**Государственное бюджетное профессиональное образовательное учреждение**

**«Нижегородский Губернский колледж»**

**«ТЕХНИЧЕСКОЕ ЗАДАНИЕ»**

МДК 01.03 РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

Выполнил:

Студент группы 41П

Кукина Анастасия Игоревна

Нижний Новгород

2025 г.

**Содержание**

[1. Общие сведения 3](#_Toc209035519)

[2. Назначение и цели создания системы 3](#_Toc209035520)

[2.1. Назначение системы 3](#_Toc209035521)

[2.2. Цели создания системы 4](#_Toc209035522)

[3. Характеристика объектов автоматизации 6](#_Toc209035523)

[4. Требования к системе 7](#_Toc209035524)

[4.1. Требования к системе в целом 7](#_Toc209035525)

[4.2. Требования к функциям, выполняемым системой 7](#_Toc209035526)

[4.3. Требования к видам обеспечения 10](#_Toc209035527)

[5. Состав и содержание работ по созданию системы 19](#_Toc209035528)

[6. Порядок контроля и приёмки системы 20](#_Toc209035529)

[7. Требования к документированию 21](#_Toc209035530)

[8. Источники разработки 22](#_Toc209035531)

# 1. Общие сведения

Наименование системы PlantBuddy мобильное приложение такси.

**Плановые сроки начала и окончания работы:**

Сроки уточняются

**Порядок оформления и предъявления заказчику результатов работ**

Работы по созданию системы сдаются Разработчиком поэтапно в соответствии с календарным планом Проекта. По окончании каждого из этапов работ Разработчик сдает Заказчику соответствующие отчетные документы этапа, состав которых определены Договором.

# 2. Назначение и цели создания системы

## 2.1. Назначение системы

Система PlantBuddy представляет собой мобильное приложение, предназначенное для предоставления консультаций и помощи в уходе за комнатными растениями.

Система создает базу знаний о потребностях различных видов растений, автоматизирует процесс получения информации об уходе и предоставляет возможности для мониторинга состояния растений.

**Объекты автоматизации:**

Основные объекты автоматизации: пользователи, комнатные растения, информация об уходе.

Для пользователей:

Автоматизация процессов получения информации о потребностях растений, организации ухода и ведения дневника наблюдений.

1. Поиск информации о растении (указание названия растения или идентификация по фотографии).
2. Отображение информации о потребностях растения (полив, освещение, температура, удобрения, влажность воздуха).
3. Настройка уведомлений о необходимости ухода (полив, подкормка, пересадка).
4. Ведение дневника наблюдений за ростом и развитием растения (добавление заметок и фотографий).
5. Получение информации о возможных болезнях и вредителях, а также о способах борьбы с ними.

Для системы:

Автоматизация процессов получения, хранения и предоставления информации о потребностях растений.

1. Получение данных о растении из API или локальной базы данных.
2. Анализ введенных пользователем данных и предоставление рекомендаций по уходу.
3. Отправка push-уведомлений пользователям.
4. Хранение данных пользователя и настроек (предпочтения, дневник наблюдений).

## 2.2. Цели создания системы

**Основные цели**

1. Повышение удобства и эффективности ухода за комнатными растениями для пользователей: Предоставление простого и интуитивно понятного интерфейса для получения информации о потребностях растений и организации ухода.
2. Улучшение выживаемости комнатных растений: Предоставление пользователям знаний и инструментов для правильного ухода, что способствует более здоровому росту и увеличению продолжительности жизни растений.
3. Повышение уровня знаний пользователей о растениях: Предоставление доступа к информации о различных видах растений, их потребностях и особенностях.
4. Формирование осознанного отношения к комнатным растениям: Мотивирование пользователей к более внимательному и ответственному уходу за своими растениями.

В результате создания хранилища данных должны быть улучшены значения следующих показателей:

1. Время, затрачиваемое пользователем на поиск информации об уходе за растением.
2. Понимание потребностей растения пользователем (оценка до/после использования приложения).
3. Количество правильно выполненных действий по уходу (полив, подкормка) за определенный период времени.
4. Состояние растений (оценка по фотографиям или субъективным ощущениям пользователя).

**Критерии оценки достижения целей создания системы**

1. Количество установок приложения и активных пользователей: Показатель востребованности сервиса;
2. Среднее время, затрачиваемое пользователем на поиск информации о растении (снижение времени): Показатель удобства и эффективности приложения;
3. Удовлетворенность пользователей (по результатам опросов): Показатель восприятия пользователями качества предоставляемой информации и удобства использования приложения;
4. Субъективная оценка состояния растений пользователями (по фотографиям или опросам): Показатель влияния приложения на здоровье растений;
5. Количество растений, о которых пользователи регулярно получают информацию и ухаживают с помощью приложения: Показатель вовлеченности пользователей.

# 3. Характеристика объектов автоматизации

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| Структурное подразделение | Наименование процесса | Возможность автоматизации | Решение об автоматизации в ходе проекта |
| Пользователь | Поиск информации о растении | Высокая | Будет автоматизирован |
| Пользователь | Получение информации о потребностях растения (полив, освещение, удобрение и т.д.) | Высокая | Будет автоматизирован |
| Пользователь | Настройка напоминаний об уходе | Высокая | Будет автоматизирован |
| Пользователь | (Опционально) Идентификация растения по фотографии | Средняя/Высокая (зависит от реализации) | Будет автоматизирован (если реализовано) |
| Система (API/База данных) | Предоставление информации о растениях | Высокая | Будет автоматизирован |
| Система (Push-уведомления) | Отправка уведомлений пользователям | Высокая | Будет автоматизирован |
| Система (Хранилище данных) | Хранение данных пользователя (предпочтения, дневник наблюдений) | Высокая | Будет автоматизирован |

# 4. Требования к системе

## 4.1. Требования к системе в целом

1. Безопасность:

1. Шифрование персональных данных пользователей.
2. Защита от несанкционированного доступа к данным.

2. Масштабируемость:

1. Возможность расширения базы данных растений.
2. Оптимизация работы с API в случае увеличения числа запросов.

3. Производительность:

1. Время отклика на запросы пользователя не более 2 секунд.
2. Быстрая загрузка информации о растениях.

4. Удобство использования:

1. Интуитивно понятный и простой интерфейс.
2. Адаптивный дизайн (корректное отображение на разных устройствах с Android).
3. Минимальное количество действий для получения нужной информации.

5. Интеграция:

1. API для получения информации о растениях (например, PlantNet API, если используется).
2. API для распознавания изображений (например, TensorFlow Lite, Google Cloud Vision API, если используется).
3. Сервис push-уведомлений (Firebase Cloud Messaging).

6. Соответствие законодательству:

1. Соблюдение законодательства о защите персональных данных (GDPR, если пользователи находятся в ЕС).
2. Соблюдение требований к обработке изображений (если реализована функция распознавания растений по фотографиям).

## 4.2. Требования к функциям, выполняемым системой

Система «PlantBuddy» состоит из следующих подсистем:

**Подсистема «Пользовательское приложение»**

1. Регистрация и авторизация пользователя.
2. Поиск информации о растении по названию.
3. (Опционально) Идентификация растения по фотографии.
4. Отображение информации о потребностях растения (полив, освещение, температура и т.д.).
5. Настройка напоминаний об уходе.
6. Ведение дневника наблюдений (заметки, фотографии).
7. Просмотр истории ухода за растениями.
8. Управление профилем пользователя.
9. Просмотр информации о болезнях и вредителях растений.

**Подсистема «База данных растений (API)»**

1. Предоставление информации о растениях по запросу пользователя.
2. Предоставление информации о потребностях растения.

**Очередь внедрения**

1-я очередь: Подсистема «Пользовательское приложение» (базовый функционал: поиск, отображение информации, напоминания).

2-я очередь: Подсистема «База данных растений (API)» (если используется внешний API).

3-я очередь: Расширенный функционал пользовательского приложения (идентификация по фото, информация о болезнях).

**Временный регламент**

1. Поиск информации о растении: Не более 2 секунд с момента ввода запроса.
2. Отображение информации о растении: Не более 1 секунды после получения данных.
3. Отправка уведомления пользователю: Не более 3 секунд после события (например, наступления времени полива).

**Требования к качеству реализации функций**

1. Функция «Отображение информации о растении»:

1. Достоверность: Информация должна быть актуальной и соответствовать потребностям растения.
2. Полнота: Предоставление информации обо всех основных аспектах ухода (полив, освещение, температура, удобрения).
3. Время выполнения: Не более 1 секунды.

2. Функция «Настройка напоминаний»:

1. Точность: Отправка уведомлений в установленное пользователем время.
2. Надежность: Уведомления должны доставляться пользователю в 99.9% случаев.
3. Гибкость: Возможность настройки различных параметров уведомлений (время, дни недели, текст сообщения).

3. Функция «Распознавание растения по фотографии (если реализовано)»:

1. Точность: Правильное определение вида растения в 90% случаев.
2. Время выполнения: Не более 5 секунд.

Форма представления выходной информации: Информация должна быть представлена в удобном и интуитивно понятном виде для пользователей.

**Перечень и критерии отказов для функций требования по надежности**

1. Функция «Отображение информации о растении»:

Информация должна быть актуальной и соответствовать потребностям растения.

Предоставление информации обо всех основных аспектах ухода (полив, освещение, температура, удобрения).

Время выполнения: Не более 1 секунды.

2. Функция «Настройка напоминаний»:

Отправка уведомлений в установленное пользователем время.

Уведомления должны доставляться пользователю в 99.9% случаев.

3. Функция «Распознавание растения по фотографии»:

Правильное определение вида растения в 90% случаев.

Время выполнения: Не более 5 секунд.

## 4.3. Требования к видам обеспечения

Минимальное математическое обеспечение:

Система не требует сложного математического обеспечения, кроме базовых арифметических операций (для расчета времени до полива, количества удобрений) и, опционально, алгоритмов обработки изображений (для распознавания растений по фотографии).

Надежное и структурированное информационное обеспечение:

Это ключевой аспект. Необходимо четко определить, какие данные собираются (информация о растениях, пользовательские настройки, записи в дневнике наблюдений), как они хранятся (в базе данных или с использованием внешнего API), как передаются между компонентами системы, и как обеспечивается их безопасность и целостность. Важно обеспечить совместимость с внешними сервисами (API для растений, API для распознавания изображений) и соответствие законодательству (в частности, в отношении защиты персональных данных).

Требования к информационному обмену между компонентами системы

Данные системы PlantBuddy должны быть организованы с использованием следующих структур:

1. Данные о растениях:

Может храниться в локальной базе данных (Room) или получаться из внешнего API (например, PlantNet API).

Должна содержать информацию о названии растения, потребностях в поливе, освещении, температуре, удобрениях, влажности воздуха, а также фотографии растения.

Структура данных должна быть оптимизирована для быстрого поиска информации по названию растения или по фотографии (если реализовано).

2. Данные пользователя:

Хранится в локальной базе данных (Supabase).

Должна содержать информацию о предпочтениях пользователя, списке растений, о которых он хочет получать информацию, настройках уведомлений и записях в дневнике наблюдений.

Должна обеспечивать возможность безопасного хранения и доступа к персональным данным пользователя.

**Требования к информационной совместимости со смежными системами**

На этапе проектирования (эскизный и технический проекты) определить конкретные данные, которые будут передаваться/получаться для каждой смежной системы:

API для растений: Определить формат запросов и ответов, структуру данных о растениях.

API для распознавания изображений: Определить формат изображений, которые будут передаваться для распознавания, а также формат результатов распознавания.

Сервис push-уведомлений: Определить формат сообщений, которые будут отправляться пользователям.

Учитывать потребности каждой смежной системы и требования к формату данных.

Предоставить механизмы для экспорта данных из системы (например, для резервного копирования).

Использовать стандартные и общепринятые форматы данных (например, JSON).

Обеспечить возможность загрузки данных, полученных от смежных систем (например, импорт списка растений из файла).

Предусмотреть механизмы для проверки и валидации импортируемых данных.

**Требования по использованию классификаторов, унифицированных документов и классификаторов**

Использовать общепринятые классификаторы растений (например, систематику APG IV) для обеспечения единообразия представления информации.

Использовать справочники по уходу за растениями (например, рекомендации по поливу, освещению, удобрениям) для предоставления пользователям проверенной и актуальной информации.

**Требования по применению систем управления базами данных**

Использовать реляционную СУБД (Supabase) для хранения данных о растениях и пользователях.

Обеспечить возможность масштабирования СУБД для хранения большого количества данных.

Обеспечить безопасность и целостность данных, хранящихся в СУБД.

**Требования к структуре процесса сбора, обработки, передачи данных в системе и представлению данных**

1. Сбор информации:

Информация о растениях собирается из внешних API (если используются) или вводится вручную администратором базы данных.

Информация о пользователе собирается при регистрации и в процессе использования приложения.

2. Обработка информации:

Полученная информация обрабатывается и структурируется в соответствии с форматом, требуемым системой.

Данные о потребностях растения анализируются для предоставления пользователю рекомендаций по уходу.

3. Передача информации:

Информация передается между компонентами системы с использованием стандартных протоколов (например, HTTP).

Данные передаются в безопасном режиме (например, с использованием шифрования).

4. Хранение информации:

Данные о растениях и пользователях хранятся в реляционной СУБД (Supabase).

Обеспечивается защита данных от несанкционированного доступа.

**Требования к защите данных от разрушений при авариях и сбоях в электропитании системы**

Использовать отказоустойчивые технические средства (например, облачное хранилище с автоматическим резервированием).

Предусмотреть меры по защите данных от повреждения в случае перебоев с электропитанием.

**Требования к контролю, хранению, обновлению и восстановлению данных**

Обеспечить возможность контроля целостности данных.

Обеспечить безопасное хранение данных.

Предусмотреть механизмы для обновления данных (например, добавление новых растений в базу данных).

**Требования к лингвистическому обеспечению**

1. Языки Программирования (Мобильный Клиент):
2. (Android): Kotlin.

2. Языки Программирования (Серверная Часть):

1. Kotlin.
2. SQL (для работы с базой данных).

3. Языки и Стандарты Взаимодействия (Мобильный Клиент - Сервер):

1. JSON: Для обмена данными между мобильным приложением и сервером (основной формат).
2. RESTful API: Стандартная архитектура для взаимодействия между мобильным клиентом и сервером.

4. Кодирование и Декодирование Данных:

UTF-8: Для обмена данными между мобильным приложением и сервером, а также для хранения данных в базе данных (более современный и универсальный, чем CP1251).

CP1251: Только если требуется интеграция с устаревшими системами, которые используют эту кодировку. В этом случае необходимо предусмотреть конвертацию.

5. Язык Манипулирования Данными (Сервер):

1. SQL: Для запросов к базе данных.

6. Средства Описания Предметной Области:

UML: Для моделирования бизнес-логики и взаимодействия компонентов системы.

**Требования к программному обеспечению**

1. Состав Покупного ПО (Серверная Часть):
2. СУБД, PostgreSQL
3. TensorFlow Lite
4. Firebase Cloud Messaging (FCM)
5. PlantNet API

Обеспечение кроссплатформенности мобильного приложения (Android) достигается за счет выбора соответствующих фреймворков разработки

1. Требования к Качеству ПО (Как для Серверной, так и для Мобильной Части):

Архитектура

1. UI (Activity/Fragment): Отображение данных и обработка действий пользователя.
2. ViewModel: Управление состоянием UI, получение данных из репозитория и передача их в UI. Выполнение логики, связанной с UI.
3. Repository: Абстракция доступа к данным. Реализует логику получения данных из Supabase и кэширования.
4. Supabase Client: Клиентская библиотека для взаимодействия с Supabase.
5. Supabase Database: Облачная база данных Supabase.

Функциональность: Все подсистемы (мобильный клиент, и сервер, если используется) должны выполнять все предусмотренные проектом функции. Это включает:

1. Регистрацию и авторизацию пользователей.
2. Поиск информации о растениях.
3. Распознавание растений по фотографии.
4. Отображение информации об уходе за растениями.
5. Настройку уведомлений.
6. Ведение дневника наблюдений.

Надежность: Приложение должно стабильно работать без сбоев и ошибок.

Должны быть предусмотрены механизмы обработки исключений для предотвращения аварийного завершения работы приложения.

Должно быть обеспечено максимальное покрытие кода тестами (юнит-тесты, UI-тесты).

Легкость применения:

Мобильное приложение должно иметь интуитивно понятный и простой в использовании интерфейс.

Пользователь должен иметь возможность легко находить нужную информацию и выполнять необходимые действия.

Интерфейс должен быть адаптивным и корректно отображаться на различных устройствах Android.

Эффективность: Принятие верных и обоснованных решений на всех этапах разработки ПО и системы в целом.

Сопровождаемость:

Высокое качество документации (API, архитектура системы, код).

Описание объектов и комментарии в программном коде (как на сервере, так и в мобильном приложении).

Использование осмысленных (мнемонических) и устойчиво различимых имен объектов.

Производительность:

Сервер: Оптимизация запросов к базе данных, масштабируемость.

Мобильное приложение: Быстрая загрузка, отзывчивость интерфейса, эффективное использование ресурсов устройства.

Безопасность: Защита данных пользователей, безопасное взаимодействие с API, устойчивость к взлому.

Конфиденциальность: Обеспечение конфиденциальности личных данных, в том числе локационных данных.

Доступность: Обеспечение стабильной работы сервиса 24/7.

Масштабируемость: Возможность увеличения мощности серверной части, количества поддерживаемых устройств без потери производительности.

Обновляемость: Обеспечение возможности обновления мобильного приложения без обязательной переустановки.

Тестируемость: Архитектура должна быть построена таким образом, чтобы компоненты можно было легко протестировать.

Юзабилити: Соответствие мобильного приложения стандартам юзабилити.

**Требования к техническому обеспечению**

Сервер Базы Данных:

CPU: 16 (32 core)

RAM: 128 Gb

HDD: 500 Gb

Network Card: 2 (2 Gbit)

Fiber Channel: 4

Сервер Сбора, Обработки и Загрузки Данных:

CPU: 8 (16 core)

RAM: 32 Gb

HDD: 100 Gb

Network Card: 2 (1 Gbit)

Fiber Channel: 2

Сервер Приложений:

CPU: 6 (12 core)

RAM: 64 Gb

HDD: 300 Gb

Network Card: 3 (1 Gbit)

Сеть Хранения Данных (SAN):

Дисковый массив HP XP.

Минимальный объем свободного пространства: 100 Тб.

Функциональные, Конструктивные и Эксплуатационные Характеристики:

1. Серверы должны быть подключены к дисковому массиву HP XP с организацией сети хранения данных (SAN).
2. Параметры серверов, указанные выше, представляют собой минимальные требования.
3. При необходимости (в зависимости от нагрузки) может потребоваться увеличение ресурсов.
4. Необходимо обеспечить надежное электропитание и охлаждение серверов.
5. Необходимо предусмотреть систему мониторинга состояния серверов и сети хранения данных.

**Требования к организационному обеспечению**

1. Кто использует: Пользователи мобильного приложения (люди, ухаживающие за растениями). Администратор базы данных растений

2. Кто поддерживает: Разработчик приложения (в рамках сопровождения проекта).

4. Безопасность:

1. Аутентификация (логин/пароль + ограничение доступа).
2. Понятное руководство для всех.
3. Аудит изменений.

5. Удобство: Простой интерфейс, подтверждения действий.

6. Конфиденциальность: Минимум личных данных для диспетчеров.

**Требования к патентной чистоте**

1. Лицензии: Все используемые компоненты (API, библиотеки, SDK) должны иметь корректные лицензии, разрешающие их использование в коммерческих или некоммерческих целях.
2. Патентная чистота: Использование всех компонентов не должно нарушать патенты третьих лиц.

# 5. Состав и содержание работ по созданию системы

Проектирование:

1. Разработка концепции системы и определение ключевых функций.
2. Проектирование архитектуры системы (серверная часть, мобильное приложение, API).
3. Разработка схемы базы данных.
4. Проектирование пользовательского интерфейса мобильного приложения (UI/UX).
5. Проектирование API для взаимодействия между мобильным приложением и сервером.
6. Разработка спецификаций на все компоненты системы.

Разработка и Адаптация

1. Разработка серверной части системы (backend).
2. Разработка мобильных приложений для Android (frontend).
3. Разработка и тестирование API.
4. Интеграция всех компонентов системы.
5. Проведение модульного и интеграционного тестирования.

# 6. Порядок контроля и приёмки системы

Виды, Состав, Объем и Методы Испытаний:

1. Модульное тестирование: Проверка отдельных модулей (80% покрытия кода, автоматизированное тестирование).
2. Интеграционное тестирование: Проверка взаимодействия модулей (ручное и автоматизированное тестирование основных сценариев).
3. Системное тестирование: Проверка системы в целом (соответствие техническому заданию, функциональные и нефункциональные требования).
4. Приемочное тестирование: Проверка системы Заказчиком в реальных условиях (пилотное тестирование, сбор отзывов).
5. Тестирование безопасности: Выявление уязвимостей (тестирование на проникновение, сканирование уязвимостей).

# 7. Требования к документированию

Минимальный список документов:

1. Техническое Задание (ТЗ): Что приложение должно делать.
2. Руководство Пользователя: Как пользоваться приложением (для водителей и пассажиров).
3. Описание API: Как приложение взаимодействует с другими сервисами (картами, платежами).
4. Исходный Код: Сам код приложения.
5. Описание архитектуры: Основные технические решения
6. План тестирования: Как будем проверять, что приложение работает правильно.

Для каждой библиотеки, SDK или API, которые используете:

1. Запишите точное название и версию.
2. Добавьте ссылку на официальную документацию.
3. Кратко опишите, как этот компонент используется в вашем приложении.
4. Укажите лицензию (например, MIT, Apache 2.0).

Во время разработки:

1. Регулярно обновляйте все документы.
2. Используйте систему контроля версий (Git) для кода и документации.

# 8. Источники разработки

1. ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы [<https://www.prj-exp.ru/patterns/pattern_tech_task.php>] (дата обращения 17.09.2025).
2. Энциклопедия [https://leplants.ru/plant-encyclopedia/?ysclid=mfoa8j7ati980197616] (дата обращения 17.09.2025).