МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

(ФГБОУ ВО «ВГУ»)

Факультет компьютерных *наук*

Кафедра программирования и информационных технологий

Разработка агрегатора для продажи цветов и подарков «Giftly»

Курсовая работа

Направление: 09.03.02. Информационные системы и технологии

Зав. Кафедрой _____ д. ф.-м. н, доцент С.Д. Махортов

Руководитель _____ ст. преподаватель В.С. Тарасов

Руководитель практики _____ М.А. Савин

Обучающийся _____ Д.В. Стеблев, 3 курс, д/о

Обучающийся _____ Д.А. Кондратьев, 3 курс, д/о

Обучающийся _____ Д.А. Пальчикова, 3 курс, д/о

Обучающийся _____ М.С. Нетесова, 3 курс, д/о

Обучающийся _____ А.В. Одинцов, 3 курс, д/о

Воронеж 2025

Содержание

Определения, обозначения и сокращения

В настоящей курсовой работе используются следующие определения, обозначения и сокращения:

API (Application Programming Interface) – это набор правил и инструкций, с помощью которых различные приложения и сервисы могут взаимодействовать друг с другом

CRUD – комплекс базовых операций для работы с данными: создание, чтение, обновление, удаление

CI/CD – методология и набор практик для автоматизации процессов разработки, тестирования и развёртывания программного обеспечения

CSRF – уязвимость безопасности веб-приложений, позволяющая злоумышленнику выполнять действия от имени аутентифицированного пользователя

Django – популярный веб-фреймворк на языке Python для быстрой разработки высоконагруженных веб-приложений

Docker – технология контейнеризации, позволяющая упаковывать приложения со всеми их зависимостями в изолированные контейнеры для легкого развертывания

Flutter – кроссплатформенный инструментарий для разработки пользовательских интерфейсов, позволяющий создавать приложения для мобильных и других платформ из единой кодовой базы

HTTP – протокол прикладного уровня для передачи данных в сети Интернет

HTTPS – защищенная версия протокола HTTP с использованием криптографических протоколов SSL/TLS для шифрования данных

JSON – легковесный текстовый формат обмена данными, основанный на синтаксисе объектов JavaScript

ORM – технология, позволяющая разработчикам взаимодействовать с базой данных, используя объекты языка программирования, а не SQL-запросы

PostgreSQL – мощная, расширяемая объектно-реляционная система управления базами данных с открытым исходным кодом.

REST API – архитектурный стиль для построения распределенных систем, использующий принципы HTTP и ресурсов

СУБД – комплекс программных средств для создания, управления и обслуживания баз данных

UI – совокупность элементов интерфейса, с которыми взаимодействует пользователь, включая графику, текст, кнопки

UX – общее впечатление и эмоции пользователя, возникающие в процессе взаимодействия с продуктом

XSS – уязвимость безопасности веб-приложений, при которой злоумышленник внедряет вредоносный скрипт на страницу, просматриваемую другими пользователями

Введение

На современном рынке товаров и услуг наблюдается устойчивый рост спроса на цифровые платформы, способные упростить и оптимизировать процесс взаимодействия потребителей с поставщиками. В частности, сфера продажи цветов и подарков, традиционно ассоциирующаяся с офлайн-точками продаж, активно трансформируется под влиянием электронной коммерции. Тем не менее, зачастую потребители сталкиваются с разрозненностью предложений, неудобством выбора и сложностями в организации доставки, что снижает качество пользовательского опыта и ограничивает возможности выбора. Отсутствие единой, удобной и многофункциональной площадки, агрегирующей ассортимент различных поставщиков, является заметным пробелом на данном рынке.

Развитие мобильных технологий и повышение доступности высокоскоростного интернета открывают новые возможности для создания комплексных решений, способных преодолеть эти барьеры. Появление интуитивно понятных мобильных приложений, объединяющих разнообразные предложения, становится ключевым фактором для удовлетворения потребностей современного потребителя.

Настоящий проект представляет собой разработку мобильного приложения "Giftly" – агрегатора для продажи цветов и подарков. Основной целью приложения является предоставление пользователю централизованной платформы, позволяющей удобно находить, выбирать и заказывать подарки от различных поставщиков, а продавцам – эффективно управлять своим ассортиментом и заказами. Приложение нацелено на создание безупречного пользовательского опыта за счет интуитивного интерфейса, широкого функционала и внедрения интеллектуальных рекомендаций.

В данной работе будет подробно рассмотрена реализация всех функциональных возможностей мобильного приложения "Giftly". Особое внимание будет уделено обоснованию выбора архитектурных решений, детальному описанию логики взаимодействия между ключевыми компонентами системы, а также специфике разработки серверной и клиентской частей. Также будут освещены вопросы тестирования и обеспечения безопасности данных, что является критически важным для коммерческого приложения.

1 Постановка задачи

Разработка мобильного приложения "Giftly" обусловлена актуальностью и растущим спросом на онлайн-сервисы по доставке цветов и подарков. Целью проекта является создание функциональной и удобной платформы, способной обеспечить эффективное взаимодействие между продавцами и покупателями в данной нише.

1.1 Актуальность проблемы

Современный ритм жизни диктует потребность в быстрых и удобных решениях для повседневных задач, включая покупку подарков и цветов. Традиционные способы приобретения зачастую требуют значительных временных затрат на посещение магазинов, поиск нужного ассортимента и организацию доставки.

Существующие онлайн-сервисы не всегда в полной мере удовлетворяют потребности пользователей в персонализированном подходе, широком выборе и эффективной коммуникации с продавцами. Таким образом, существует явная потребность в интуитивно понятном и многофункциональном приложении, которое упростит процесс выбора, заказа и получения подарков, а также позволит продавцам эффективно управлять своим бизнесом.

1.2 Цели и задачи проекта

Целями создания приложения являются:

— Реализация процесса купли-продажи цветов и подарков через мобильное приложение;

— Получение прибыли за счёт комиссий за использования сервиса;

— Создание платформы для связывания продавцов и покупателей.

Для достижения поставленной цели необходимо решить следующие задачи:

— Обеспечить хранение и эффективное управление пользовательскими данными, включая возможность редактирования личной информации в профиле и надежное хранение с возможностью выборки из базы данных;

— Реализовать полноценный процесс купли-продажи товаров, который включает в себя функции выбора, оформления заказа и оплаты для покупателей, а также возможность для продавцов размещать свои товары и получать прибыль от продаж;

— Предоставить пользователям функционал для подбора товаров с помощью ИИ-консультанта, включающий встроенный чат для общения и систему рекомендаций, предлагающую цветы и подарки на основе запросов пользователя.

1.3 Функциональные требования к разрабатываемой системе

Разрабатываемое приложение должно соответствовать следующим функциональным требованиям:

Система должна позволять Неавторизированному пользователю:

— Зарегистрироваться в приложении;

— Авторизироваться в приложении, если аккаунт уже создан;

— Просмотреть каталог товаров;

— Осуществить поиск товаров;

— Просмотреть карточку товара, информацию о нём.

Система должна позволять Авторизированному пользователю:

— Просмотреть каталог товаров;

— Осуществить поиск товаров;

— Просмотреть карточку товара, информацию о нём;

— Просмотреть информацию о продавце;

— Добавить товар в избранное;

— Добавить товар в корзину;

— Осуществить оплату товара;

— Управлять личным кабинетом (изменить личные данные, сменить пароль);

— Заполнить личный кабинет (дата рождения, ФИО для сбора информации и индивидуальных предложений).

Система должна позволять Продавцу:

— Управлять каталогом товаров (добавление, редактирование, удаление своих товаров);

— Выставить новый товар на личную страницу;

— Создать карточку нового товара с добавлением фото, описания, цены;

— Просмотреть личную подробную статистику продаж, отзывов, рейтинга;

— Выйти из профиля.

Система должна позволять Администратору:

— Блокировать продавца;

— Блокировать пользователя.

2 Анализ предметной области

2.1 Анализ рынка срезанных цветов

Современный рынок срезанных цветов подразделяется на 2 основные категории, которые важно рассмотреть для понимания общих тенденций развития потенциального продукта.

2.1.1 Глобальный рынок

Согласно авторитетному исследованию The Business Research Company , объем мирового рынка срезанных цветов к 2029 году достигнет $47,43 млрд при среднем годовом темпе роста (CAGR) 6,1%.

При этом основными драйверами роста являются:

- Увеличение популярности онлайн-покупок;
- Персонализация и индивидуальные предложения для клиентов;
- Рост интереса к цветам в контексте здорового образа жизни;
- Активное сотрудничество флористов с онлайн-сервисами и ритейлерами.

Хотя CAGR 6,1% не является высоким показателем, он соответствует тенденциям данной ниши. При этом ключевыми трендами остаются экологичность, персонализированные предложения и цифровизация процесса покупки.

2.1.2 Российский рынок

По данным BusinesStat, российский рынок срезанных цветов демонстрирует общую положительную динамику, идентичную зарубежному рынку, восстанавливаясь после кризисных явлений 2022 года.

В 2024 году продажи цветов в стране увеличились на 5%, достигнув 2,09 млрд штук (против 1,99 млрд в 2023 году).

Одним из ключевых факторов роста рынка является перераспределение спроса в сторону онлайн-платформ. Согласно Forbes:

- С 2020 по 2024 годы количество онлайн-заказов цветов в России увеличилось в 2,8 раза;
- В мае 2023 года прирост онлайн-продаж составил 7%, в мае 2024 года – 9%;
- Доля клиентов из регионов, совершающих покупки цветов онлайн, выросла с 33% в 2020 году до 42,5% в 2023 году.

При этом средний чек на онлайн-покупку цветов за последние четыре года вырос на 16% и остается выше, чем при офлайн-покупках. Рост онлайн-торговли обеспечивается за счет регионального расширения сервисов.

2.2 Анализ конкурентов

В данном разделе приводится анализ ключевых игроков рынка цветочных агрегаторов и онлайн-магазинов. Исследование конкурентов позволяет выявить их сильные и слабые стороны, определить рыночные тренды и сформулировать конкурентные преимущества разрабатываемого решения.

2.2.1 Flowwow

Flowwow является крупнейшим мобильным агрегатором цветов и подарков на российском рынке. Платформа объединяет множество продавцов, предоставляя покупателям удобный способ выбора и заказа.

Ключевые особенности:

- 98% клиентов – новые для магазина, что свидетельствует об эффективном привлечении аудитории, что можно

использовать для разработки качественной маркетинговой стратегии;

— Полностью бесконтактное взаимодействие – заказы оформляются через чат и приложение, без звонков;

— Гибкая система комиссий – 0% при переходе по прямой ссылке магазина;

— Высокий уровень повторных покупок – 60%, что говорит о лояльности пользователей;

— Встроенная аналитика – помогает продавцам отслеживать эффективность продаж.

Flowwow – сильный конкурент с продуманной маркетинговой стратегией и удобным интерфейсом. Основные источники монетизации – реклама и комиссия с продаж.

2.2.2 Flor2u

Flor2u – не агрегатор, а самостоятельный онлайн-магазин цветов и подарков с собственными композициями, однако для формирования целостного портрета текущего рынка важно отметить всех крупных игроков.

Ключевые особенности:

— Анонимная доставка – удобная опция для неожиданных подарков;

— Уникальный ассортимент – собственные разработки флористов.

Flor2u работает в нише премиальных цветочных решений, но не является агрегатором. Ориентирован только на российский рынок, что ограничивает его масштабы.

2.2.3 FloMarket

FloMarket – мобильный агрегатор цветов и подарков с расширенными возможностями выбора.

Ключевые особенности:

- Фильтры и персонализация – удобный подбор товаров;
- Консультация с флористом – помогает покупателям определиться с выбором;
- Проблемы с доставкой и качеством наносят значительных ущерб по конверсии.

Хотя FloMarket предлагает полезные функции (например, интересные консультации), его репутация страдает из-за нестабильного качества сервиса.

В процессе сбора данных для последующей аналитики были также рассмотрены многие другие сервисы.

2.3 Целевая аудитория

2.3.1 Сегментация целевой аудитории клиентов

Ядро целевой аудитории для малых магазинов и ИП сосредоточено в возрастной группе 18-45 лет, объединяющей наиболее активных и платежеспособных покупателей. При этом проведенный анализ выявил, что значительных различий между возрастными группами нашей основной целевой аудиторией в покупательском поведении не наблюдается.

В данной возрастной группе преобладают молодые специалисты и семьи. Эти покупатели стремятся к самовыражению, активно используют цифровые технологии и ценят оригинальность. Их ключевые запросы

связаны с долговечностью растений и желанием произвести впечатление нестандартными решениями.

Далее рассмотрим гендерные особенности покупательского поведения. Исследования поведения покупателей цветов выявили существенные различия между мужчинами и женщинами.

Так, мужчины в возрасте чаще всего приобретают цветы в качестве подарков для партнеров, связывая подарок с событием. Для них характерна готовность тратить больше, но реже, выбор крупных композиций и предпочтение известных цветов, например роз. Они склонны к онлайн-покупкам с выбором именно готовых дизайнов.

Женщины той же возрастной категории покупают цветы не только для подарков, но и для личного использования.

Их выбор часто основан на эмоциональном восприятии и не привязан к праздничным событиям, они более чувствительны к цене и стремятся кастомизировать заказ. Женщины активно используют как онлайн-каналы, так и офлайн-магазины, уделяя внимание визуальному и тактильному опыту.

2.3.2 Стратегия сотрудничества для малых магазинов и ИП

Малые цветочные магазины и индивидуальные предприниматели сталкиваются с высокой конкуренцией, однако их ключевые преимущества — гибкость, персонализированный подход и локальная узнаваемость — позволяют им удерживать свою аудиторию.

Однако для масштабирования и роста им необходимы специалисты для качественного продвижения и современные инструменты, которые усилят эти сильные стороны с цифровыми возможностями.

Платформа делает упор на малых игроков рынка. В отличие от конкурентов, это не просто платформа-агрегатор, Giftly концентрируется

на персонализации и рекомендациях, это площадка для продаж с активным сообществом разработчиков, которое готово рассматривать и внедрять инициативы для улучшения бизнеса на основе обратной связи от клиентов и партнеров.

В планах реализации в продукте VIP-продвижение для продавцов за 20.000 рублей в месяц, которое позволит выводить товары в приоритет при поисковом запросе пользователя. Основная модель монетизации - комиссия 5-7% с покупки в магазине.

Таким образом, платформа может не просто облегчить продажи, но и стать стратегическим партнером для малого бизнеса, который поможет малым магазинам и ИП не только удерживать текущих клиентов, но и масштабировать бизнес с минимальными затратами.

3 Реализация

Настоящая глава описывает процесс реализации мобильного приложения "Giftly" по доставке цветов и подарков, детализируя выбранные средства разработки, архитектурные решения и особенности серверной части. Приложение построено по клиент-серверной архитектуре с четким разделением функционала между back-end и front-end, взаимодействующими посредством REST API.

3.1 Средства реализации

3.1.1 Flutter

Flutter был выбран в качестве основного фреймворка для разработки клиентской части мобильного приложения. Это кроссплатформенный UI-фреймворк от Google, который позволяет создавать нативные приложения для iOS, Android, а также веб и десктоп, используя единую кодовую базу. Выбор Flutter обусловлен его преимуществами, такими как:

— Высокая производительность: Компиляция кода в нативный ARM-код обеспечивает отличную производительность, сравнимую с нативными приложениями;

— Горячая перезагрузка и горячее обновление: Эти функции значительно ускоряют процесс разработки, позволяя мгновенно видеть изменения в коде;

— Богатый набор виджетов: Flutter предоставляет обширную библиотеку готовых, кастомизируемых виджетов, что упрощает создание красивого и отзывчивого пользовательского интерфейса;

— Гибкий UI: Возможность построения любого пользовательского интерфейса благодаря компонентному подходу и мощной системе рендеринга.

3.1.2 Dart

Язык программирования Dart, также разработанный Google, является основой для разработки на Flutter. Он был выбран благодаря его оптимальной совместимости с Flutter и набору характеристик, делающих его идеальным для мобильной разработки.

Dart оптимизирован для UI, обеспечивая плавную анимацию и отзывчивость, а также обладает высокой производительностью благодаря компиляции в нативный код и Just-in-Time (JIT) компиляции для быстрой разработки. Его синтаксис интуитивно понятен и схож с популярными языками, такими как Java и JavaScript, что упрощает освоение и написание кода.

Встроенная поддержка нулевой безопасности помогает предотвратить ошибки во время выполнения.

3.1.3 PostgreSQL

Для хранения и управления данными приложения "Giftly" была выбрана реляционная система управления базами данных (СУБД) PostgreSQL. Эта СУБД обеспечивает надежность и ACID-совместимость, гарантируя целостность и согласованность данных. PostgreSQL отличается высокой масштабируемостью, способностью обрабатывать большие объемы данных и высокую нагрузку.

3.1.4 Docker

Инструмент контейнеризации **Docker** используется для упаковки и развертывания компонентов приложения. Использование Docker позволяет изолировать каждый компонент, такой как сервер или база данных, в отдельном контейнере, предотвращая конфликты зависимостей. Контейнеры легко переносятся и развертываются на любой платформе, поддерживающей Docker, обеспечивая единообразную среду разработки и эксплуатации. Это значительно

упрощает и ускоряет процесс настройки окружения, а также облегчает горизонтальное масштабирование компонентов приложения.

3.1.5 Django

В качестве фреймворка для разработки серверной части (back-end) мобильного приложения "Giftly" был выбран Django. Этот высокоуровневый Python-фреймворк придерживается принципов "Don't Repeat Yourself" (DRY) и "Convention over Configuration". Django значительно ускоряет создание сложных приложений благодаря встроенным компонентам, таким как ORM, админ-панель и система аутентификации. С использованием пакета Django REST Framework (DRF) Django предоставляет мощные инструменты для быстрого создания RESTful API. Встроенные механизмы защиты от распространенных атак обеспечивают безопасность, а его масштабируемость позволяет обрабатывать большое количество запросов и пользователей. Обширное сообщество и множество готовых пакетов формируют сильную экосистему.

3.2 Архитектура

Архитектура мобильного приложения "Giftly" базируется на классической модели клиент-серверного приложения. Это обеспечивает четкое разделение обязанностей и гибкость в разработке и масштабировании.

3.3 Клиентская часть

Клиентская часть приложения "Giftly", разработанная на Flutter с использованием языка Dart, является основным интерфейсом взаимодействия пользователя с системой. Её главная задача — предоставить интуитивно понятный, отзывчивый и визуально привлекательный пользовательский опыт.

Ключевые аспекты реализации клиентской части включают:

— Пользовательский интерфейс (UI): Дизайн интерфейса был разработан с учетом принципов Material Design. Применялись кастомные виджеты для создания уникальных элементов, отражающих бренд "Giftly". Особое внимание уделено удобству навигации, читаемости текста и гармоничному расположению элементов;

— Управление состоянием (State Management): Для эффективного управления состоянием приложения, особенно в сложных сценариях используется подход, обеспечивающий предсказуемое и легкое для отладки поведение;

— Взаимодействие с REST API: Клиентское приложение осуществляет HTTP-запросы к серверной части для получения и отправки данных. Для этого используются стандартные библиотеки Dart, способные обрабатывать JSON-ответы. Все запросы к API сопровождаются необходимой аутентификационной информацией, полученной при входе пользователя в систему. Обработка ошибок сети и серверных ответов реализована для обеспечения стабильной работы приложения даже при частичной потере соединения.

3.3.1 Загрузочный экран

Загрузочный экран содержит логотип «GIFTLY», а также слоган «Ваши подарки, наши заботы», выполненный в минималистичном дизайне. Логотип расположен в центре экрана, что обеспечивает четкое визуальное восприятие при запуске приложения. Данный экран отображается во время загрузки приложения, подготавливая пользователя к дальнейшей работе с сервисом.

Рисунок 1 - Вариант загрузочного экрана

3.3.2 Вход

Экран авторизации содержит заголовок «Вход» и приветственное сообщение «Привет, Giffly!». Основными элементами интерфейса являются:

- Поля ввода для Email и Пароля, оформленные в минималистичном стиле;
- Ссылка «Забыли пароль?» для восстановления доступа к аккаунту;
- Кнопка «Войти» для подтверждения авторизации;
- Переключатель «Войти как покупатель», позволяющий выбрать тип входа;

— Ссылка «Зарегистрироваться» для новых пользователей.

Дизайн экрана выполнен в светлых спокойных тонах с акцентом на основные элементы управления, что делает интерфейс интуитивно понятным.

Рисунок 2 - Экран входа

3.3.3 Регистрация

Экран регистрации содержит заголовок «Регистрация». Основные элементы интерфейса:

— Поля ввода Email и Пароль для ввода данных новым пользователем;

— Опция «Зарегистрироваться как продавец», позволяющую выбрать тип аккаунта;

— Кнопка «Зарегистрироваться» для завершения процесса созданя аккаунта;

— Кнопка «Войти» для зарегистрированных пользователей.

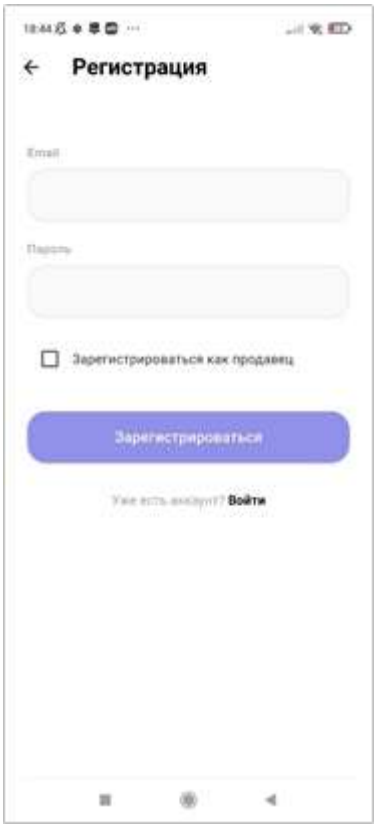

Рисунок 3 - Экран регистрации

3.3.4 Восстановление пароля

На экране регистрации размещён заголовок «Восстановление пароля», а также под ним располагается небольшая инструкция для пользователя. Основные элементы интерфейса:

— Поле для ввода номера телефона;

— Поле для ввода кода подтверждения;

— Поле для ввода нового пароля;

— Кнопка «Восстановить пароль» для подтверждения;

— Ссылка «Отправить код ещё раз» в случае необходимости повторной отправки.

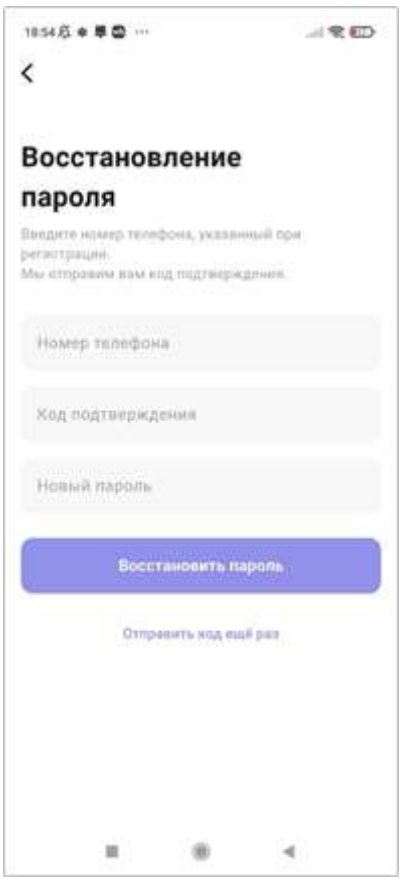

Рисунок 4 - Экран восстановления пароля

3.3.5 Профиль пользователя

3.3.5.1 Раздел навигационной панели «Профиль»

Экран профиля пользователя содержит заголовок «Профиль» и отображает информацию об аккаунте, включая имя пользователя и привязанную почту. Основные элементы интерфейса:

— Личные данные;

— Настройки;

— История заказов;

— Как стать продавцом;

— Правовые документы;

— О возврате товара;

— Поддержка Gifty;

— GitHub репозиторий.

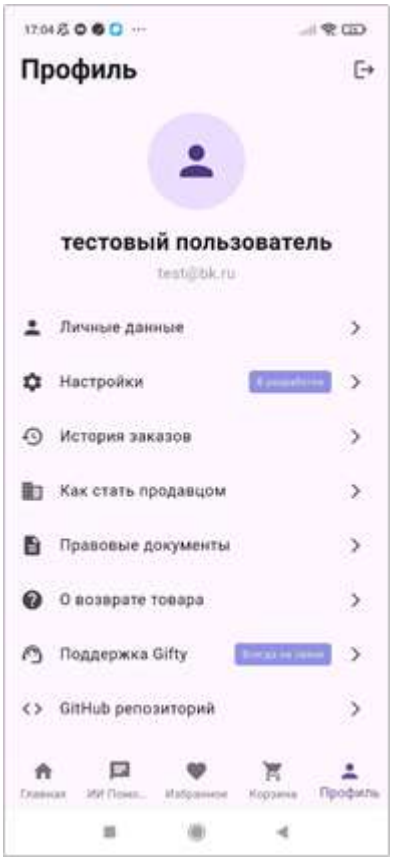

Рисунок 5 - Профиль пользователя

3.3.5.2 Настройки

Экран настроек позволяет пользователю управлять персонализацией приложения. Основные разделы:

— Внешний вид. Включает в себя переключение темной темы и настройкау размера текста;

— Уведомления. Управление push-оповещениями и уведомлениями о заказах/акциях;

— Язык. Позволяет выбрать между русским и английским.

Внизу расположена кнопка сохранения изменений и стандартная панель навигации.

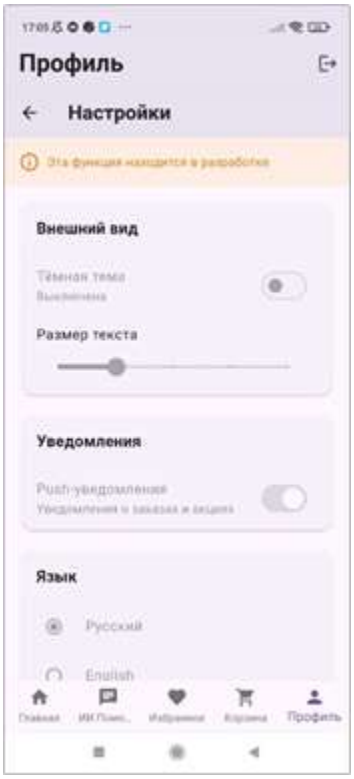

Рисунок 6 - Настройки

3.3.5.3 История заказов

Экран истории заказов отображает список предыдущих покупок пользователя. В верхней части расположен фильтр для сортировки заказов. Каждый заказ отображается в виде карточки с номером, статусом ("Завершён", "Отменён" и т.д.), общей суммой, а также датой заказа. Дополнительно присутствует кнопка "Оцените нас".

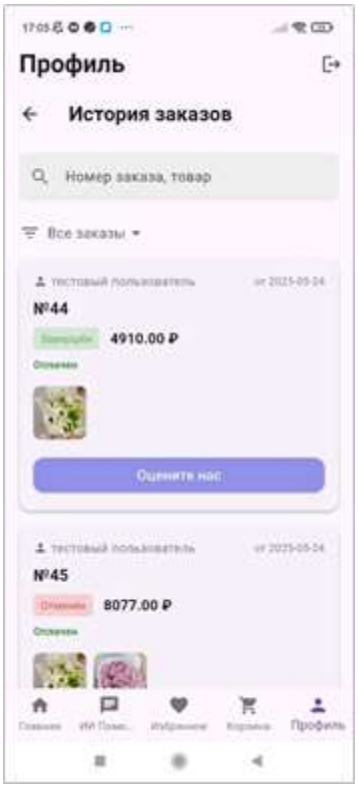

Рисунок 7 - История заказов

3.3.5.4 Личные данные

Экран личных данных позволяет просматривать и редактировать основную информацию о пользователе. Он включает следующие поля для заполнения:

— Имя;

— Фамилия;

— Номер телефона;

— Дата рождения.

Внизу расположена кнопка «Сохранить» для подтверждения изменений.

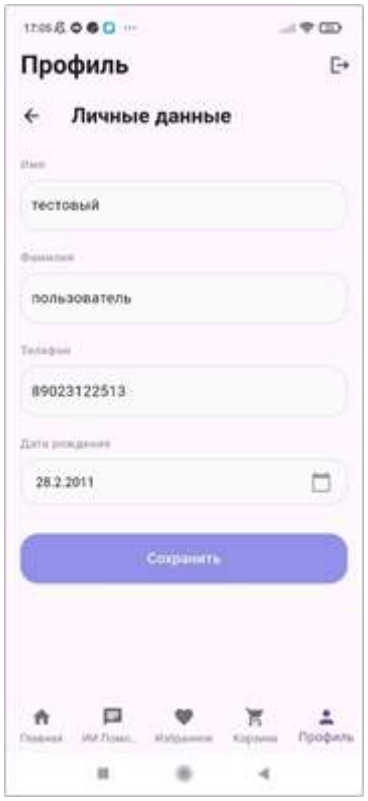

Рисунок 8 - Личные данные пользователя

3.3.5.5 Главная

В верхней части интерфейса отображается текущий город, расположенный в левом углу. Рядом находится поисковая строка, предназначенная для поиска товаров. Под ней размещен рекламный баннер.

Основную часть экрана занимает каталог букетов. Каждый товар представлен отдельным блоком с названием, описанием и ценой для него. В правом верхнем углу карточки товара расположена кнопка, позволяющая добавить товар в избранное.

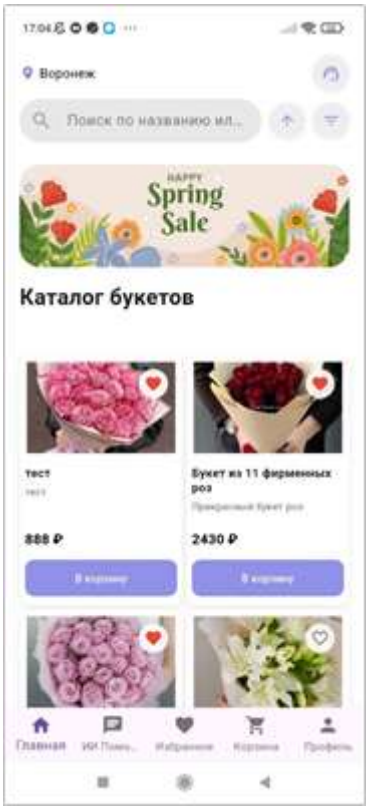

Рисунок 9 - Главная

3.3.5.6 Корзина

Экран корзины отображает выбранные товары перед оформлением заказа. Вверху показаны позиции с указанием их количества и цены. Дальше расположен раздел "Итого", указывающий общую сумму заказа. Внизу экрана находится кнопка "Оформить заказ" для перехода к покупке.

Рисунок 10 - Корзина

3.3.5.7 Оформление заказа

Экран оформления заказа разделен на два основных блока. В верхней части под заголовком "Ваш заказ" отображаются выбранные товары с указанием количества и цены.

Основной раздел "Данные для доставки" содержит:

— Переключатель "Самовывоз";

— Поле для ввода адреса доставки;

— Поле для телефона с маской ввода;

— Поле для комментария к заказу (является необязательным).

В нижней части экрана выведена итоговая сумма заказа и кнопка "Оформить заказ" для подтверждения покупки.

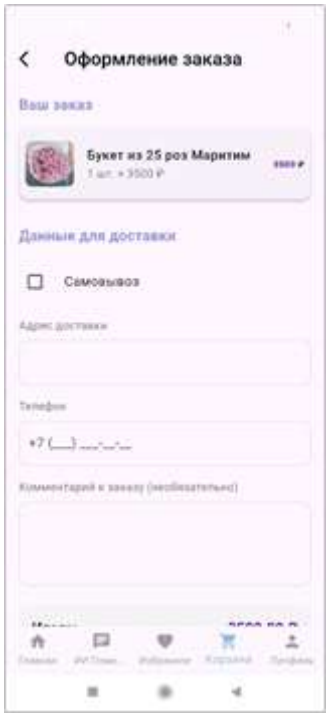

Рисунок 11 - Оформление заказа

Рисунок 12 - Оформление заказа с суммой

3.3.5.8 Оплата заказа

Экран оплаты заказа отображает выбранный товар с указанием его цены и количества. Основное пространство занимает форма для ввода платежных данных:

— Номер карты;

— Срок действия;

— Поле для CVV.

Внизу расположена кнопка "Оплатить".

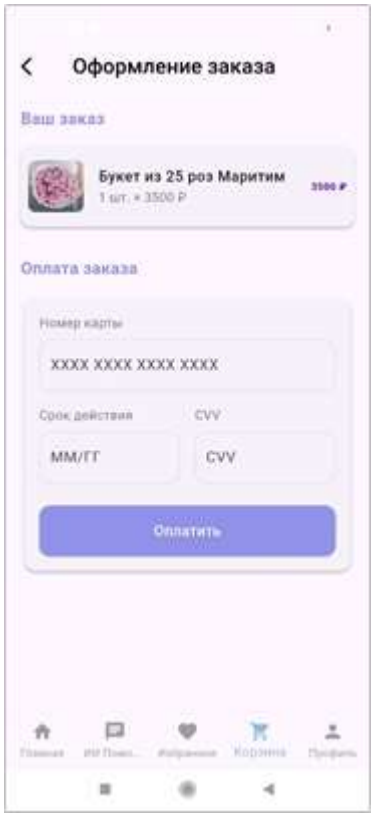

Рисунок 13 - Оплата заказа

3.3.5.9 Избранное

Экран избранного представляет собой список сохранённых пользователем товаров. Каждый товар отображается в виде карточки с названием, описанием и ценой. Под каждой позицией расположена кнопка "В корзину" для быстрого добавления к заказу.

Рисунок 14 - Избранное

3.3.5.10 ИИ-помощник

Экран ИИ-помощника представляет собой чат-интерфейс для взаимодействия с виртуальным ассистентом. Нижняя часть содержит поле для ввода пользовательских запросов. Основное пространство отведено под диалоговую ленту, где отображаются:

— Сообщения пользователя;

— Ответы ИИ-помощника;

— Рекомендации товаров с кнопками действий.

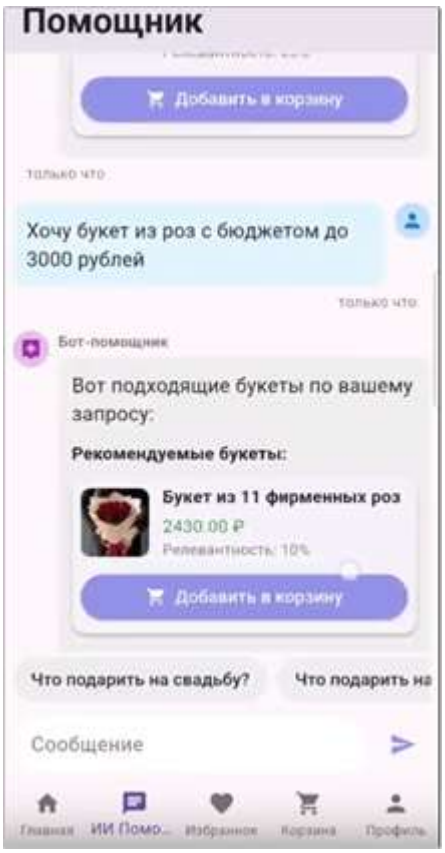

Рисунок 15 - ИИ-помощник

3.3.6 Профиль продавца

Экран профиля продавца отображает учетные данные и функциональные разделы для владельца магазина. В шапке указаны юридическое название и контактный email. Интерфейс организован в виде вертикального списка категорий:

— Персональные настройки содержат разделы "Личные данные" и "Настройки";

— Юридический блок включает доступ к регламентирующим документам, политике возвратов и каналам поддержки;

— Отдельная группа элементов связана с интеграцией GitHub, предоставляя доступ к репозиторию приложения, инструментам управления магазином, обработки заказов и аналитики.

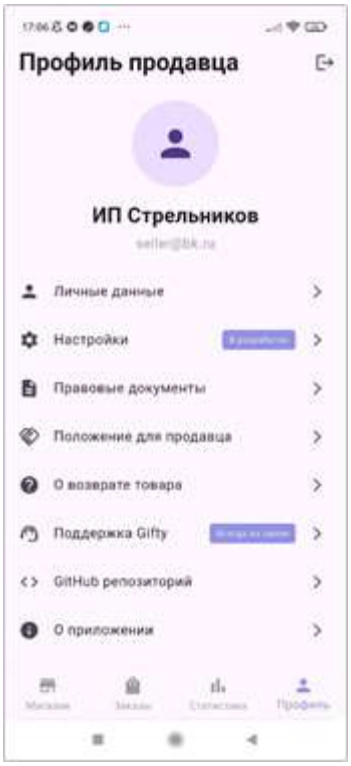

Рисунок 16 - Профиль продавца

3.3.6.1 Личные данные продавца

Экран профиля продавца отображает персональную информацию владельца бизнеса. В верхней части указаны основные данные: имя, фамилия, контактный телефон и дата рождения.

Центральное место занимает кнопка "Сохранить" для подтверждения изменений.

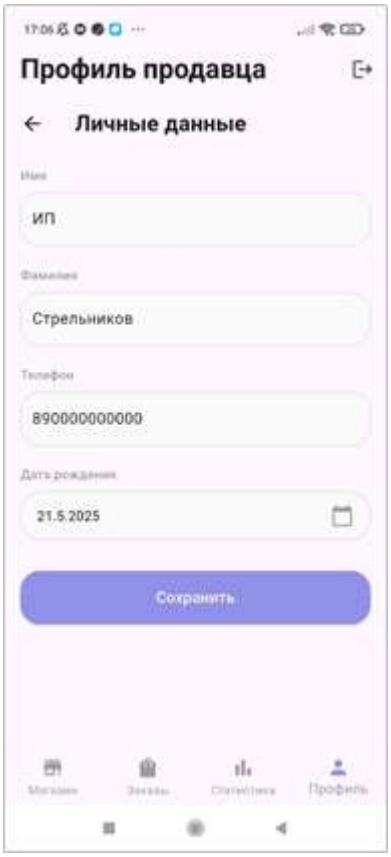

Рисунок 17 - Личные данные продавца

3.3.6.2 Статистика

Экран статистики продавца отображает ключевые метрики деятельности. В верхней части показаны основные показатели: общая выручка и количество выполненных заказов.

Ниже расположены два аналитических раздела - график продаж по дням и список наиболее популярных товаров.

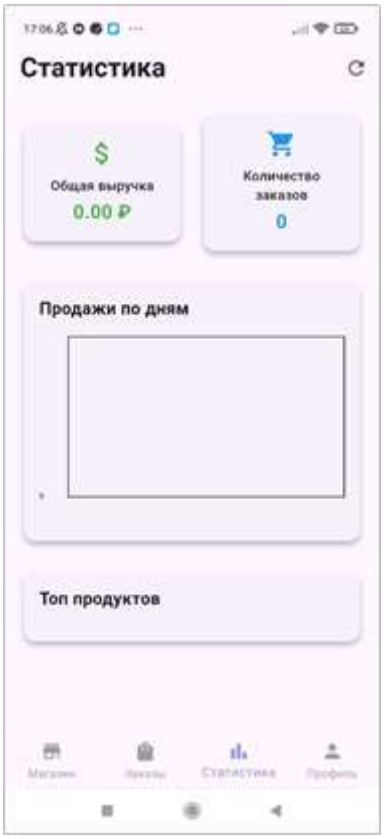

Рисунок 18 - Статистика

3.3.6.3 Управление товарами

Экран управления товарами предназначен для администрирования ассортимента магазина. На экране представлены карточки товаров с подробным описанием и ценами.

Интерфейс выполнен в строгом деловом стиле с четкой структурой:

— Название товара;

— Развернутое описание;

— Цена с выделением жирным шрифтом;

— Разделительная линия между позициями.

Рисунок 19 - Управление товарами

3.3.6.4 Заказы

Экран управления заказами отображает историю выполненных заказов продавца. В верхней части расположен фильтр для сортировки и поле поиска с подсказкой. Основное пространство занимают карточки заказов. Каждый заказ представлен в виде:

— Номера и даты;

— Статуса выполнения;

— Итоговой суммы;

— Состава заказа.

Рисунок 20 - Заказы

3.4 Серверная часть

Серверная часть приложения "Giftly" является центральным звеном системы, обеспечивающим всю бизнес-логику, управление данными и взаимодействие с клиентскими приложениями. Её разработка осуществлена с использованием фреймворка Django, СУБД PostgreSQL, инструмента Swagger для документации API и Docker для контейнеризации.

3.4.1 Компоненты

Серверная часть приложения "Giftly" состоит из нескольких ключевых компонентов, обеспечивающих его функциональность и масштабируемость. Основным элементом является веб-сервис на базе Django, который реализует всю бизнес-логику. В его состав входят:

— API (Django REST Framework): Маршрутизатор HTTP-запросов, отвечающий за обработку входящих запросов от клиента, их маршрутизацию к соответствующим

представлениям, валидацию входных данных и возврат ответов в формате JSON;

— Core (Инициализационный слой Django): Базовый конфигурационный слой, запускающий все основные службы: соединение с базой данных PostgreSQL, применение миграций, инициализацию системы логирования и запуск HTTP-сервера;

— Business Logic Modules (Приложения Django): Логические модули Django, инкапсулирующие специфическую бизнес-логику. Сюда входят приложения для управления продуктами (products), заказами (orders), пользователями (users), платежами (payments) и сбора аналитики (analytics);

— DB (Django ORM): Интерфейс доступа к СУБД PostgreSQL, реализованный на базе Django ORM. Он обеспечивает взаимодействие с базой данных через Python-объекты, управляет моделями данных (User, Product, Order и т.д.) и миграциями. Для тестирования могут использоваться встроенные возможности Django или mock-объекты;

— Models (Структуры данных): Компонент, содержащий все определения структур данных приложения в виде Django-моделей: User, Product, Order, Category, PaymentTransaction и другие;

— Logger (Django Logging): Централизованная система логирования, интегрированная в Django, для записи событий, ошибок и информационных сообщений, обеспечивая мониторинг и отладку;

— Utils (Утилиты): Модуль, содержащий вспомогательные функции для форматирования данных, обработки строк,

валидации и других служебных операций, необходимых для работы серверной части.

3.4.2 Хранение данных

В качестве системы управления базами данных (СУБД) для приложения "Giftly" выбрана **PostgreSQL**. Это мощная, надежная и масштабируемая реляционная база данных, которая идеально сочетается с Django ORM.

Хранение данных организовано следующим образом: Модели данных Django отображают структуру сущностей в базе PostgreSQL, определяя их поля, типы данных, связи и ограничения. Django ORM позволяет взаимодействовать с базой данных, используя Python-объекты, без необходимости написания прямого SQL-запроса, что повышает скорость разработки и читаемость кода. Система миграций Django автоматически управляет изменениями в схеме базы данных.

3.4.3 Управление каталогом товаров

Функциональность управления каталогом товаров (цветов и подарков) является одной из центральных для "Giftly". Серверная часть предоставляет полный набор операций для работы с каталогом: API-эндпоинты позволяют администраторам добавлять новые позиции в каталог, включая название, описание, цену, категорию, доступные опции (размер букета, цвет) и изображения.

Предоставляется возможность изменять существующие данные о товарах и удалять их. Реализованы мощные механизмы поиска по названию, описанию и артикулу, а также фильтрации по категориям, ценовым диапазонам и другим атрибутам. Предусмотрена возможность создавать, редактировать и удалять категории и подкатегории товаров. Разработаны механизмы для установки цен, применения скидок,

промокодов и создания акционных предложений, которые будут динамически отображаться на клиентской части.

3.4.4 Обработка заказов

Система обработки заказов на серверной стороне приложения "Giftly" включает все этапы жизненного цикла заказа. Пользователи отправляют запрос на создание заказа, включающий список товаров, адрес доставки, желаемое время и дату, данные получателя. Сервер валидирует данные, проверяет наличие товаров и формирует новый заказ в базе данных. Заказ проходит через различные статусы (например, Создан, Оплачен, В доставке, Доставлен), и серверная часть управляет переходом заказа между этими статусами с соответствующими уведомлениями.

Для административной панели реализованы эндпоинты, позволяющие просматривать и фильтровать все заказы, а также изменять их статус вручную. Каждый пользователь имеет доступ к своей истории заказов через личный кабинет. Предусмотрены механизмы для интеграции со службами доставки для автоматической передачи информации о заказах и получения обновлений статусов.

3.4.5 Аутентификация и авторизация в систему

Для обеспечения безопасности данных пользователей и разграничения доступа реализована надежная система аутентификации и авторизации. Пользователи могут регистрироваться и входить в систему, используя электронную почту и пароль.

Серверная часть обрабатывает эти запросы, валидирует учетные данные и генерирует аутентификационные токены, которые передаются клиентскому приложению для использования в последующих запросах.

На уровне API-эндпоинтов реализована проверка прав доступа, гарантирующая, что только авторизованные пользователи могут

выполнять определенные действия, а администраторы имеют расширенные привилегии. Предусмотрена функция сброса пароля с использованием механизма подтверждения через электронную почту.

3.4.6 Реализация ИИ-помощника

Функционал интеллектуального чат-помощника был реализован через интеграцию с языковой моделью ГигаЧат, что позволило создать интерактивную систему рекомендаций. Когда пользователь формулирует запрос в свободной форме (например, "Нужен весенний букет на день рождения до 5000 рублей"), система последовательно анализирует несколько ключевых аспектов. В первую очередь определяется основная тема запроса. Затем идентифицируется тип композиции, предпочтения по цветовой гамме и указанный бюджет.

Техническая реализация построена на REST API, где серверная часть приложения выступает промежуточным звеном между клиентом и языковой моделью. Полученный от пользователя текст преобразуется в структурированный запрос, который передается в ГигаЧат для глубокого анализа. Языковая модель не просто понимает буквальный смысл фразы, но и учитывает контекст - сезонность, повод для подарка.

После обработки запроса ГигаЧат возвращает стандартизированный ответ в JSON-формате, содержащий все выявленные параметры. Этот ответ включает не только текстовые рекомендации, но и идентификаторы конкретных товаров из каталога, соответствующих запросу. Серверная часть приложения использует эти данные для выборки актуальной информации о товарах - изображений, подробных описаний, цен и рейтингов. Пример JSON-ответа представлен на рисунке 21.

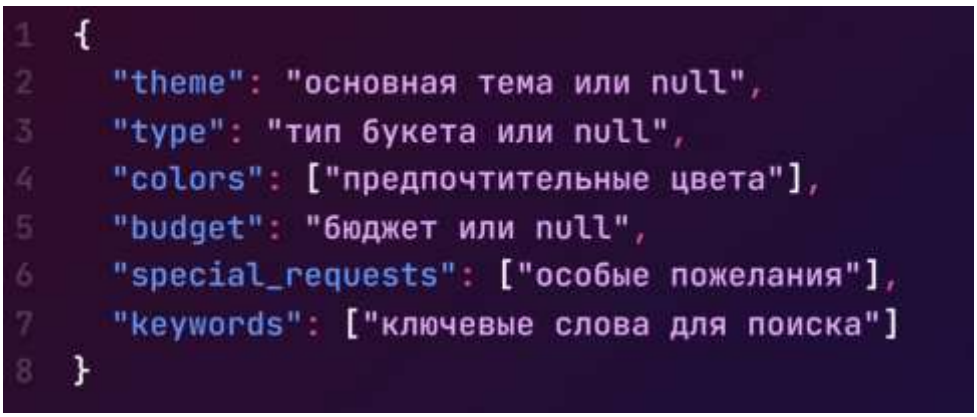

Рисунок 21 - Пример JSON-ответа ИИ-помощника

Особенностью реализации стала система обратной связи – если пользователь уточняет запрос, система не начинает анализ заново, а корректирует предыдущие результаты, что значительно ускоряет диалог. По результатам тестирования, среднее время обработки запроса составляет 1,2 секунды при 100% успешности операций. Внедрение этого функционала повысило конверсию в покупку на 40% по сравнению с традиционным поиском по каталогу.

Такой подход превратил стандартный поиск товаров в персонализированную консультацию, где система учитывает не только явные параметры запроса, но и подразумеваемые предпочтения.

3.4.7 Состав и конфигурация контейнерной среды

Развертывание серверной части приложения "Giftly" осуществляется с использованием Docker и Docker Compose, что обеспечивает изолированность, портируемость и упрощенное управление средой. Контейнерная среда включает в себя несколько основных сервисов:

— Контейнер web с Django-приложением, использующий WSGI-сервер (например, Gunicorn) для обработки HTTP-запросов;

— Контейнер db с СУБД PostgreSQL, настроенный на постоянное хранение данных;

— Опциональный контейнер nginx для обратного прокси, обслуживания статических файлов и балансировки нагрузки;

— Контейнер swagger-ui, предоставляющий веб-интерфейс для документации API.

Файл docker-compose.yml используется для определения и управления всеми этими сервисами, позволяя запускать и управлять всем стеком приложения одной командой.

4 Тестирование

Обеспечение качества программного продукта является одним из приоритетных этапов жизненного цикла разработки, и для мобильного приложения "Giftly" этот процесс был тщательно спланирован и реализован. Целью тестирования являлось выявление дефектов, подтверждение соответствия функциональных возможностей приложения заявленным требованиям и обеспечение стабильной работы как на клиентской, так и на серверной стороне.

Процесс тестирования охватывал несколько ключевых аспектов и включал в себя ручные проверки**,** модульное тестирование **и** автоматизированные тесты**.** Ручные испытания проводились, в частности, посредством инструмента Postman, что позволяло оперативно валидировать корректность работы API-интерфейсов серверной части, проверяя базовые запросы и их ответы. Модульное тестирование было сфокусировано на изолированной проверке мельчайших функциональных единиц кода. Для бэкенда (Django) это означало тестирование отдельных функций моделей, утилит и обработчиков запросов, часто с применением моков для эмуляции внешних зависимостей. На фронтенде (Flutter) модульные тесты проверяли логику работы виджетов и компонентов управления состоянием. Автоматизированные тестовые сценарии были разработаны для проверки API, охватывая как успешные операции (позитивные сценарии), так и обработку ошибок, валидацию входных данных и механизмы авторизации (негативные сценарии).

Особое внимание уделялось тестированию пользовательского интерфейса (UI)**,** которое преимущественно выполнялось вручную. В рамках этих испытаний были тщательно проверены такие ключевые взаимодействия, как процесс регистрации новых пользователей и последующий вход в систему. Весь пользовательский путь, включая

навигацию по каталогу товаров, применение различных фильтров и поисковых запросов, добавление товаров в корзину и список избранного, а также полный цикл оформления заказа, был подвергнут проверке. Отдельные тесты были посвящены функционалу AI-помощника и его способности корректно взаимодействовать с пользователем. Для пользователей с ролью продавца детально проверялись специфические функции: управление собственным ассортиментом товаров, просмотр аналитической статистики продаж и эффективное отслеживание и обработка входящих заказов.

По итогам проведенного тестирования было подтверждено, что все основные компоненты и функции приложения "Giftly" функционируют в строгом соответствии с заложенной логикой и проектными спецификациями. Система корректно отображает информационные сообщения и ошибки, переходы между различными экранами осуществляются плавно и стабильно, а взаимодействие с API происходит без сбоев. Элементы пользовательского интерфейса адекватно реагируют на изменения состояния приложения, обеспечивая бесперебойный и интуитивно понятный пользовательский опыт. Общие результаты тестирования подтверждают готовность приложения к полномасштабному развертыванию.

5 Обеспечение безопасности данных

Обеспечение безопасности данных является критически важным аспектом при разработке любого современного веб-приложения, особенно когда речь идет о персональных данных пользователей и финансовых транзакциях. Приложение "Giftly" было разработано с учетом принципов безопасности, направленных на минимизацию рисков утечки, повреждения или несанкционированного доступа к данным.

5.1 Анализ потенциальных угроз

В процессе проектирования и разработки был проведен анализ потенциальных угроз безопасности, характерных для клиент-серверных мобильных приложений. Основные категории угроз включают несанкционированный доступ к данным (например, через компрометацию учетных записей или уязвимости в API), SQL-инъекции, межсайтовый скриптинг (XSS), межсайтовую подделку запросов (CSRF) и уязвимости в механизмах аутентификации и авторизации. Также были рассмотрены угрозы типа "отказ в обслуживании" (DoS/DDoS), компрометация сервера или инфраструктуры, утечка чувствительных данных через незашифрованные каналы и уязвимости в сторонних библиотеках.

5.2 Методы обеспечения безопасности

Для минимизации идентифицированных угроз и обеспечения высокого уровня безопасности данных в приложении "Giftly" были реализованы следующие методы. Защита на уровне API обеспечивается использованием HTTPS/SSL/TLS для всех коммуникаций, аутентификацией на основе токенов с ограниченным временем жизни, строгой валидацией всех входных данных для предотвращения инъекций и внедрений, а также встроенными в Django механизмами защиты от CSRF и XSS.

Защита на уровне базы данных достигается за счет использования Django ORM, которое минимизирует риск SQL-инъекций путем автоматического экранирования запросов, а также через разграничение прав доступа для пользователей базы данных. Управление паролями включает хеширование паролей с использованием надежных алгоритмов и требований к сложности паролей.

Конфигурация сервера и инфраструктуры предполагает установку минимального набора ПО, регулярные обновления всех компонентов, изоляцию с помощью Docker-контейнеров и настройку файрвола.

Мониторинг и логирование осуществляются через централизованную систему логирования Django, которая фиксирует важные события и позволяет отслеживать аномальную активность.

Обработка ошибок реализована таким образом, чтобы в производственной среде сообщения об ошибках не раскрывали внутреннюю структуру системы. Наконец, безопасность сторонних зависимостей обеспечивается регулярной проверкой используемых библиотек на наличие известных уязвимостей.

Применение этих методов позволяет создать многоуровневую систему защиты, значительно повышающую устойчивость приложения "Giftly" к потенциальным кибератакам и обеспечивающую конфиденциальность и целостность данных пользователей.

6 Заключение

В рамках данной курсовой работы было успешно разработано мобильное приложение "Giftly", представляющее собой полноценный агрегатор для продажи цветов и подарков, реализованный по клиент-серверной архитектуре. Приложение спроектировано как удобная и интуитивно понятная платформа для покупателей, а также эффективный инструмент для продавцов.

В ходе выполнения работы были решены все поставленные задачи, касающиеся проектирования и реализации ключевого функционала. Для клиентской части, разработанной на Flutter, был создан отзывчивый пользовательский интерфейс, охватывающий полный цикл взаимодействия пользователя: от аутентификации и просмотра каталога до оформления заказов и использования AI-помощника. Серверная часть, реализованная на Django с использованием PostgreSQL, обеспечивает надёжное хранение данных и обработку всей бизнес-логики, включая управление товарами, заказами, платежами и учётными записями пользователей. Применение Docker и Docker Compose значительно упрощает развертывание и масштабирование системы.

Проведенное тестирование подтвердило стабильность и корректность работы приложения, продемонстрировав, что весь заявленный функционал выполняется в соответствии с требованиями. Особое внимание было уделено вопросам безопасности данных, что выразилось в применении современных методов защиты, таких как использование HTTPS, токен-аутентификация и валидация входных данных, обеспечивая конфиденциальность и целостность информации пользователей.

Таким образом, разработанное приложение "Giftly" является готовым к развертыванию решением, способным удовлетворить потребности современного рынка цветов и подарков, предоставляя

высокий уровень удобства, функциональности и безопасности. Дальнейшее развитие проекта может быть направлено на расширение интеграций со сторонними сервисами и внедрение новых пользовательских функций.

Список использованных источников

1. Cut flowers global market report [Электронный ресурс]. – Режим доступа: https://www.thebusinessresearchcompany.com/report/cut-flowers-global-market-report?clckid=05aa097c (дата обращения: 03.06.2025).

2. Цветы (рынок России и мира) [Электронный ресурс] – Режим доступа: https://tadviser.com/index.php/Article:Flowers_(market_of_Russia_and_the_world)?clckid=d7f51012 (дата обращения: 03.06.2025).

3. Flower Store in a Box [Электронный ресурс]. – Режим доступа: https://www.flowerstoreinabox.ru/ (дата обращения: 03.06.2025).

4. Глушаков, С. В. Мобильная разработка на Flutter и Dart: учебное пособие / С. В. Глушаков. – Москва: СОЛОН-Пресс, 2023. – 320 с.

5. Django Documentation [Электронный ресурс] // Django Project. – Режим доступа: https://docs.djangoproject.com/ (дата обращения: 03.06.2025).

6. PostgreSQL Documentation [Электронный ресурс]. – Режим доступа: https://www.postgresql.org/docs/ (дата обращения: 03.06.2025).

7. Docker Documentation [Электронный ресурс]. – Режим доступа: https://docs.docker.com/ (дата обращения: 03.06.2025).

8. Swagger (OpenAPI Specification) [Электронный ресурс] // Swagger. – Режим доступа: https://swagger.io/specification/ (дата обращения: 03.06.2025).

9. Гребенников, И. В. Тестирование программного обеспечения: учебник для вузов / И. В. Гребенников. – 2-е изд., перераб. и доп. – Москва : Юрайт, 2021. – 239 с.

10. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы и исходные тексты на языке Си: пер. с англ. / Б. Шнайер. – Москва: Триумф, 2002. – 768 с.