

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа «Программная инженерия»

СОГЛАСОВАНО

Приглашенный преподаватель ФКН
НИУ ВШЭ

_____ Н. И. Веселко
«__» _____ 2026 г.

УТВЕРЖДЕНО

Академический руководитель
образовательной программы
«Программная инженерия», старший
преподаватель департамента
программной инженерии

_____ Н. А. Павлов
«__» _____ 2026 г.

**ГЕЙМИФИЦИРОВАННОЕ МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ
УПРАВЛЕНИЯ ЗАДАЧАМИ И ВИЗУАЛИЗАЦИИ ПРОГРЕССА - TASKORIA**

Пояснительная записка

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.06.05-01 81 01-1-ЛУ

Исполнитель:

Студент группы БПИ247

_____/ В. А. Путинцева /

«__» _____ 2026 г.

Подп. и дата	
Инв.№ дубл.	
Взам. инв.№	
Подп. и дата	
Инв.№ подл.	

УТВЕРЖДЕН

RU.17701729.06.05-01 81 01-1-ЛУ

**ГЕЙМИФИЦИРОВАННОЕ МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ
УПРАВЛЕНИЯ ЗАДАЧАМИ И ВИЗУАЛИЗАЦИИ ПРОГРЕССА - TASKORIA**

Пояснительная записка

RU.17701729.06.05-01 81 01-1

Листов 29

Инв.№ подл	Подп. и дата	Взам. инв.№	Инв.№ дубл.	Подп. и дата

АННОТАЦИЯ

Данный программный документ представляет собой пояснительную записку к программному проекту «Геймифицированное мобильное приложение для управления задачами и визуализации прогресса - Taskoria».

Документ содержит разделы: «Введение», «Назначение разработки», «Технические характеристики», «Описание пользовательского интерфейса», «Тестирование», «Технико-экономические показатели» и «Заключение».

В разделе «Введение» указано наименование программы и основание для разработки.

В разделе «Назначение разработки» описаны функциональное и эксплуатационное назначение программы, целевая аудитория и сравнение с аналогами.

В разделе «Технические характеристики» приведены постановка задачи, состав программного продукта, выбранные технические средства и архитектурные решения.

В разделе «Описание пользовательского интерфейса» описаны основные экраны мобильного приложения и пользовательские сценарии.

В разделе «Тестирование» приведены сведения о проверках серверной и клиентской части.

В разделе «Технико-экономические показатели» приведены предполагаемая потребность, преимущества по сравнению с аналогами и ограничения текущей версии.

Настоящий документ разработан с учетом требований:

1. ГОСТ 19.101-77 [1]: Виды программ и программных документов.
2. ГОСТ 19.102-77 [2]: Стадии разработки.
3. ГОСТ 19.103-77 [3]: Обозначения программ и программных документов.
4. ГОСТ 19.104-78 [4]: Основные надписи.
5. ГОСТ 19.105-78 [5]: Общие требования к программным документам.
6. ГОСТ 19.106-78 [6]: Требования к программным документам, выполненным печатным способом.
7. ГОСТ 19.404-79 [10]: Пояснительная записка. Требования к содержанию и оформлению.

Изменения к данной пояснительной записке оформляются согласно ГОСТ 19.603-78 [12], ГОСТ 19.604-78 [13].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

СОДЕРЖАНИЕ

АННОТАЦИЯ.....	2
СОДЕРЖАНИЕ.....	3
1. ВВЕДЕНИЕ	5
1.1. Наименование программы.....	5
1.2. Основание для разработки.....	5
1.3. Актуальность темы	5
1.4. Краткая характеристика области применения	5
2. НАЗНАЧЕНИЕ РАЗРАБОТКИ	6
2.1. Функциональное назначение.....	6
2.2. Эксплуатационное назначение	6
2.3. Целевая аудитория.....	7
3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ.....	8
3.1. Постановка задачи на разработку программы	8
3.2. Описание и обоснование выбора технических и программных средств.....	9
3.3. Архитектура программного продукта.....	10
3.3.1. Архитектура серверной части.....	11
3.3.2. Архитектура клиентской части.....	13
3.3.3. Паттерны проектирования.....	14
3.4. Организация данных.....	14
3.5. Алгоритм функционирования программы.....	15
3.6. Изометрический город.....	16
4. ОПИСАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА	18
4.1. Авторизация и регистрация.....	18
4.2. Главный экран города	18
4.3. Задачи.....	19
4.4. Календарь, магазин и достижения	21
4.5. Профиль, уведомления и настройки	21
5. ТЕСТИРОВАНИЕ	23
6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ.....	24
6.1. Предполагаемая потребность	24
6.2. Преимущества перед аналогами	24

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

RU.17701729.06.05-01 81 01-1

6.3. Ограничения текущей версии	25
7. ЗАКЛЮЧЕНИЕ	26
8. СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ.....	27
ПРИЛОЖЕНИЕ. ССЫЛКИ НА АНАЛОГИ.....	28

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

1. ВВЕДЕНИЕ

1.1. Наименование программы

Наименование программы: «Геймифицированное мобильное приложение для управления задачами и визуализации прогресса - Taskoria».

Наименование программы на английском языке - «Gamified mobile app for task management and progress visualization - Taskoria».

1.2. Основание для разработки

Разработка ведется на основании учебного плана подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденной темы курсового проекта.

1.3. Актуальность темы

Пользователи мобильных приложений для планирования часто сталкиваются с проблемой снижения мотивации после первоначального периода использования. Классический список задач помогает фиксировать дела, но не всегда дает ощущение накопленного результата. Из-за этого пользователь может перестать регулярно возвращаться к приложению, даже если сама задача планирования остается актуальной.

Геймификация позволяет усилить вовлеченность за счет понятной обратной связи: очков опыта, уровней, наград, достижений и визуального прогресса. В Taskoria выполнение задач связано не только с изменением статуса задачи, но и с развитием изометрического города. Такой подход делает результат действий пользователя более наглядным.

1.4. Краткая характеристика области применения

Программа относится к области мобильных инструментов персональной продуктивности, планирования и самоорганизации. Основная область применения - индивидуальное управление учебными, рабочими, личными и связанными со здоровьем задачами.

Приложение может использоваться студентами, начинающими специалистами и другими пользователями, которым требуется простой инструмент для фиксации задач, отслеживания выполнения, получения краткой статистики и поддержания регулярности работы с делами.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2. НАЗНАЧЕНИЕ РАЗРАБОТКИ

2.1. Функциональное назначение

Мобильное приложение Taskoria предназначено для индивидуального управления повседневными задачами с использованием игровых механик и визуализации прогресса. Пользователь создает задачи, указывает название, описание, категорию, приоритет, сложность и дедлайн, после чего может просматривать задачи в списке, применять поиск и фильтры, открывать детальную карточку задачи, работать с подзадачами и завершать выполненные задачи.

Для снижения сложности планирования в приложении предусмотрено разбиение задачи на подзадачи с использованием ИИ-подсистемы. Данная функция помогает пользователю получить более понятную структуру работы над крупной задачей. При создании и завершении задач серверная часть рассчитывает награды, обновляет состояние пользователя и возвращает актуальные данные клиентскому приложению.

Ключевой особенностью приложения является геймифицированная визуализация прогресса. За выполнение задач пользователь получает XP и Coins. XP влияет на уровень профиля, а Coins используются для покупки декоративных объектов в магазине. Выполнение задач разных категорий связано с развитием зданий изометрического города: учебные задачи развивают школу, рабочие задачи - офис, задачи здоровья - госпиталь, личные задачи - личное здание. Ратуша используется для отображения краткой сводки профиля и уровней зданий.

Дополнительно приложение включает календарь активности, экран достижений, профиль пользователя, внутренний экран уведомлений и настройки. Календарь позволяет просматривать активность по датам, достижения фиксируют важные результаты пользователя, а профиль отображает уровень, XP, Coins, streak, количество выполненных задач и количество открытых достижений.

2.2. Эксплуатационное назначение

Программа предназначена для индивидуального использования на Android-устройстве. Клиентское приложение обеспечивает пользовательский интерфейс, а серверная часть отвечает за хранение данных, авторизацию, обработку задач, начисление наград, обновление города, работу магазина и предоставление REST API.

Текущая версия проекта рассчитана на локальное развертывание серверной части и базы данных. Android-приложение может запускаться на физическом устройстве или эмуляторе и подключаться к локальному backend.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2.3. Целевая аудитория

Целевая аудитория приложения включает студентов, школьников, молодых специалистов, пользователей, испытывающих трудности с мотивацией к выполнению повседневных задач, а также пользователей, интересующихся продуктивностью и саморазвитием. Общими характеристиками аудитории являются активное использование смартфонов, знакомство с базовыми функциями мобильных приложений и интерес к игровым механикам.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

3.1. Постановка задачи на разработку программы

Целью разработки является создание программного продукта, который объединяет функции планировщика задач, игровую систему мотивации и визуализацию прогресса пользователя в виде изометрического города. Постановка задачи включает разработку мобильного Android-приложения и серверной части, обеспечивающей хранение данных, обработку пользовательских действий и расчет игрового прогресса.

Входными данными программы являются сведения об аккаунте пользователя, данные создаваемых задач и подзадач, выбранные категории, приоритеты, сложность, дедлайны, а также действия пользователя в разделах города, магазина, достижений и профиля. Ожидаемым результатом работы программы является сохранение задач, начисление XP и Coins за продуктивность, обновление уровней зданий, отображение достижений и предоставление пользователю наглядной картины личного прогресса.

В состав разрабатываемого функционала входят:

1. Управление аккаунтом пользователя:

- регистрация пользователя;
- вход в приложение;
- выход из аккаунта с очисткой локального состояния сессии.

2. Управление задачами:

- создание задач с указанием названия, описания, категории, приоритета, сложности и дедлайна;
- просмотр списка активных и выполненных задач;
- поиск и фильтрация задач;
- просмотр детальной карточки задачи;
- редактирование, удаление и завершение задачи.

3. Работа с подзадачами:

- получение списка подзадач для задачи;
- создание подзадач вручную;
- формирование подзадач через ИИ-подсистему;
- изменение статуса подзадачи.

4. Геймифицированная система прогресса:

- начисление XP и Coins за выполнение задач;
- расчет уровня пользователя;
- учет streak;

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

- открытие достижений;
- отображение краткой статистики пользователя.

5. Изометрический город:

- отображение карты города;
- развитие зданий в зависимости от прогресса по категориям задач;
- отображение карточек зданий и ратуши;
- точное позиционирование зданий и декоративных объектов на карте.

6. Магазин и декоративные объекты:

- отображение доступных товаров магазина;
- покупка объектов за Coins;
- размещение купленных объектов на закрепленных позициях карты.

7. Дополнительные разделы приложения:

- календарь активности;
- экран достижений;
- профиль пользователя;
- внутренние уведомления;
- настройки.

3.2. Описание и обоснование выбора технических и программных средств

Серверная часть приложения разработана на языке Python 3.11 с использованием фреймворка FastAPI. Python выбран благодаря развитой экосистеме библиотек для веб-разработки, работы с базами данных и интеграции с внешними сервисами. FastAPI обеспечивает удобную разработку REST API, поддержку асинхронной обработки запросов, встроенную работу с Pydantic-схемами и автоматическую генерацию Swagger/OpenAPI-документации.

Для хранения данных используется PostgreSQL. Данная СУБД подходит для структурированных данных проекта: пользователей, задач, подзадач, зданий, достижений, статистики и покупок. Работа с базой данных организована через SQLAlchemy, что позволяет описывать таблицы как модели и снижает риск ошибок при формировании SQL-запросов. Для управления изменениями структуры базы данных используется Alembic.

Flutter выбран для реализации клиентской части, поскольку интерфейс Taskoria содержит большое количество кастомных визуальных элементов: изометрическую карту, pixel-art здания, карточки задач, игровые панели прогресса и адаптивные экраны для разных размеров Android-устройств. Использование Flutter позволяет строить такие интерфейсы на единой кодовой базе и точно контролировать расположение элементов, что особенно важно для карты города, где здания

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

должны визуально совпадать с координатами ячеек.

Riverpod выбран как механизм управления состоянием, так как приложение работает с несколькими связанными наборами данных: сессией пользователя, списком задач, состоянием города, достижениями, магазином и профилем. В отличие от простого локального состояния виджетов, Riverpod позволяет централизованно обновлять данные после действий пользователя. Например, после завершения задачи должны измениться список задач, баланс Coins, XP, уровень пользователя, достижения и состояние города. Такое обновление удобнее реализовывать через провайдеры, а не через передачу состояния между экранами вручную.

Dio используется для HTTP-взаимодействия с backend, поскольку клиентское приложение должно выполнять авторизованные запросы к REST API, передавать JWT-токен и обрабатывать ответы сервера в режиме core_api. Наличие отдельного сетевого клиента упростило поддержку двух режимов работы приложения: mock-режима для демонстрации интерфейса без сервера и core_api-режима для работы с реальной серверной частью.

Для локального развертывания предусмотрена Docker Compose-конфигурация, позволяющая запускать backend и PostgreSQL в контейнерах.

3.3. Архитектура программного продукта

Система построена по клиент-серверной архитектуре. Пользователь взаимодействует с мобильным Android-приложением. Клиентская часть отвечает за отображение экранов, навигацию, локальное состояние и отправку запросов. Серверная часть принимает HTTP-запросы, выполняет бизнес-операции, обращается к базе данных и возвращает ответы в формате JSON. База данных PostgreSQL хранит основное состояние приложения.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

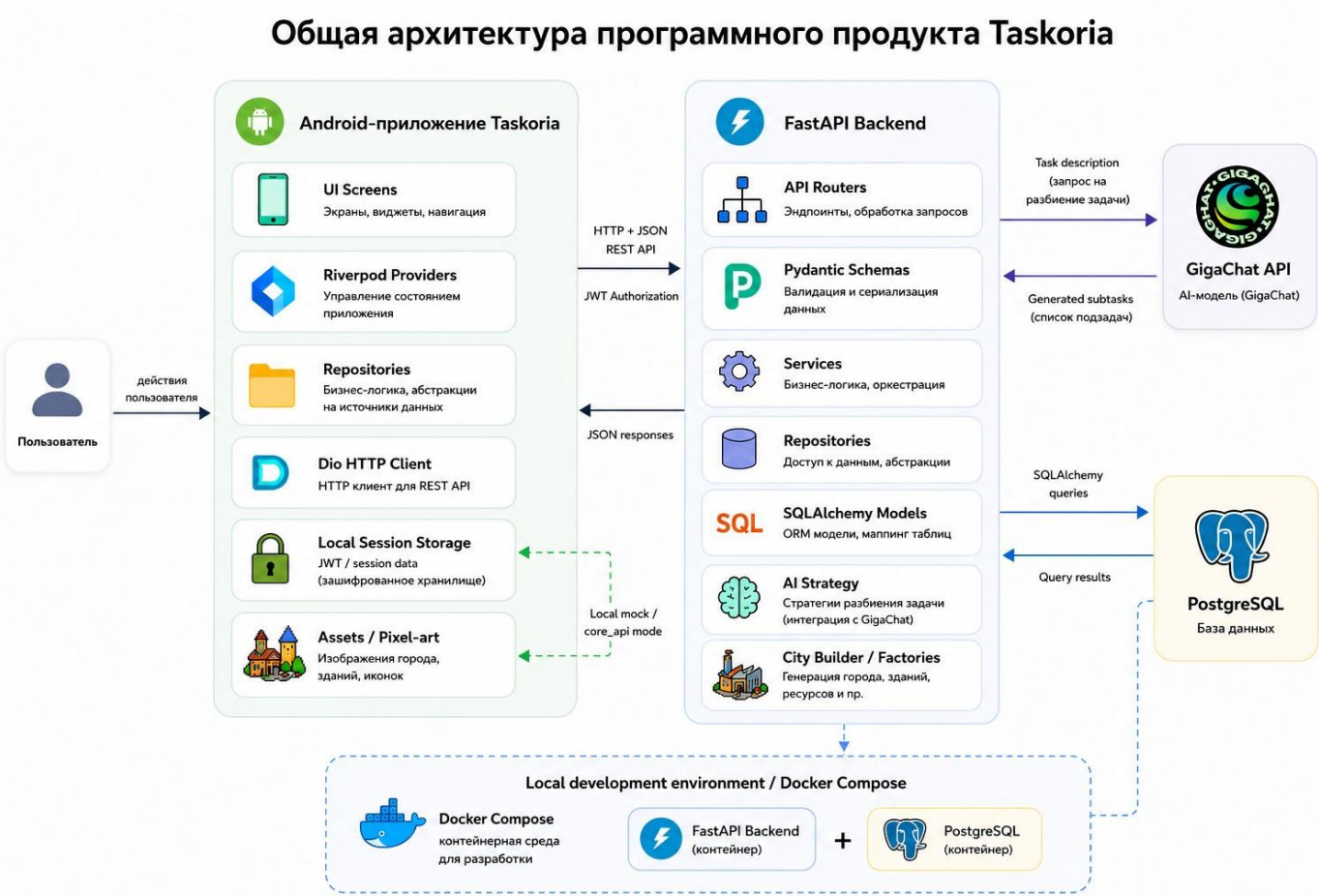


Рис. 1. Общая архитектура программного продукта

На верхнем уровне проект состоит из двух основных каталогов: backend и productivity_city. В каталоге backend расположена серверная часть на FastAPI, включая API-роутеры, модели базы данных, схемы, репозитории, сервисы, миграции и тесты. В каталоге productivity_city расположено Flutter-приложение, включая экраны, тему, маршрутизацию, модели, провайдеры, репозитории, сетевой клиент и графические ресурсы.

Обмен данными между клиентом и сервером выполняется через REST API. В режиме core_api мобильное приложение отправляет запросы к backend и получает реальные данные пользователя. В mock-режиме используются демонстрационные данные без обращения к серверу, что упрощает проверку интерфейса.

3.3.1. Архитектура серверной части

Серверная часть реализована по многоуровневой архитектуре. Уровень API содержит роутеры, принимающие запросы авторизации, задач, подзадач, города, магазина, достижений и пользователя.

Уровень схем описывает входные и выходные данные, а также используется для валидации запросов.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Уровень сервисов содержит основные бизнес-правила: создание и завершение задач, расчет наград, обновление прогресса пользователя, синхронизацию города, покупку товаров и работу с ИИ-подсистемой.

Уровень репозитория отвечает за доступ к данным и изолирует бизнес-логику от деталей работы с базой данных. Модели базы данных описывают таблицы пользователей, задач, подзадач, зданий, достижений, статистики и покупок. Миграции Alembic используются для фиксации изменений структуры базы данных. Автоматические тесты серверной части расположены в каталоге backend/tests и проверяют ключевые сценарии: завершение задач, начисление наград, работу города и магазина.

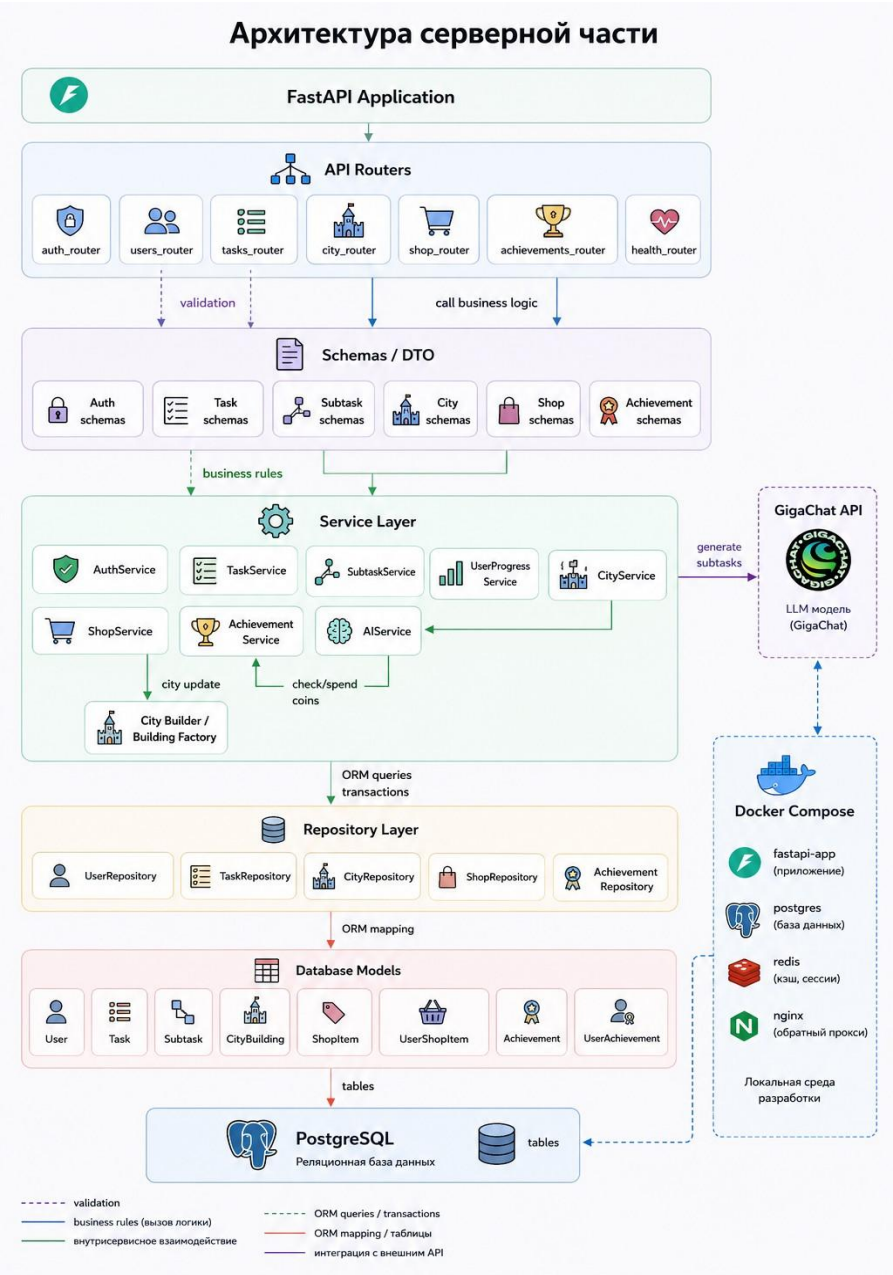


Рис. 2. Архитектура серверной части

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.3.2. Архитектура клиентской части

Клиентская часть Flutter-приложения разделена на уровни приложения, функциональных модулей и общих компонентов. Уровень приложения содержит тему оформления, маршрутизацию и точку сборки приложения. Функциональные модули включают экраны авторизации, onboarding, города, задач, календаря, магазина, достижений, профиля, уведомлений и настроек. Общий слой содержит модели данных, провайдеры, репозитории, сетевой клиент, управление сессией, пути к ресурсам и переиспользуемые виджеты.

Для работы с данными используется Repository-подход. Экраны не обращаются к API напрямую: они получают данные через провайдеры состояния, а провайдеры используют репозитории. Репозитории имеют две реализации: mock-реализацию для демонстрационных данных и API-реализацию для подключения к backend.

Отдельным модулем реализована изометрическая карта города. В нем описаны характеристики зданий, декоративных объектов, дорог и точек привязки изображений к ячейкам карты. Это позволяет вынести настройки визуального размещения объектов из основной логики экрана и поддерживать карту в согласованном виде.

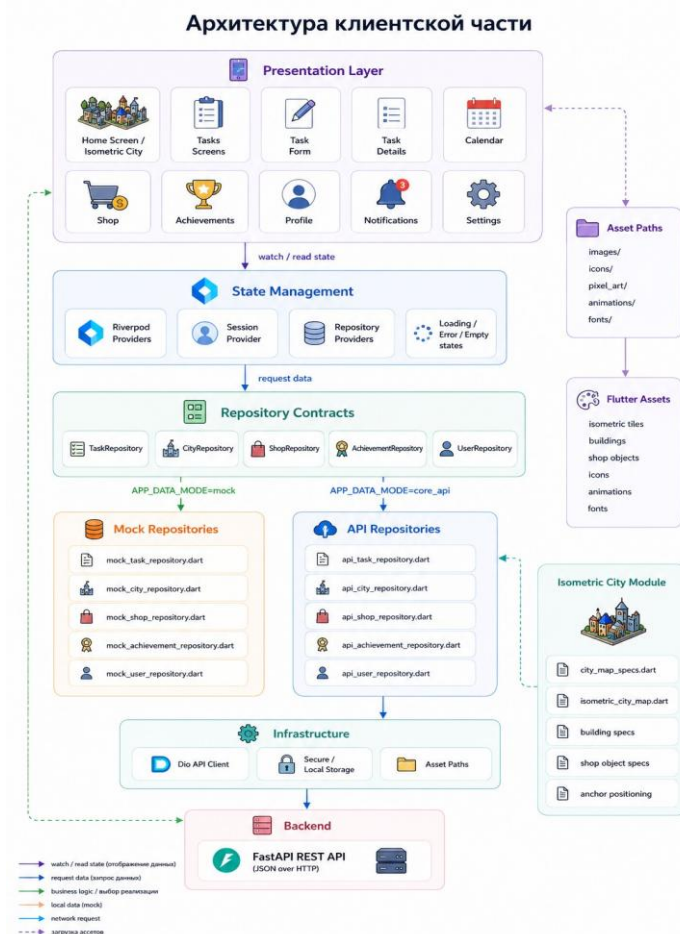


Рис. 3. Архитектура клиентской части

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.3.3. Паттерны проектирования

В проекте применены архитектурные подходы и паттерны, соответствующие клиент-серверной структуре приложения. На серверной части используются Repository для отделения доступа к данным, Service Layer для выделения бизнес-логики, Dependency Injection для передачи зависимостей, Factory и Builder для работы с городом, Strategy для ИИ-подсистемы. На клиентской части применяются Repository-подход, провайдеры состояния и единая обработка состояний загрузки, ошибки и пустого результата.

3.4. Организация данных

Основными сущностями системы являются пользователь, задача, подзадача, здание, достижение, связь пользователя с достижением, статистика и состояние покупок пользователя. Основные транзакционные таблицы спроектированы с учетом требований третьей нормальной формы. Данные пользователя, задач, подзадач, зданий и достижений вынесены в отдельные отношения, а связь многие-ко-многим между пользователями и достижениями реализована через ассоциативную таблицу.

Таблица статистики используется как агрегированная модель для быстрого отображения календаря активности и краткой статистики пользователя. Каталог товаров магазина в текущей версии задан в коде приложения, а в базе данных хранится состояние покупок пользователя.

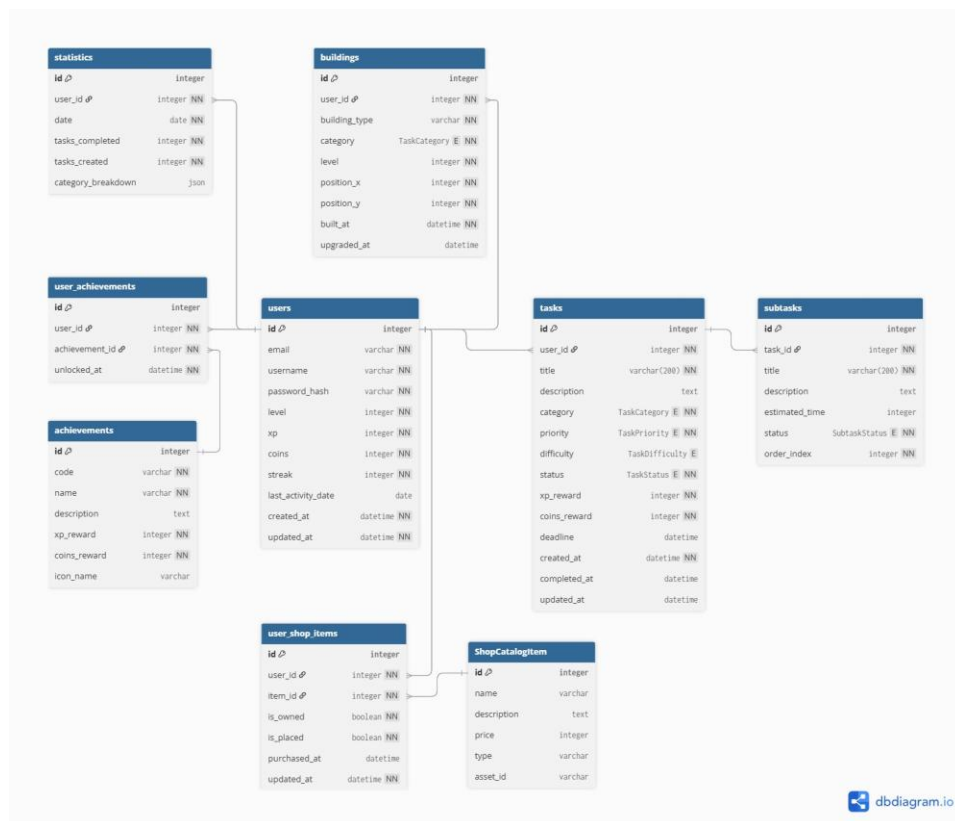


Рис. 4. ER-диаграмма базы данных

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.5. Алгоритм функционирования программы

При первом запуске приложения отображается экран загрузки, после чего пользователь переходит к экрану входа или регистрации. Если пользователь уже авторизован, приложение открывает основной экран города и загружает связанные данные: профиль, задачи, достижения, состояние города и магазин. В mock-режиме данные берутся из локальных mock-репозитория, а в режиме core_api клиент выполняет HTTP-запросы к backend.

Основной сценарий работы начинается с создания задачи. Пользователь вводит название, описание, категорию, приоритет, сложность и дедлайн. После отправки формы клиент проверяет обязательные поля и передает данные на сервер. Сервер сохраняет задачу, при необходимости формирует подзадачи через ИИ-подсистему, рассчитывает награды XP и Coins и возвращает результат клиентскому приложению. После получения ответа список задач обновляется, а пользователь может открыть карточку задачи для просмотра деталей.

При завершении задачи клиент отправляет запрос на backend. Сервер проверяет, что задача существует, принадлежит текущему пользователю и ранее не была завершена. После этого выполняется расчет наград, обновляются XP, Coins, уровень пользователя, streak, статистика и достижения. Также обновляется состояние города: здания, связанные с категориями задач, могут перейти на новый уровень. В ответ backend возвращает обновленные данные, а frontend перерисовывает список задач, профиль, достижения и карту города.

Отдельная ветка алгоритма связана с магазином. Пользователь выбирает товар, после чего backend проверяет баланс Coins. Если средств недостаточно, покупка отклоняется. Если средств достаточно, Coins списываются, покупка сохраняется, а объект становится доступным для размещения на карте. После размещения декоративный объект отображается в изометрическом городе.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Основной пользовательский путь в приложении Taskoria

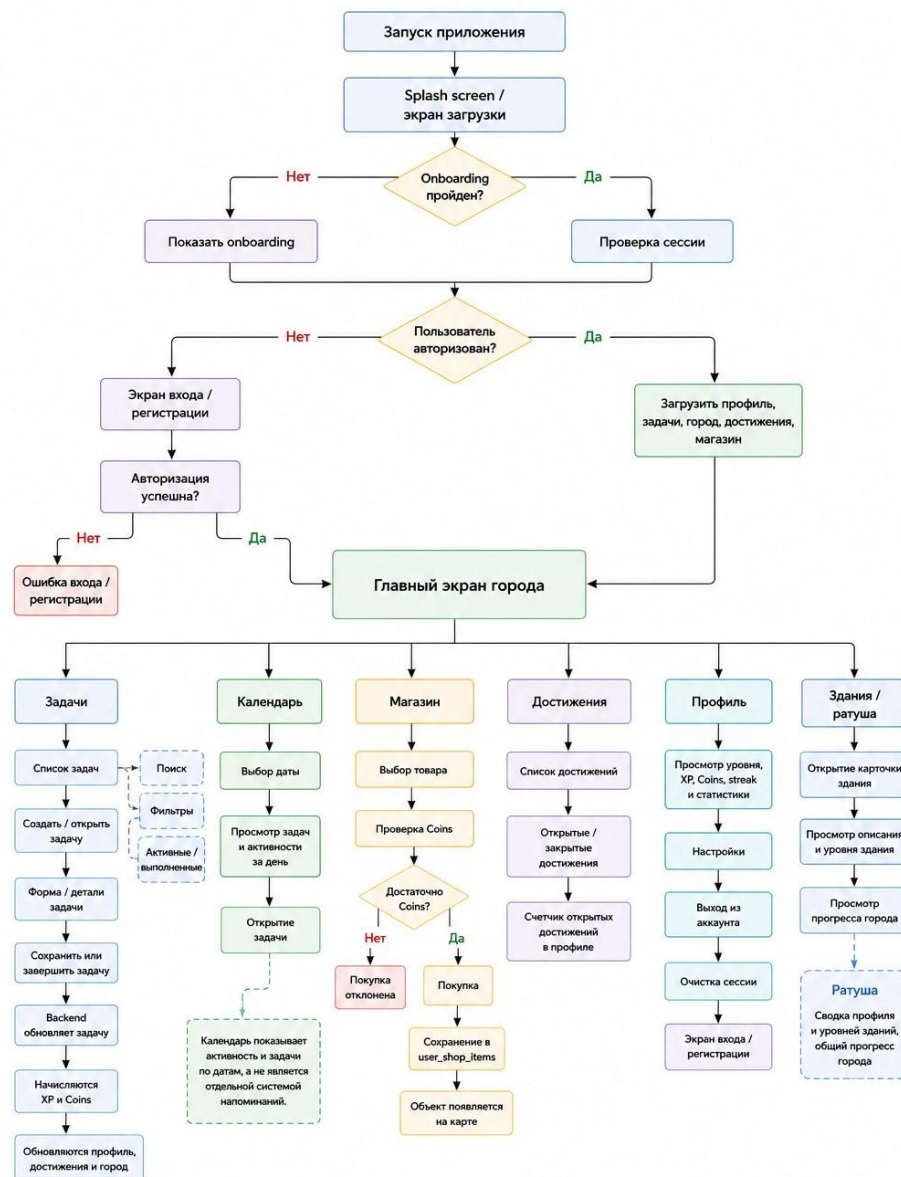


Рис. 5. Основной пользовательский путь

3.6. Изометрический город

Изометрический город реализован как отдельный визуальный слой клиентского приложения. Карта строится на основе сетки ячеек, где каждая ячейка имеет координаты X и Y. Дороги задаются явно через набор занятых ячеек, а здания и декоративные объекты размещаются по спецификациям. Такой подход позволяет не подбирать координаты вручную на экране, а связывать визуальный объект с логической позицией на карте.

Для каждого здания задается footprint - прямоугольная область ячеек, которую занимает объект на карте. Footprint используется для описания места здания в логической сетке и для запрета

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

размещения других объектов поверх занятых областей. Например, здание может занимать диапазон ячеек по X и Y, соответствующий его основанию на карте. Это позволяет учитывать не только визуальную картинку здания, но и область, которую оно занимает в городе.

Точная привязка изображения к карте выполняется через anchor-точку. Для каждого изображения была измерена точка внутри PNG-файла, соответствующая нижнему углу нужной ячейки. В спецификации здания указываются координаты anchorPxX и anchorPxY внутри изображения, а также cellX и cellY на карте. При отрисовке приложение совмещает anchor-точку изображения с экранной позицией указанной ячейки. Благодаря этому здания разных уровней, имеющие разный размер и форму, могут оставаться привязанными к одной логической области города.

Порядок наложения объектов определяется с учетом их положения на изометрической сетке. Объекты, расположенные визуальнее дальше, должны рисоваться раньше, а объекты ближе к пользователю - позже. Для этого используется сортировка по координатам ячеек, в первую очередь по сумме X и Y, а при необходимости - по дополнительному приоритету слоя. Такой подход позволяет избежать ситуаций, когда дальнее здание перекрывает ближнее. Для отдельных зданий также учитывается ручной приоритет слоя, если их визуальные области пересекаются.

Развитие города связано с категориями задач. Учебные задачи влияют на школу, рабочие - на офис, задачи здоровья - на госпиталь, личные - на личное здание. При росте уровня категории изображение здания заменяется на следующий уровень, но логическая привязка остается управляемой через anchor-точку. Таким образом, город является визуальным отображением прогресса пользователя.

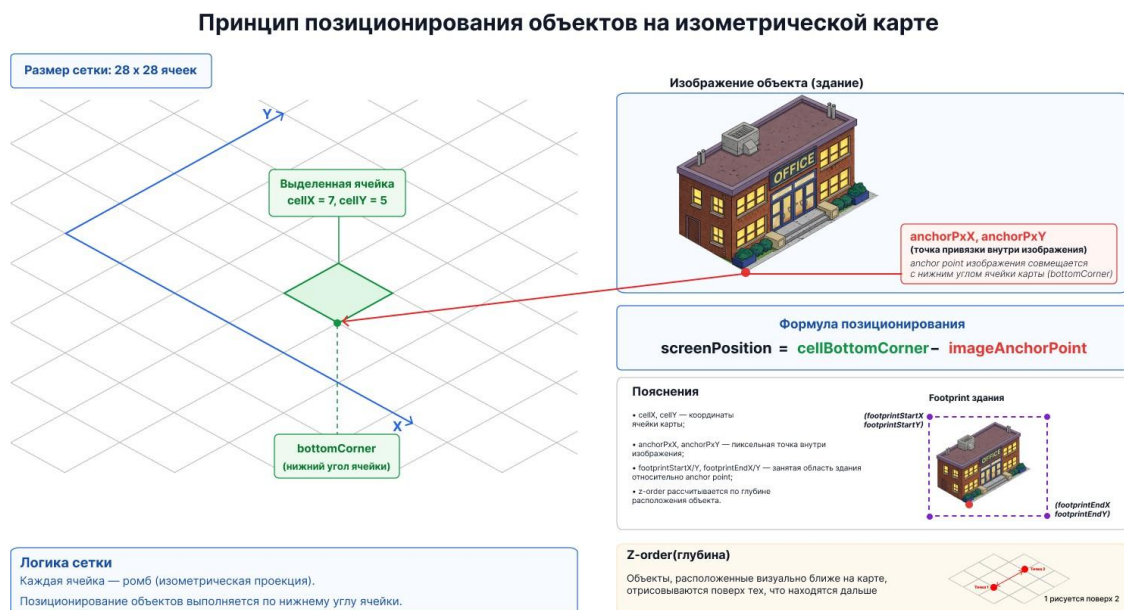


Рис. 6. Принцип позиционирования объектов на изометрической карте

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

4. ОПИСАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Пользовательский интерфейс выполнен в игровом стиле с использованием pixel-art ресурсов. Основная навигация расположена в нижней панели и включает разделы города, задач, календаря, магазина и профиля.

4.1. Авторизация и регистрация

Рис. 7. Экран входа

Рис. 8. Экран регистрации

4.2. Главный экран города

Главный экран отображает изометрическую карту города, баланс Coins, уровень пользователя, панель навигации, здания и декоративные объекты. При нажатии на здание открывается карточка с описанием и уровнем здания, а ратуша показывает краткую сводку профиля и прогресса.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



Рис. 9. Главный экран города



Рис. 10. Карточка здания

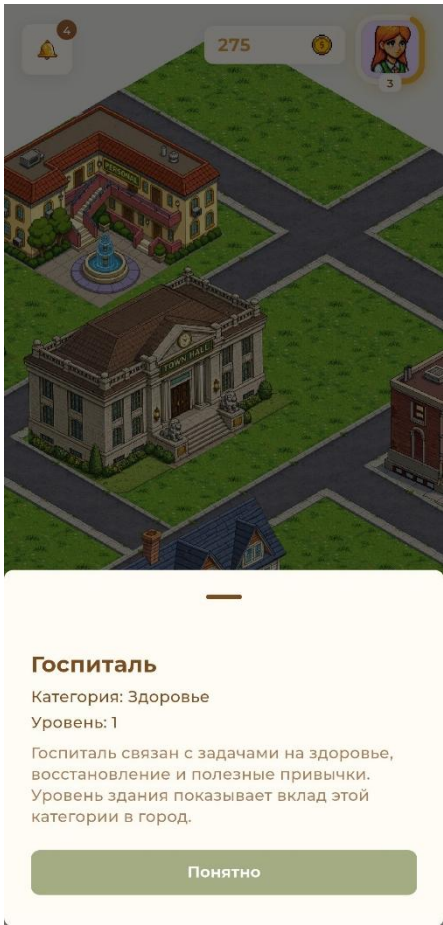


Рис.11. Карточка ратуши

4.3. Задачи

Экран задач содержит поиск, фильтры, переключение активных и выполненных задач, карточки задач и переход к детальному просмотру. Экран создания задачи позволяет указать название, описание, приоритет, категорию, сложность и дедлайн.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

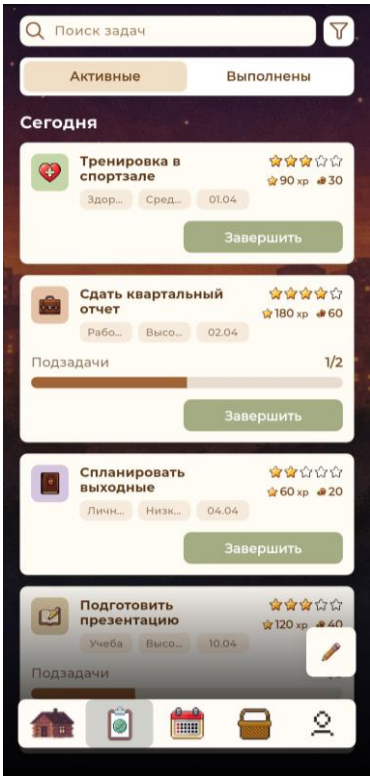


Рис. 12. Экран списка задач

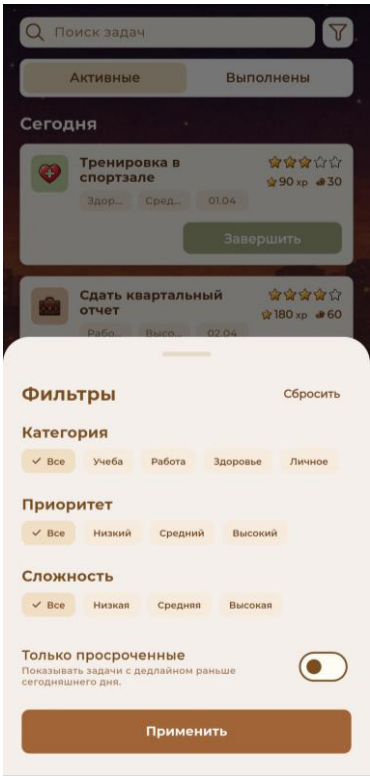


Рис. 13. Фильтрация задач



Рис. 14. Экран создания задачи



Рис. 15. Экран детального просмотра задачи

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

4.4. Календарь, магазин и достижения

Календарь отображает активность пользователя по датам. Магазин позволяет покупать декоративные объекты за Coins. Экран достижений показывает открытые и закрытые достижения, награды и прогресс пользователя.

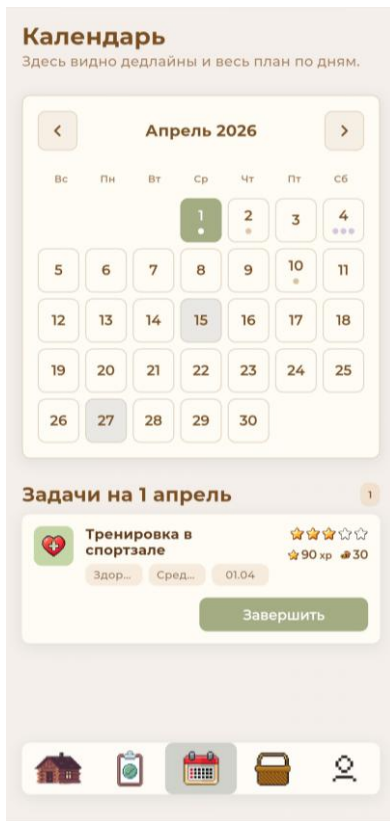


Рис. 16. Экран календаря

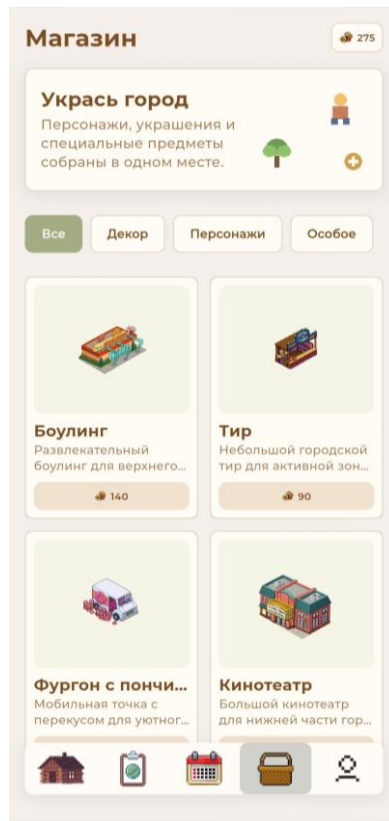


Рис. 17. Экран магазина



Рис. 18. Экран достижений

4.5. Профиль, уведомления и настройки

Профиль отображает уровень, XP, Coins, streak, краткую статистику, активность и количество открытых достижений. Экран уведомлений используется для просмотра внутренних событий приложения. В настройках пользователь может выполнить выход из аккаунта.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



Рис. 19. Экран профиля



Рис. 20. Дополнительная информация профиля



Рис. 21. Экран уведомлений



Рис. 22. Экран настроек

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

5. ТЕСТИРОВАНИЕ

В ходе проверки программы выполнялись функциональные, интеграционные и пользовательские испытания. Проверялись основные сценарии работы приложения: регистрация и вход пользователя, создание и завершение задач, начисление наград, обновление города, покупка товаров магазина, отображение достижений и работа клиентского приложения в mock- и core_api-режимах.

Дополнительно выполнялась проверка backend-тестов через Pytest, статический анализ Flutter-кода и запуск клиентского приложения на Android-устройстве. При ручном тестировании отдельно проверялись экраны города, задач, календаря, магазина, достижений, профиля, уведомлений и настроек.

№	Проверяемое условие	Действия	Ожидаемый результат	Фактический результат
1	Регистрация пользователя	Ввести имя, email и пароль, отправить форму регистрации	Пользователь создается, выполняется переход в приложение	Соответствует ожидаемому результату
2	Создание задачи	Заполнить форму задачи и нажать кнопку создания	Задача сохраняется и появляется в списке задач	Соответствует ожидаемому результату
3	Завершение задачи	Открыть задачу и выполнить действие завершения	Начисляются XP и Coins, задача получает статус выполненной	Соответствует ожидаемому результату
4	Защита от повторного завершения	Повторно отправить завершение уже выполненной задачи	Повторное начисление наград не выполняется	Соответствует ожидаемому результату
5	Обновление города	Выполнить задачи определенной категории	Уровень соответствующего здания обновляется	Соответствует ожидаемому результату
6	Покупка товара магазина	Купить объект при достаточном количестве Coins	Coins списываются, объект становится купленным и отображается на карте	Соответствует ожидаемому результату
7	Покупка при недостатке Coins	Попытаться купить объект без достаточного баланса	Покупка отклоняется, баланс не изменяется	Соответствует ожидаемому результату
8	Отображение достижений	Открыть экран достижений после выполнения условий	Полученные достижения отмечены как открытые, счетчик синхронизирован с профилем	Соответствует ожидаемому результату
9	Работа mock-режима	Запустить приложение без подключения к backend	Основные экраны открываются на демонстрационных данных	Соответствует ожидаемому результату
10	Работа core_api-режима	Запустить backend и приложение с API_BASE_URL	Клиент получает данные с backend и выполняет операции через API	Соответствует ожидаемому результату

Таблица 1. Основные тестовые сценарии

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

6.1. Предполагаемая потребность

Потребность в разработке приложения обусловлена отсутствием на рынке гибридного продукта, сочетающего функции планировщика задач, элементы геймификации, визуальное развитие виртуального города и ИИ-помощь при разбиении задач на подзадачи.

Существующие аналоги либо делают акцент на классическом списке задач, либо используют игровые механики без развитой визуализации города и без автоматизации планирования. Taskoria ориентирована на пользователей, которым важно видеть не только список дел, но и наглядный результат своей активности.

6.2. Преимущества перед аналогами

Основные преимущества Taskoria связаны с сочетанием практических функций планировщика и игровых механик. Приложение поддерживает задачи, подзадачи, категории, календарь активности, достижения, игровую валюту, уровни пользователя и развитие изометрического города. Выполнение задач в разных категориях связано с развитием соответствующих зданий, а магазин позволяет дополнительно изменять внешний вид города.

В таблице приведено сравнение функциональных характеристик Taskoria и аналогичных приложений. Знак «+» означает наличие функции, знак «-» - отсутствие функции, знак «±» - частичную или ограниченную реализацию.

Функция	Taskoria	Habitica	Forest	Avocation	Habit Hunter	Do It Now	LifeUp	Todoist Karma
ИИ-разбиение задач на подзадачи	+	-	-	-	-	-	-	-
Виртуальный город с развитием	+	-	-	-	-	-	-	-
Система уровней и XP	+	+	-	-	+	+	+	+
Игровая валюта	+	+	-	-	+	+	+	-

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Streak-система	+	+	+	-	+	+	±	+
Достижения	+	+	-	-	+	+	+	-
Статистика продуктивности	+	+	±	±	±	+	+	+
Категории задач	+	+	-	+	+	+	+	+
Подзадачи	+	+	-	-	+	+	-	+
Календарь/планирование	+	±	-	+	+	+	+	+
Напоминания	+	+	+	+	+	+	+	+
Визуализация прогресса	+	+	+	+	+	+	+	+
Социальные функции	+	+	-	-	-	-	+	-
Адаптивная система наград	+	±	-	-	±	±	±	±
Итого	14	10	3	4	9	10	9	8

Таблица 2. Сравнение функциональных характеристик

Из сравнения видно, что Taskoria объединяет функции планировщика, геймификации, визуализации прогресса и ИИ-помощи. При этом приложение сохраняет индивидуальный фокус: пользователь развивает собственный город без необходимости участвовать в социальных активностях.

6.3. Ограничения текущей версии

Текущая версия проекта рассчитана на локальное развертывание backend и PostgreSQL. Публичное серверное развертывание, публикация приложения в магазине приложений, полноценный офлайн-режим с автоматической синхронизацией и социальные функции не входят в состав реализованной версии. Эти ограничения не препятствуют демонстрации основных функций приложения, но определяют направления дальнейшего развития.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

7. ЗАКЛЮЧЕНИЕ

В ходе работы разработан программный продукт Taskoria, включающий мобильное Android-приложение и серверную часть. Реализованы основные функции управления задачами, игровые награды, прогресс профиля, изометрический город, магазин декоративных объектов, достижения, календарь активности и интеграция с backend.

Архитектура проекта построена с разделением ответственности между клиентской частью, серверной частью и базой данных. Такое разделение облегчает сопровождение, тестирование и дальнейшее развитие программы.

В дальнейшем проект может быть расширен за счет публичного развертывания серверной части, подготовки APK-файла, добавления новых объектов города, развития системы достижений, расширения аналитики и улучшения ИИ-подсистемы.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

8. СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77: Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77: Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77: Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78: Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78: Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78: Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78: Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.301-79: Программа и методика испытаний. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.401-78: Текст программы. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.404-79: Пояснительная записка. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 19.505-79: Руководство оператора. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. ГОСТ 19.603-78: Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
13. ГОСТ 19.604-78: Правила внесения изменений в программные документы, выполненные печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

ПРИЛОЖЕНИЕ. ССЫЛКИ НА АНАЛОГИ

Приложение	Ссылка
Habitica	https://habitica.com/
Forest	https://www.forestapp.cc/
Avocation	https://avocation.app/
Habit Hunter	https://habithunter.net/
Do It Now	https://do-it-now.app/
LifeUp	https://www.lifeupapp.fun/en/index.html
Todoist Karma	https://www.todoist.com/ru/karma

Дата обращения: 01.05.26.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.05-01 81 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

[illegible]