МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра математической теории упругости и биомеханики

Разработка компонентов бэкенда для помощника по уходу за растениями

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 442 группы

направления 09.03.03 – Прикладная информатика

механико-математического факультета

Зайцева Андрея Владимировича

Научный руководитель
доцент, к.ю.н. Р.В. Амелин

Зав. кафедрой
зав. кафедрой, д.ф.-м.н., профессор Л.Ю. Коссович

Саратов 2025

Введение. Стартап «Помощник по уходу за растениями на базе моделей ИИ» предлагает решение для любителей растений и профессиональных садоводов. Идея проекта – разработка чат-бота на основе искусственного интеллекта, который помогает определять виды растений, информировать о здоровье и предоставлять персонализированные рекомендации по уходу. Предлагаемый продукт будет способствовать улучшению состояния растений и повышению осведомленности пользователей в области ботаники, отвечая на растущий интерес к экологии и помогая сохранять природу.

Актуальность проекта в целом обосновывается тем, что около 80% хозяйств и садоводов в России имеют комнатные растения. Кроме того, в условиях растущего интереса к устойчивому потреблению и инициативам по сохранению растений, проект обладает потенциалом востребованности и за пределами Российской Федерации. Исследования рынка приложений данного направления показали малый процент конкурентов на рынке отечественного производства, также было найдено и изучено несколько зарубежных продуктов, анализ которых позволил не только сформулировать требования к разрабатываемому приложению, но и сформулировать предложения по улучшению предлагаемого функционала и стиля по сравнению с существующими аналогами.

Основная цель проекта – разработка приложения, позволяющего удовлетворить потребность пользователя в быстрой идентификации растений и их болезней по фотографии и поиску информации по уходу, профилактике и лечению заболеваний.

Целью данной бакалаврской работы является создание серверной части (бэкенда) для проекта, удовлетворяющей современным требованиям разработки веб-приложений.

Разработка бэкенда представляет собой неотъемлемую составляющую создания современных веб и мобильных приложений, обеспечивающую обработку данных, выполнение бизнес-логики и взаимодействие с базами данных. В условиях стремительного развития цифровых технологий и

широкого распространения облачных вычислений проектирование и реализация серверной архитектуры становятся важными задачами. Все это определяет актуальность темы бакалаврской работы.

Одними из требований, предъявляемых к современным приложениям, являются требования к безопасности, гибкости и расширяемости. Проект будет соответствует стандарту CIA информационной безопасности, которая обеспечивается за счёт внедрения современных инструментов разработки, позволяющих достичь высокой скорости обработки данных, стабильности и безопасной работы приложения. За счёт использования современных фреймворков и гибкой структуры серверной части в связке с искусственным интеллектом имеется возможности своевременно менять или дорабатывать отдельные модули и компоненты.

Для достижения целей проекта в целом и данной бакалаврской работы, в частности, были поставлены следующие **задачи:**

– провести маркетинговый анализ рынка;

– составить требования к приложению;

– разработать структуру серверной части;

– реализовать необходимые модули в соответствии с требованиями к проекту;

– запустить хостинг приложения.

Структура и объем работы. Работа состоит из введения, трех разделов, заключения, трёх приложений и содержит 51 страницу. Список использованных источников включает 25 наименований.

Раздел 1. Анализ рынка и разработка требований.

Раздел 2. Разработка архитектуры серверной части.

Раздел 3. Реализация компонентов бэкенда.

Основное содержание работы. Работа посвящена разработке серверной части для проекта-стартапа «Помощник по уходу за растениями на базе моделей ИИ».

Во введении описана идея проекта в целом, обоснованы его актуальность и цель, кроме того, поставлена цель бакалаврской работы – создание бэкенда для проекта. Обоснована актуальность разработки бэкенда для современных приложений, произведена постановка задач для достижения целей бакалаврской работы и проекта.

В первом разделе раскрыто понятие анализа рынка, исследован вопрос роли проекта для общества и конечного потребителя, рынка, сформулированы требования к приложению.

В качестве потребителей были выбраны частные лица и малый бизнес, к примеру, магазины, которые продают горшечные растения или удобрения. Последние также являются основными партнерами на данный момент. Далее потребители рассматриваются более конкретно, определяются их основные характеристики, а затем на основе этого создается портрет потребителя. В двух предпоследних пунктах первого раздела рассмотрен вопрос о наличии на рынке конкурентов и о соответствующей ценовой стратегии, сделан вывод о том, что отечественные конкуренты на данный момент фактически есть, но за счет преимуществ в виде выбранного стиля, использования недавно выпущенных моделей искусственного интеллекта, высокой точности, современного дизайна, данный проект может лучше привлечь клиента. Также были рассмотрены варианты продвижения продукта и ценовой политики, показано, что оптимальным вариантом использования приложения для потребителей является короткий бесплатный период, затем предложение по переходу на подписку по различным тарифам, где самый дешевый будет по цене меньше средней по рынку. В итоге сформулированы функциональные и нефункциональные требования к приложению.

На рисунке 1 представлена диаграмма C4, описывающая взаимодействие пользователя с различными частями системы.

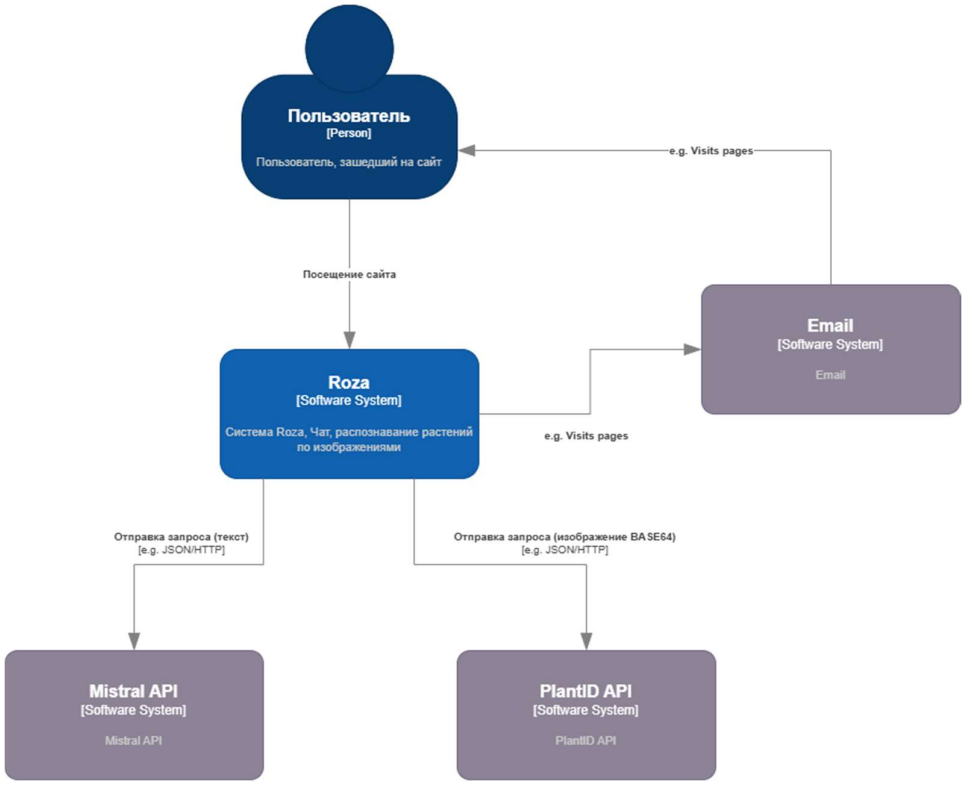

Рисунок 1 – Контекстный уровень приложения

Основными элементами приложения являются:

– пользователь – человек, использующий систему;

– Roza – приложение, которое обрабатывает запросы пользователя;

– Email – отвечает за отправку писем пользователям (например, уведомления);

– Mistral API – внешний API, с которым взаимодействует Roza, отправляя запросы по протоколу HTTP в формате JSON;

– PlantID API – внешний API для идентификации растений, также работает по протоколу HTTP в формате JSON.

Процесс взаимодействия пользователя с приложением можно описать следующим образом:

– пользователь заходит в приложение и загружает фото растения, войдя перед этим в свой аккаунт, если у пользователя нет аккаунта, то он регистрирует новый и подтверждает его по почте;

– серверная часть приложения обрабатывает полученную информацию и отправляет запросы на Mistral API и PlantID API;

– получив ответы от API, серверная часть направляет полученные результаты в чат пользовательского интерфейса.

Во втором разделе представлена архитектура серверной части. В начале описаны основные понятия бэкенда, которые важны для формирования структуры. Показано, что типичная серверная архитектура включает в себя несколько ключевых элементов, каждый из которых играет важную роль в обеспечении производительности, надежности и масштабируемости:

– серверы;

– базы данных;

– API;

– кэширование;

– асинхронный обмен.

Среди ключевых принципов проектирования архитектуры выделены:

– разделение на небольшие, слабо связанные компоненты;

– микросервисная архитектура;

– масштабирование;

– безопасность.

На рисунке 2 приведена структура бэкенда. В данном случае бэкенд состоит из следующих компонентов:

– App.py;

– config.py

– email_services.py

– db.py

– info.env

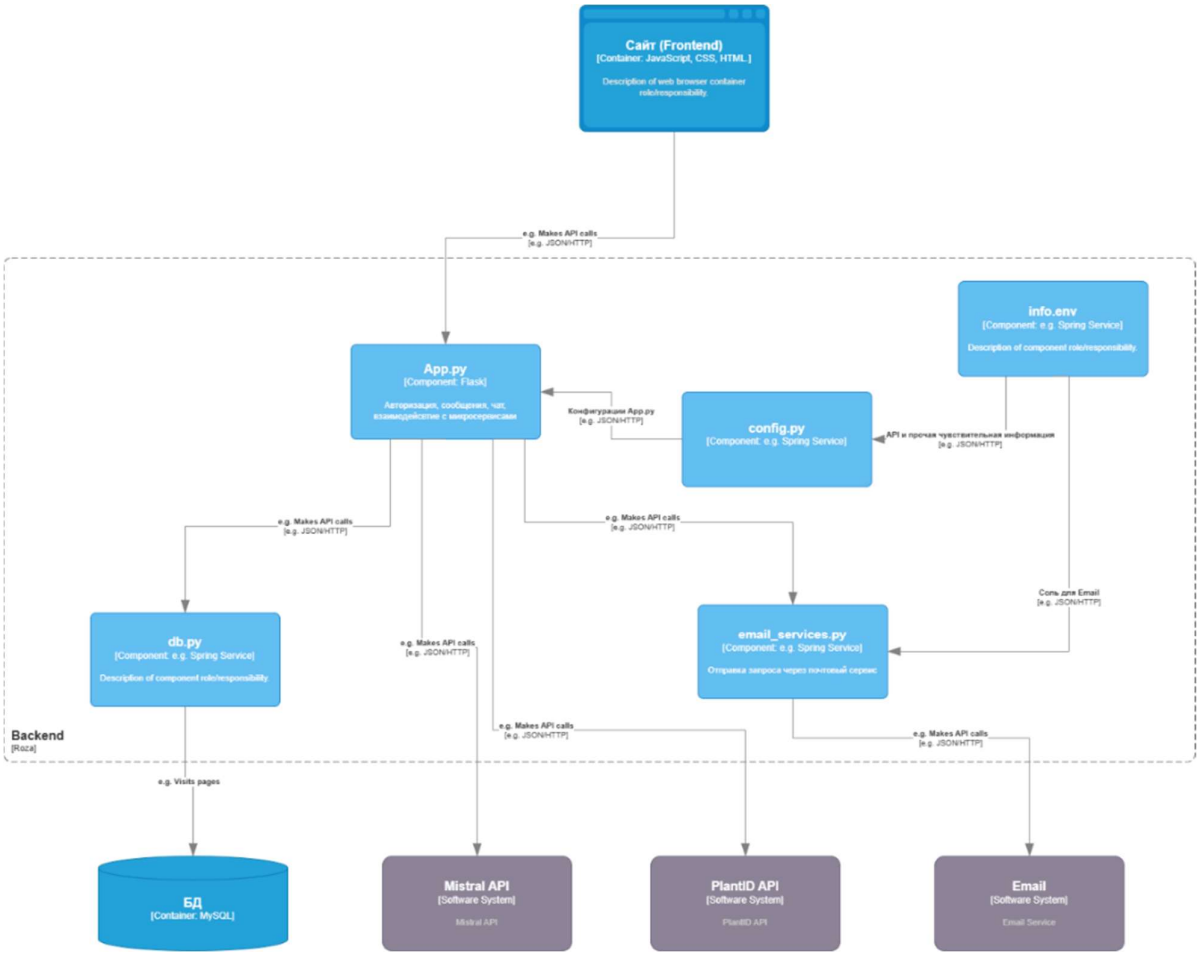

Рисунок 2 – Компонентный уровень приложения

App.py выполняет центральную функцию в приложении. Этот модуль отвечает за управление аутентификацией и передачей данных между различными частями приложения. Взаимодействие с остальными компонентами осуществляется посредством API-вызовов с применением протокола JSON/HTTP. config.py участвует в конфигурации приложения, включая параметры безопасности и сетевые настройки. email_services.py отвечает за обработку и отправку электронных сообщений. db.py служит для взаимодействия с базой данных, реализуя безопасное хранение и извлечение информации. Взаимодействие с App.py происходит через API-вызовы. info.env выполняет функцию хранения конфигурационных переменных среды, которые используются различными частями приложения. Он позволяет централизованно управлять параметрами конфигурации без необходимости их жесткого кодирования в исходных файлах.

В последнем пункте второго раздела обоснован выбор системы управления баз данных (СУБД) и рассмотрена архитектура самой базы данных. Была выбрана MySQL, обладающая высокой скоростью работы, масштабируемостью, надежностью и поддержкой репликации данных. Она совместима с различными языками программирования, такими как Python, PHP, Java и C++ и является частью экосистемы LAMP (Linux, Apache, MySQL, PHP/Python/Perl). Также она поддерживается pythonanywhere, который поддерживает Flask. В итоге выбор основан на взаимосвязи фреймворков и технических возможностях. База данных состоит из четырех таблиц, а именно: message, users, user_plant_classification, subscription_types. Последние две таблицы связаны с первой через таблицу users.

В третьем разделе описана практическая часть работы, включающая реализацию идеи в коде. Основа бэкенда - фреймворк Flask. Flask подходит для небольших и средних проектов, которые не требуют сложности большого фреймворка. В основе архитектурной концепции Flask лежит принцип минимализма, подразумевающий включение в базовую структуру фреймворка исключительно необходимых компонентов с возможностью расширения по мере необходимости. Основу Flask составляют два ключевых компонента:

– Werkzeug – библиотека реализации WSGI (Web Server Gateway Interface), представляющая собой стандартизированный интерфейс взаимодействия веб-серверов с Python-приложениями. Данный компонент обеспечивает низкоуровневую обработку HTTP-запросов и ответов;

– Jinja2 – система шаблонизации, использующая синтаксические конструкции, интегрированные в HTML-документы для динамической генерации контента. Jinja2 реализует механизм компиляции шаблонов в оптимизированный Python-код для повышения производительности рендеринга.

Далее более подробно разобраны модули бэкенда с точки зрения логики работы и кода.

Файл info.env загружается в приложение при старте и обеспечивает доступ к переменным этого файла настройкам из кода.

Модуль db.py осуществляет интеграцию базы данных MySQL с приложением.

Функция get_db_connection является центральной в модуле и отвечает за создание и возвращение соединения с базой данных MySQL. Она реализует паттерн singleton, используя объект g из Flask для хранения соединения внутри контекста запроса. Если соединение уже существует, функция просто возвращает его. Если нет, создается новое соединение с параметрами из переменных окружения, устанавливается режим автоматической фиксации изменений, и соединение сохраняется в g.db.

Модуль email_services.py отвечает за отправку электронных писем для верификации адреса электронной почты пользователя. Функция принимает email адрес получателя, имя пользователя и URL для верификации как параметры.

Модуль config.py представляет собой конфигурационный файл приложения, сосредоточенный на настройках безопасности и управлении сессиями.

Модуль App.py является главным, т.к. включает в себя все модули, описанные выше, и участвует в процессе взаимодействия с пользователем. В нём создаётся объект app как экземпляра класса Flask, после чего производится инициализация базы данных путем вызова метода init_app для объекта db. Подключение защиты от атак типа CSRF осуществляется с использованием CSRFProtect, который привязывается к приложению.

Кроме того, в третьем разделе подробно описано взаимодействие модулей. Описаны формы, через которые пользователь регистрируется и как данные отправляются в БД. Обоснованы причины выбора хостинга pythonanywhere, а именно: встроенная поддержка веб-фреймворков Django и Flask позволяет быстро развертывать веб-сайты и API без необходимости настраивать серверное окружение вручную. PythonAnywhere предоставляет возможности для развертывания серверных приложений, в том числе хостинга

веб-сайтов и работы с базами данных. В качестве систем управления базами данных платформа поддерживает MySQL и PostgreSQL.

Заключение. В ходе выполнения бакалаврской работы был проведен анализ рынка потенциальных потребителей, анализ конкурентов, составлены требования к приложению «Помощник по уходу за растениями на базе моделей ИИ» с учетом современных тенденций и стандартов разработки программного обеспечения, разработана структура серверной части приложения, реализованы необходимые модули и запущен хостинг приложения. Работа выполнена в рамках проекта стартап как диплом. Разрабатываемое приложение предназначено для удовлетворения потребностей пользователей в оперативном и точном получении информации о состоянии растений и рекомендациях по уходу.

Проект отвечает современным требованиям к веб-приложениям, включая безопасность, масштабируемость и гибкость архитектуры, что делает его конкурентоспособным как на отечественном, так и на международном рынке. Разработка серверной части осуществлялась с учетом стандартов информационной безопасности, включая модель CIA, что обеспечит защиту пользовательских данных, стабильность работы приложения и ее отказоустойчивость. Использование современных инструментов и фреймворков позволило создать адаптивную и легко расширяемую серверную архитектуру, способную интегрироваться с алгоритмами искусственного интеллекта для анализа растений. Гибкость предложенной системы обеспечивает возможность оперативного внесения изменений и доработки отдельных модулей. Данное решение успешно интегрировано в общий проект.

В дальнейшем предложенное проект может развиваться во многих направлениях за счет своей гибкой структуры и на данный момент уже выявлены возможности его использования работы в сельскохозяйственном направлении за счет изменения API модели. Таким образом, цель бакалаврской работы достигнута и все поставленные задачи выполнены.