Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б. Н. Ельцина» (УрФУ)
Институт радиоэлектроники и информационных технологий – РТФ

ОТЧЕТ
о проектной работе
по теме: Мобильное приложение Thesis
по дисциплине: Проектный практикум 1А

Команда: Vmeste

Тимлид: Гущин Андрей Александрович РИ-140941
Аналитик: Гатаулин Данил Валерьевич РИ-140946
Дизайнер: Тюрина Арина Александровна РИ-140941
Фронтенд-разработчик: Иванов Дмитрий Игоревич РИ-140946
Бэкенд-разработчик: Кулбусинов Еркен Абаевич РИ-140941

Екатеринбург
2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современной образовательной среде студенты сталкиваются с серьезными вызовами в организации своего времени и продуктивности. Высокая учебная нагрузка, плотное расписание, множественные дедлайны и необходимость балансировать между учебой, работой и личной жизнью создают хронический стресс и снижают эффективность. Большинство студентов испытывает трудности с планированием, распределением времени. Существующие инструменты тайм-менеджмента часто требуют ручного ввода данных, не учитывают индивидуальные особенности пользователя, остаются пассивными инструментами. Использующие же искусственный интеллект приложения являются платными и сложными в освоении. Поэтому разработка приложения, использующего искусственный интеллект для помощи в планировании и доступного для студентов, является актуальной задачей.

Целью данного проекта является разработка мобильного приложения для планирования с помощью искусственного интеллекта “Thesis”.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить потребностей студентов и выявление основных проблем, с которыми они сталкиваются в сфере планирования своего времени.

2. Изучение существующих решений на рынке планировщиков задач для выявления основных функциональных возможностей и недостатков.

3. Разработка интуитивно понятного и привлекательного интерфейса мобильного приложения.

4. Реализация основной функциональности приложения, интеграция ИИ

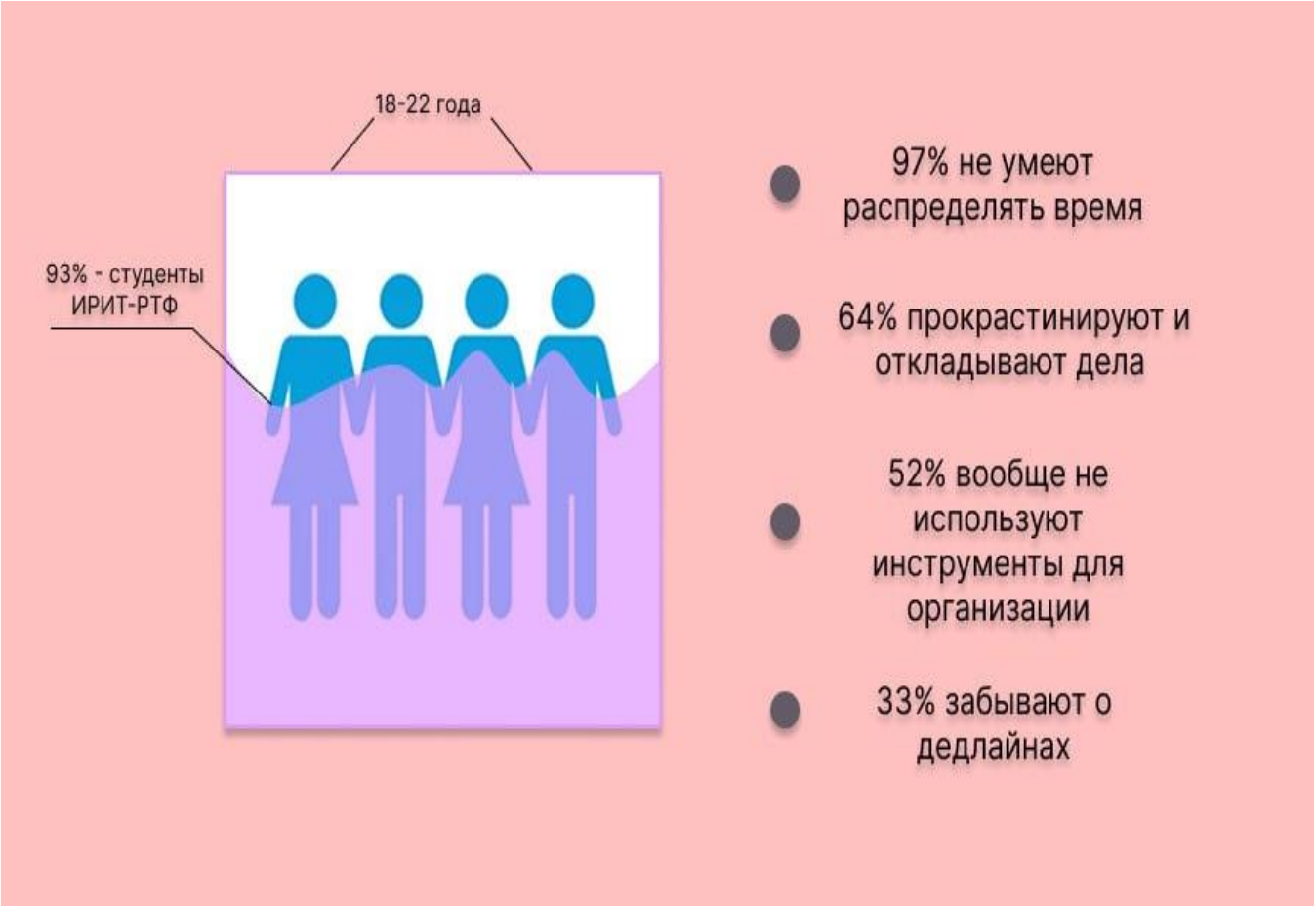
5. Проведение тестирования приложения на различных устройствах для выявления и исправления ошибок

2. ЦЕЛЕВАЯ АУДИТОРИЯ

Нашей целевой аудиторией являются студенты ИРИТ-РтФ, возрастом 18–22 года.

По данным опроса[3], участие в котором принимало 42 человека (из них 93% опрошенных - студенты) 97% не умеют распределять время, 64% прокрастинируют и откладывают дела, 52% вообще не используют инструменты для организации, 33% забывают о дедлайнах.

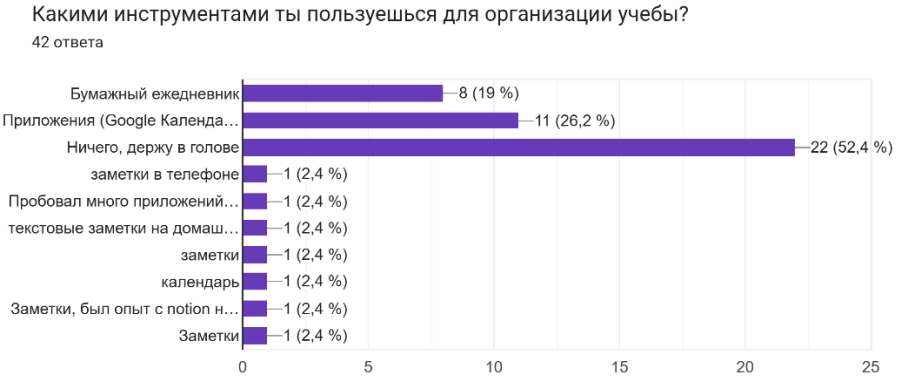
Рисунок 1 – Графическое представление целевой аудитории

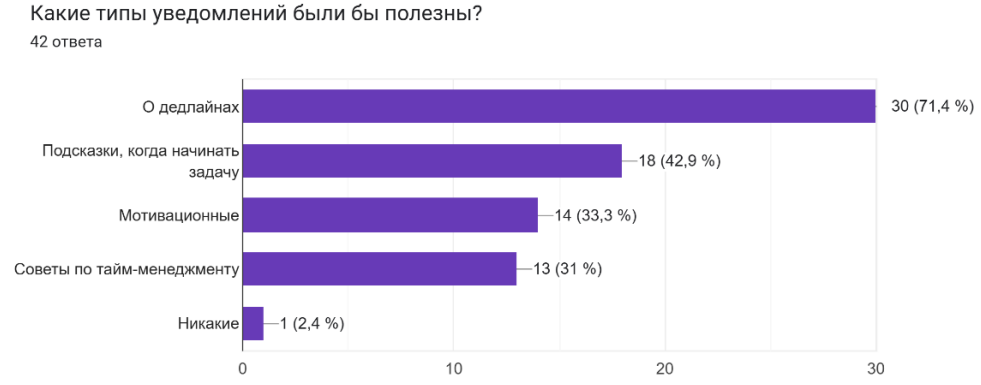


3.ОПРЕДЕЛЕНИЕ ПРОБЛЕМЫ

Наша команда провела опрос [3], в котором приняло участие 42 человека (93% - студенты). Опрос содержал 15 вопросов, связанных с планированием, сложностями с учебой, а также предпочтительными функциями будущего приложения

Рисунки 2, 3, 4 - Примеры вопросов из опроса и ответы на них:





Через анализ целевой аудитории и анализ конкурентов мы выявили такие проблемы:

1. Плохой тайм-менеджмент - В современной образовательной среде плохой тайм-менеджмент стал одной из ключевых проблем, существенно влияющих на качество обучения и психологическое благополучие студентов. По опросу: 97% не умеют распределять время, 64% прокрастинируют и откладывают дела, 33% забывают о дедлайнах.

2. Сложность использования приложений для планирования у новичков - больше половины опрошенных никогда не пользовались приложениями для планирования, в них же представлен перегруженный функционал, который “путает” пользователя, так же чаще всего приложения имеют ограниченную бесплатную версию.

4. ПОДХОДЫ К РЕШЕНИЮ ПРОБЛЕМЫ

Проблема: Плохой тайм-менеджмент

Варианты решения:

Удобные инструменты для планирования - разделение задачи, напоминания о дедлайнах помогут пользователям контролировать свое время

Плюсы: метод разделения задач снижает когнитивную нагрузку, автоматические напоминания уменьшают количество пропущенных дедлайнов, технологическая реализуемость - не требует сложных алгоритмов ИИ для базовой реализации

Минусы: Требуют сознательных усилий от пользователя

Геймификация - Использование игровых механик (баллы, уровни, награды) для мотивации к выполнению задач.

Плюсы: упрощает формирование привычек, снижает прокрастинацию через немедленное вознаграждение

Минусы: риск "игры ради игры" - выполнение бесполезных задач ради баллов, может раздражать пользователей.

Визуализация последствий - Показ негативных результатов бездействия.

Плюсы: работает с эмоциональным интеллектом

Минусы: может повышать тревожность, не заменяет другие методы мотивации

Нами было выбрано создание удобных инструментов для планирования, как наиболее подходящее, с возможностью несложной реализации, но при этом эффективное для пользователя.

Проблема: Сложность использования приложений для планирования у новичков

Варианты решения:

Умный ИИ-ассистент - чат-бот, который обучает работе с приложением в диалоге.

Плюсы: работает через естественный язык, может предлагать советы.

Минусы: движок требует больших вычислительных ресурсов, техническая сложность в реализации.

Разработка удобного и понятного интерфейса, без лишних функций

Плюсы: успешные приложения сделали простоту своим ключевым преимуществом, экономия ресурсов - меньше багов, проще тестирование

- Минусы: Риск необходимости полного редизайна при расширении функционала

Нами было выбрана разработка удобного и понятного интерфейса, как наиболее подходящее решение.

5.АНАЛИЗ АНАЛОГОВ

Сравнительный анализ приложений для планирования и учебы ([4], [7])

1. Motion

- Функционал:
  • Автоматическое распределение задач по календарю
  • Интеграция с Google Calendar и Zoom
  • Учет приоритетов и сроков выполнения
- Особенности:
  • Множество настроек и параметров
  • Стоимость подписки: $34/месяц
  • Требуется время на освоение

2. Reclaim.ai

- Функционал:
  • Синхронизация задач из Todoist, Google Tasks
  • Выделение временных блоков для регулярных активностей
- Особенности:
  • Бесплатная версия: 5 календарей
  • Отсутствует мобильное приложение
  • Часть функций - только в платных тарифах

3. SkedPal

- Функционал:
  • Учет индивидуальных паттернов продуктивности
  • Система цветовых меток и фильтров

- Особенности:

  • Возможность работы оффлайн

  • Стоимость: от $9.95/месяц

4. Shapa

- Функционал:

  • Анализ знаний пользователя

  • Подбор учебных материалов

  • Адаптивные тестовые задания

- Особенности:

  • Отсутствуют инструменты планирования

  • Ориентировано только на образовательные задачи

5. Notion

- Функционал:

  • Организация данных различного типа

  • Генерация текстов и чек-листов

- Особенности:

  • Требуется изучение возможностей

  • Нет автоматического планирования

Ключевые характеристики

1. Модель монетизации:

   - Платная подписка

   - Ограниченный бесплатный функционал

2. Интерфейс:

- Множество настроек и параметров

- Требуется время на освоение

3. Специализация:

- Ориентация на конкретные типы задач

4. Требования:

- Необходимость интернет-соединения

Наше приложение

- Основные особенности:

• Упрощенный интерфейс

• Полный бесплатный доступ

• Оффлайн-работа

• Базовые функции ИИ для учебы

- Отличия:

• Отсутствие рекламы

• Нет платного контента

• Все функции доступны после установки

Таблица 1 –сравнение нашего приложения и конкурентов

Критерий	Наше приложение	Motion	Reclaim.ai	SkedPal	RescueTime	Shapa	Notion
Основной функционал	Удобный интерфейс, расчет времени, уведомления, ИИ-анализ задач	Автопланирование задач, интеграции	Синхронизация задач, защита времени, статистика	Планирование по энергии и приоритетам	Трекинг времени, блокировка отвлекающих сайтов	ИИ-ментор, адаптивные тесты, трекинг прогресса	Гибкие страницы, ИИ-помощник, шаблоны
Плюсы	✅ Минимализм, нет рекламы, без подписки, ИИ-анализ	✅ Автоматизация планирования	✅ Гибкость, бесплатный тариф	✅ Глубокая кастомизация, оффлайн-режим	✅ Автоматический трекинг, блокировка отвлекающих факторов	✅ Персонализированные рекомендации, адаптивное обучение	✅ Многофункциональность, интеграции, ИИ-генерация текста
Минусы	❌ Пока мало известен	❌ Дорого ($34/мес), сложный интерфейс	❌ Нет мобильного приложения, ограничения в бесплатной версии	❌ Сложный интерфейс, цена ($9.95/мес)	❌ Нет планирования задач	❌ Узкая специализация (только для учебы)	❌ Требует времени на освоение
Подходит для учебы	✅ Да (ИИ-анализ задач, уведомления)	❌ Нет (акцент на бизнес-задачи)	❌ Нет (общее планирование)	❌ Нет (общее планирование)	❌ Нет (только трекинг)	✅ Да (адаптивное обучение)	✅ Да (шаблоны, ИИ-помощник)
Использование ИИ	✅ Да (анализ времени, рекомендации)	✅ Да (автопланирование)	✅ Да (умные временные блоки)	❌ Нет	❌ Нет	✅ Да (анализ слабых мест, персонализация)	✅ Да (генерация текста, суммаризация)
Цена	🆓 Бесплатно	💰 $34/месяц	💰 Бесплатно (ограничено), платно — от $8/мес	💰 От $9.95/месяц	💰 Бесплатно (базовый), платно — от $6/мес	💰 Уточняется	💰 Бесплатно (базовый), платно — от $8/мес

6.КАЛЕНДАРНЫЙ ПЛАН ПРОЕКТА

Название проекта: Vmeste

Руководитель проекта: Гущин Андрей Александрович

Таблица 2 – Календарный план

№	Название	Ответственный	Длительность	Дата начала	Временные рамки проекта(недели)															
					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Анализ																				
1.1	*Определение проблемы*	Гатаулин Данил	1 неделя	01.03.25																
1.2	*Выявление целевой аудитории*	Гатаулин Данил	1 неделя	04.03.25																
1.3	*Конкретизац ия проблемы*	Гущин Андрей	1 неделя	08.03.25																
1.4	*Подходы к решению проблемы*	Гатаулин Данил	1 неделя	10.03.25																
1.5	*Анализ аналогов*	Гатаулин Данил	1 неделя	15.03.25																

1.6	*Формулирование требований к MVP продукта*	Гущин Андрей	1 неделя	21.03.25																
1.7	*Формулировка цели*	Гатаулин Данил	1 неделя	16.03.25																
1.8	*Формулирование требований к продукту*	Гущин Андрей	1 неделя	01.04.25																
1.9	*Определение задач*	Гущин Андрей	1 неделя	18.03.25																

Прототипирование																				
2.1	*Разработка сценариев использования системы*	Гущин Андрей	1 неделя	9.04.25																
2.2	*Прототипы интерфейсов*	Тюрина Арина	1 неделя	21.04.25																
2.3	*Дизайн- макеты*	Тюрина Арина	2 недели	23.04.25																
Разработка																				
3.1	*Написание кода*	Кулбусинов Еркен Иванов Дмитрий	4 недели	22.04.25																
3.2	*Тестирование приложения*	Кулбусинов Еркен	1 неделя	17.05.25																
Завершение работы																				
4.1	*Оформление продукта*	Иванов Дмитрий	1 неделя	25.05.25																
4.2	*Написание отчета*	Гущин Андрей Гатаулин Данил	1 неделя	5.06.25																
4.3	*Оформление презентации*	Гущин Андрей	1 неделя	7.06.25																
Защита																				
	Защита проекта	Гущин Андрей	1 неделя	19.06 - 22.06																

7. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

Сценарий использования продукта (use case)[2] представлен в формате блок-схем в приложении 1. Составлен с помощью источника [5].

8. СТЕК ДЛЯ РАЗРАБОТКИ

Приложение будет разрабатываться для платформы Android, исходя из результатов опроса[3].

Разберем некоторые платформы для разработки[8]:

Android Studio

Плюсы:

Полная поддержка языков – работает с Kotlin, Java и C++.

Встроенный эмулятор – можно тестировать на разных устройствах и версиях Android.

Интеграция с сервисами Google – Firebase, Google Cloud и Material Design легко подключаются.

Минусы:

Требует мощного компьютера – «тяжелая» среда, особенно при работе с эмулятором.

Медленная сборка и индексация – на больших проектах возможны задержки.

IntelliJ IDEA

Плюсы:

Основана на той же платформе, что и Android Studio, но без узкоспециализированных Android-инструментов.

Работает быстрее и потребляет меньше ресурсов, чем Android Studio.

Поддерживает множество языков: Kotlin, Java, Python, JavaScript и другие.

Минусы:

Нет встроенного эмулятора Android (нужно использовать сторонний, например, Genymotion или ADB).

Меньше встроенных инструментов для Android-разработки

(требуются дополнительные плагины).

Flutter

Плюсы:

Горячая перезагрузка (изменения в коде сразу отображаются без пересборки).

Хорошая производительность

Встроенные Material Design (Android) и Cupertino (iOS) виджеты.

Минусы:

Не все нативные API доступны "из коробки" (иногда нужны плагины).

Размер приложения больше, чем у нативных решений.

В итоге нашей командой была выбрана Android Studio.

Почему выбрали:

Хорошая производительность

Наличие доступа к индивидуальным возможностям платформы

Язык программирования: Java

Почему выбрали его:

Огромное количество материалов для обучения

Широкая поддержка и совместимость - Все API Android изначально разрабатывались для Java.

9.ТРЕБОВАНИЯ К ПРОДУКТУ И К MVP

Требования к MVP

Функции:

Добавление удаление/редактирование задач

Добавление задач через календарь

Уведомления о дедлайнах

Оценки времени на задачу с помощью ИИ (Финальное время = Время от ИИ *(Коэффициент сложности (от 1 до 5) * Уровень навыка (У каждого уровня навыка есть свой понижающий коэффициент))

Требования к продукту

Приложение должно быть с понятным и простым интерфейсом.

Время отклика должно составлять <10 мс.

Приложение должно иметь возможность загрузки и изменения расписания пользователя, самостоятельно пользователем либо ИИ помощником.

Приложение должно регулярно отправлять уведомления пользователю, напоминая о предстоящих дедлайнах и оптимальном времени для начала выполнения задач.

Приложение должно предлагать сценарии для отправки уведомлений.

Нефункциональные требования

Требования к интерфейсу

Интерфейс Системы должен быть прост, нагляден, интуитивно понятен и легок в освоении

В Приложении должна быть возможность ввода данных с клавиатуры в следующих местах: Чат с ИИ, Добавление и редактирование задач.

Клавиатура в приложении будет такой, какая выбрана у пользователя в системе Android, реализации собственной клавиатуры в приложении не будет.

Требования к расчету времени для уведомлений об оптимальном времени начала выполнения задачи.

Оптимальное время, будет зависеть от приоритета задачи, а также от времени, в которое ИИ оценил задачу. Расчет будет производиться по формуле t=Дедлайн-(Оценка ИИ + Буферное время)

- Для приоритета Low – буферное время = 20% от Оценки ИИ
- Medium – 50% от Оценки ИИ
- High – 80% от Оценки ИИ

Функциональные требования

Требования к добавлению и изменению расписания

Добавление задач и редактирование задач будет осуществляться на вкладке «Задачи на сегодня», а также на вкладке «Обзор задач».

Требования по отображению задач

У всех задач будет 3 кнопки: разбиение на подзадачи, оценка времени, удаление.

Мотивационные уведомления не должны отвлекать пользователя поэтому изначально они будут выключены.

Требования ко вкладке «Обзор задач»

- Ее можно будет открыть из панели навигации внизу экрана кнопка «Документ».
- На этой вкладке по умолчанию задачи будут сгруппированы по дням, поэтому над каждым днем должна выводиться дата.
- В верхней части экрана должна быть лента с датами, чтобы можно было удобно переходить к интересующим дням посредством нажатия на интересующий день.
- Также задачи можно будет обозревать с помощью прокрутки списка вверх/вниз.

С подробными требованиями к приложению в форме технического задания вы можете ознакомиться в Приложении 2.

10. ПРОТОТИПИРОВАНИЕ

Первый прототип: бот для помощи людям с деменцией

1. Бот-напоминатель с адаптивным интерфейсом

-Проблема: Люди с деменцией часто забывают о важных вещах: приеме лекарств, встречах, повседневных делах.

- Решение: Бот, который отправляет напоминания в удобной форме:

- Голосовые сообщения (нейронная сеть может генерировать голос, похожий на голос близкого человека).

- Простые текстовые сообщения с крупным шрифтом.

- Адаптивные напоминания (например, если пользователь не подтвердил выполнение задачи, бот повторяет напоминание).

- Нейронная сеть: Для анализа поведения пользователя и адаптации напоминаний (например, если человек чаще забывает утром, бот усиливает утренние уведомления).

- Проблема: Деменция связана с ухудшением памяти, и регулярные тренировки могут замедлить прогрессирование заболевания.

- Решение: Бот, который предлагает простые упражнения для тренировки памяти:

- Игры на запоминание (например, "Запомни последовательность картинок").

- Вопросы о прошлом (например, "Как зовут твоего внука?").

- Адаптивные задания, которые усложняются или упрощаются в зависимости от прогресса.

- Нейронная сеть: Для анализа прогресса пользователя и персонализации заданий.

С командой отказались от реализации этого прототипа из-за нужды в точности информации, так как речь идет о здоровье и здесь нужна экспертная оценка.

Второй прототип: бот для помощи в планировании

Функции:

1. Автоматическое планирование дня

- Сценарий: Пользователь загружает расписание и добавляет список задач. ИИ анализирует: - Время между занятиями. - Сложность задач. - Приоритеты (например, дедлайны). - Что делает ИИ: - Составляет оптимальный график на день, распределяя задачи между "окнами" в расписании. - Предлагает, сколько времени уделить каждой задаче. - Учитывает предпочтения пользователя (например, "я лучше работаю утром"). - Пример: "У тебя есть 2 часа между лекциями. Я предлагаю потратить 1 час на подготовку к экзамену и 30 минут на написание конспекта."

2. Прогнозирование времени на выполнение задач - Сценарий: Пользователь добавляет задачу, например, "написать реферат на 10 страниц". - Что делает ИИ: - Анализирует прошлые задачи пользователя (например, сколько времени он тратил на написание рефератов). - Предсказывает, сколько времени займёт новая задача. - Предлагает начать выполнение задачи заранее. - Пример: "Обычно ты пишешь реферат на 10 страниц за 5 часов. Начни за 2 дня до дедлайна, чтобы успеть." ---

3. Анализ учебных материалов - Сценарий: Пользователь загружает учебные материалы (например, конспекты, статьи, PDF-файлы). - Что делает ИИ: - Анализирует текст и выделяет ключевые моменты. - Создаёт краткие summary (краткое содержание). - Формирует чек-листы для повторения материала. - Пример: "Я проанализировал твой конспект по математике. Вот ключевые темы, которые нужно повторить: 1) Интегралы, 2) Дифференциальные уравнения."

4. Персонализированные рекомендации по обучению - Сценарий: Пользователь регулярно добавляет задачи и отмечает их выполнение. - Что делает ИИ: - Анализирует, какие темы или типы задач вызывают у пользователя больше всего трудностей. - Предлагает дополнительные материалы (видео, статьи, задачи для тренировки). - Рекомендует, на что обратить больше внимания. - Пример: "Я заметил, что у тебя возникают

сложности с задачами по физике. Вот несколько видео и задач, которые помогут тебе разобраться."\

Этот прототип стал основой нашего итогового продукта. Решили, что для реализации такого функционала телеграм-бот не подойдет и переключились на разработку мобильного приложения. Отказались от ряда функции из-за технических сложностей реализации.

11.ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА СИСТЕМЫ

Приложение разрабатывалось с использованием этих источников: [9], [10], [11], [12], [6].

Архитектура приложения

Архитектура мобильного приложения состоит из Пользовательского интерфейса, Управления, Базы данных и Искусственного интеллекта. Со схемой работы приложения вы можете ознакомиться на рисунке...

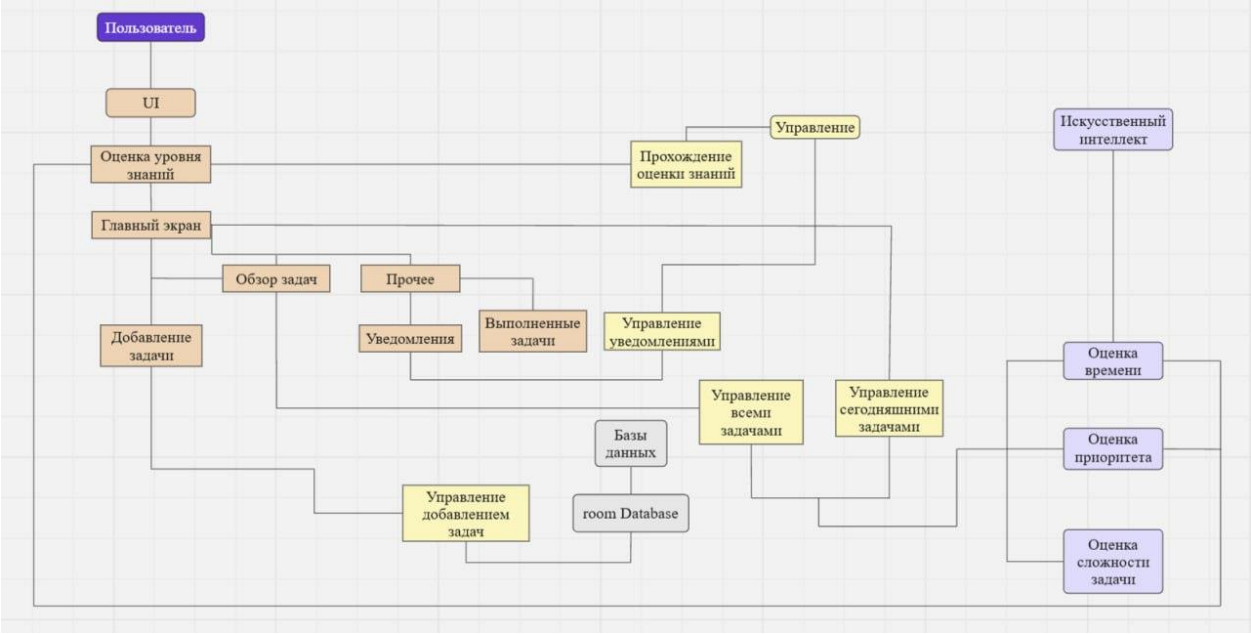


Рисунок - Схема работы приложения

1. Пользовательский интерфейс (UI)

UI включает в себя экраны с контентом, который может наблюдать пользователь.

Страницы, доступные пользователю: Оценка уровня знаний, Главный экран, Обзор задач, Прочее, Добавление задачи, Уведомления, Выполненные задачи.

При первом открытии приложения пользователь проходит Оценку уровня знаний. В последующие разы открывается сразу Главный экран. На главном экране пользователь видит задачи на сегодня. Между всеми экранами (кроме экрана Оценки уровня знаний) пользователь может переключаться с помощью кнопок. Из Главного экрана можно перейти в Обзор задач и Прочее

через кнопки снизу экрана (они связаны между собой, можно переходить из одного в другой. Также можно из Уведомлений и Выполненных сразу перейти в Главный экран и Обзор задач). В Главном экране и Обзоре задач с помощью “+” происходит переход в Добавление задач. Из экрана Прочее можно перейти в Уведомления, Выполненные задачи, Экспорт расписания.

2. Управление

В оценке знаний пользователь выбирает ответ на вопрос в низу экрана, выбор ответов можно скрыть/показать. Управление позволяет взаимодействовать с разделами приложения. В Главном экране можно узнать примерное время выполнения задачи, перейти к редактированию этой задачи, удалить задачу, отметить задачу как выполненную. В экране Обзора задач у пользователя есть возможность выбрать дату (+- 2 дня), также можно открыть календарь и уже выбрать дату там. С задачами можно делать все то же самое, что и на Главном экране. В экране Прочее можно поменять своё имя. В Уведомлениях можно включить/отключить разные виды уведомлений. В Выполненных задачах можно выбрать дату, посмотреть задачи, выполненнные в эту дату.

3. Базы данных

В приложении используется 1 база данных для хранения информации: room Database

4. Искусственный интеллект

Используется DeepSeek-R1-Distill-Llama-70B-free с помощью together.ai; ии оценивает сложность задач, базовое время выполнения задачи, разбивает задачи на подзадачи.

Подробный функционал приложения:

Добавление задач:

Добавление задач происходит с помощью нажатия на “+” на Главном

экране или в Обзоре задач. Можно вписать название задачи, описать ее, написать подзадачи. Сохранение происходит кнопкой “Сохранить” в правом верхнем углу.

Код добавления задачи:

```
     @Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    addTaskBtn = findViewById(R.id.addTaskButton);
    addTaskBtn.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new
Intent(TasksActivity.this, AddTaskActivity.class);
            startActivity(intent);
        }
    });
}
```

Код сохранения задачи:

```
     private void saveTask() {
    String title =
titleEditText.getText().toString().trim();
    String description =
descriptionEditText.getText().toString().trim();

    if (title.isEmpty()) {
        Toast.makeText(this, "Введите название задачи",
Toast.LENGTH_SHORT).show();
```

```
            return;
        }

        TaskDataModel task = new TaskDataModel(title,
description);
        if (taskId != -1) {
            task.setId(taskId); // Для обновления
существующей задачи
        }

        AppDatabase.databaseWriteExecutor.execute(() -> {
            if (taskId == -1) {

AppDatabase.getDatabase(this).taskDao().insert(task);
            } else {

AppDatabase.getDatabase(this).taskDao().update(task);
            }
            finish();
        });
}
```

Удаление задач:

Удалить задачи можно на знак корзины возле задачи в Главном экране или Обзоре задач.

Код удаления задач:

```
     private void
showDeleteConfirmationDialog(TaskDataModel task, int
position) {
    new AlertDialog.Builder(context)
```

```
            .setTitle("Удаление задачи")
            .setMessage("Вы уверены, что хотите удалить
задачу \"" + task.getTitle() + "\"?")
            .setPositiveButton("Удалить", (dialog,
which) -> {
                // Удаляем из базы данных
                new Thread(() -> {
                    taskDao.delete(task);
                    // Удаляем из списка и обновляем UI
на главном потоке
                    ((Activity)
context).runOnUiThread(() -> {
                        tasks.remove(position);
                        notifyItemRemoved(position);

notifyItemRangeChanged(position, tasks.size());
                    });
                }).start();
            })
            .setNegativeButton("Отмена", null)

.setIcon(android.R.drawable.ic_dialog_alert)
            .show();
}
```

Примерное время выполнения задач:

Когда происходит нажатие на часы возле задачи в Главном экране или Обзоре задач, всплывает окно, в котором написано примерное время выполнения задач. Оно рассчитывается по формуле: Финальное время = (Базовое время от ИИ) × (Коэффициент сложности / Уровень навыка).

Код подачи запроса ИИ:

```
     private String createPrompt(String title, String
description) {
    return "Оцени время выполнения задачи и ее
сложность. " +
            "Задача: " + title + "\nОписание: " +
description + "\n\n" +
            "Ответь строго в формате JSON:
{\"base_time\": X, \"complexity\": Y} " +
            "где X - время в минутах (только число), Y
- сложность от 1 до 5 (только число)";
}
```

Код извлечения ответа ИИ:

```
     private void
handleSuccessfulResponse(Response<ChatCompletionRespons
e> response, TimeEstimationCallback callback) {
    String content =
response.body().choices.get(0).message.content;
    String jsonString = extractJson(content);

    if (jsonString == null) {
        callback.onError("Не удалось извлечь JSON из
ответа ИИ");
        return;
    }

    try {
        JSONObject json = new JSONObject(jsonString);
        int baseTime = json.getInt("base_time");
```

```
            int complexity = json.getInt("complexity");
            callback.onEstimated(baseTime, complexity);
        } catch (Exception e) {
            callback.onError("Неверный формат JSON: " +
e.getMessage());
        }
}

private String extractJson(String text) {
    String regex = "\\{.*?\\}";
    Pattern pattern = Pattern.compile(regex);
    Matcher matcher = pattern.matcher(text);
    if (matcher.find()) {
        return matcher.group(0);
    }
    return null;
}
```

Расчет приоритета выполнения задач:

Чем выше приоритет у задачи, тем выше она будет в списке. Приоритет считается по формуле: Приоритет = Важность / (Сложность * 0.5 + Время * 0.2)

Расчет приоритета в коде:

```
     private void
estimateTaskComplexityAndTime(TaskDataModel task, int
skillLevel) {
    AlertDialog loadingDialog = new
AlertDialog.Builder(context)
            .setTitle("Оценка параметров задачи")
            .setView(R.layout.dialog_loading) //
```

```
Создайте этот layout или используйте свой
            .setCancelable(false)
            .create();
    loadingDialog.show();

    estimateTaskTime(task.getTitle(),
task.getDescription(), new TimeEstimationCallback() {
        @Override
        public void onEstimated(int baseTime, int
complexity) {
            task.setBaseTime(baseTime);
            task.setComplexity(complexity);
            task.calculatePriority(skillLevel);
            updateTaskInDatabase(task);
            loadingDialog.dismiss();
            Toast.makeText(context,
                    "Приоритет рассчитан: " +
String.format("%.2f", task.getPriority()),
                    Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onError(String error) {
            loadingDialog.dismiss();
            Toast.makeText(context, "Ошибка: " + error,
Toast.LENGTH_LONG).show();
        }
    });
}
     public void calculatePriority(int userSkillLevel)
```

```
{
    double complexityFactor = 0.5 + (complexity *
0.25);
    double skillMultiplier =
getSkillMultiplier(userSkillLevel);
    double finalTime = baseTime * (complexityFactor /
skillMultiplier);
    this.priority = (float) (importance / (complexity *
0.5 + finalTime * 0.2));
}
```

Изменение состояния задачи:

Задачу можно пометить выполненной. Для этого нужно нажать на пустой кружок слева от задачи, тогда он пометиться галочкой. Повторное нажатие будет убирать галочку.

Логика в коде:

```
     // Геттер для статуса выполнения
public boolean isCompleted() {
    return isCompleted;
}

// Сеттер для статуса выполнения
public void setCompleted(boolean completed) {
    isCompleted = completed;
}
```

Уведомления:

Можно включить следующие уведомления: мотивационные, о дедлайнах, рекомендации. Они будут приходить раз в 2 часа.

Код для уведомлений:

```
    public class NotificationReceiver extends
BroadcastReceiver {
    private static final int MOTIVATION_NOTIFICATION_ID
= 1;
    private static final String[] MOTIVATIONAL_MESSAGES
= {
            "Даже если кажется, что всё сложно, помни:
каждый большой успех начинается с маленького шага.
Продолжай!",
            "Ты уже так близко к цели! Осталось совсем
немного — соберись, и всё получится!",
            "Не откладывай на завтра то, что можно
сделать сегодня — и ты уже на пути к успеху!",
            "Работа важна, но не забывай отдыхать.
Лучшие идеи приходят в моменты расслабления!",
            "Успех — это движение от неудачи к неудаче
без потери энтузиазма. Продолжай идти!",
            "Не говори, что у тебя мало времени. У тебя
ровно столько же часов в сутках, сколько было у великих
людей!",
            "Будущее зависит от того, что ты делаешь
сегодня. Начни прямо сейчас!",
            "Сложные задачи — как пазлы: если разобрать
на кусочки, они становятся проще.",
            "Прежде чем сдаться, вспомни, зачем ты
начал(а). Эта причина всё ещё важна, правда?",
            "Рост не всегда виден сразу. Даже когда ты
не замечаешь изменений — они есть. Продолжай."
    };
```

```
    @Override
    public void onReceive(Context context, Intent
intent) {
        try {
            NotificationManager notificationManager =
                    (NotificationManager)
context.getSystemService(Context.NOTIFICATION_SERVICE);

            if (notificationManager == null) {
                Log.e("NOTIFY", "NotificationManager is
null");
                return;
            }

            // Создаем канал для Android 8.0+
            if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.O) {
                NotificationChannel channel = new
NotificationChannel(
                        "VMESTE_NOTIFICATIONS",
                        "Уведомления Vmeste",

NotificationManager.IMPORTANCE_HIGH
                );
                channel.setDescription("Мотивационные
уведомления");

notificationManager.createNotificationChannel(channel);
            }
```

```
            // Генерируем уникальный ID для каждого
уведомления
            int notificationId = (int)
System.currentTimeMillis();

            Notification notification = new
NotificationCompat.Builder(context,
"VMESTE_NOTIFICATIONS")

.setSmallIcon(R.drawable.ic_notification)

.setContentTitle(intent.getStringExtra("title"))

.setContentText(getRandomMotivationalMessage())

.setPriority(NotificationCompat.PRIORITY_HIGH)
                    .setAutoCancel(true)
                    .build();

            notificationManager.notify(notificationId,
notification);

        } catch (Exception e) {
            Log.e("NOTIFY", "Ошибка показа
уведомления", e);
        }
    }

    private String getRandomMotivationalMessage() {
        String[] messages = {
```

```
                "Даже если кажется, что всё сложно,
помни: каждый большой успех начинается с маленького
шага. Продолжай!",
                "Ты уже так близко к цели! Осталось
совсем немного — соберись, и всё получится!",
                "Не откладывай на завтра то, что можно
сделать сегодня — и ты уже на пути к успеху!",
                "Работа важна, но не забывай отдыхать.
Лучшие идеи приходят в моменты расслабления!",
                "Успех — это движение от неудачи к
неудаче без потери энтузиазма. Продолжай идти!",
                "Не говори, что у тебя мало времени. У
тебя ровно столько же часов в сутках, сколько было у
великих людей!",
                "Будущее зависит от того, что ты
делаешь сегодня. Начни прямо сейчас!",
                "Сложные задачи — как пазлы: если
разобрать на кусочки, они становятся проще.",
                "Прежде чем сдаться, вспомни, зачем ты
начал(а). Эта причина всё ещё важна, правда?",
                "Рост не всегда виден сразу. Даже когда
ты не замечаешь изменений — они есть. Продолжай."
        };
        return messages[new
Random().nextInt(messages.length)];
    }
}
```

Оценка знаний:

Оценка знаний происходит в виде диалога с ботом. Он задает

вопросы, пользователь отвечает на них. По окончанию оценки приложение получает уровень знаний пользователя, оно будет использоваться в формулах расчета примерного времени выполнения задачи и в приоритетах.

Код примера хранения вопросов/ответов:

```
    private final String[] questions = {
        "Какой язык программирования используется в
Android?",
        "Столица Франции?",
        "2 + 2 = ?"
};

private final String[][] answerOptions = {
        {"Kotlin", "Java", "Python", "C++"},
        {"Париж", "Лондон", "Берлин", "Мадрид"},
        {"4", "5", "22", "0"}
};
```

Код проверки правильности ответов:

```
    private void processAnswer(String selectedAnswer)
{
    addMessage(selectedAnswer, false);

    // Проверка правильного ответа
    boolean isCorrect =
selectedAnswer.equals(answerOptions[currentQuestionInde
x][0]);
    if (isCorrect) {
        correctAnswersCount++;
        addMessage("✓ Верно!", true);
    } else {
```

```
            addMessage("✗ Неверно. Правильный ответ: " +
answerOptions[currentQuestionIndex][0], true);
        }

        currentQuestionIndex++;
        answerOptionsContainer.setVisibility(View.GONE);

        if (currentQuestionIndex < questions.length) {
            new
Handler().postDelayed(this::showNextQuestion, 1500);
        } else {
            showFinalResults();
        }
}
```

ЗАКЛЮЧЕНИЕ

Проблема плохого тайм-менеджмента среди студентов остается одной из ключевых в современной образовательной среде. Высокая учебная нагрузка, плотное расписание, множественные дедлайны и необходимость балансировать между учебой, работой и личной жизнью создают хронический стресс и снижают эффективность. Существующие инструменты планирования зачастую требуют ручного ввода данных, не учитывают индивидуальные особенности пользователей и не помогают бороться с прокрастинацией, а доступные ИИ-решения оказываются либо платными, либо слишком сложными в использовании.

Разработка мобильного приложения "Thesis", использующего искусственный интеллект для автоматизации планирования, является актуальным и востребованным решением. Основная цель проекта — создать удобный, интуитивно понятный и доступный инструмент, который поможет студентам эффективно распределять учебную нагрузку, сокращать время на организацию задач и снижать уровень стресса.

Для достижения этой цели были поставлены и последовательно решались следующие задачи:

1. Анализ потребностей студентов — выявление ключевых проблем в тайм-менеджменте и прокрастинации.
2. Исследование рынка — изучение существующих решений, их сильных и слабых сторон.
3. Проектирование интерфейса — создание удобного и современного дизайна, ориентированного на студентов.
4. Разработка функционала — реализация автоматического планирования, интеграция ИИ и инструментов борьбы с прокрастинацией.
5. Тестирование — проверка стабильности работы приложения на различных устройствах и устранение ошибок.

В отличие от существующих аналогов, приложение ориентировано именно на студентов, учитывая их потребности, цифровые привычки и психологические особенности.

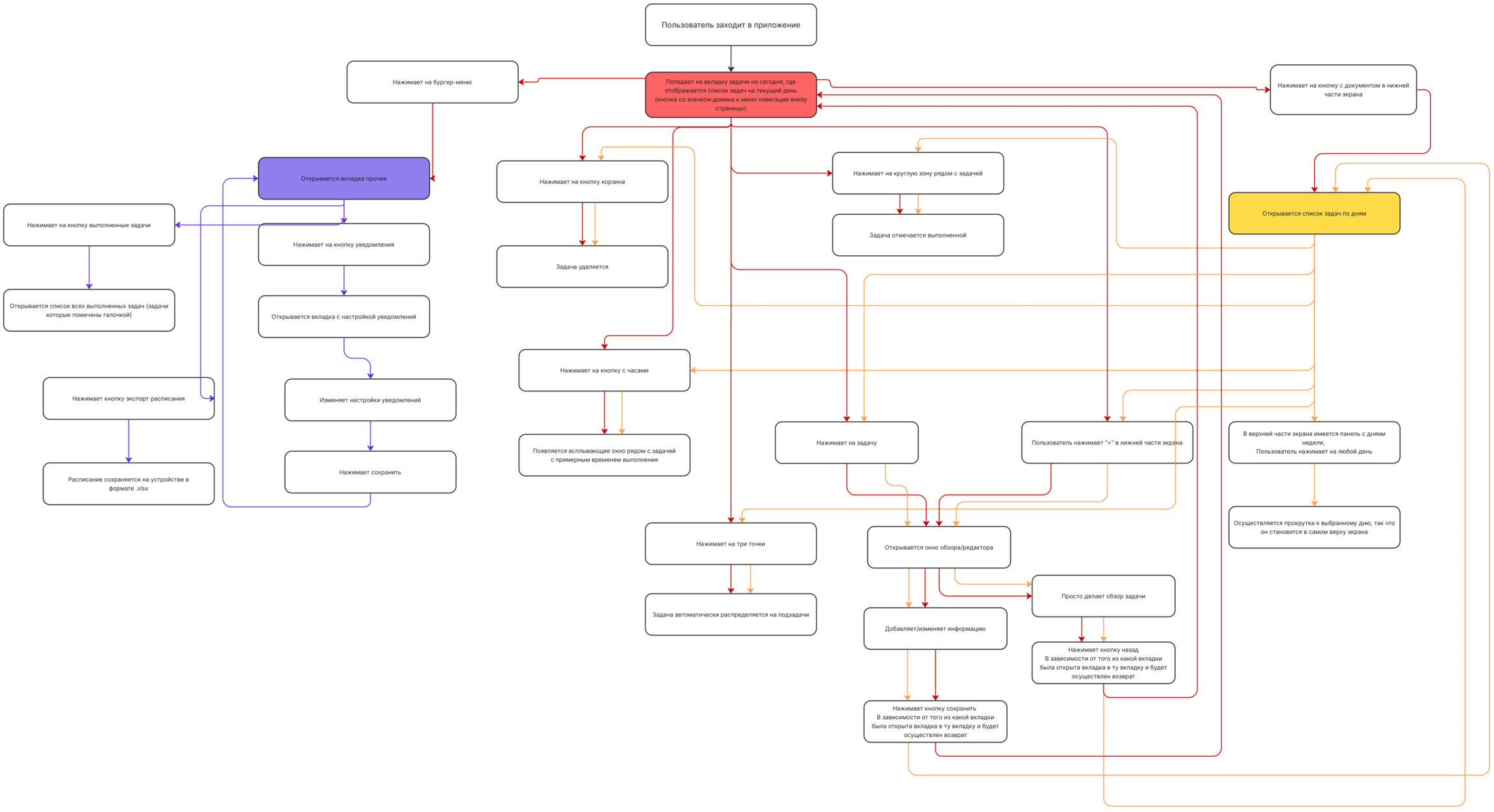
Таким образом, проект "Thesis" не только решает актуальную проблему плохого тайм-менеджмента, но и создает новый стандарт в инструментах для планирования — интеллектуальный, адаптивный и доступный помощник, который помогает студентам учиться эффективнее и с меньшим стрессом.

Дальнейшее развитие проекта включает расширение функционала, интеграцию с новыми образовательными платформами и улучшение алгоритмов ИИ для еще более точного прогнозирования и персонализации.

СПИСОК ИСТОЧНИКОВ

1. Доска в Figma – 2025. – URL: https://www.figma.com/design/ii3B0EY84UzCm0qSliT3WP/Untitled--Copy-?node-id=0-1&p=f&t=5u1EK3D3eC6dRPmx-0 (дата создания: 28.03.2025).
2. UseCase – 2025. – URL: https://miro.com/app/board/uXjVI-qM74Q=/?userEmail=guschinandrey06@gmail.com&shareBoard=danilgataulin9329@gmail.com&track=true&utm_source=notification&utm_medium=email&utm_campaign=request-for-access&utm_content=open-sharing-settings&flow_feature=access_board&flow_type=request&lid=xuboqkm6jgvs (дата создания: 22.04.2025).
3. Опрос – 2025. – URL: https://docs.google.com/forms/d/e/1FAIpQLSdZEPQlS4mAGlsUvcOeNNx6V5boSRUVpPxAOGsiFb1EdGQzzA/viewform (дата создания: 30.03.2025)
4. Эффективный тайм-менеджмент. Топ-15 планировщиков задач на любой случай. – 2022. – URL: https://blog.eldorado.ru/publications/effektivnyytaym-menedzhment-top-15-planirovshchikov-zadach-na-lyuboy-sluchay-30617 (дата обращения: 03.04.2025).
5. Варианты на все случаи жизни: как написать полезный use case – 2022. – URL: https://practicum.yandex.ru/blog/chto-takoe-use-case-kak-ih-napisat/ (дата обращения: 21.04.2025).
6. Android App Development in Java All-in-One Tutorial Series – 2020. – https://www.youtube.com/watch?v=tZvjSl9dswg&t=3385s (дата обращения 24.04.2025)
7. 15 лучших ИИ инструментов для учебы в 2025 году – 2025. – URL: https://vc.ru/services/1742860-15-luchshih-ii-instrumentov-dlya-ucheby-v-2025-godu (дата обращения: 25.03.2025).

8. Top 10 Best Android App Development Languages for 2024 – 2024. – URL: https://www.designveloper.com/blog/android-app-developmentlanguages/ (дата обращения: 28.03.2025).
9. РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ НА ЯЗЫКЕ JAVA С ИСПОЛЬЗОВАНИЕМ ANDROID STUDIO – 2020. – https://cchgeu.ru/upload/iblock/65d/fedcmxcahj8k3o4s2xh9ooqsx35z461i/Uchebn_posobie-Razrabotka-mobilnykh-prilozheniy-na-yazyke-JAVA-s-ispolzovaniem-ANDROID-STUDIO.pdf.pdf (дата обращения: 21.04.2025).
10. Первый проект в Android Studio – 2023. – https://metanit.com/java/android/1.2.php (дата обращения: 20.04.2025).
11. Уроки Android Studio с нуля – 2022. – https://www.youtube.com/watch?v=6cnWnHUScp4 (дата обращения 23.04.2025)
12. Что такое ListView и Adapters? Создание списков – 2022. – https://www.youtube.com/watch?v=jIRtqRIsFlg&t=2469s (дата обращения 23.04.2025)


Пользователь заходит в приложение
Попадает на вкладку задачи на сегодня, где отображается список задач на текущий день (кнопка со значком домика к меню навигации внизу страницы)
Нажимает на бургер-меню
Открывается вкладка прочее
Нажимает на кнопку выполненные задачи
Открывается список всех выполненных задач (задачи которые помечены галочкой)
Нажимает на кнопку уведомления
Открывается вкладка с настройкой уведомлений
Изменяет настройки уведомлений
Нажимает сохранить
Нажимает кнопку экспорт расписания
Расписание сохраняется на устройстве в формате .xlsx
Нажимает на кнопку корзина
Задача удаляется
Нажимает на круглую зону рядом с задачей
Задача отмечается выполненной
Нажимает на кнопку с часами
Появляется всплывающее окно рядом с задачей с примерным временем выполнения
Нажимает на задачу
Пользователь нажимает "+" в нижней части экрана
Нажимает на три точки
Задача автоматически распределяется на подзадачи
Открывается окно обзора/редактора
Просто делает обзор задачи
Добавляет/изменяет информацию
Нажимает кнопку назад
В зависимости от того из какой вкладки была открыта вкладка в ту вкладку и будет осуществлен возврат
Нажимает кнопку сохранить
В зависимости от того из какой вкладки была открыта вкладка в ту вкладку и будет осуществлен возврат
Нажимает на кнопку с документом в нижней части экрана
Открывается список задач по дням
В верхней части экрана имеется панель с днями недели,
Пользователь нажимает на любой день
Осуществляется прокрутка к выбранному дню, так что он становится в самом верху экрана

ПРИЛОЖЕНИЕ 2

Техническое задание

на разработку приложения для планирования с помощью искусственного интеллекта

Команда: Vmeste

Тимлид: Гущин Андрей Александрович РИ-140941
Аналитик: Гатаулин Данил Валерьевич РИ-140946
Дизайнер: Тюрина Арина Александровна РИ-140941
Фронтенд-разработчик: Иванов Дмитрий Игоревич РИ-140946
Бэкенд-разработчик: Кулбусинов Еркен Абаевич РИ-140941

Екатеринбург
2025

1.Общие положения

1.1 Полное наименование приложения

Полное наименование – мобильное приложение для планирования с помощью искусственного интеллекта “Thesis”.

1.2 Определения, обозначения и сокращения

Приложение - мобильное приложение для планирования с помощью искусственного интеллекта “Thesis”.
Пользователь – студент учебного заведения.

ИИ – Искусственный интеллект.

2. Назначения и цели создания приложения

2.1.1 Назначение приложения

Приложение предназначено для планирования своих дел в едином пространстве и контроля сроков выполнения задач.

2.1.2 Назначение ИИ.

Искусственный интеллект в приложении предназначен для:

1. Автоматического распределения задач на основе расписания пользователя, приоритетов и дедлайнов.

2. Прогнозирования времени на выполнение задач с учетом истории пользователя и сложности заданий.

2.2 Цели создания приложения

Цель создания – повышение качества жизни пользователя за счет:
1. Обеспечения контроля и отслеживания прогресса выполнения задач и достижения поставленных целей.
2. Создания возможности для планирования временем и задачами.
3. Снижения вероятности упущения сроков и задач.
4. Сокращения времени, затраченного на планирование и управление задачами.

3. Требования к приложению.

1. Приложение должно быть с понятным и простым интерфейсом.

2. Приложение должно быть разработано под Android OS начиная с версии Android 7.0

3. Приложение должно хранить некоторые данные пользователя локально на устройстве.

4. Время отклика должно составлять <10 мс.

5. Приложение должно иметь возможность загрузки и изменения расписания пользователя, самостоятельно пользователем либо ИИ помощником.

6. Приложение должно регулярно отправлять уведомления пользователю, напоминая о предстоящих дедлайнах и оптимальном времени для начала выполнения задач.

7. Приложение должно предлагать сценарии для отправки уведомлений.

3.1 Нефункциональные требования

3.1.1 Требования к интерфейсу

Интерфейс Системы должен быть прост, нагляден, интуитивно понятен и легок в освоении, и должен отвечать следующим требованиям:

- Он должен сочетать в себе цвета, которые помогают успокоиться (снять стресс), а также которые стимулируют человека работать. Такими цветами являются: #000000 (Чёрный), #FF8D71 (Тёплый кораллово-розовый), #FFE8DF (Светлый персиковый), #FFFDFD (Белый с лёгким розовым подтоном)
- Шрифт – Jost**.**

3.1.2. Требования к техническим параметрам приложения

Приложение должно работать на устройствах, удовлетворяющих требованиям:

- Свободное место на диске 800 мб.

- Оперативная память 1гб.
- Приложение должно быстро открываться и выполнять основные операции с минимальными задержками, не заметными для пользователя.
- Приложение должно быть разработано под Android OS начиная с версии Android 7.0
- Обязателен доступ к интернету

3.1.3. Требования к навигации.

- При входе в приложение будет открываться вкладка «Расписание на сегодня». В приложении будет 3 основных вкладки «Задачи на сегодня», «Обзор задач», «Прочее».
- Переход между вкладками будет осуществляться с помощью меню навигации в нижней части экрана.
- На вкладках «Задачи на сегодня», «Обзор задач», «Выполненные задачи», а также в редакторе задач будет реализована прокрутка скроллом.
- В меню будут представлены кнопки:
    - «Домик» - Задачи на сегодня
    - «Документ» - Обзор задач
    - «Бургер-меню» - Прочее

3.1.4. Требования к вводу данных

- В Приложении должна быть возможность ввода данных с клавиатуры в следующих местах: Чат с ИИ, Добавление и редактирование задач.
- Клавиатура в приложении будет такой, какая выбрана у пользователя в системе Android, реализации собственной клавиатуры в приложении не будет.

3.1.5 Требования к ИИ

1. Производительность: прогноз времени для задачи: <10 секунд,

разбиение задачи на подзадачи: <20 секунд
2. Погрешность прогнозирования для времени выполнения задач: ±20% от реального затраченного времени.

3.ИИ должен поддерживать русский язык.

4.Совместимость: Android OS версии 7.0 и выше.

3.1.6. Требования к распределению задач.

Распределение задач будет происходить с помощью алгоритма, основанного на принципе «Most Important Tasks»: Выделение 1–3 самых важных задач на день, которые обязательно нужно выполнить. Шаг 1. Автоматический выбор MIT: ИИ анализирует задачи и назначает MIT на основе:

- Дедлайнов
- Сложности
- Приоритетов

Шаг 2. Определение «окон» для задач: ИИ ищет свободные промежутки в расписании.

3.1.7. Требования к расчету времени для уведомлений об оптимальном времени начала выполнения задачи.

Оптимальное время, будет зависеть от приоритета задачи, а также от времени, в которое ИИ оценил задачу. Расчет будет производиться по формуле t=Дедлайн-(Оценка ИИ + Буферное время)

- Для приоритета Low – буферное время = 20% от Оценки ИИ
- Medium – 50% от Оценки ИИ
- High – 80% от Оценки ИИ

3.2 Функциональные требования

3.2.1. Требования к добавлению и изменению расписания

- Добавление задач и редактирование задач будет осуществляться на вкладке «Задачи на сегодня», а также на вкладке «Обзор задач».

- При нажатии на существующую задачу открывается окно или кнопку «+» будет открываться «Редактор задач» и если мы добавляем задачу, то в полях будут значения по умолчанию (имя задачи – «Новая задача», описание – «Введите описание задачи», подзадачи – «Подзадачи еще не добавлены), иначе в полях будет информация о задаче.
- Задача подгоняется по ширине к полю, слова не помещающиеся на текущей строке переносятся на следующую.
- Если задача длиннее чем поля в редакторе задач, у поля будет появляться полоса прокрутки с правого края.

3.2.2. Требования по отображению задач

- Задачи будут отображаться на двух вкладках: «Задачи на сегодня», а также на вкладке «Обзор задач», где можно будет увидеть как предстоящие задачи, так и выполненные.
- На вкладке «Обзор задач» задачи будут отображаться списком, в котором будет представлен день, и задачи, которые на него назначены.
- На вкладке «Задачи на сегодня» будет представлен блок задач, над которым будет подписан текущее число и день недели, а также количество задач.
- Рядом с задачей на вкладке «Задачи на сегодня» будет кружок, по нажатии на который задача будет отмечаться выполненной, и рядом с ней будет появляться галочка
- У всех задач будет 3 кнопки: разбиение на подзадачи, оценка времени, удаление.

3.2.3. Требования к настройке напоминаний

- Настройка напоминаний осуществляется в разделе «Прочее» => «Уведомления». В нем можно будет выбрать какой тип напоминаний можно включить/выключить.

- Напоминания отключаются с помощью тумблера рядом с видом напоминания.

3.2.4. Требования к мотивационным уведомлениям

- Мотивационные уведомления не должны отвлекать пользователя поэтому изначально они будут выключены.
- Они не должны мешать пользователю, поэтому они должны приходить 2 раза в день.
- В тексте уведомления будут выводиться предварительно прописанные слова мотивации.

3.2.5. Требования к уведомлениях о дедлайнах

- Напоминание должно состоять из «Заголовка» - название задачи, и текст уведомления – время которое осталось до окончания задачи.
- Количество напоминаний зависит от приоритета задачи:
    - Приоритет Low – 1 напоминание в день дедлайна за 3 часа
    - Приоритет Medium – напиминание за день и за 3 часа до дедлайна
    - Приоритет High – напоминание за 3 дня, за день, за 3 часа до дедлайна

3.2.6. Требования к уведомлениям об оптимальном начале задачи

- Данный тип уведомлений отключен по умолчанию
- Если пользователь воспользуется функцией оценки времени на задачу, то тогда ему придет уведомление об оптимальном времени для начала задачи.

3.2.4. Требования ко вкладке «Задачи на сегодня»

- При открытии приложения данная вкладка будет активна.

- Ее можно будет открыть из панели навигации внизу экрана кнопка «Домик».
- Она должна содержать блок задач, и текущую дату.
- Если задач больше чем позволяет увидеть экран, то задачи можно будет пролистать свайпом вверх/вниз.

3.2.5. Требования ко вкладке «Обзор задач»

- Ее можно будет открыть из панели навигации внизу экрана кнопка «Документ».
- На этой вкладке по умолчанию задачи будут сгруппированы по дням, поэтому над каждым днем должна выводиться дата.
- В верхней части экрана должна быть лента с датами, чтобы можно было удобно переходить к интересующим дням посредством нажатия на интересующий день.
- Также задачи можно будет обозревать с помощью прокрутки списка вверх/вниз.

3.2.9. Требования к вкладке прочее

- На вкладку можно попасть по нажатию на бургер-меню в панели навигации
- На данной вкладке будет 2 кнопки «Уведомления» «Выполненные задачи».

3.2.10. Требования к вкладке «Выполненные задачи»

- Попасть на эту вкладку можно через «Прочее» => «Выполненные задачи»
- На данной вкладке будет представлен список задач по дням, с условием фильтрации – «задача выполнена»
- Перед задачами каждого дня должна быть дата.

- Никаких кнопок около задачи не будет, а также в этой вкладке нельзя будет убрать отметку, что задача выполнена, на ней просто можно обозревать выполненные задачи.

3.2.6. Требования к отправке уведомлений

- Напоминания присылаются в виде уведомлений.
- По нажатию на уведомление открывается приложение.

3.2.7. Требования к оценке времени на задачу

- ИИ анализирует задачу и возвращает примерное время для выполнения задачи
- Оценка происходит на основе использования прошлых результатов пользователя для прогноза времени на новые задачи.
- Результаты анализа помещается в специальное поле внутри задачи.

3.2.8. Требования к экспорту расписания.

- При нажатии на кнопку «Экспорт расписания» открывается проводник с возможностью выбора папки для сохранения файла формата .ics и кнопка «Сохранить».

В случае, если Пользователь нажал на кнопку «Сохранить», вся информация

из расписания конвертируется в формат .ics и сохраняется в файле в выбранной

директории, окно проводника закрывается, а в Приложении выводится

текстовое оповещение: «Расписание было успешно экспортировано в формате ICalendar.».

В случае, если Пользователь не нажал на кнопку «Сохранить», файл нигде не сохраняется.

Название файла должно иметь вид: schedule _(текущая дата и время).ics.