МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ

КАЛИНИНГРАДСКОЙ ОБЛАСТИ

ПРОФЕССИОНАЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ ОРГАНИЗАЦИЯ

«КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СТРОИТЕЛЬСТВА»

(ГБУ КО ПОО «КИТиС»)

|  |  |  |
| --- | --- | --- |
| Курсовой проект  допущен к защите  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата)  Зам.директора по УМР  \_\_\_\_\_\_\_\_\_\_\_\_\_Павленко Г.Я.  (подпись) |  | Курсовой проект  защищен с оценкой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  Руководитель работы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Пояснительная записка к курсовому проекту

по дисциплине: МДК 01.01 Разработка программных модулей

Тема Разработка приложения по уходу за домашними питомцами

Специальность 09.02.07 «Информационные системы и программирование»

|  |  |
| --- | --- |
| Выполнила: | студентка 3 курса,  группы ИСп22-1  Филиппова Алёна Дмитриевна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Руководитель: | Большакова-Стрекалова Анна Викторовна, преподаватель ГБУ КО ПОО КИТиС  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Калининград

2025

Аннотация

Мобильное приложение «MyApplicat» предназначено для удобного учёта и автоматизации ухода за домашними животными. Приложение позволяет хранить данные о питомцах, создавать напоминания о важных мероприятиях, вести заметки и получать уведомления. Решение работает автономно, не требует регистрации и обеспечивает быстрый доступ к информации о питомцах.

Объём работы: 61 страница.

Приложение: «MyApplicat» (APK-файл).

Среда разработки: Android Studio.

GitHub:

Содержание

[Введение 4](#_Toc196824159)

[1 Анализ предметной области 5](#_Toc196824160)

[1.1 Аналоги разрабатываемого приложения 6](#_Toc196824161)

[1.2 Техническое задание 9](#_Toc196824162)

[1.3 Описание структуры приложения 12](#_Toc196824163)

[2 Описание разработки приложения 16](#_Toc196824164)

[2.1 Обоснование средств разработки 16](#_Toc196824165)

[2.2 Разработка интерфейса 19](#_Toc196824166)

[2.3 Разработка логики работы приложения. Схема взаимодействия компонентов проекта. 30](#_Toc196824167)

[2.3.1 Разработка логики работы приложения. 30](#_Toc196824168)

[2.3.2 Схема взаимодействия компонентов проекта 32](#_Toc196824169)

[2.4 Описание переменных, компонентов, классов и подпрограмм 32](#_Toc196824170)

[3 Тестирование и установка приложения 38](#_Toc196824171)

[3.1 Интеграционное тестирование 38](#_Toc196824172)

[3.2 Установка приложения 39](#_Toc196824173)

[Заключение 40](#_Toc196824174)

[Список использованной литературы 41](#_Toc196824175)

[Приложения А 45](#_Toc196824176)

[Приложение Б 50](#_Toc196824177)

# Введение

Домашние животные играют важную роль в жизни человека, способствуя эмоциональному и психологическому благополучию. Однако уход за питомцами требует регулярного выполнения множества процедур: кормления, вакцинации, приёма лекарств, гигиенических мероприятий и визитов к ветеринару. При наличии нескольких животных систематизация этих задач становится сложной и подверженной ошибкам.

В последние годы наблюдается рост числа владельцев домашних животных и, соответственно, спроса на цифровые инструменты для оптимизации ухода. Большинство существующих мобильных приложений либо требуют постоянного интернет-соединения и регистрации, либо перегружены ненужным функционалом, не соответствующим реальным потребностям пользователей.

Актуальность работы обусловлена необходимостью создания простого, автономного мобильного приложения для Android, позволяющего владельцам домашних животных эффективно организовывать уход без привязки к онлайн-сервисам и без обязательной регистрации. Проект ориентирован на пользователей без специальной технической подготовки и обеспечивает выполнение базовых задач по уходу за питомцем в удобной форме.

Новизна проекта заключается в разработке локального Android-приложения с интуитивно понятным интерфейсом и гибкой архитектурой, позволяющей масштабировать функциональность без существенных изменений в структуре программы.

Цель работы — спроектировать и реализовать мобильное приложение для Android, автоматизирующее и упрощающее рутинные процессы ухода за домашними животными.

1. Анализ предметной области

Предметная область автоматизации ухода за домашними животными охватывает широкий спектр задач, связанных с хранением информации о питомцах, планированием мероприятий по уходу, напоминаниями и ведением истории здоровья. Приложение позволяет автоматизировать и централизовать учет всех аспектов ухода за питомцами, включая напоминания, расписание процедур, хранение заметок и получение уведомлений.

Основные функции приложения:

* Ведение информации о питомцах

Пользователь может добавлять новых питомцев, указывая имя, вид, породу, дату рождения, вес и другую важную информацию. Для каждого питомца можно прикрепить фотографию и просматривать полную карточку с данными.

* Планирование и напоминания о мероприятиях

Приложение позволяет создавать напоминания о важных событиях: кормление, прогулки, вакцинация, визиты к ветеринару, гигиенические процедуры и другие задачи. Пользователь может выбрать дату и время, а также получить уведомление на устройство.

* Ведение истории здоровья и процедур

Для каждого питомца можно сохранять записи о проведённых процедурах, посещениях ветеринара, прививках и других событиях, что помогает отслеживать состояние здоровья и своевременно выполнять необходимые действия.

* Управление заметками и дополнительной информацией

Пользователь может добавлять текстовые заметки, связанные с уходом за питомцем, особенностями поведения или рекомендациями специалистов.

* Гибкая настройка уведомлений

Приложение информирует пользователя о предстоящих событиях и напоминаниях, а также предупреждает о необходимости обновить или добавить информацию, что способствует своевременному уходу за питомцем.

* Интеграция с системными сервисами

Возможность быстро связаться с ветеринарной клиникой или службой поддержки через телефон или мессенджеры (например, Telegram) облегчает решение возникающих вопросов и повышает уровень заботы о питомце.

## Аналоги разрабатываемого приложения

Приложение PetDesk (Рисунок 1) представляет собой комплексное средство планирования мероприятий по уходу за питомцем, включая функции напоминания о визитах в ветеринарные учреждения, вакцинации и гигиенических процедурах, а также ведения медицинской карты животного. Одним из значимых преимуществ данного решения является возможность интеграции с ветеринарными клиниками, что позволяет синхронизировать информацию о медицинских осмотрах и рекомендованных процедурах. Однако среди существенных ограничений необходимо отметить недостаточную гибкость в настройке персонализированных напоминаний, а также отсутствие локализации на русский язык, что значительно снижает удобство использования приложения русскоязычными пользователями.



Рисунок 1 — Интерфейс первой страницы PetDesk

Мобильное приложение 11pets (Рисунок 2) ориентировано на более детальный подход к уходу за питомцем, предоставляя расширенные инструменты для мониторинга состояния здоровья, планирования приёма лекарств и фиксации результатов обследований. Его сильной стороной считается наличие обширной системы учёта и отчётности, позволяющей владельцу отслеживать динамику самочувствия животного и получать визуализированные данные. Вместе с тем избыточность интерфейса и перегруженность функционала могут вызывать трудности у пользователей, не обладающих достаточным уровнем цифровой грамотности.

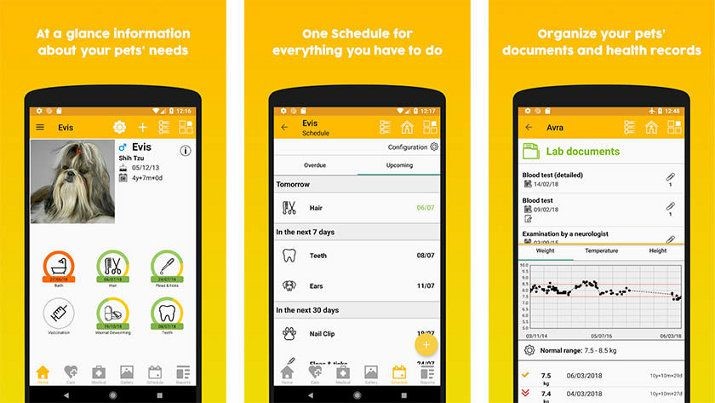


Рисунок 2 — Интерфейс страниц 11pets

В отличие от вышеперечисленных решений, приложение DogLog (Рисунок 3) ориентировано на выполнение повседневных задач, таких как кормление, выгул и гигиенические процедуры. Его интерфейс отличается простотой и интуитивно понятным дизайном, что делает его особенно привлекательным для широкой аудитории. Однако отсутствие модуля для ведения медицинских данных, а также ограниченность функционала в части планирования профилактических мероприятий снижают его универсальность и применимость в более широком контексте ухода за животными.



Рисунок 3 — Интерфейс страниц DogLog

Анализ рассмотренных решений позволяет выявить ряд существенных ограничений, сохраняющихся на рынке подобных цифровых продуктов:

отсутствие полнофункционального русскоязычного интерфейса в большинстве представленных приложений;

недостаточная возможность настройки гибких и индивидуализированных напоминаний;

Раздельное представление функционала, связанного с медицинским учётом и выполнением повседневных процедур, что затрудняет комплексное ведение информации о питомце.

Таким образом, выявленные недостатки и незакрытые потребности пользователей формируют основу для разработки нового мобильного приложения, способного объединить в едином программном решении простоту использования, индивидуальную настройку задач и интеграцию медицинского мониторинга, при этом обеспечивая полноценную автономную работу и адаптацию для русскоязычной аудитории.

* 1. Техническое задание

Общее описание проекта

* Разрабатываемое мобильное приложение «PetCareApp» представляет собой цифровую платформу для управления информацией о домашних животных, позволяющую пользователям вести учет данных о питомцах, создавать напоминания, отслеживать процедуры по уходу и получать уведомления о важных событиях. Приложение предназначено для автоматизации процессов ухода за домашними животными. Интерфейс реализован на Android с упором на простоту, автономность и функциональность.

Цель проекта

Создать удобное в использовании мобильное приложение, с помощью которого пользователь сможет:

* Хранить и управлять информацией о питомцах
* Создавать и получать напоминания о важных событиях
* Вести учет процедур по уходу
* Отслеживать историю здоровья питомца
* Получать своевременные уведомления о запланированных мероприятиях

Функциональные требования

1. Управление питомцами

Добавление нового питомца с указанием:

* Имени
* Вида животного
* Породы
* Даты рождения
* Веса
* Фотографии
* Редактирование информации о питомце
* Удаление питомца
* Просмотр списка всех питомцев
* Детальный просмотр информации о каждом питомце

Система напоминаний

Создание напоминаний с указанием:

1. Описания события

* Даты и времени
* Типа события (кормление, прогулка, ветеринар и т.д.)
* Привязки к конкретному питомцу
* Редактирование существующих напоминаний
* Удаление напоминаний
* Просмотр списка всех напоминаний
* Фильтрация напоминаний по питомцам

Ведение истории

1. Добавление записей о:

* Посещениях ветеринара
* Вакцинациях
* Процедурах
* Изменениях веса
* Просмотр истории по каждому питомцу
* Возможность добавления заметок к записям

1. Система уведомлений

* Отправка уведомлений о предстоящих событиях
* Настройка времени предварительного уведомления
* Возможность отключения уведомлений
* Группировка уведомлений по питомцам

1. Дополнительные функции

* Добавление заметок о питомцах
* Быстрый доступ к важной информации
* Возможность экстренного звонка ветеринару
* Сохранение контактов ветеринарных клиник

1. Настройки приложения

* Управление профилем пользователя
* Настройка уведомлений
* Выбор формата отображения даты и времени
* Настройка приоритетов напоминаний

1. Нефункциональные требования

Технические требования

* Минимальная версия Android: 7.0 (API 24)
* Автономная работа без подключения к интернету
* Локальное хранение данных
* Оптимизация использования памяти устройства
* Требования к интерфейсу
* Интуитивно понятный дизайн
* Адаптивный интерфейс для различных размеров экрана
* Поддержка светлой и темной темы
* Удобная навигация между разделами

Требования к производительности

* Быстрый запуск приложения
* Плавная анимация переходов
* Минимальное потребление ресурсов
* Стабильная работа уведомлений

Требования к безопасности

* Защита персональных данных
* Безопасное хранение информации
* Резервное копирование данных
* Восстановление данных при сбоях

1. Ограничения

* Приложение работает только на Android устройствах
* Все данные хранятся локально на устройстве
* Не требуется подключение к интернету
* Не требуется регистрация пользователя
  1. Описание структуры приложения

Мобильное приложение «PetCareApp» разрабатывается с целью предоставить пользователям удобный инструмент для управления информацией о домашних животных, создания напоминаний и отслеживания ухода за питомцами.

Структура приложения:

Главный экран (MainActivity)

* Отображает список всех питомцев пользователя
* Содержит кнопку добавления нового питомца (FAB)
* Использует RecyclerView и PetAdapter для отображения карточек питомцев
* При клике на карточку открывается детальная информация о питомце
* Нижняя навигация для переключения между основными разделами

Экран добавления питомца (AddPetActivity)

* Форма для ввода информации о новом питомце:
* Имя
* Вид животного
* Порода
* Дата рождения
* Вес
* Загрузка фотографии
* Данные сохраняются в локальное хранилище через PreferencesManager
* При успешном добавлении происходит возврат на главный экран

Детальная информация о питомце (PetDetailsActivity)

* Отображает полную информацию о выбранном питомце
* Возможность редактирования данных
* Доступ к истории процедур и записей
* Список напоминаний для данного питомца
* Кнопки быстрых действий (добавить напоминание, записать вес и т.д.)

Управление напоминаниями (ReminderActivity)

* Создание новых напоминаний с указанием:
* Описания
* Даты и времени
* Типа события
* Привязки к питомцу
* Список всех активных напоминаний
* Возможность редактирования и удаления
* Использует ReminderManager для работы с уведомлениями

Менеджер напоминаний (ReminderManager)

* Управляет всеми напоминаниями в приложении
* Планирует системные уведомления через NotificationManager
* Обрабатывает периодические напоминания
* Сохраняет данные через PreferencesManager

История и записи (HistoryActivity)

* Отображает историю всех событий:
* Посещения ветеринара
* Вакцинации
* Изменения веса
* Процедуры
* Использует RecyclerView с фильтрацией по типам записей
* Возможность добавления новых записей

Менеджер данных (PreferencesManager)

* Отвечает за хранение всех данных приложения
* Использует SharedPreferences для локального хранения
* Сериализация/десериализация объектов через Gson
* Управление резервным копированием данных

Система уведомлений (NotificationService)

* Создает и отправляет системные уведомления
* Обрабатывает различные типы напоминаний
* Использует NotificationManager и AlarmManager
* Поддерживает группировку уведомлений по питомцам

Настройки приложения (SettingsActivity)

* Управление параметрами уведомлений
* Настройка отображения информации
* Управление резервным копированием
* Контакты экстренной связи с ветеринаром

Утилиты и вспомогательные классы

* DateTimeUtils - работа с датами и временем
* ImageUtils - обработка изображений
* ValidationUtils - проверка вводимых данных
* Constants - константы приложения

Компоненты пользовательского интерфейса

* PetCard - карточка питомца для списка
* ReminderItem - элемент списка напоминаний
* CustomDialogs - диалоговые окна приложения
* LoadingIndicators - индикаторы загрузки

Обработка данных

* Все данные хранятся локально на устройстве
* Использование SharedPreferences для хранения настроек
* Сериализация объектов в JSON формат
* Периодическое резервное копирование

Навигация

* BottomNavigationView для основных разделов
* Кнопка "Назад" для возврата к предыдущему экрану
* Быстрые действия через FloatingActionButton
* Контекстное меню для дополнительных операций
* Данная структура обеспечивает:
* Модульность и расширяемость кода
* Удобную навигацию между экранами
* Надежное хранение данных
* Эффективную работу с уведомлениями
* Интуитивно понятный пользовательский интерфейс

1. Описание разработки приложения
2. Обоснование средств разработки

Для реализации мобильного приложения «PetCareApp» были выбраны следующие инструменты и технологии:

* Язык программирования: Kotlin

Kotlin официально поддерживается Google и является предпочтительным языком для Android-разработки благодаря лаконичному синтаксису, безопасности типов и полной совместимости с Java [4].



Рисунок 4 – Логотип Kotlin

Среда разработки: Android Studio

Android Studio предоставляет богатый набор инструментов для написания, отладки, тестирования и сборки Android-приложений, включая визуальный редактор интерфейса, эмуляторы устройств и средства анализа производительности.



Рисунок 5– Логотип Android Studio

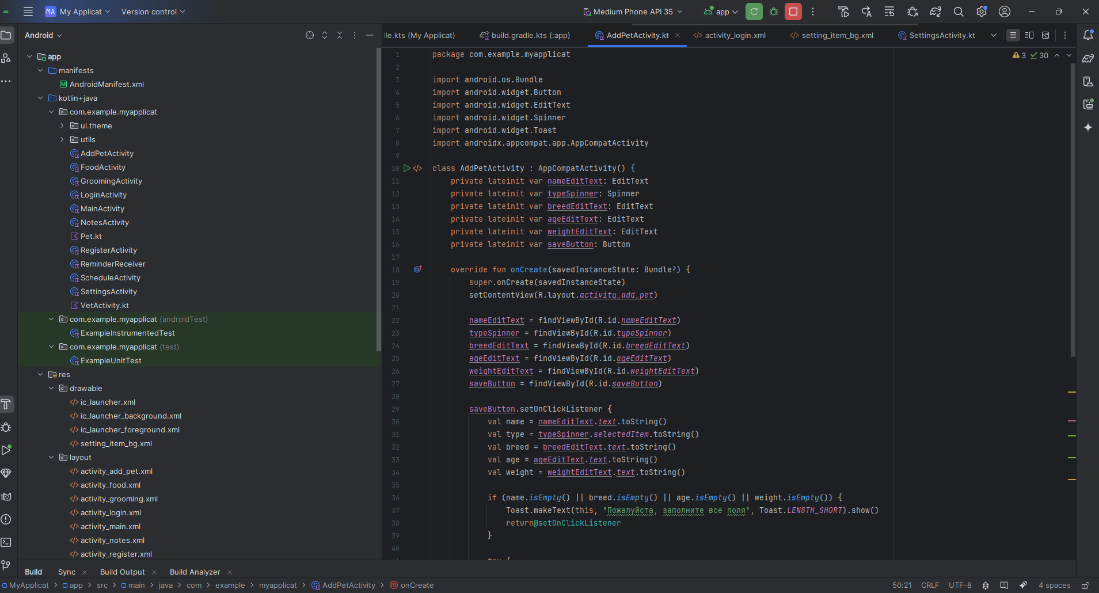


Рисунок 6 - Интерфейс «Android Studio»

Хранение данных

SharedPreferences + Gson

* Локальное хранение данных о питомцах и напоминаниях
* Сериализация объектов в JSON формат
* Быстрый доступ к часто используемым данным
* Надежное сохранение пользовательских настроек

UI и дизайн

XML + Material Design

* Современный и интуитивный интерфейс
* Адаптивные макеты для разных экранов
* Использование стандартных компонентов Material Design
* Поддержка светлой и темной темы

Навигация и структура

Activities + BottomNavigationView

* MainActivity как основной контейнер
* Отдельные Activity для ключевых функций
* BottomNavigationView для основной навигации
* RecyclerView для отображения списков

Управление уведомлениями

NotificationManager + AlarmManager

* Планирование и отправка уведомлений
* Группировка по категориям
* Настраиваемые напоминания
* Фоновая обработка событий

Интеграция с системными сервисами

* Телефон: Intent.ACTION\_DIAL для звонка ветеринару
* Календарь: Интеграция с календарем устройства для напоминаний
* Камера: Intent для фотографирования питомцев
* Галерея: Выбор изображений из галереи

Дополнительные библиотеки

* Gson: для сериализации данных
* AndroidX: основные компоненты
* Material Components: элементы интерфейса
* Glide: работа с изображениями

Figma — это инструмент для дизайна пользовательских интерфейсов (UI) и прототипирования, который работает в браузере и позволяет командам работать совместно в реальном времени. Это один из самых популярных и мощных инструментов для дизайна в последние годы, особенно среди UI/UX дизайнеров и команд разработчиков.



Рисунок 7 – Логотип Figma

* 1. Разработка интерфейса

Изначально макет был разработан в Figma.

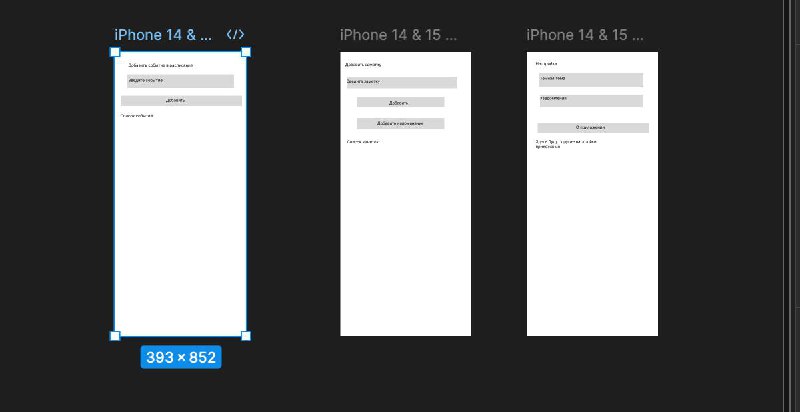


Рисунок 8 - Макет интерфейса

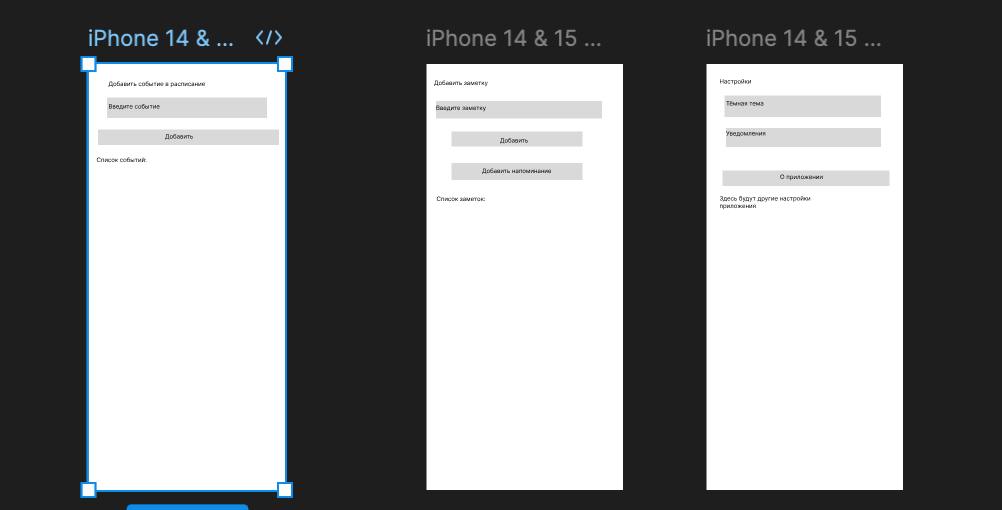


Рисунок 9 - Макет интерфейса

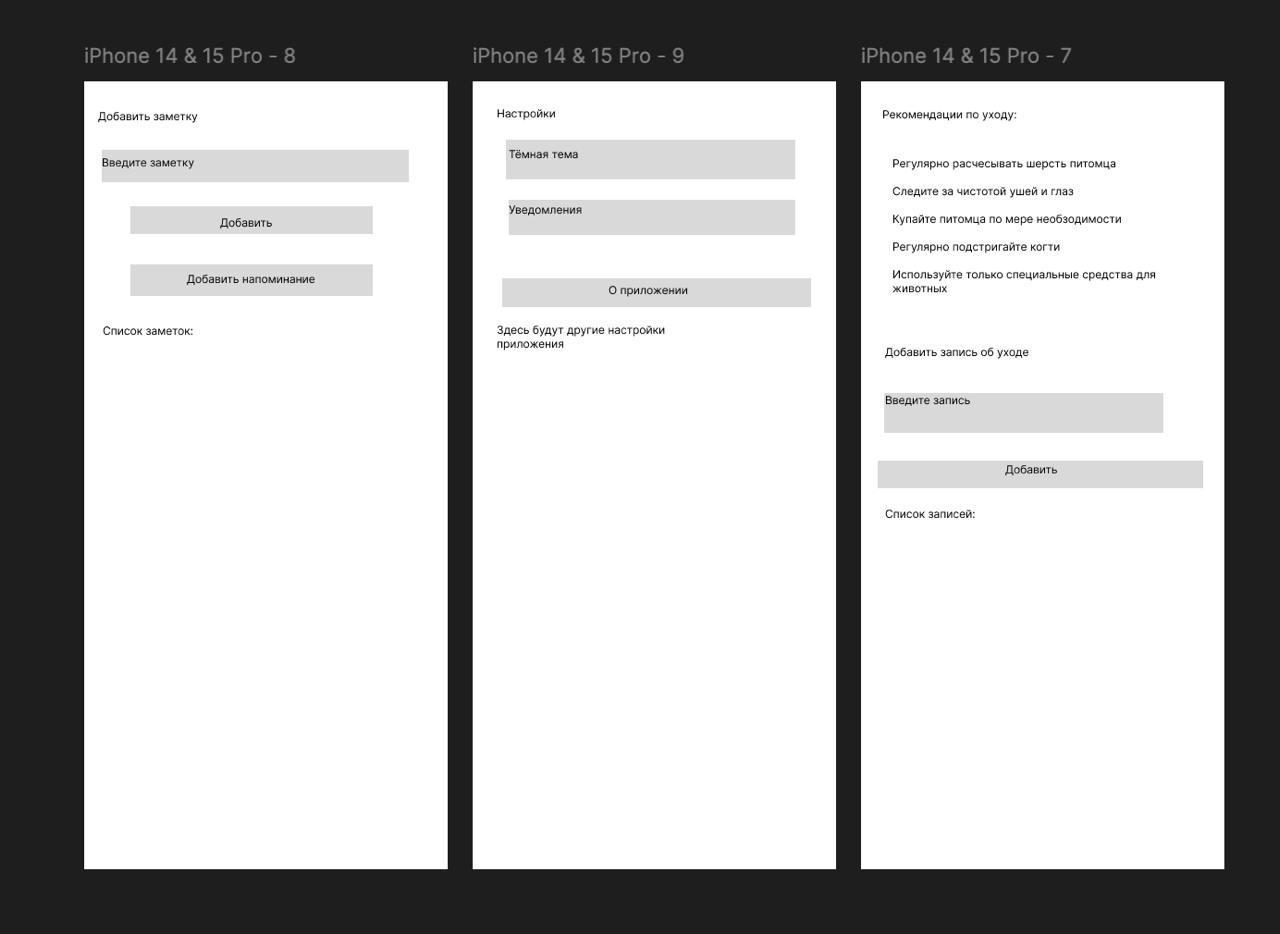
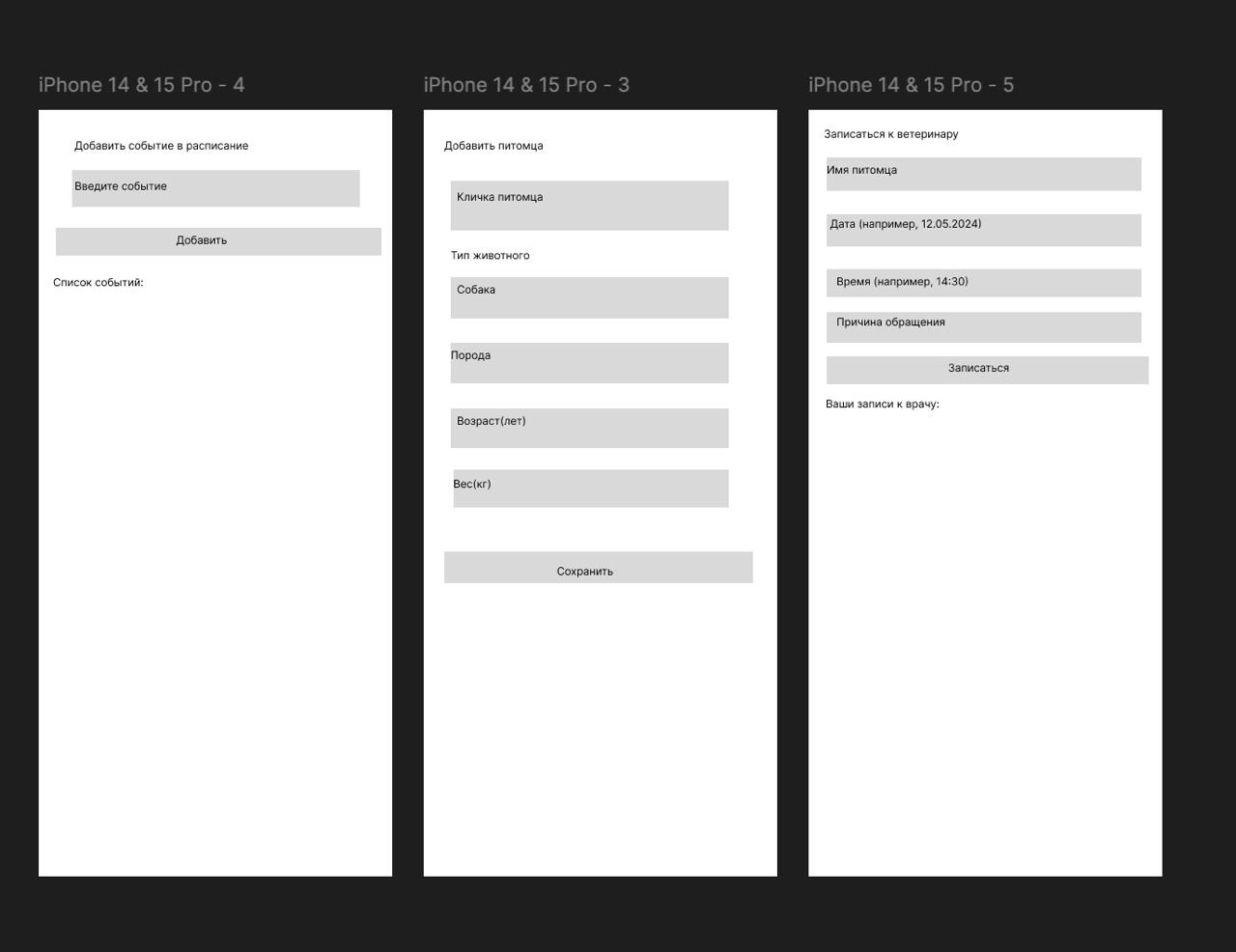


Рисунок 10 - Макет интерфейса

Рисунок 10 - Макет интерфейса

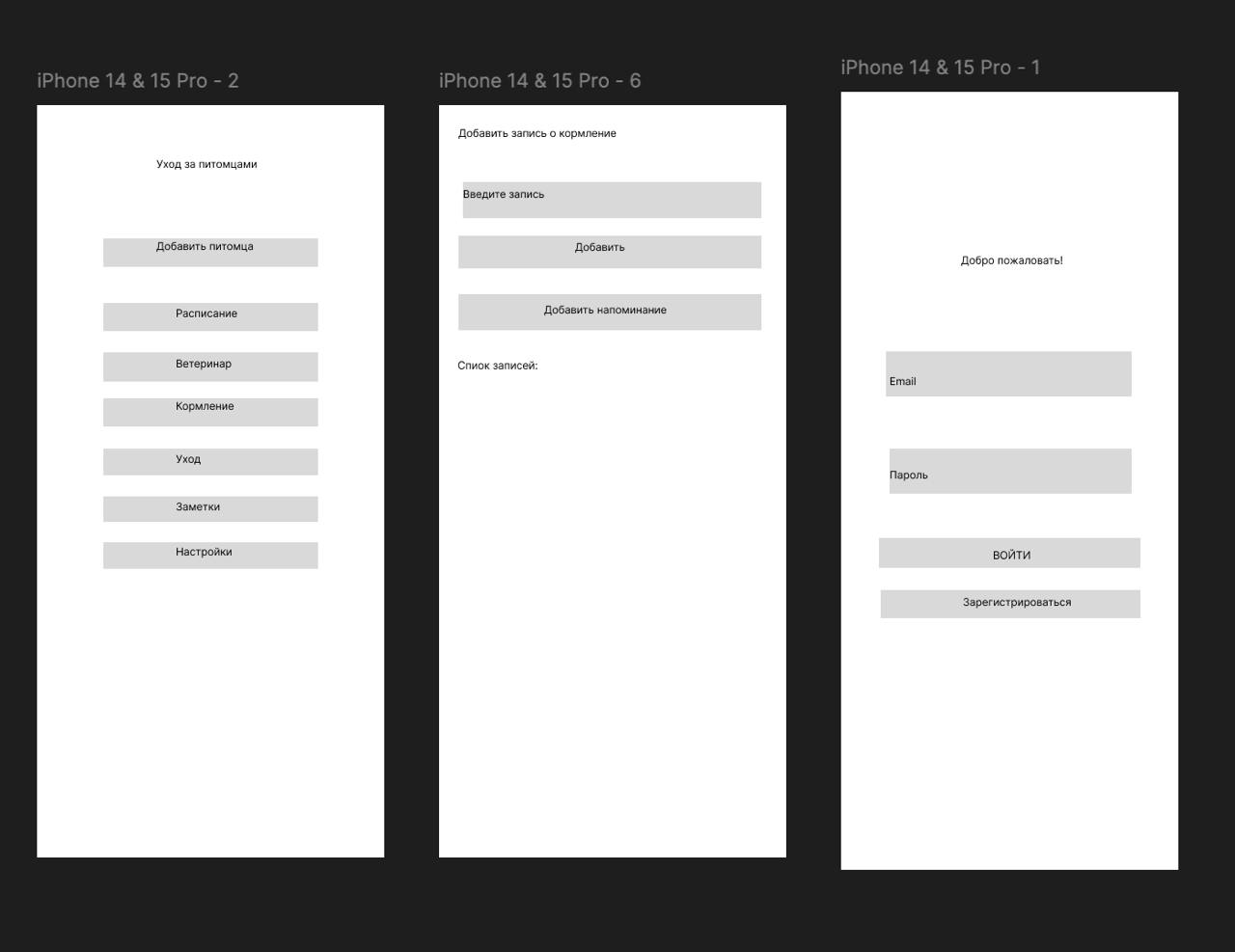


Рисунок 10 - Макет интерфейса

Макеты, которые были сделаны в Android Studio:

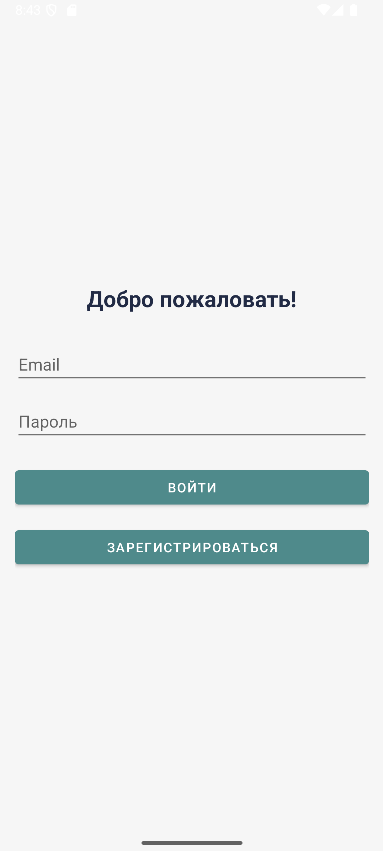


Рисунок 11 – Регистрация пользователя

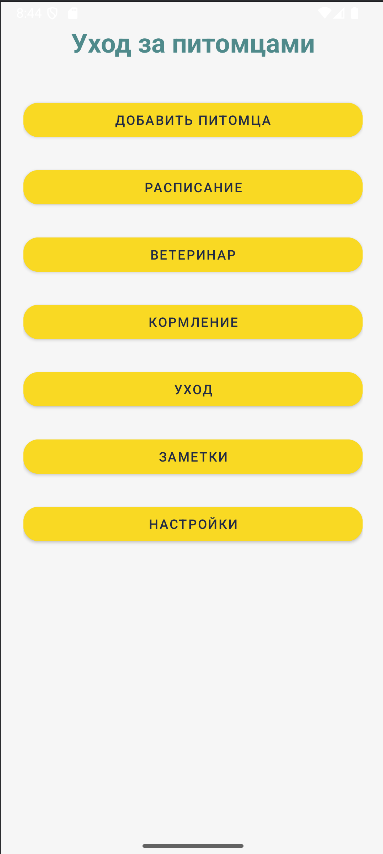


Рисунок 12 – Главный экран

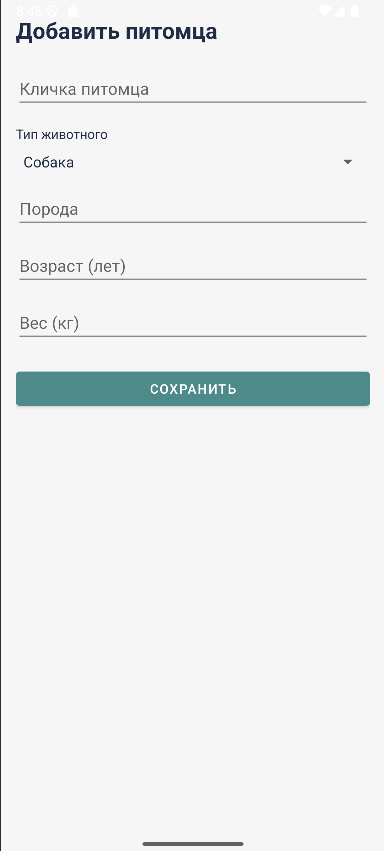


Рисунок 13 – Добавить питомца



Рисунок 14 – Добавить событие в расписание

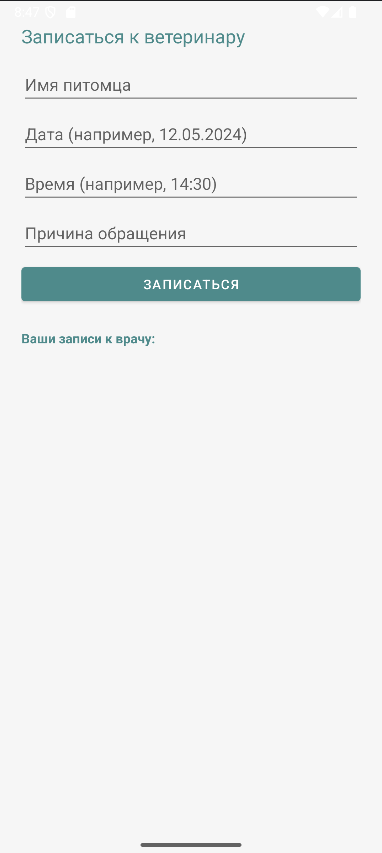


Рисунок 15 – Запись к ветеринару

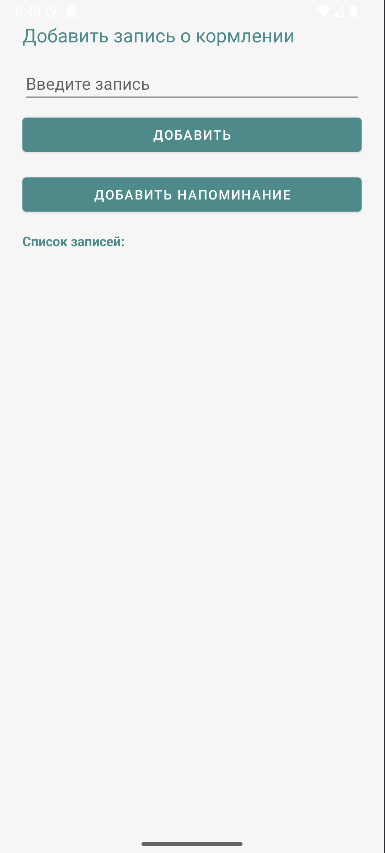


Рисунок 16 - Добавление записи о кормлении

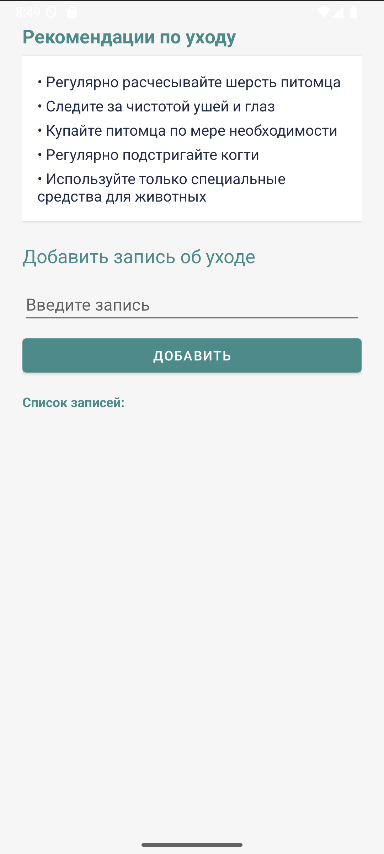


Рисунок 17 –Рекомендации по уходу

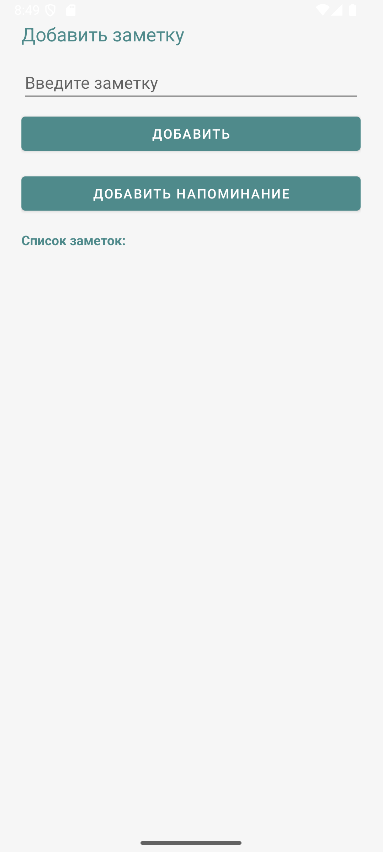


Рисунок 18 – Добавление заметок

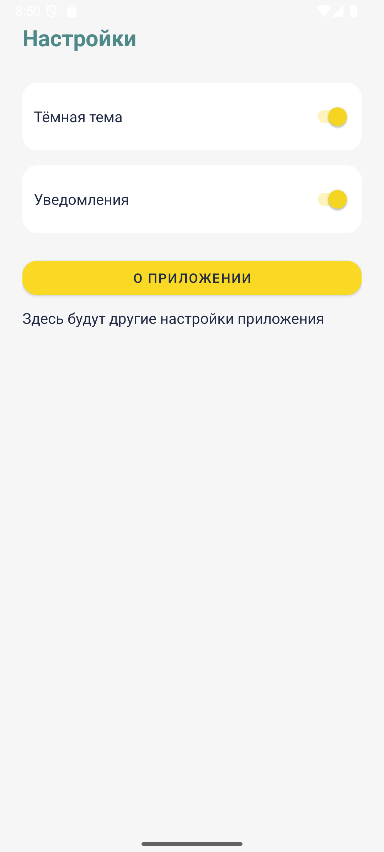


Рисунок 19 – Настройки

* 1. Разработка логики работы приложения. Схема взаимодействия компонентов проекта.
     1. Разработка логики работы приложения.

Основной поток работы:

Запуск приложения

* Открытие главного экрана с меню
* Загрузка сохраненных данных о питомцах

Управление питомцами

* Добавление нового питомца (имя, тип, порода, возраст, вес)
* Сохранение через PreferencesManager
* Отображение списка питомцев

Система напоминаний

* Создание напоминаний о важных событиях
* Планирование уведомлений через NotificationService
* Отправка системных уведомлений

Основные функции

* Ведение расписания
* Учет посещений ветеринара
* Контроль кормления
* Запись процедур ухода
* Создание заметок

Хранение данных

* Локальное хранение через SharedPreferences
* Сериализация данных с помощью Gson
* Резервное копирование

Обработка ошибок

* Валидация вводимых данных
* Вывод уведомлений об ошибках
* Обработка исключительных ситуаций
  + 1. Схема взаимодействия компонентов проекта

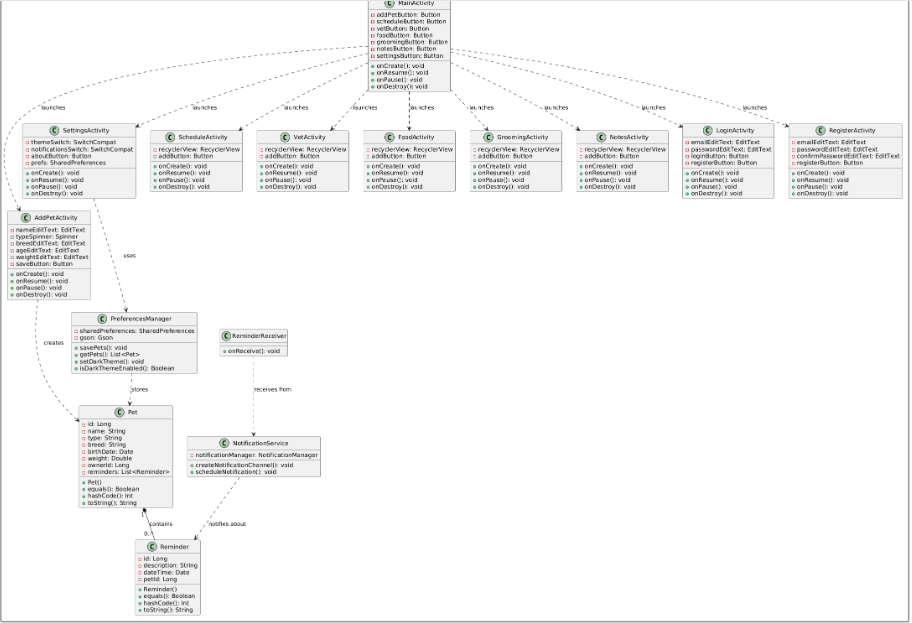


Рисунок 4 — UML диаграмма приложения

* 1. Описание переменных, компонентов, классов и подпрограмм

Таблица 1 - Таблица переменных

|  |  |  |
| --- | --- | --- |
| Переменная | Где используется | Назначение |
| petList | MainActivity | Список всех питомцев |
| petNameInput | AddPetActivity | Поле ввода клички питомца |
| breedInput | AddPetActivity | Поле ввода пород |
| petTypeSpinner | AddPetActivity | Выбор типа животного |
| ageInput | AddPetActivity | Поле ввода возраста |
| weightInput | AddPetActivity | Поле ввода веса |
| saveButton | AddPetActivity | Сохранение данных питомца |
| reminderText | ReminderActivity | Текст напоминания |
| reminderDate | ReminderActivity | Дата напоминания |
| reminderTime | ReminderActivity | Время напоминания |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| notificationManager | NotificationService | Управление уведомлениями |
| sharedPreferences | PreferencesManager | Работа с хранилищем |
| vetNotes | VetActivity | Заметки о посещении ветеринара |
| feedingTime | FeedingActivity | Время кормления |
| notificationsEnabled | SettingsActivity | Включение уведомлений |

Таблица 2 – Таблица компонентов

|  |  |  |
| --- | --- | --- |
| Компонент | Где используется | Назначение |
| Button "добавить питомца" | MainActivity | Переход к экрану добавления питомца |
| Button "расписание" | MainActivity | Переход к управлению расписанием |
| Button "ветеренар" | MainActivity | Переход к ветеринарным записям |
| Button "кормление" | MainActivity | Переход к графику кормления |
| Spinner "Тип животного" | AddPetActivity | Выбор вида животного |
| EditText "порода" | AddPetActivity | Ввод породы питомца |
| EditText "возраст" | AddPetActivity | Ввод возраста питомца |
| EditText "вес" | AddPetActivity | Ввод веса питомц |
| Button "сохранить" | AddPetActivity | Сохранение данных питомца |

Продолжение таблицы 2

|  |  |  |
| --- | --- | --- |
| Button "заметки" | MainActivity | Переход к заметкам о питомцах |
| Button "настройки" | MainActivity | Переход к настройкам приложения |
| EditText "Кличка питомца" | AddPetActivity | Ввод имени питомца |
| RecyclerView | MainActivity | Отображение списка питомцев |
| NotificationManager | NotificationService | Отправка системных уведомлений |
| Button "уход" | MainActivity | Переход к процедурам ухода |
| Button "заметки" | MainActivity | Переход к заметкам о питомцах |
| Button "настройки" | MainActivity | Переход к настройкам приложения |
| EditText "Кличка питомца" | AddPetActivity | Ввод имени питомца |

Таблица 3 – Таблица классов

|  |  |  |
| --- | --- | --- |
| Класс | Тип | Назначение |
| MainActivity | Activity | Главный экран приложения с основным меню и навигацией |
| AddPetActivity | Activity | Экран для добавления и редактирования информации о питомце |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
| PetAdapter | | Adapter | Адаптер для отображения списка питомцев в RecyclerView |
| PreferencesManager | Utility | Класс для работы с локальным хранилищем данных |
| VetActivity | Activity | Экран для ведения ветеринарных записей |
| FeedingActivity | Activity | Экран управления графиком кормления |
| SettingsActivity | Activity | Экран настроек приложения |
| ReminderActivity | Activity | Экран управления напоминаниями и расписанием |
| Pet | Data Class | Модель данных питомца (имя, тип, возраст, вес и т.д.) |
| Reminder | data class | Модель данных напоминания (текст, дата, время) |
| NotificationService | Service | Сервис для создания и отправки уведомлений |
| ReminderManager | Utility | Класс для управления системой напоминаний |
| DateTimeUtils | Utility | Вспомогательный класс для работы с датой и временем |
| ValidationUtils | Utility | Класс для проверки вводимых данных |
| Constants | Object | Object |

Таблица 4 – Таблица подпрограмм

|  |  |  |
| --- | --- | --- |
| Подпрограмма | Где используется | Назначение |
| onCreate() | MainActivity | Инициализация главного экрана и компонентов меню |

Продолжение таблицы 4

|  |  |  |
| --- | --- | --- |
| createNotificationChannel() | NotificationService | Создание канала уведомлений |
| deletePet() | MainActivity | Удаление питомца из списка |
| deletePet() | MainActivity | Удаление питомца из списка |
| addNewPet() | AddPetActivity | Сохранение данных нового питомца |
| showDateTimePicker() | ReminderActivity | Отображение диалога выбора даты и времени |
| onPetClick() | PetAdapter | Обработка нажатия на карточку питомца |
| saveSettings() | SettingsActivity | Сохранение настроек приложения |
| formatDateTime() | DateTimeUtils | Форматирование даты и времени для отображения |
| showError() | Все Activity | Отображение сообщения об ошибке |
| validatePetData() | AddPetActivity | Проверка корректности введенных данных о питомце |

Продолжение таблицы 4

|  |  |  |
| --- | --- | --- |
| loadPetList( | MainActivity | Загрузка списка питомцев из хранилища |
| savePetToStorage( | PreferencesManager | Сохранение информации о питомце в локальное хранилище |
| createReminder( | ReminderActivity | Создание нового напоминания |
| scheduleNotification() | NotificationService | Планирование уведомления на заданное время |
| savePetToStorage( | PreferencesManager | Сохранение информации о питомце в локальное хранилище |

1. Тестирование и установка приложения
   1. Интеграционное тестирование
2. Модуль управления питомцами:

* Сохранение и загрузка данных
* Обновление информации
* Удаление питомцев
* Целостность данных

1. Модуль напоминаний:

* Создание напоминаний
* Точность уведомлений
* Привязка к питомцам
* Отображение информации

1. Валидация данных:

* Проверка возраста и веса
* Обязательные поля
* Обработка ошибок

Пользовательские сценарии

1. Управление питомцами:

* Добавление питомца
* Редактирование данных
* Удаление

1. Напоминания:

* Создание напоминания
* Проверка уведомлений
* Редактирование

1. Дополнительные функции:

* Ведение расписания
* Запись к ветеринару
* График кормления
* Создание заметок

1. Установка

Android 7.0+

* Размер: ~10MB
* Автономная работа
* Установка через APK
  1. Установка приложения

1. Создание APK:
   * В Android Studio выберите "Build" -> "Generate App Bundles or APKs-> " Generate APKs" [1].
   * Это создаст APK-файл вашего приложения, который(-ые) вы можете использовать для установки.
2. Перенос APK на устройство:
   * Перенесите APK-файл на ваше устройство через USB, облачное хранилище или любой другой метод передачи файлов.
3. Установка приложения:
   * На вашем устройстве перейдите в настройки безопасности и разрешите установку приложений из неизвестных источников, если это ещё не сделано.
   * Используйте файловый менеджер на устройстве, чтобы найти и открыть APK-файл приложения.
   * Следуйте инструкциям по установке, чтобы завершить установку приложения.

# Заключение

В ходе выполнения курсовой работы была достигнута поставленная цель — разработка мобильного приложения для управления информацией о домашних животных на платформе Android. Приложение реализует базовый функционал, необходимый для удобного ухода за питомцами: хранение информации о животных, управление напоминаниями, ведение расписания процедур, а также учет посещений ветеринара и контроль кормления.В процессе разработки были изучены и применены современные технологии мобильной разработки: язык программирования Kotlin, платформа Android Studio, SharedPreferences для локального хранения информации, а также принципы проектирования пользовательского интерфейса на основе концепции Material Design. Это позволило создать простое, интуитивно понятное и надёжное в работе приложение.Разработанное приложение автоматизирует множество процессов, связанных с уходом за домашними животными, что делает его полезным инструментом для владельцев питомцев. Пользователи получают возможность экономить время за счёт автоматизации напоминаний и систематизации информации, а также повышать качество ухода за питомцами благодаря своевременному выполнению всех необходимых процедур.

# Список использованной литературы

1. Разработчики Android. Создание вашего первого приложения // developer.android.com. — URL: <https://developer.android.com/training/basics/firstapp> (дата обращения: 04.04.2025). — Текст: электронный.
2. Документация по языку Kotlin // Kotlinlang.org. — URL: <https://kotlinlang.org/docs/home.html> (дата обращения: 04.04.2025). — Текст: электронный.
3. Обзор Android Jetpack // developer.android.com. — URL: <https://developer.android.com/jetpack> (дата обращения: 04.04.2025). — Текст: электронный.
4. Основы Jetpack Compose // developer.android.com. — URL: <https://developer.android.com/jetpack/compose/tutorial> (дата обращения: 04.04.2025). — Текст: электронный.
5. Gson: библиотека сериализации/десериализации Java // GitHub.com. — URL: <https://github.com/google/gson> (дата обращения: 04.04.2025). — Текст: электронный.
6. Справочник по API SharedPreferences // developer.android.com. — URL: <https://developer.android.com/reference/android/content/SharedPreferences> (дата обращения: 04.04.2025). — Текст: электронный.
7. Документация по классу AlarmManager // developer.android.com. — URL: <https://developer.android.com/reference/android/app/AlarmManager> (дата обращения: 04.04.2025). — Текст: электронный.
8. Справочник по построителю уведомлений NotificationCompat // developer.android.com. — URL: <https://developer.android.com/reference/androidx/core/app/NotificationCompat.Builder> (дата обращения: 04.04.2025). — Текст: электронный.
9. Класс TimePicker // developer.android.com. — URL: <https://developer.android.com/reference/android/widget/TimePicker> (дата обращения: 04.04.2025). — Текст: электронный.
10. Руководство по RecyclerView // developer.android.com. — URL: <https://developer.android.com/guide/topics/ui/layout/recyclerview> (дата обращения: 04.04.2025). — Текст: электронный.
11. Руководство по дизайну материалов // material.io. — URL: <https://m3.material.io> (дата обращения: 04.04.2025). — Текст: электронный.
12. Дизайн пользовательского интерфейса для мобильных приложений // Nielsen Norman Group. — URL: <https://www.nngroup.com/articles/mobile-app-usability/> (дата обращения: 04.04.2025). — Текст: электронный.
13. Чистая архитектура для Android // developer.android.com. — URL: <https://developer.android.com/jetpack/guide> (дата обращения: 04.04.2025). — Текст: электронный.
14. Использование BroadcastReceiver в Android // Habrahabr. — URL: <https://habr.com/ru/articles/347156/> (дата обращения: 04.04.2025). — Текст: электронный.
15. Библиотека Room для хранения данных // developer.android.com. — URL: <https://developer.android.com/training/data-storage/room> (дата обращения: 04.04.2025). — Текст: электронный.
16. Руководство по архитектуре приложений Android // developer.android.com. — URL: <https://developer.android.com/jetpack/guide> (дата обращения: 04.04.2025). — Текст: электронный.
17. Тенденции разработки мобильных приложений в 2025 году // TechRadar. — URL: <https://www.techradar.com/news/app-development-trends> (дата обращения: 04.04.2025). — Текст: электронный.
18. PetDesk — мобильное приложение для ухода за домашними животными // Petdesk.com. — URL: <https://petdesk.com/> (дата обращения: 04.04.2025). — Текст: электронный.
19. 11pets: приложение для ухода за домашними животными // 11pets.com. — URL: <https://www.11pets.com/en/> (дата обращения: 04.04.2025). — Текст: электронный.
20. DogLog — трекер активности домашних животных // Doglogapp.com. — URL: <https://www.doglogapp.com/> (дата обращения: 04.04.2025). — Текст: электронный.
21. Разработка приложений для Android: советы и рекомендации // UXDesign.cc. — URL: <https://uxdesign.cc/android-app-design> (дата обращения: 04.04.2025). — Текст: электронный.
22. Документация Android Studio // developer.android.com. — URL: <https://developer.android.com/studio> (дата обращения: 04.04.2025). — Текст: электронный.
23. Обзор корутин Kotlin // kotlinlang.org. — URL: <https://kotlinlang.org/docs/coroutines-overview.html> (дата обращения: 04.04.2025). — Текст: электронный.
24. Создание уведомлений в Android // Android-Developer.ru. — URL: <https://android-developer.ru/notifications/> (дата обращения: 04.04.2025). — Текст: электронный.
25. Базы данных SQLite в Android // StartAndroid.ru. — URL: <https://startandroid.ru/ru/lessons/complete-list/585-lesson-110-sqlite.html> (дата обращения: 04.04.2025). — Текст: электронный.
26. Компонент навигации Jetpack // developer.android.com. — URL: <https://developer.android.com/guide/navigation> (дата обращения: 04.04.2025). — Текст: электронный.
27. Использование Gson в Android // JournalDev. — URL: <https://www.journaldev.com/13639/gson-example-tutorial> (дата обращения: 04.04.2025). — Текст: электронный.
28. Оптимизация производительности мобильных приложений // Medium.com. — URL: <https://medium.com/mobile-performance-android> (дата обращения: 04.04.2025). — Текст: электронный.
29. Советы по созданию пользовательского интерфейса в Jetpack Compose // Tproger.ru. — URL: <https://tproger.ru/articles/jetpack-compose-ui> (дата обращения: 04.04.2025). — Текст: электронный.

# Приложения А

Листинг программы

MainActivity.kt (главный экран):

package com.example.myapplicat

import android.content.Intent

import android.os.Bundle

import android.widget.Button

import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

val addPetButton = findViewById<Button>(R.id.addPetButton)

val scheduleButton = findViewById<Button>(R.id.scheduleButton)

val vetButton = findViewById<Button>(R.id.vetButton)

val foodButton = findViewById<Button>(R.id.foodButton)

val groomingButton = findViewById<Button>(R.id.groomingButton)

val notesButton = findViewById<Button>(R.id.notesButton)

val settingsButton = findViewById<Button>(R.id.settingsButton)

addPetButton.setOnClickListener {

startActivity(Intent(this, AddPetActivity::class.java))

}

scheduleButton.setOnClickListener {

startActivity(Intent(this, ScheduleActivity::class.java))

}

vetButton.setOnClickListener {

startActivity(Intent(this, VetActivity::class.java))

}

foodButton.setOnClickListener {

startActivity(Intent(this, FoodActivity::class.java))

}

groomingButton.setOnClickListener {

startActivity(Intent(this, GroomingActivity::class.java))

}

notesButton.setOnClickListener {

startActivity(Intent(this, NotesActivity::class.java))

}

settingsButton.setOnClickListener {

startActivity(Intent(this, SettingsActivity::class.java))

}

}

}

2. AddPetActivity.kt (экран добавления питомца):

package com.example.myapplicat

import android.os.Bundle

import android.widget.Button

import android.widget.EditText

import android.widget.Spinner

import android.widget.Toast

import androidx.appcompat.app.AppCompatActivity

class AddPetActivity : AppCompatActivity() {

private lateinit var nameEditText: EditText

private lateinit var typeSpinner: Spinner

private lateinit var breedEditText: EditText

private lateinit var ageEditText: EditText

private lateinit var weightEditText: EditText

private lateinit var saveButton: Button

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_add\_pet)

nameEditText = findViewById(R.id.nameEditText)

typeSpinner = findViewById(R.id.typeSpinner)

breedEditText = findViewById(R.id.breedEditText)

ageEditText = findViewById(R.id.ageEditText)

weightEditText = findViewById(R.id.weightEditText)

saveButton = findViewById(R.id.saveButton)

saveButton.setOnClickListener {

val name = nameEditText.text.toString()

val type = typeSpinner.selectedItem.toString()

val breed = breedEditText.text.toString()

val age = ageEditText.text.toString()

val weight = weightEditText.text.toString()

if (name.isEmpty() breed.isEmpty() age.isEmpty() || weight.isEmpty()) {

Toast.makeText(this, "Пожалуйста, заполните все поля", Toast.LENGTH\_SHORT).show()

return@setOnClickListener

}

try {

val currentYear = java.util.Calendar.getInstance().get(java.util.Calendar.YEAR)

val birthYear = currentYear - age.toInt()

val birthDate = "$birthYear-01-01"

val pet = Pet(

name = name,

type = type,

breed = breed,

birthDate = birthDate,

weight = weight.toDouble(),

ownerId = 1 // TODO: Получить ID текущего пользователя

)

// TODO: Сохранить питомца в базу данных

Toast.makeText(this, "Питомец успешно добавлен", Toast.LENGTH\_SHORT).show()

finish()

} catch (e: NumberFormatException) {

Toast.makeText(this, "Пожалуйста, введите корректные числовые значения", Toast.LENGTH\_SHORT).show()

}

}

}

}

3. Pet.kt (модель питомца):

package com.example.myapplicat

import com.google.gson.annotations.SerializedName

data class Reminder(

@SerializedName("id")

val id: Long = 0,

@SerializedName("description")

val description: String,

@SerializedName("dateTime")

val dateTime: String,

@SerializedName("petId")

val petId: Long

)

data class Pet(

@SerializedName("id")

val id: Long = 0,

@SerializedName("name")

val name: String,

@SerializedName("type")

val type: String,

@SerializedName("breed")

val breed: String,

@SerializedName("birthDate")

val birthDate: String,

@SerializedName("weight")

val weight: Double,

@SerializedName("ownerId")

val ownerId: Long,

@SerializedName("reminders")

val reminders: List<Reminder> = emptyList()

)

4. PreferencesManager.kt (работа с данными):

package com.example.myapplicat.utils

import android.content.Context

import android.content.SharedPreferences

import com.google.gson.Gson

import com.google.gson.reflect.TypeToken

import com.example.myapplicat.Pet

class PreferencesManager(context: Context) {

private val sharedPreferences: SharedPreferences = context.getSharedPreferences(PREFS\_NAME, Context.MODE\_PRIVATE)

private val gson = Gson()

companion object {

private const val PREFS\_NAME = "PetCarePrefs"

private const val KEY\_PETS = "pets"

private const val KEY\_DARK\_THEME = "dark\_theme"

}

fun savePets(pets: List<Pet>) {

val json = gson.toJson(pets)

sharedPreferences.edit().putString(KEY\_PETS, json).apply()

}

fun getPets(): List<Pet> {

val json = sharedPreferences.getString(KEY\_PETS, null)

return if (json != null) {

val type = object : TypeToken<List<Pet>>() {}.type

gson.fromJson(json, type)

} else {

emptyList()

}

}

fun setDarkTheme(enabled: Boolean) {

sharedPreferences.edit().putBoolean(KEY\_DARK\_THEME, enabled).apply()

}

fun isDarkThemeEnabled(): Boolean {

return sharedPreferences.getBoolean(KEY\_DARK\_THEME, false)

}

}

5. NotificationService.kt (уведомления):

package com.example.myapplicat.utils

import android.app.NotificationChannel

import android.app.NotificationManager

import android.app.PendingIntent

import android.content.Context

import android.content.Intent

import android.os.Build

import androidx.core.app.NotificationCompat

import com.example.myapplicat.MainActivity

import com.example.myapplicat.R

import com.example.myapplicat.ReminderReceiver

class NotificationService(private val context: Context) {

private val notificationManager = context.getSystemService(Context.NOTIFICATION\_SERVICE) as NotificationManager

companion object {

private const val CHANNEL\_ID = "pet\_care\_channel"

private const val CHANNEL\_NAME = "Уведомления о питомцах"

}

init {

createNotificationChannel()

}

private fun createNotificationChannel() {

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {

val channel = NotificationChannel(

CHANNEL\_ID,

CHANNEL\_NAME,

NotificationManager.IMPORTANCE\_DEFAULT

).apply {

description = "Канал для уведомлений о питомцах"

}

notificationManager.createNotificationChannel(channel)

}

}

fun showNotification(title: String, message: String) {

val notification = NotificationCompat.Builder(context, CHANNEL\_ID)

.setSmallIcon(R.drawable.ic\_launcher)

.setContentTitle(title)

.setContentText(message)

.setPriority(NotificationCompat.PRIORITY\_DEFAULT)

.setAutoCancel(true)

.build()

notificationManager.notify(System.currentTimeMillis().toInt(), notification)

}

}

# Приложение Б

Руководство пользователя

Приложение PetCareApp предназначено для упрощения и систематизации ухода за домашними животными. С помощью приложения вы можете хранить информацию о питомцах, создавать напоминания о важных мероприятиях (кормление, вакцинация, визиты к ветеринару), вести заметки, управлять расписанием и получать рекомендации по уходу. Данное руководство описывает основные функции приложения и порядок их использования.

1. Установка приложения
2. Скачивание приложения:

Загрузите APK-файл приложения с официального источника или магазина приложений.

1. Установка:

Откройте загруженный APK-файл.

Следуйте инструкциям на экране для завершения установки.

При необходимости разрешите установку приложений из неизвестных источников в настройках устройства.

1. Первый запуск:

После установки откройте приложение.

При первом запуске приложение может запросить разрешения (например, доступ к уведомлениям). Подтвердите запросы для корректной работы напоминаний.

3.Интерфейс приложения

3.1 Главный экран

Навигация:

На главном экране расположены кнопки для перехода к основным функциям: добавление питомца, расписание, ветеринар, кормление, уход, заметки, настройки.

Список питомцев:

Для просмотра и управления питомцами используйте соответствующие разделы приложения.

Кнопка "Добавить питомца":

Для добавления нового питомца нажмите на кнопку "Добавить питомца". Откроется форма для ввода информации о питомце.

Кнопка "Расписание":

Для создания напоминания или планирования события используйте кнопку "Расписание".

3.2 Экран добавления питомца

Введите имя, вид, породу, возраст и вес питомца.

После заполнения всех полей нажмите кнопку "Сохранить". Новый питомец появится в общем списке.

3.3 Экран создания напоминания

Введите описание события (например, "Вакцинация").

Выберите время срабатывания напоминания с помощью диалога выбора времени.

Нажмите кнопку "Установить напоминание". В выбранное время вы получите уведомление.

3.4 Экран заметок, расписания, кормления, ухода и ветеринарных записей

В каждом разделе вы можете добавлять, просматривать и редактировать соответствующую информацию (заметки, события, записи о кормлении, уходе и визитах к ветеринару).

3.5 Экран настроек

Включайте или отключайте тёмную тему.

Управляйте уведомлениями.

Просматривайте информацию о приложении.

4. Дополнительные функции

История здоровья:

Добавляйте записи о визитах к ветеринару, вакцинациях и других событиях.

Заметки:

Ведите заметки по каждому питомцу.

Профиль пользователя:

Изменяйте свои данные и просматривайте статистику по питомцам (если реализовано).

Рекомендации:

Получайте советы по уходу за разными видами питомцев (если реализовано).

Заключение

PetCareApp — это современное и удобное приложение для организации ухода за домашними животными. Благодаря интуитивному интерфейсу и широкому набору функций, приложение станет вашим надежным помощником в ведении информации о питомцах, напоминаниях о важных событиях и получении рекомендаций по уходу. Заботьтесь о своих питомцах легко и с удовольствием вместе с PetCareApp!

1. Регистрация пользователя:

-Ведите имя пользователя.

* Введите почту, которая существует.
* Введите пароль.
* Нажать на кнопку "Регистрация".

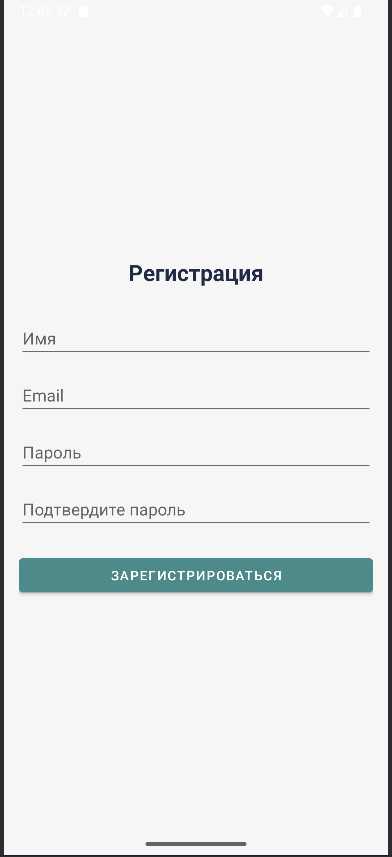


Рисунок 1 – Экран регистрации

1. Авторизация:

* Введите логин и пароль.
* Нажать на кнопку "Войти".

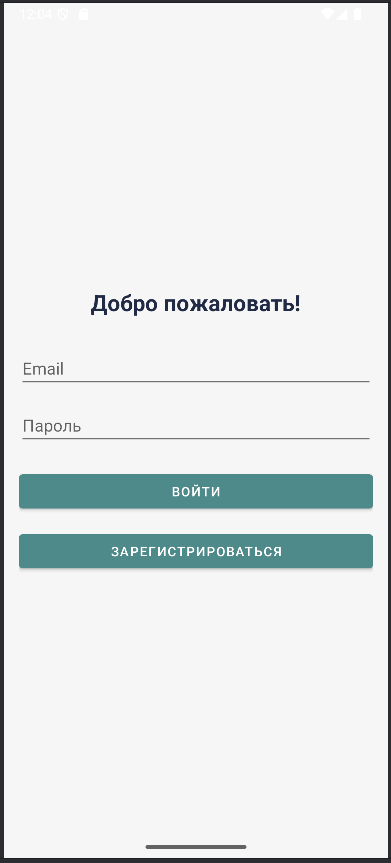


Рисунок 3 – Экран авторизации

1. Главный экран:

* "Добавить питомца" - для добавления нового питомца
* "Расписание" - для управления напоминаниями и событиями
* "Ветеринар" - для ведения записей о посещениях ветеринара
* "Кормление" - для управления графиком кормления
* "Уход" - для записей о процедурах ухода
* "Заметки" - для хранения заметок о питомцах
* "Настройки" - для настройки приложения

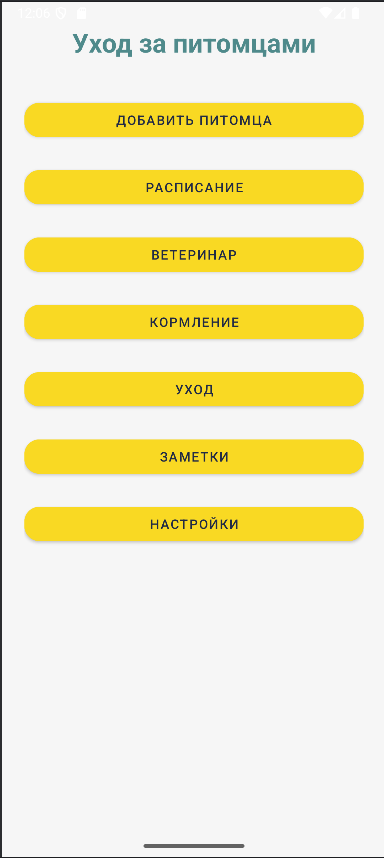


Рисунок 4 – Главный экран

1. Добавление питомца:

Переход на экран добавления питомца

* Открывается новый экран (AddPetActivity)
* В заголовке отображается текст "Добавить питомца"

Отображаются поля для ввода информации:

* Поле "Кличка питомца" (текстовое поле)
* Выпадающий список "Тип животного" (с вариантами: Собака, Кошка и т.д.)
* Поле "Порода" (текстовое поле)
* Поле "Возраст (лет)" (числовое поле)
* Поле "Вес (кг)" (числовое поле с десятичными значениями)

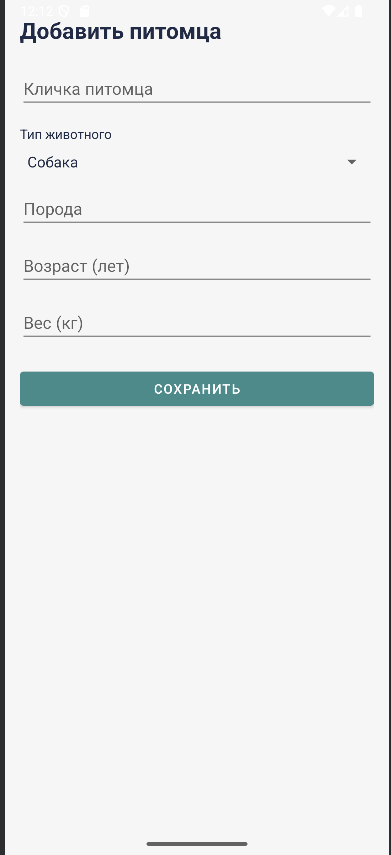


Рисунок 5 – Добавление питомца

1. Расписание:

* Кнопка "Добавить напоминание" (в верхней части)
* Список существующих напоминаний
* Календарь для просмотра напоминаний по датам

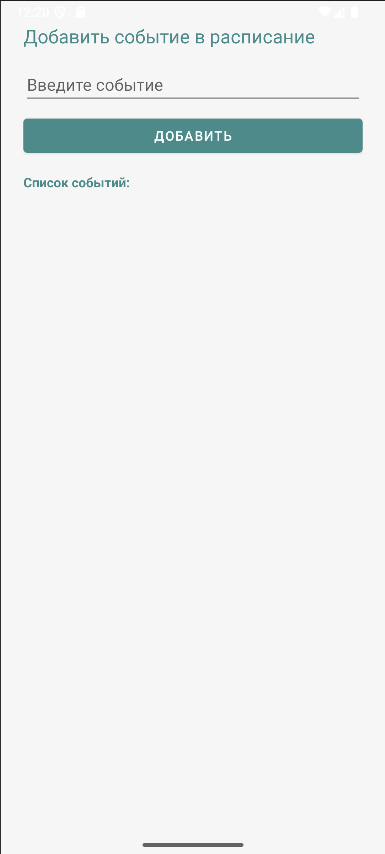


Рисунок 6 – Расписание

1. Ветеринар:

* Кнопка "Имя питомца" (вверху экрана)
* Дата посещения ветеринара
* Время посещения ветеринара
* Причина обращения

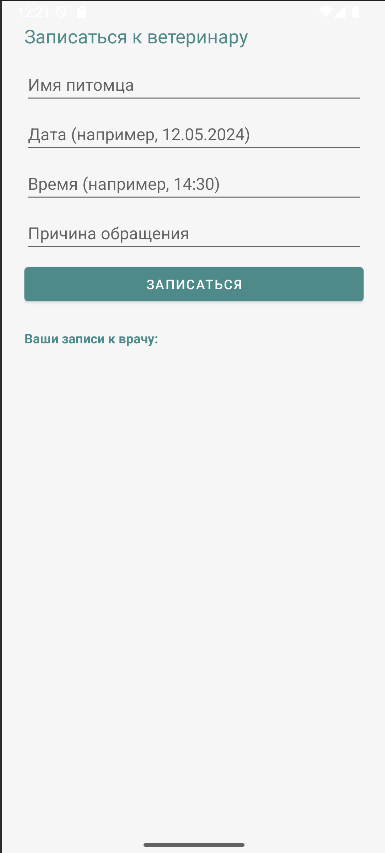


Рисунок 7 – Ветеринар

1. Кормление:

* Поле "Введите запись" (текстовое поле)
* Кнопка "Добавить" (зеленого цвета)
* Кнопка "Добавить напоминание" (зеленого цвета)

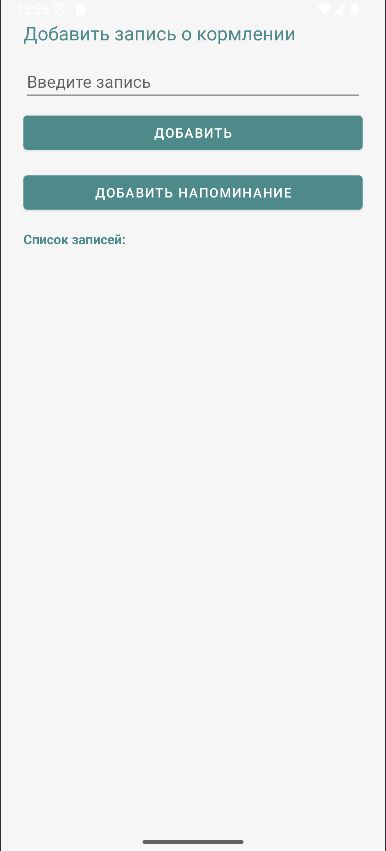


Рисунок 8 – Кормление

1. Уход:

* Заголовок "Рекомендации по уходу"
* Блок рекомендаций
* Раздел "Добавить запись об уходе"
* Поле "Введите запись" (текстовое поле)
* Кнопка "Добавить" (зеленого цвета)
* Раздел "Список записей:"

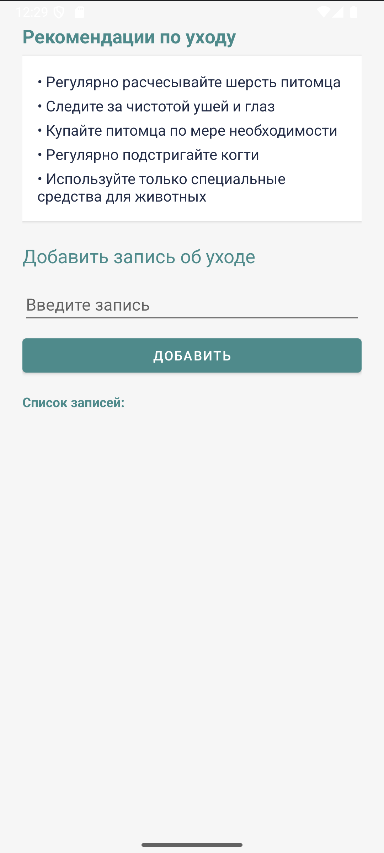


Рисунок 9 - Уход

1. Заметки:

* Заголовок "Добавить заметку"
* Поле "Введите заметку" (текстовое поле)
* Кнопка "Добавить" (зеленого цвета)
* Кнопка "Добавить напоминание" (зеленого цвета)
* Раздел "Список заметок:"

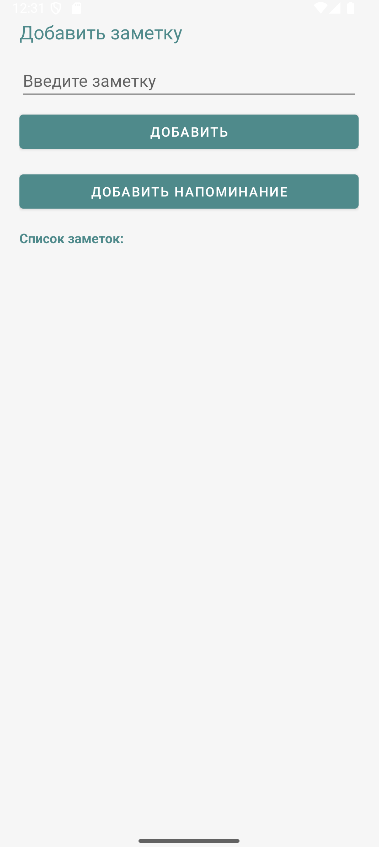


Рисунок 10 – Заметки

1. Настройки:

Заголовок "Настройки"

* Переключатель "Тёмная тема" (с индикатором)
* Переключатель "Уведомления" (с индикатором)
* Кнопка "О приложении" (желтого цвета)
* Текст "Здесь будут другие настройки приложения"

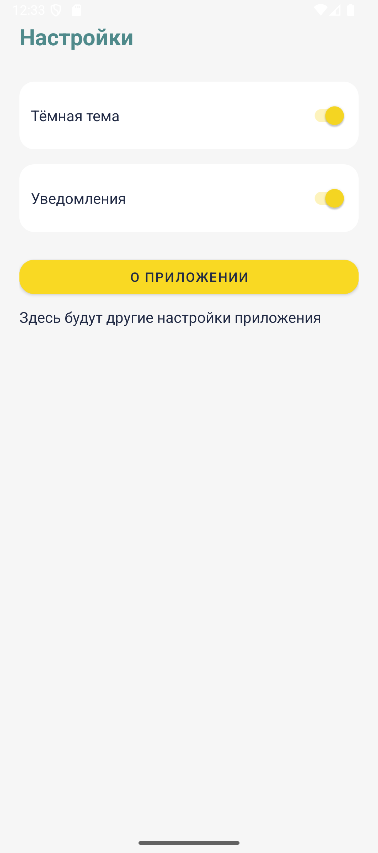


Рисунок 11 - Настройки