно дополнить панелью «Избранных фильтров» на экранах истории и статистики. Пользователь, единожды настроив сложный фильтр, сможет сохранить его под именем и применять в будущем одним тапом. Это трансформирует нелинейный многошаговый процесс в быстрый доступ к персонализированным представлениям данных. Технически это может быть реализовано через сохранение сериализованных объектов конфигурации фильтров в SharedPreferences или в отдельной таблице Room.

4. Жесты и навигация. В дополнение к точечным оптимизациям конкретных сценариев, значительный выигрыш в скорости взаимодействия может дать внедрение системы жестовой навигации. Например, свайп вниз на главном экране для быстрого доступа к форме добавления показаний или долгое нажатие на карточку счетчика для вызова контекстного меню с действиями «редактировать»/«удалить».

Эти приёмы, будучи интуитивно понятными опытным пользователям, позволяют совершать целевые действия, минуя стандартные пути через меню и кнопки.

Внедрение даже части этих оптимизаций потребует тщательного юзабилити-тестирования, так как избыточная автоматизация может привести к ошибкам и снижению чувства контроля у пользователя. Тем не менее, предложенные пути демонстрируют потенциал для эволюции интерфейса от реактивного к проактивному, где приложение не просто выполняет команды, а предугадывает намерения пользователя на основе контекста и исторических данных, сводя рутинные операции к абсолютному минимуму.

## ЗАКЛЮЧЕНИЕ

В ходе курсового проекта разработано мобильное приложение «Система учета потребления электроэнергии и воды» для Android. Цель создания — упрощение фиксации показаний счетчиков и анализ потребления ресурсов — достигнута.

Реализованы ключевые функции:

- ручной ввод показаний и автоматическое распознавание через камеру с ML Kit;
- локальное хранение данных в базе Room;
- просмотр истории и статистики с фильтрацией и графиками;
- интуитивный интерфейс на Jetpack Compose.

Приложение построено по чистой архитектуре с использованием Kotlin, Jetpack Compose, Room, CameraX и ML Kit, что обеспечило производительность и возможность расширения.

Таким образом, проект соответствует поставленным задачам и может использоваться как персональный инструмент для учета коммунальных расходов соответствует поставленным задачам и может быть использовано в качестве персонального инструмента для учета потребления ресурсов, способствуя формированию ответственного отношения к расходованию энергии и воды. Проект демонстрирует практическое применение современных технологий разработки под Android и может служить основой для дальнейшего развития — например, добавления синхронизации с облачными сервисами или интеграции с умными счетчиками.

## СПИСОК ЛИТЕРАТУРЫ

- 1 Бунина, Л. В. Разработка мобильных приложений : учебное пособие / Л. В. Бунина, А. П. Титов. — Москва : РТУ МИРЭА, 2025 — Часть 1 — 2025. — 123 с.
- 2 Коузен, К. Kotlin. Сборник рецептов / К. Коузен ; перевод с английского А. Н. Киселева. — Москва : ДМК Пресс, 2021. — 220 с.
- 3 Начало работы с Kotlin [Электронный ресурс] // Официальная документация Kotlin. — Электрон. дан. — Режим доступа: <https://kotlinlang.org/docs/getting-started.html>. — (Дата обращения: 20.10.2025).
- 4 Петросян, Л. Э. Разработка мобильных приложений на Kotlin : учебное пособие / Л. Э. Петросян, Н. А. Приходько. — Москва : РТУ МИРЭА, 2024. — 101 с. — ISBN 978-5-7339-2215-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/421091> (дата обращения: 10.11.2025). — Режим доступа: для авториз. пользователей.
- 5 Разработка приложений для Android с помощью Compose [Электронный ресурс] // Android Developers. — Электрон. дан. — Режим доступа: <https://developer.android.com/courses/android-basics-compose/course>. — (Дата обращения: 24.11.2025).
- 6 Степанов, П. В. Разработка мобильных приложений под Android на языке Kotlin : учебное пособие / П. В. Степанов, Н. А. Приходько, А. И. Запорожских. — Москва : РТУ МИРЭА, 2025. — 889 с.
- 7 MeterScanner [Электронный ресурс] : репозиторий исходного кода / lyacsevna // GitHub. — Режим доступа: <https://github.com/lyacsevna/MeterScanner> (дата обращения: 13.12.2025).

## Приложение А. Инструкция для пользователя

Данная инструкция описывает основные функции мобильного приложения «Учёт ресурсов», предназначенного для учета показаний счетчиков электроэнергии и воды. После установки приложения нажмите на его иконку для запуска. При первом запуске откроется главный экран (Рисунок А1), на котором отображается сводка по последним показаниям и кнопки быстрого доступа. Для перехода к другим разделам используйте боковое меню (Рисунок А2), открывающееся по свайпу вправо или нажатию на иконку меню в верхнем левом углу.

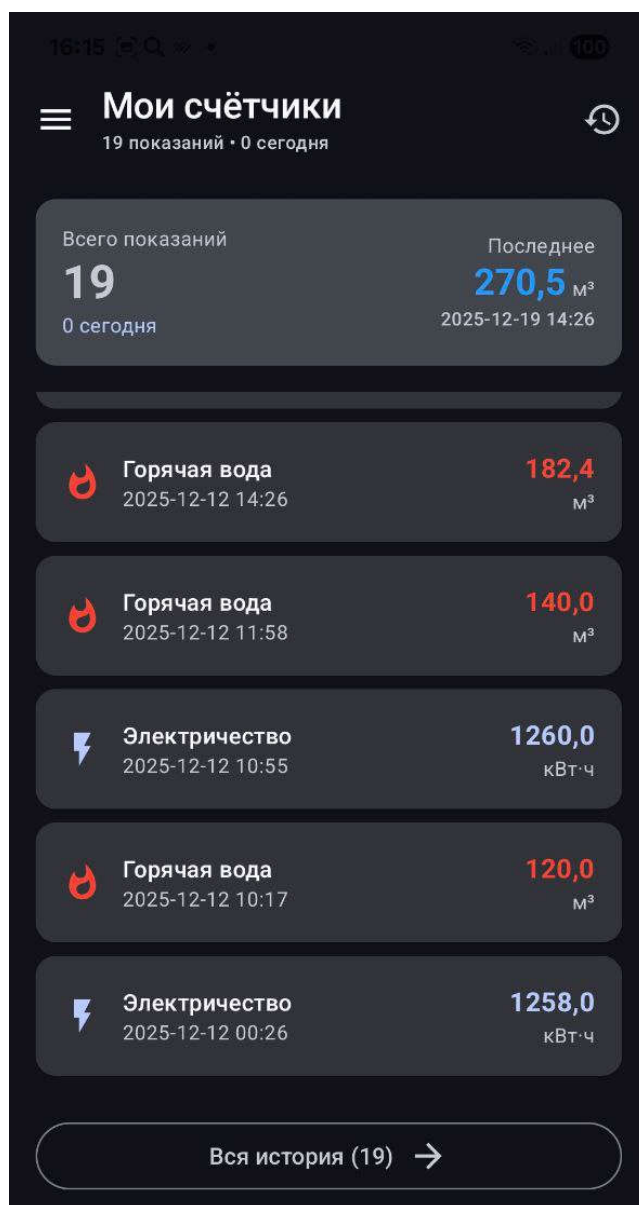


Рисунок А1 – Главное меню

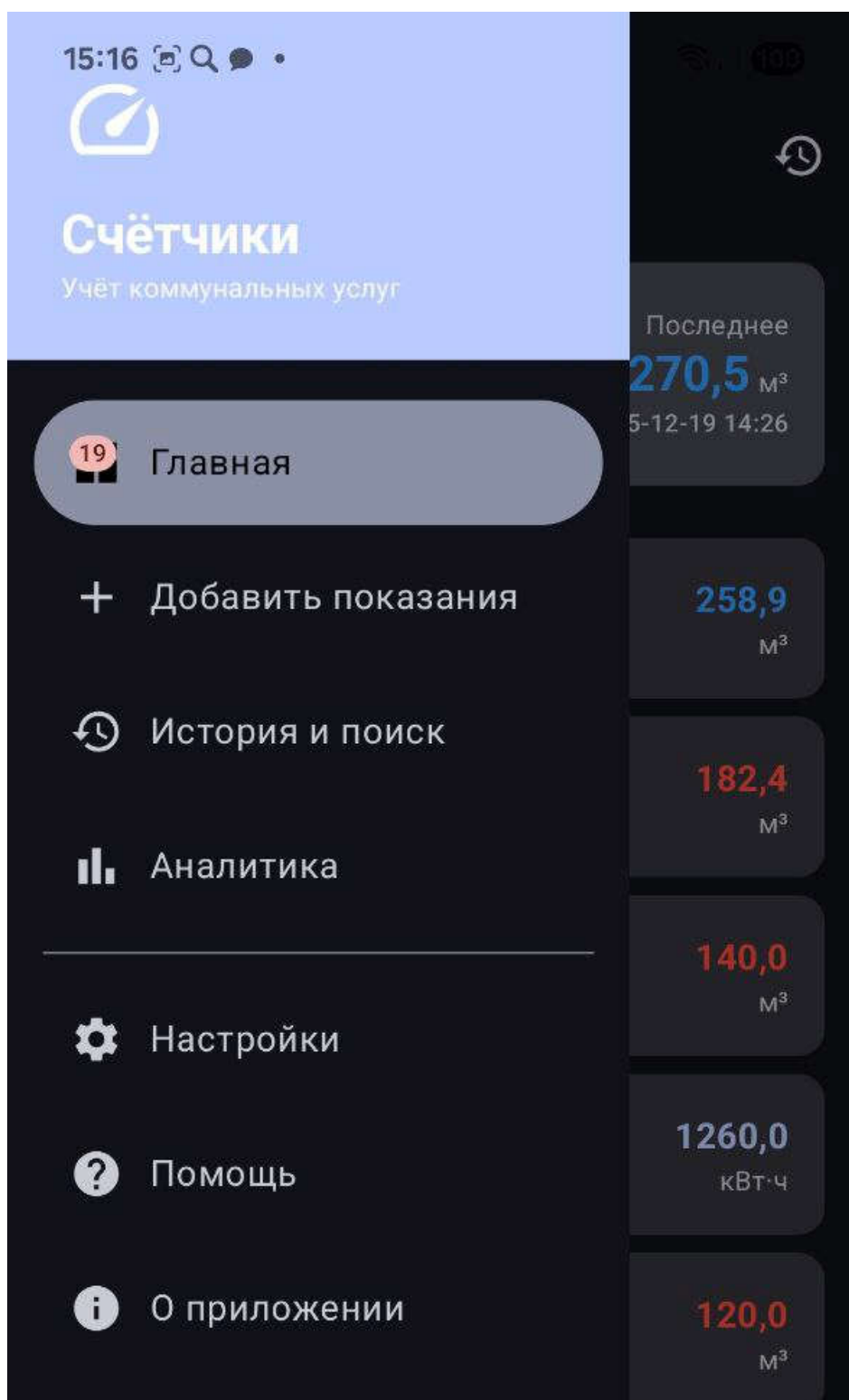


Рисунок А2 – Боковая панель навигации

Для добавления данных откройте раздел «Добавить». Существует два способа внесения показаний: ручной ввод и сканирование. На экране ввода (Рисунок А3) выберите тип счётчика из списка, введите текущие показания в соответствующее поле, при необходимости добавьте комментарий и нажмите кнопку «Сохранить».

Рисунок А3 – Ручной ввод

Добавление через сканирование. На экране ввода нажмите кнопку «Сканировать» (Рисунок А4). Наведите камеру на табло счётчика так, чтобы цифры оказались в зелёной рамке, и нажмите кнопку съёмки. Приложение распознает цифры и подставит их в поле для ввода. Подтвердите или отредактируйте значение и сохраните.

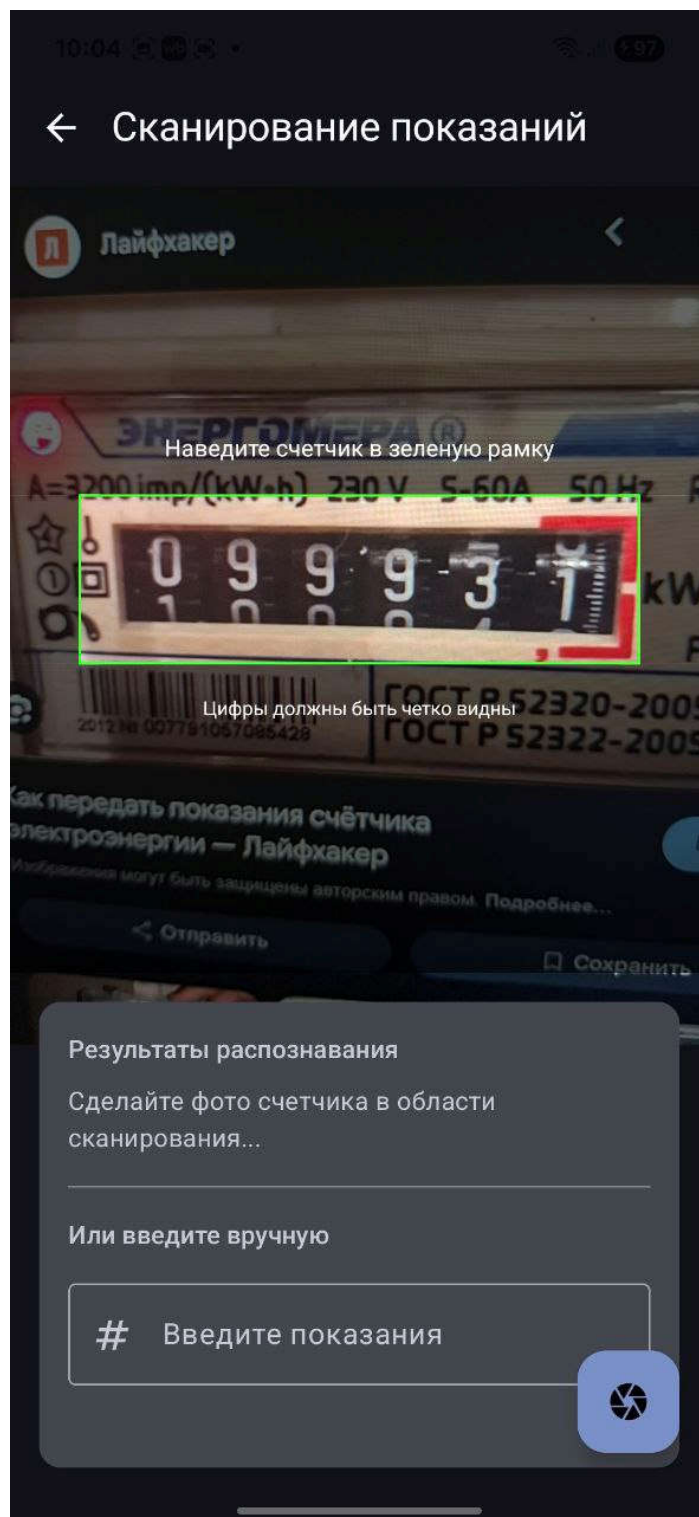


Рисунок А4 – Сканирование

В разделе «История» (Рисунок А5) отображается список всех ранее внесённых показаний. Для удобства поиска используйте строку поиска по заметкам или панель фильтров (Рисунок А6). В фильтрах можно выбрать тип счётчика, период, а также настроить сортировку и группировку записей.

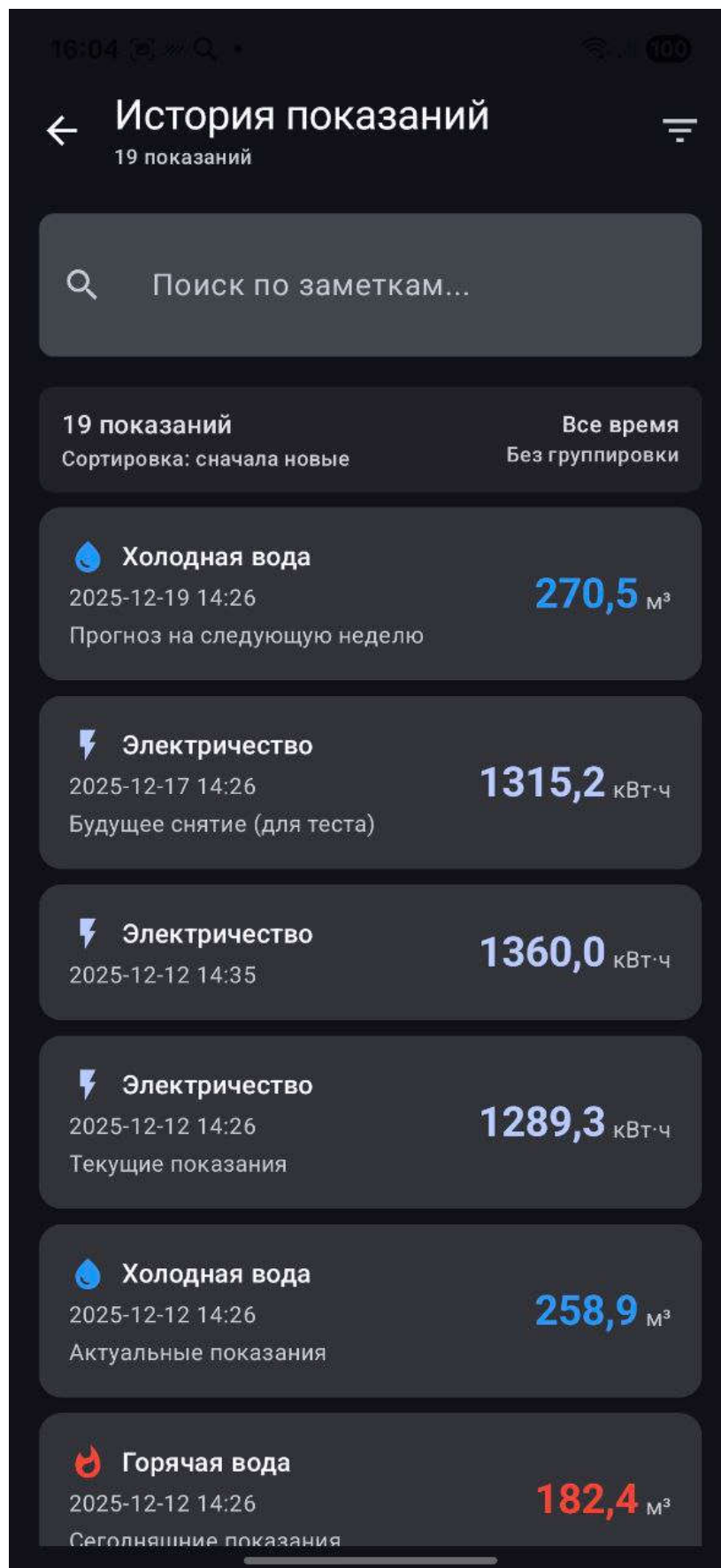


Рисунок А5 – История показаний

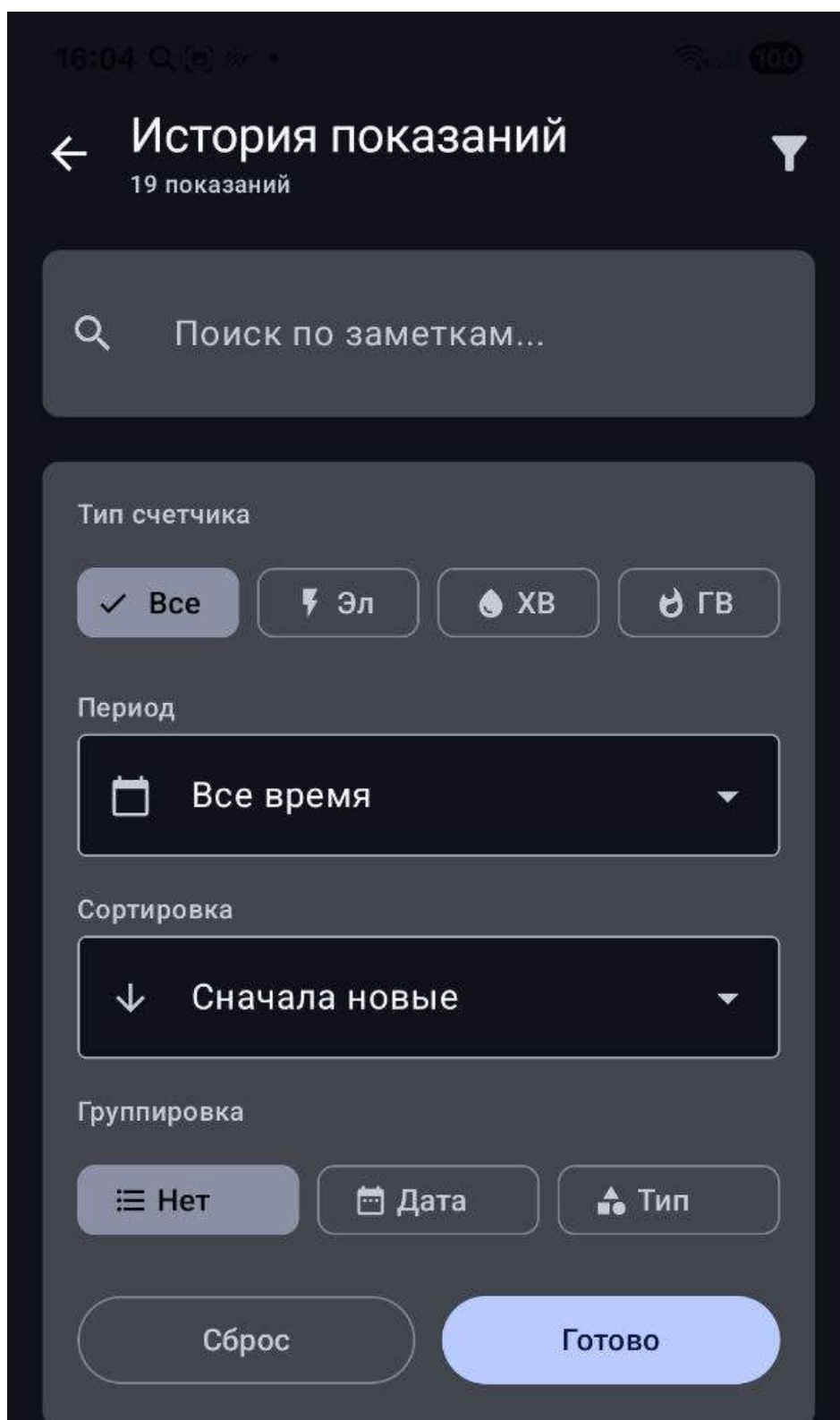


Рисунок А6 – Панель фильтров

Раздел «Статистика» (Рисунок А7) предназначен для визуального анализа потребления. Выберите интересующий период и тип ресурса. Приложение отобразит

ключевые показатели (общее потребление, среднее в день) и построит график динамики расходов.

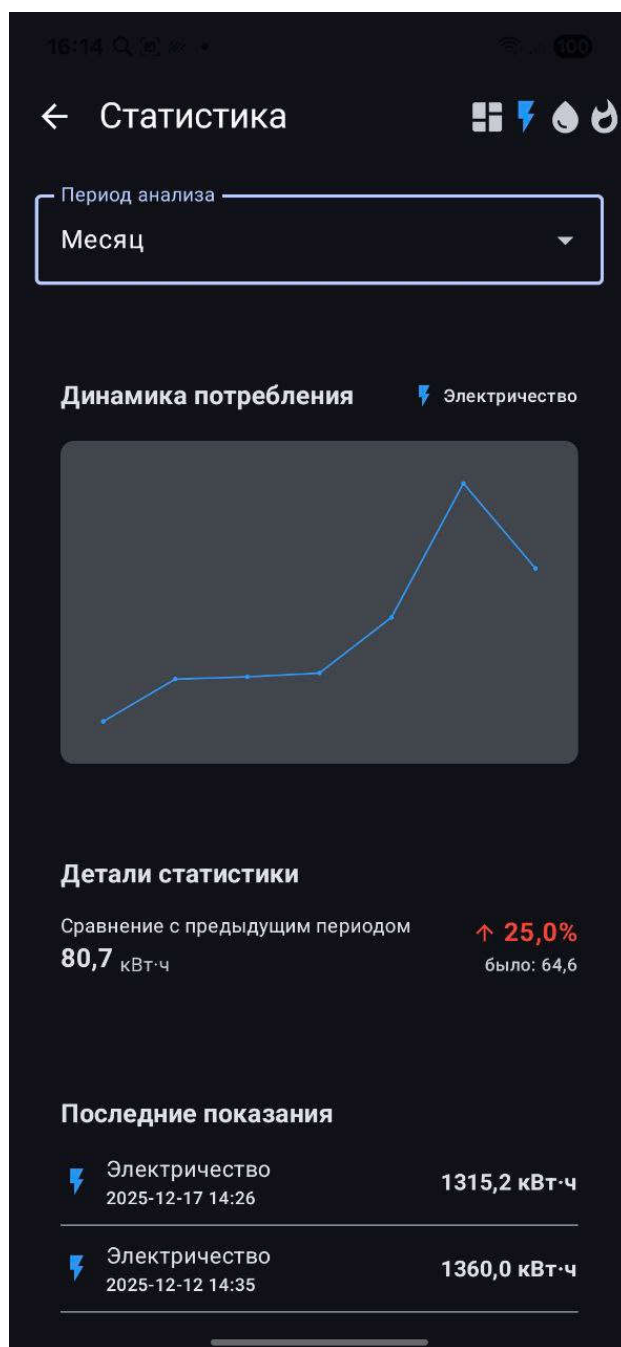


Рисунок А7 – Статистика

В разделе «Настройки» (Рисунок А8) можно включить или отключить уведомления-напоминания, выбрать тему оформления и выполнить резервное копирование данных.

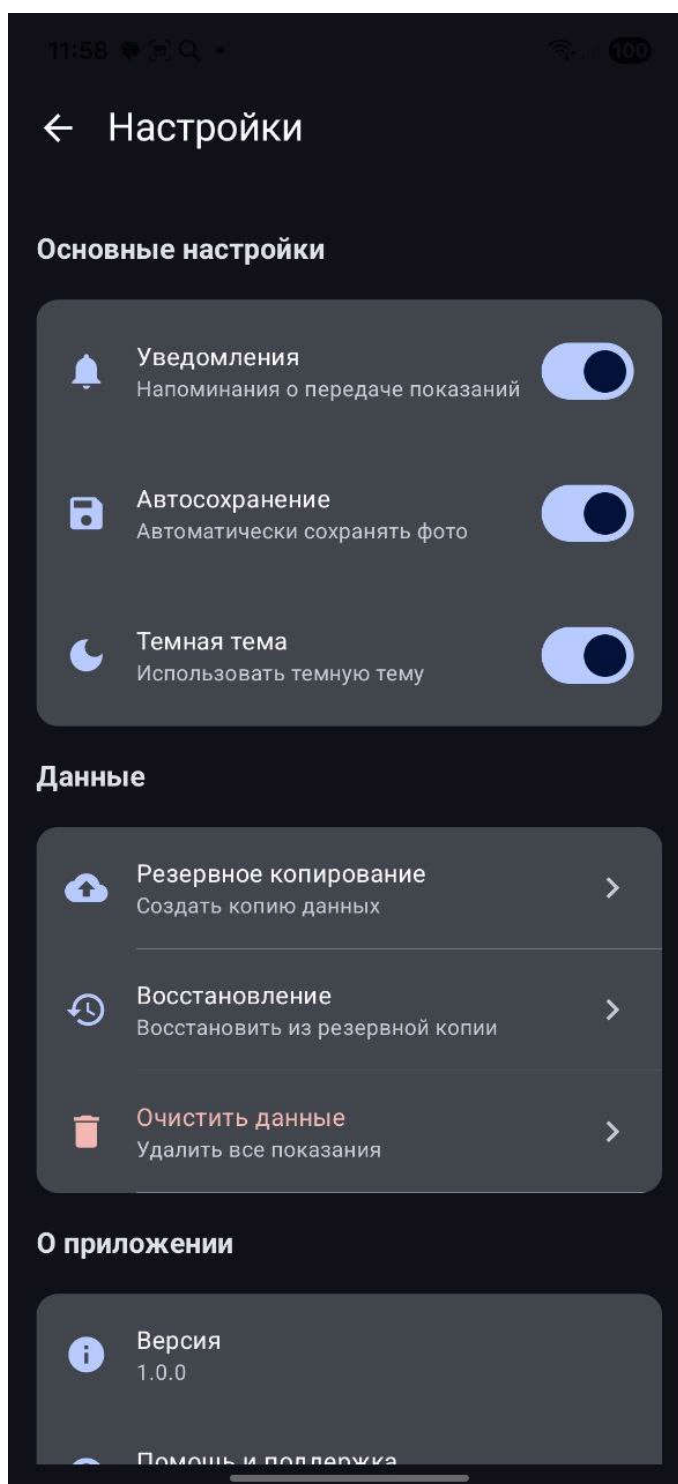


Рисунок А8 – Настройки

В разделе «Помощь» (Рисунок А9) содержатся ответы на часто задаваемые вопросы, инструкции по использованию функций и контакты поддержки. Информация о версии приложения доступна в разделе «О программе» (Рисунок А10).

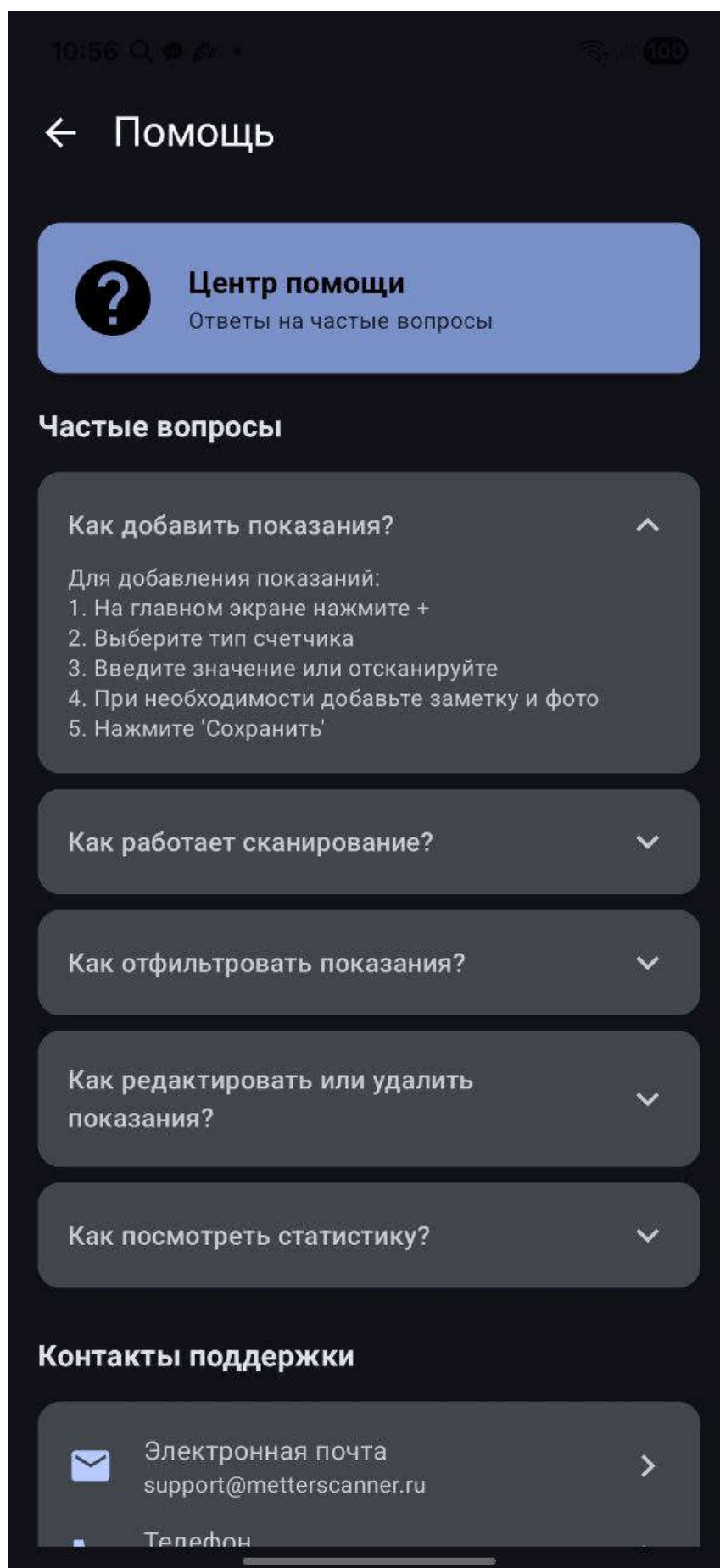


Рисунок А9 – Помощь

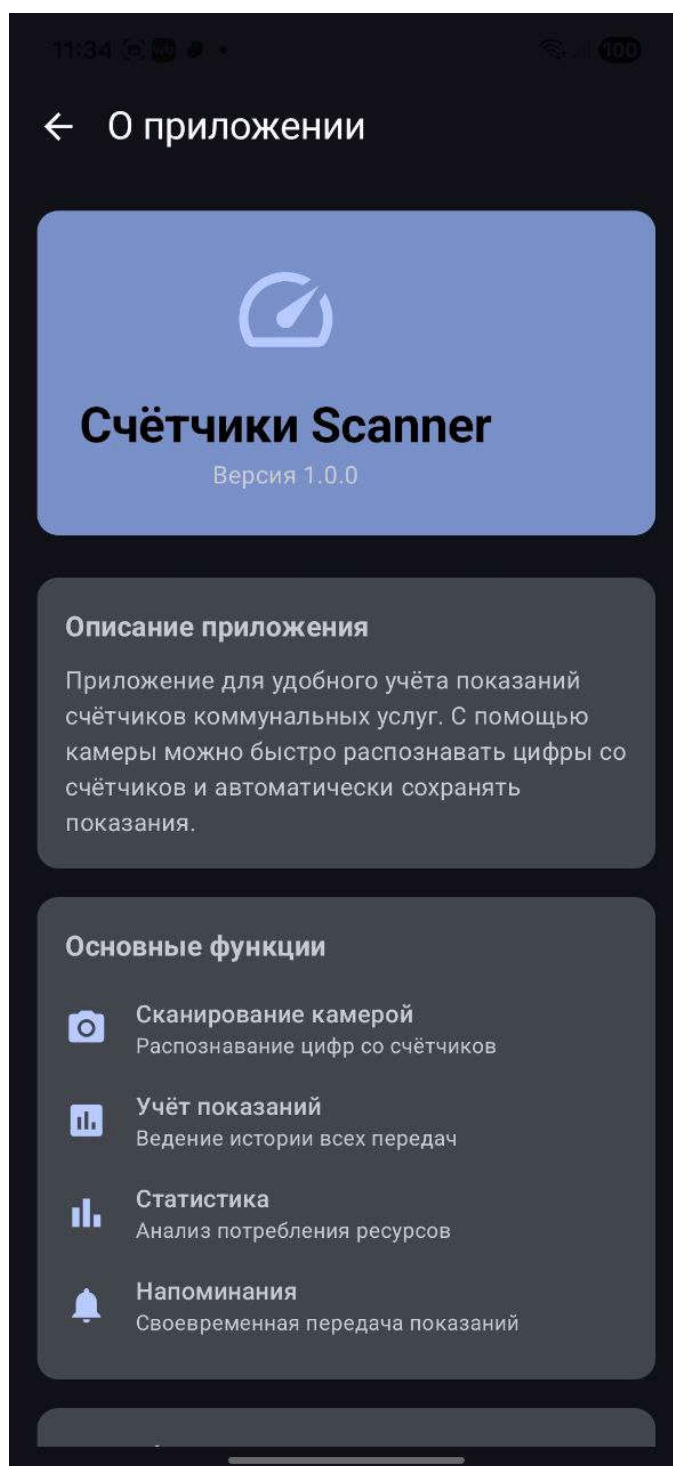


Рисунок А10 – О приложении

Приложение работает преимущественно в офлайн-режиме, все данные хранятся локально на устройстве. Для работы функции сканирования необходимы разрешение на использование камеры и установленные сервисы Google Play. Рекомендуется регулярно выполнять резервное копирование данных через настройки.

## Приложение Б. Снимки экрана приложения

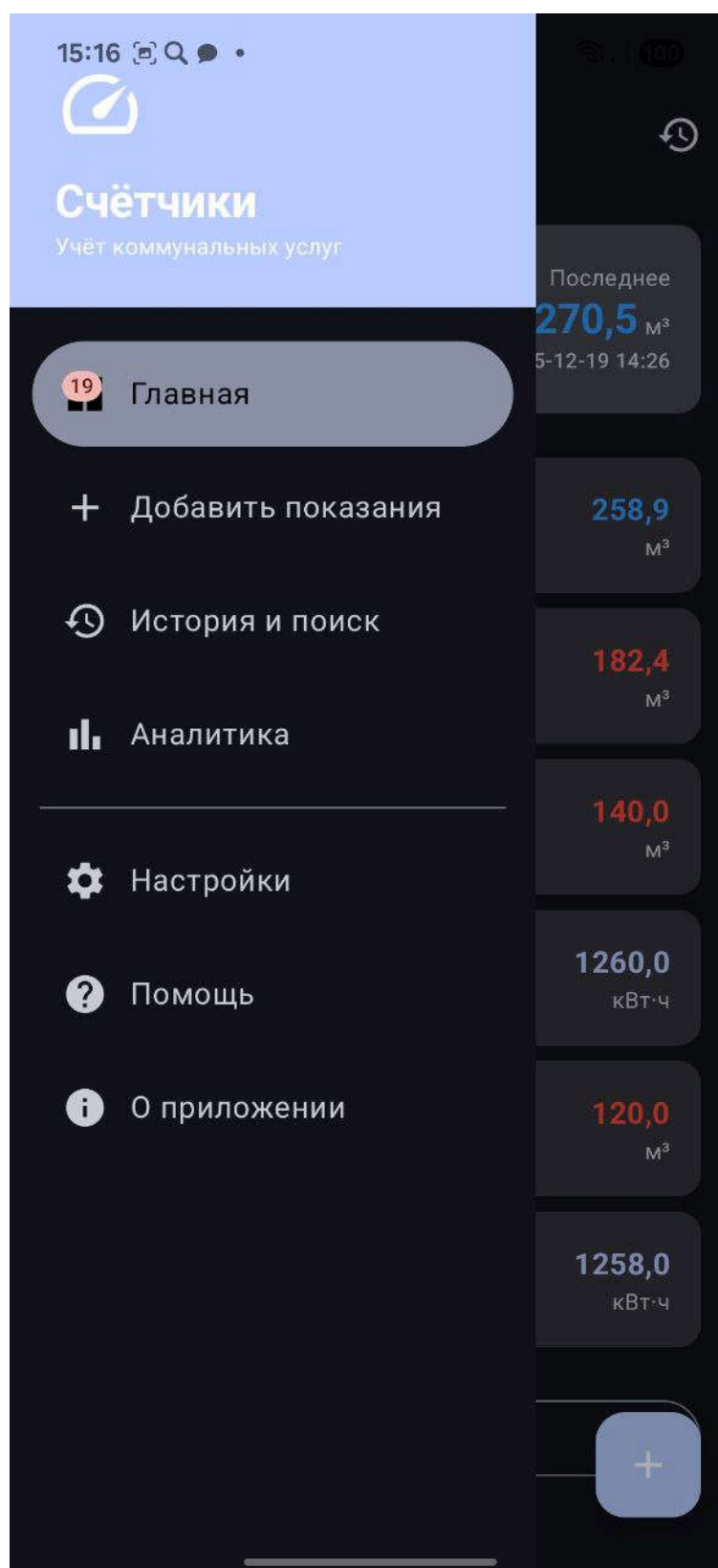


Рисунок Б1 – Боковая панель навигации

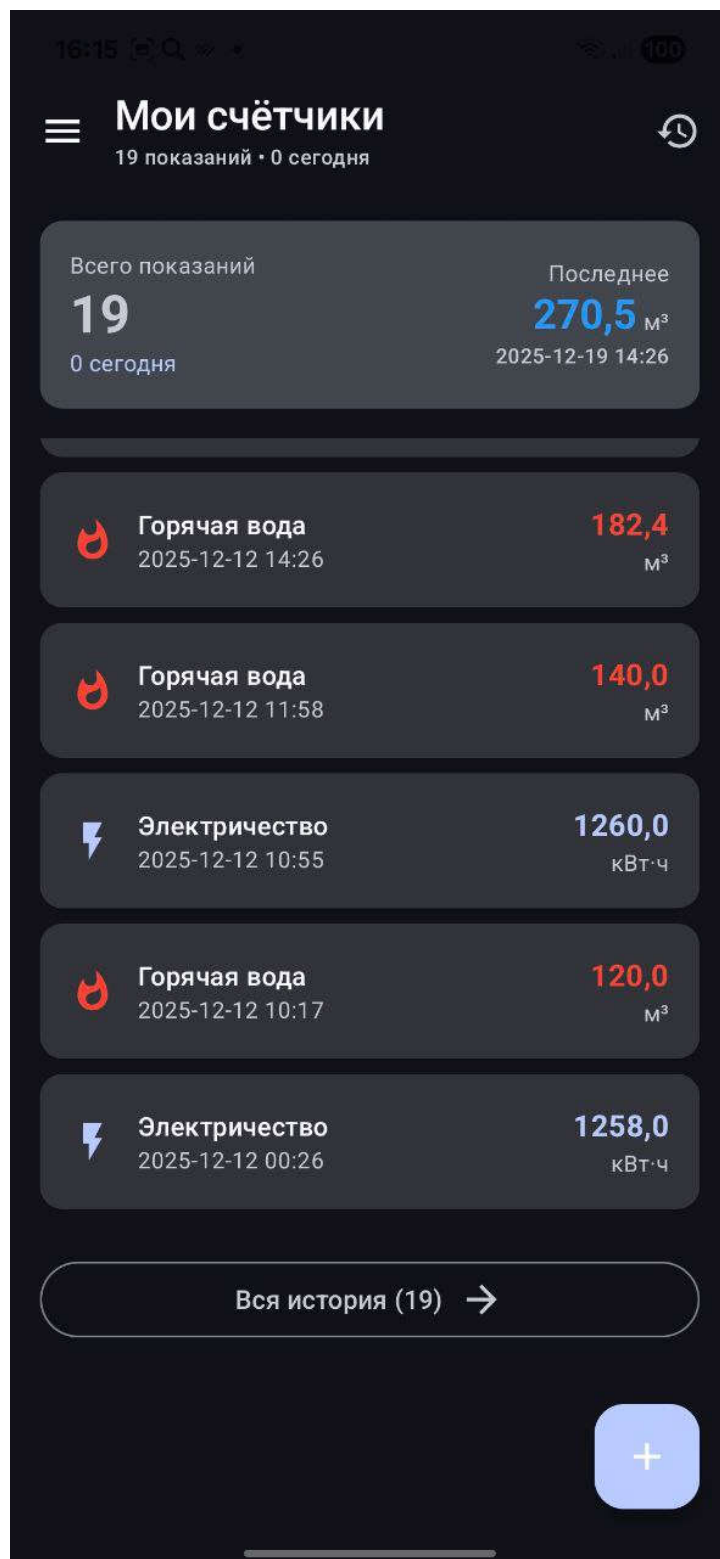


Рисунок Б2 – Главное меню

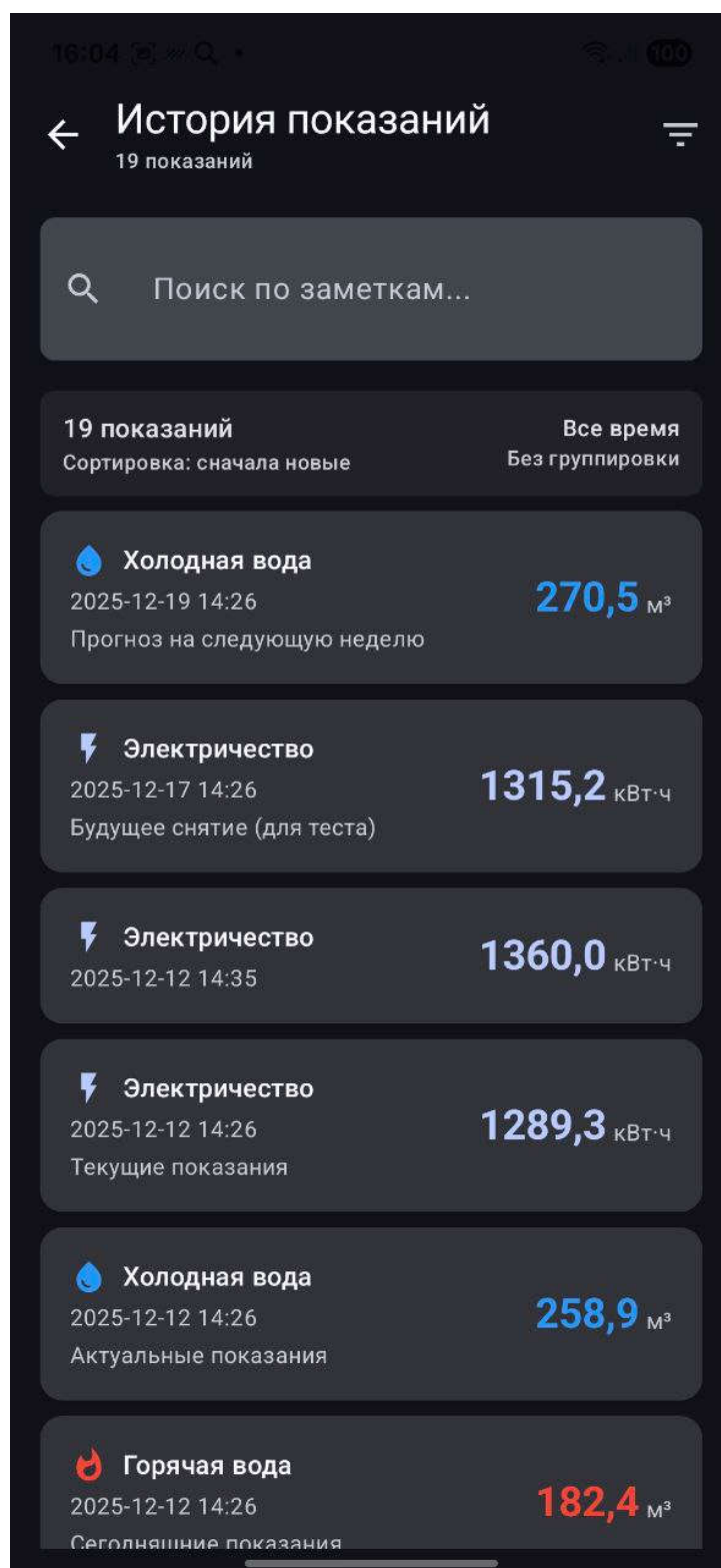


Рисунок Б3 – История показаний

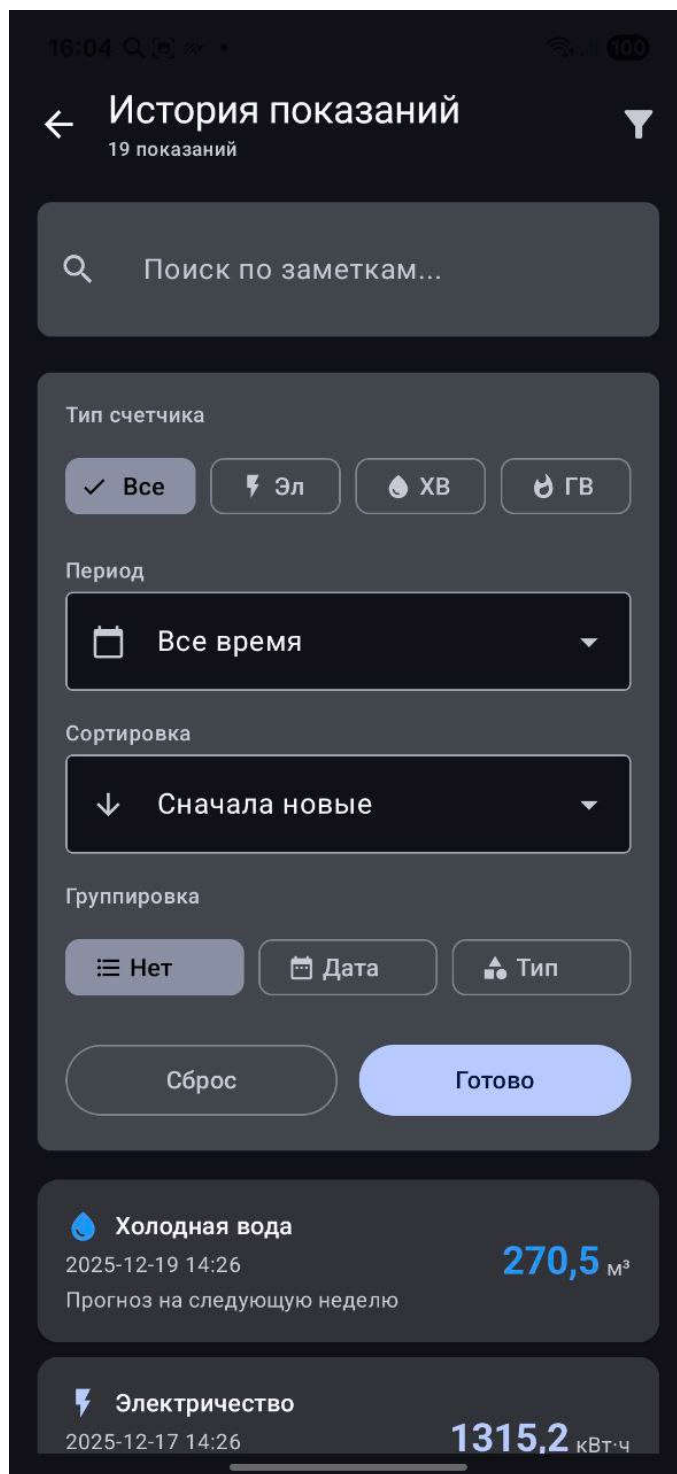


Рисунок Б4 – Параметры поиска показаний

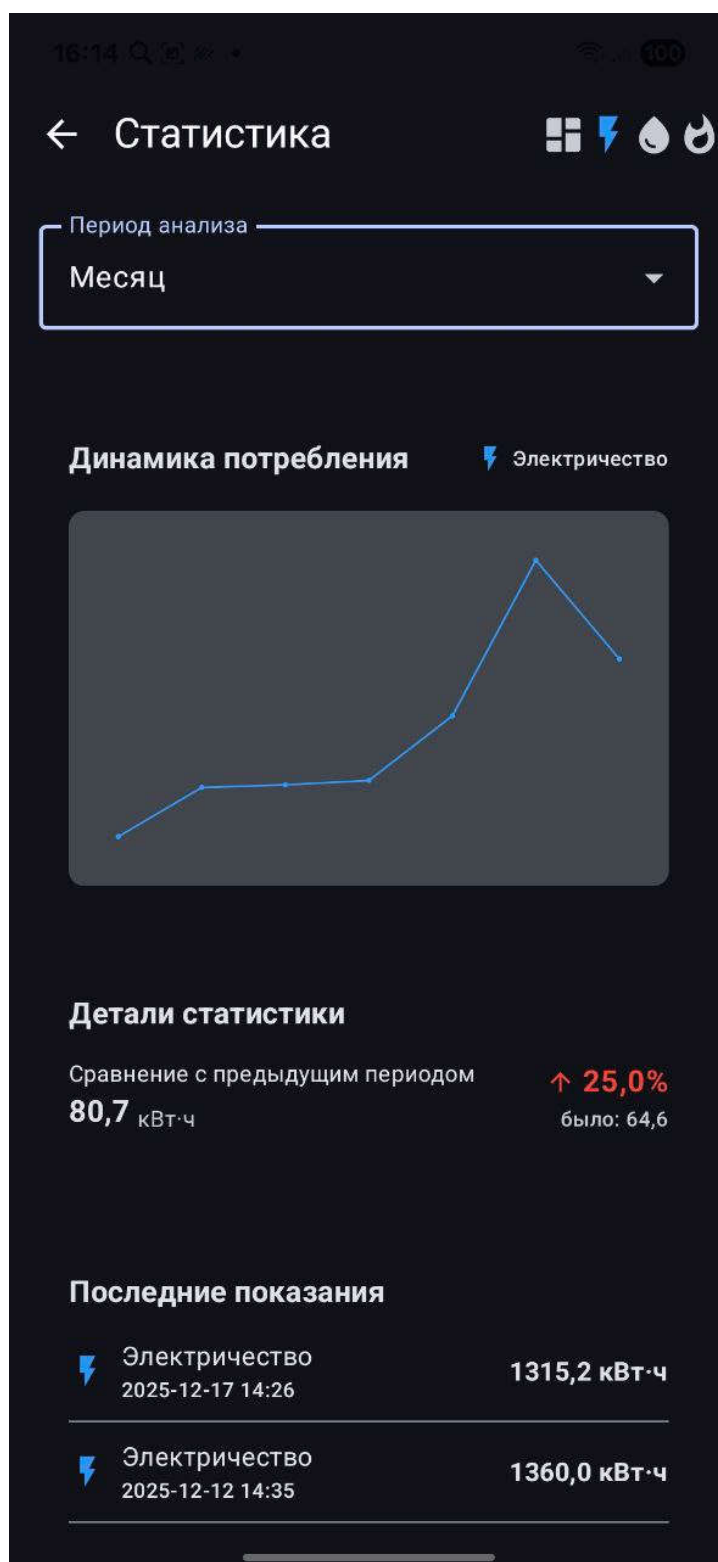



Рисунок Б5 – Статистика расходов

10:03 4G 37

← Добавить показания 

**Последнее показание** 2025-12-12 00:26

Тип: Электричество **225.0**


Заметка: Тестовая заметка


Введите значение больше 225.0

**Тип счетчика**

Выберите тип счетчика

Электричество ▾

**Показания**  Сканировать

 Например: 1234.56

**Заметка**

Необязательно

Сохранить показания

Отмена

Рисунок Б6 – Добавление показаний вручную

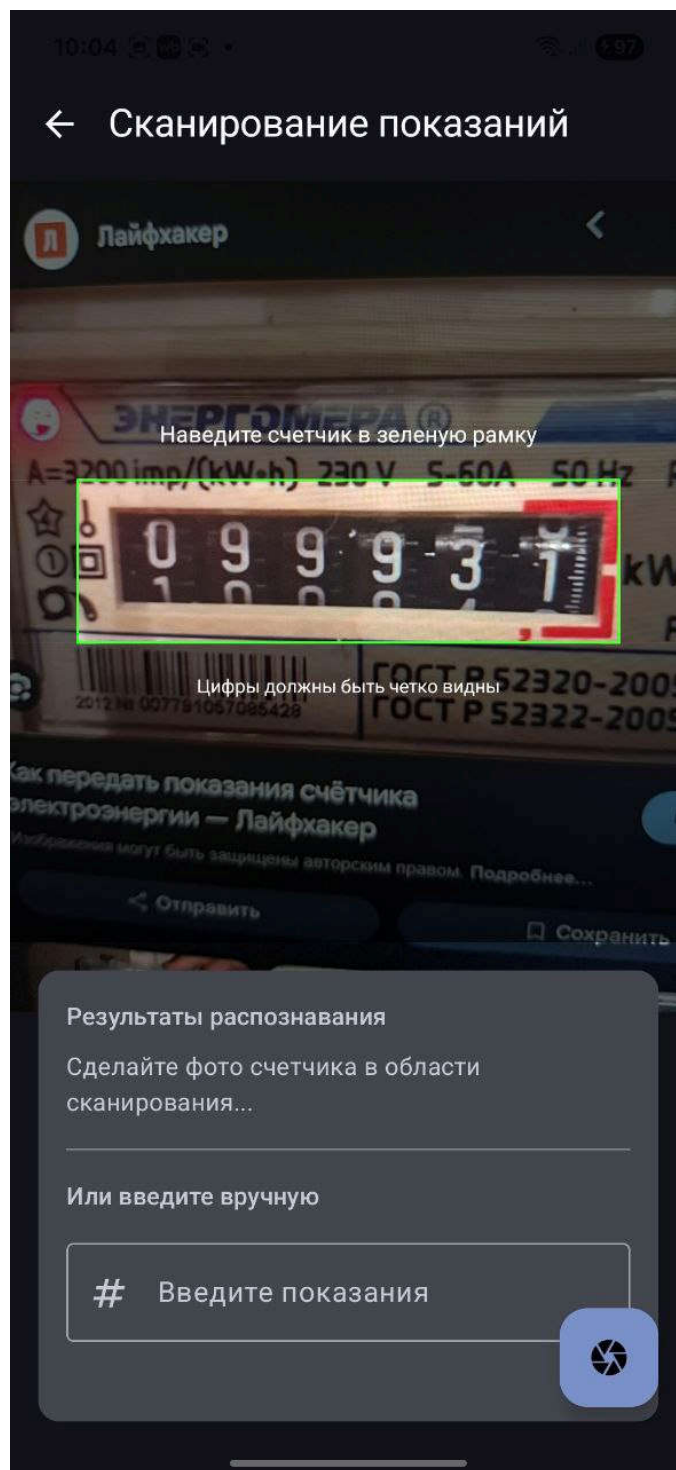


Рисунок Б7 – Сканирование показаний

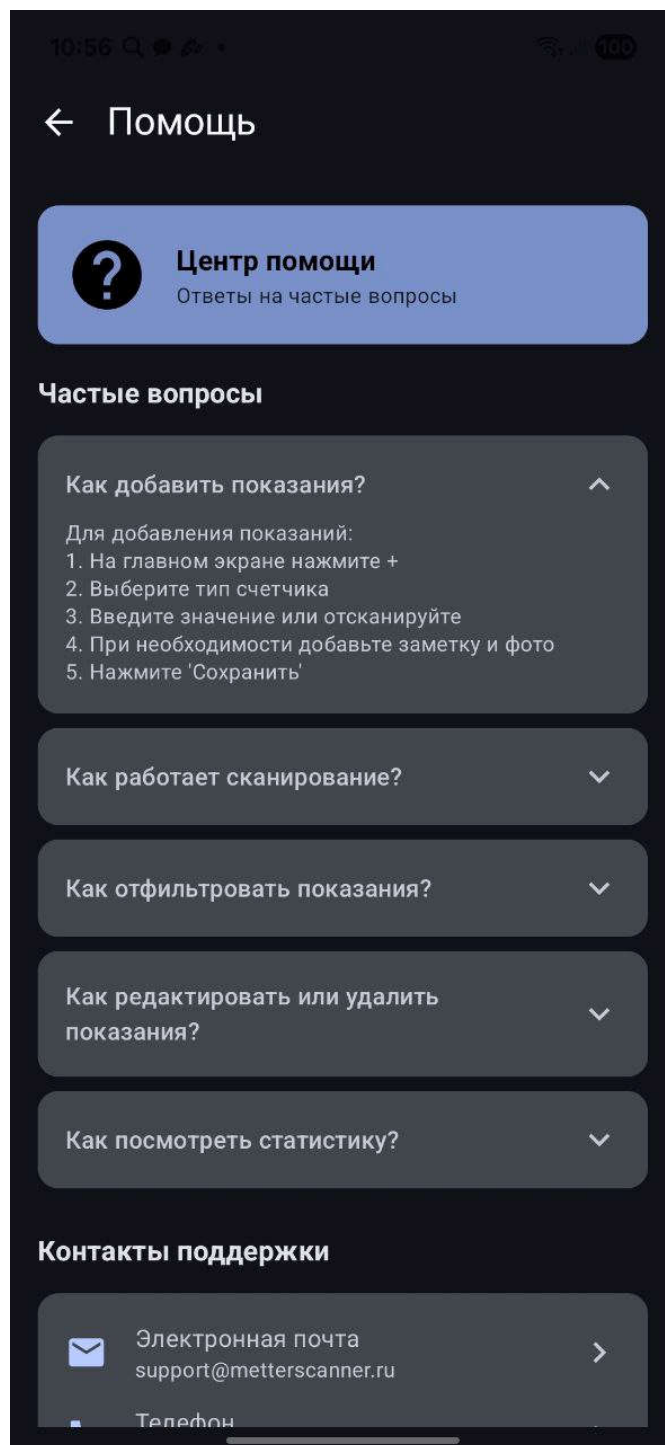


Рисунок Б8 – Окно «Помощь»

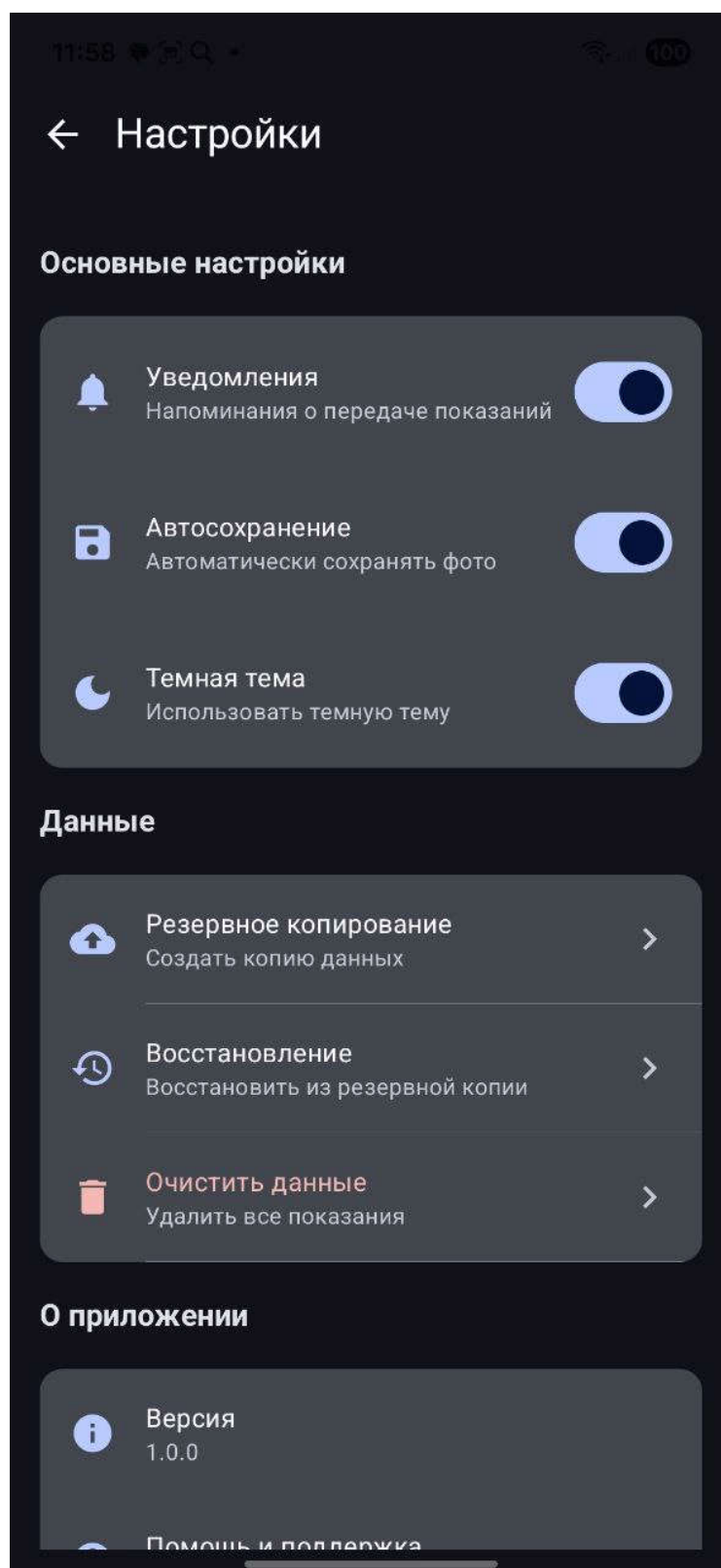


Рисунок Б9 – Настройки

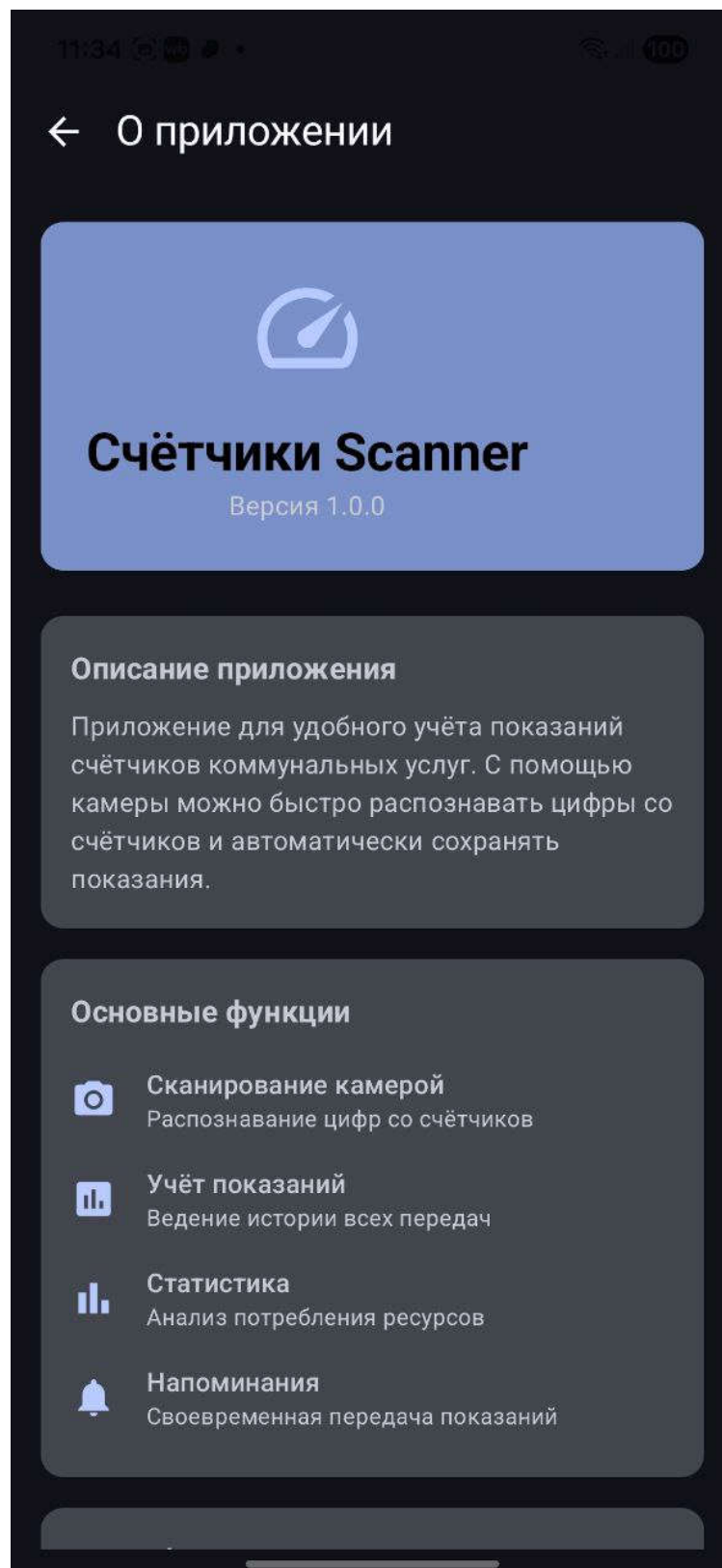


Рисунок Б10 – Окно с информацией о приложении