**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (ВлГУ)

Кафедра ВТиСУ

**КУРСОВАЯ РАБОТА**

по дисциплине «Разработка мобильных приложений»

на тему «Разработка мобильных приложений на платформе Android. Часть 14. Разработка мобильного приложения для магазина автозапчастей»

Студент: Пупанов К.Ю.

Институт: ИИТЭ

Направление: 09.03.03

Группа: ПИ-121

Руководитель: Шутов А.В.

Владимир 2024 г.

Оглавление

[**ВВЕДЕНИЕ** 3](#_Toc185906471)

[**ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ** 4](#_Toc185906472)

[1.1. Роль и место мобильных приложений в современном обществе 4](#_Toc185906473)

[1.2. Анализ популярных мобильных платформ 6](#_Toc185906474)

[1.3. Современные архитектуры разработки мобильных приложений 7](#_Toc185906475)

[**ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ** 14](#_Toc185906476)

[2.1. Постановка задачи 14](#_Toc185906477)

[2.1.1. Краткая характеристика организации 14](#_Toc185906478)

[2.1.2. Описание состава задач 14](#_Toc185906479)

[2.2. Обоснование выбора программных средств 15](#_Toc185906480)

[2.3. Краткое руководство пользователя 19](#_Toc185906481)

[**ЗАКЛЮЧЕНИЕ** 30](#_Toc185906482)

[**СПИСОК ЛИТЕРАТУРЫ** 32](#_Toc185906483)

# **ВВЕДЕНИЕ**

Современное развитие технологий оказывает значительное влияние на различные отрасли бизнеса, включая розничную торговлю. С увеличением популярности мобильных устройств и ростом интернет-торговли магазины стремятся адаптироваться к новым условиям, чтобы оставаться конкурентоспособными. Одним из эффективных инструментов для достижения этой цели является разработка мобильных приложений, которые обеспечивают удобство для пользователей и автоматизацию ряда процессов для компаний.

В условиях насыщенного рынка автозапчастей удобство взаимодействия с клиентами играет ключевую роль. Покупатели ожидают возможности быстро найти необходимые детали, проверить их наличие, сравнить цены и оформить заказ без необходимости посещения физического магазина.

Целью данной курсовой работы является разработка мобильного приложения на платформе Android для магазина автозапчастей. Приложение должно обеспечить удобный интерфейс для заказа автозапчастей, а также интеграцию с базой данных для хранения информации о товарах и заказах. В рамках работы будут рассмотрены следующие аспекты:

* Изучить современные подходы к разработке мобильных приложений.
* Провести сравнительный анализ популярных мобильных платформ.
* Определить архитектуру и инструменты для создания мобильного приложения.
* Разработать функциональную спецификацию будущего приложения.
* Подготовить краткое руководство пользователя для разработанного приложения.

Актуальность темы обусловлена растущей потребностью в цифровизации бизнес-процессов, а также высоким спросом на мобильные приложения, которые облегчают повседневные задачи потребителей. Разработка данного приложения позволит повысить конкурентоспособность магазина автозапчастей, расширить охват аудитории и упростить процесс взаимодействия с клиентами.

# **ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

## 1.1. Роль и место мобильных приложений в современном обществе

В современном обществе мобильные приложения играют ключевую роль в повседневной жизни людей. Они охватывают широкий спектр задач — от общения и развлечений до работы и обучения. Стремительное развитие мобильных технологий способствует трансформации традиционных бизнес-моделей и созданию новых форм взаимодействия с потребителями.

Одним из главных преимуществ мобильных приложений является их доступность. Благодаря широкому распространению смартфонов и стабильному интернет-соединению пользователи могут оперативно получать доступ к необходимым услугам и информации. Это позволяет компаниям предоставлять клиентам персонализированный опыт и удерживать их внимание на своих продуктах и сервисах.

Мобильные приложения играют важную роль в цифровой экономике. Они способствуют развитию электронной коммерции, автоматизации процессов и улучшению качества обслуживания. Например, приложения позволяют оптимизировать процессы заказа товаров, отслеживания поставок, взаимодействия с клиентами и анализа данных о поведении пользователей.

Мобильные приложения находят применение в различных областях:

1. **Здравоохранение**:
   * Приложения для мониторинга здоровья, такие как трекеры физической активности и контроля параметров тела.
   * Телемедицина, предоставляющая возможность удалённых консультаций с врачами.
   * Управление расписанием приёма лекарств и напоминания о медицинских процедурах.
2. **Образование**:
   * Платформы для дистанционного обучения, такие как электронные учебники и курсы.
   * Интерактивные приложения для детей, способствующие развитию навыков и знаний.
   * Видеоконференции и онлайн-классы для взаимодействия студентов и преподавателей.
3. **Транспорт**:
   * Приложения для вызова такси, бронирования билетов на поезд или авиаперелёты.
   * Навигационные программы с функцией планирования маршрутов и мониторинга трафика.
   * Каршеринг и аренда велосипедов.
4. **Финансы**:
   * Онлайн-банкинг, позволяющий пользователям управлять своими счетами и проводить операции.
   * Инструменты для управления личными финансами, такие как бюджетирование и учёт расходов.
   * Платёжные системы для быстрых и удобных транзакций.
5. **Развлечения**:
   * Стриминговые платформы для музыки, фильмов и сериалов.
   * Игровые приложения, охватывающие как казуальные, так и профессиональные игры.
   * Программы для чтения книг и прослушивания аудиокниг.
6. **Коммерция**:
   * Приложения для интернет-магазинов, предоставляющие пользователям возможность покупки товаров онлайн.
   * Программы для сравнения цен и поиска скидок.
   * Сервисы доставки еды и товаров повседневного спроса.

Таким образом, мобильные приложения активно интегрируются в различные аспекты жизни современного человека, предоставляя широкий спектр возможностей для повышения комфорта и оптимизации повседневных процессов.

## 1.2. Анализ популярных мобильных платформ

Современный рынок мобильных приложений представлен двумя основными платформами, каждая из которых обладает своими особенностями и преимуществами:

1. **Android**:
   * **Преимущества**:
     + Самая популярная операционная система в мире, обеспечивающая широкий охват аудитории.
     + Поддержка разнообразных устройств, от бюджетных смартфонов до премиальных моделей.
     + Открытый исходный код, что предоставляет разработчикам гибкость в создании приложений.
     + Наличие Google Play Market как основной платформы для распространения приложений с мощными инструментами монетизации и аналитики.
     + Доступность бесплатной среды разработки (Android Studio) и обширной документации.
   * **Недостатки**:
     + Высокая фрагментация устройств, что может усложнить оптимизацию и тестирование приложений.
     + Повышенные требования к обеспечению безопасности, поскольку открытость платформы делает её уязвимой для атак.
     + Конкуренция среди приложений в Google Play Market, что требует дополнительных усилий по продвижению.
2. **iOS**:
   * **Преимущества**:
     + Высокий уровень безопасности и стабильности благодаря строгому контролю со стороны Apple.
     + Однородность устройств, что упрощает процесс оптимизации и тестирования приложений.
     + Престижность платформы, ориентированной на пользователей с высоким уровнем дохода.
     + Качественный пользовательский интерфейс и опыт.
   * **Недостатки**:
     + Ограниченный рынок, поскольку iOS доступна только на устройствах Apple, которые менее распространены, чем Android.
     + Высокие требования к разработке, включая необходимость использования платных инструментов и лицензий.
     + Строгие правила модерации в App Store, что может замедлить публикацию приложений.

В рамках данной курсовой работы выбран именно Android как наиболее универсальная платформа, предоставляющая оптимальный баланс между функциональностью и масштабируемостью.

Выбор платформы для разработки зависит от целей и аудитории приложения. В случае данной работы Android был выбран из-за его широкого распространения, доступности инструментов и возможности охвата максимального числа пользователей.

## 1.3. Современные архитектуры разработки мобильных приложений

С развитием мобильных технологий и постоянным усложнением мобильных приложений, архитектурные подходы к их разработке также претерпели значительные изменения. Современные архитектуры играют ключевую роль в создании гибких, масштабируемых и легко поддерживаемых приложений. Архитектуры помогают структурировать код таким образом, чтобы приложение было не только функциональным, но и легко поддающимся тестированию, поддержке и расширению.

**Монолитная архитектура** представляет собой традиционный подход к разработке программного обеспечения, в котором все компоненты приложения собраны в одном едином блоке. Это означает, что все части приложения, включая бизнес-логику, интерфейс и взаимодействие с базой данных, разрабатываются и разворачиваются как единое целое.

В контексте мобильных приложений это может означать, что вся функциональность — от пользовательского интерфейса до обработки данных — находится в одном коде, который взаимодействует с одной серверной частью или одним веб-сервисом.

**Микросервисная архитектура** предлагает разделение приложения на небольшие, независимые сервисы, каждый из которых выполняет свою задачу. Каждый микросервис взаимодействует с другими через четко определенные API. Эти сервисы могут быть разработаны, развернуты и масштабированы отдельно, что позволяет повысить гибкость системы.

Для мобильных приложений это может означать, что серверная часть разделена на несколько микросервисов, которые отвечают за конкретные задачи, такие как обработка платежей, управление пользователями, обработка заказов и другие. Каждый микросервис может быть написан с использованием разных технологий, если это необходимо.

**Клиент-серверная архитектура** является одной из самых распространенных моделей взаимодействия, когда мобильное приложение (клиент) обменивается данными с сервером через сеть. В такой архитектуре сервер отвечает за обработку запросов, хранение данных и выполнение бизнес-логики, а клиент (мобильное приложение) предоставляет пользовательский интерфейс и взаимодействует с сервером для получения или отправки данных.

Мобильное приложение для магазина автозапчастей будет работать по модели клиент-сервер, где клиент — это само приложение на Android, а сервер будет отвечать за такие операции, как поиск автозапчастей, обработка заказов, управление данными пользователей и т. д.

Мобильная разработка на платформе Android также не стоит на месте, и с каждым годом появляются новые подходы и инструменты, направленные на упрощение процесса разработки. Сегодня на практике используются несколько наиболее популярных архитектурных паттернов, каждый из которых имеет свои особенности, преимущества и недостатки. В этом разделе мы рассмотрим наиболее популярные из них, чтобы лучше понять, какой из них может быть оптимальным выбором для разработки мобильного приложения для магазина автозапчастей.

**MVC (Model-View-Controller)**

Одним из самых старых и одновременно популярных подходов в разработке является архитектура **Model-View-Controller (MVC)**. Этот паттерн разделяет приложение на три основных компонента:

* **Model (Модель)** — компонент, который отвечает за данные и логику приложения. Модель работает с данными, обрабатывает их и выполняет все операции, связанные с бизнес-логикой. В контексте магазина автозапчастей, модель может представлять товары, их описание, наличие на складе и другие характеристики.
* **View (Представление)** — компонент, который отображает данные пользователю. В случае мобильных приложений это интерфейс, с которым взаимодействует пользователь. View ответственен за отображение информации на экране в удобном для восприятия виде.
* **Controller (Контроллер)** — компонент, который связывает модель и представление. Контроллер обрабатывает действия пользователя (например, нажатия кнопок или прокрутку списка) и управляет тем, какие данные модель должна предоставить для отображения в представлении.

Этот подход стал основой многих первых приложений и до сих пор находит свое применение в простых проектах, где нет сложных взаимодействий между компонентами.

**Преимущества:**

* Простота реализации, что делает эту архитектуру хорошим выбором для небольших и несложных приложений.
* Легкость в поддержке на начальных этапах разработки.

**Недостатки:**

* С увеличением сложности приложения контроллер может быстро превратиться в слишком перегруженную сущность, что делает его трудным для масштабирования и тестирования.
* Не слишком подходит для приложений, требующих сложной логики взаимодействия между компонентами.

**MVP (Model-View-Presenter)**

Архитектура MVP (Model-View-Presenter) является развитием концепции MVC и помогает разделить логику приложения более эффективно. В отличие от MVC, в MVP между моделью и представлением вставляется третий компонент — презентер.

Model (Модель) — компонент, который по-прежнему отвечает за хранение данных и реализацию бизнес-логики.

View (Представление) — отвечает исключительно за отображение данных пользователю. В отличие от MVC, представление в MVP не должно содержать никакой логики обработки данных или взаимодействия с моделью.

Presenter (Презентер) — компонент, который связывает модель и представление. Презентер получает данные от модели и обновляет представление с учетом этих данных. Он также обрабатывает пользовательские действия, такие как нажатия кнопок, и решает, что должно быть отображено.

MVP архитектура часто используется в Android-разработке, так как она позволяет легче тестировать и масштабировать приложения. Презентер можно тестировать независимо от UI, что значительно упрощает процесс обеспечения качества кода.

Таким образом, выбор архитектуры для разработки мобильного приложения зависит от множества факторов, таких как размер проекта, требования к тестированию и поддерживаемости, а также опыт команды разработчиков.

Для данного проекта была выбрана клиент-серверная архитектура.

Выбор **клиент-серверной архитектуры** для разработки мобильного приложения для магазина автозапчастей имеет несколько обоснованных причин. Рассмотрим основные из них:

1. Централизованное управление данными

В клиент-серверной архитектуре все данные (например, информация о товарах, заказах, клиентах и т. д.) хранятся на сервере. Это упрощает процесс управления и защиты данных, так как централизованное хранение позволяет легко внедрять механизмы безопасности, резервного копирования и восстановления данных. В случае с магазином автозапчастей данные о товарах и наличии на складе могут часто обновляться, и сервер обеспечит актуальность этих данных для всех пользователей.

2. Простота в масштабировании

Мобильное приложение может обслуживать большое количество пользователей, особенно если оно станет популярным. В клиент-серверной архитектуре серверная часть легко масштабируется, что позволяет обслуживать большее количество запросов от клиентов без значительных изменений в клиентской части приложения. В случае увеличения числа пользователей можно расширять серверные мощности (например, путем использования облачных решений) для повышения производительности.

3. Независимость клиента и сервера

Клиент-серверная архитектура позволяет разделить логику работы приложения на клиентскую и серверную части. Это облегчает поддержку и развитие каждой из них. Например, можно обновлять мобильное приложение на Android, не затрагивая серверную часть, и наоборот, улучшать серверные процессы, не влияя на приложение. В рамках мобильного приложения для автозапчастей сервер будет обрабатывать запросы на поиск и оформление заказов, а клиент будет лишь отображать эти данные и взаимодействовать с пользователем.

4. Обеспечение безопасности

С серверной архитектурой легко внедрить современные технологии защиты данных, такие как шифрование и аутентификация. В приложении магазина автозапчастей необходимо обеспечить безопасность пользовательских данных, таких как информация о платёжных картах и личные данные. Сервер может контролировать все операции с чувствительной информацией, гарантируя, что данные не будут храниться на мобильных устройствах пользователей, что повышает уровень безопасности.

5. Легкость в интеграции с внешними сервисами

Мобильное приложение может потребовать интеграции с различными внешними сервисами — например, для обработки платежей, получения информации о доставке, обновлений базы данных автозапчастей и т. д. В клиент-серверной архитектуре сервер может выступать в роли посредника, обеспечивая удобное взаимодействие с этими сервисами и минимизируя нагрузку на клиентскую часть приложения.

6. Поддержка широкого спектра устройств

С использованием клиент-серверной архитектуры мобильное приложение для магазина автозапчастей может поддерживать не только Android-устройства, но и другие платформы, такие как iOS или веб-приложения. Сервер будет предоставлять API, которые могут быть использованы различными клиентами, обеспечивая единый подход к обработке данных независимо от устройства, с которого осуществляется доступ.

7. Простота в тестировании и отладке

С разделением логики на клиентскую и серверную части, тестирование и отладка приложения становятся проще. Команды разработки могут работать над улучшением и тестированием как клиентской, так и серверной частей независимо друг от друга. Это ускоряет процесс разработки и позволяет быстро выявлять и устранять ошибки.

# **ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ**

## 2.1. Постановка задачи

### 2.1.1. Краткая характеристика организации

Название бренда: АвтоПланета

Сфера деятельности: Интернет-магазин автозапчастей

Краткое описание: Компания находится на рынке более 15 лет. Конкурентное преимущество обусловлено обширным ассортиментом товаров – начиная от клипс для обшивок дверей, заканчивая сложными агрегатами в составе ДВС и адекватными ценами.

Перечень продаваемых товаров: Ступичный подшипник, масляный насос, маслосъёмные колпачки, гидрокомпенсаторы, прокладка ГБЦ, термостат и тд..

Каналы продвижения: Онлайн продвижение с помощью добавления компании на карты «Яндекс» «Google», размещение таргетинговой рекламы в соц.сетях и на схожих площадках, email рассылка, публикация объявлений на «Авито» и «Юла».

Цели и задачи, которые планируется достичь посредством разрабатываемого приложения: Цель прииложения наладить онлайн продажу с возможностью заказа товара на всей территории РФ.

Целевая аудитория: Все автовладельцы, независимо от возраста, социального и материального положения.

### 2.1.2. Описание состава задач

Для разработки мобильного приложения для магазина автозапчастей необходимо решить несколько ключевых задач, которые будут обеспечивать его функциональность и удобство для пользователей. Рассмотрим основные из них:

**1. Авторизация и регистрация**

Одной из самых важных задач для обеспечения безопасности пользователей является создание системы авторизации и регистрации. Это позволит пользователям создавать аккаунты, входить в них и получать доступ к персонализированным данным, таким как история заказов, сохраненные товары и данные профиля.

**2. Каталог товаров**

Каталог товаров — это основная часть приложения, которая будет отображать весь ассортимент автозапчастей, доступных для покупки. Важно обеспечить удобный и интуитивно понятный интерфейс для поиска и фильтрации товаров, чтобы пользователи могли легко найти необходимые детали.

**3. Карточка товара**

Карточка товара — это подробное описание каждой автозапчасти, которое помогает пользователю принять решение о покупке. Эта задача включает в себя создание интерфейса с полным описанием товара и его характеристик.

**4. Интеграция с сервером**

Для каждой из этих задач необходима интеграция с серверной частью приложения. Сервер будет обрабатывать запросы от клиента, такие как:

Сервер будет также обеспечивать безопасность и защиту данных (например, защищенные каналы для передачи паролей и личной информации), а также управление сессиями пользователей для предотвращения несанкционированного доступ

Реализовав вышеперечисленные, мы создали удобное и понятное мобильное приложение, который упростит жизнь автолюбителей.

## 2.2. Обоснование выбора программных средств

Для разработки мобильного приложения для магазина автозапчастей было выбрано несколько программных средств и технологий, которые оптимально подходят для реализации всех необходимых функциональных возможностей, а также обеспечивают гибкость, масштабируемость и удобство разработки. Рассмотрим выбор каждого из используемых инструментов:

**1. Android Studio**

**Android Studio** является официальной интегрированной средой разработки (IDE) для разработки приложений на платформе Android. Это инструмент, который предлагает все необходимые средства для разработки мобильных приложений на Android, включая редактор кода, эмулятор, инструменты для профилирования и отладки, а также интеграцию с системами контроля версий.

**Обоснование выбора:**

* **Поддержка Android-платформы:** Android Studio полностью поддерживает все особенности и API Android, включая последние версии операционной системы, что позволяет разрабатывать приложения с использованием самых современных возможностей платформы.
* **Инструменты для работы с UI:** Android Studio предоставляет удобные инструменты для проектирования пользовательского интерфейса, включая визуальный редактор и систему макетов, что значительно ускоряет процесс разработки.
* **Поддержка Java и Kotlin:** Android Studio поддерживает как Java, так и Kotlin, что дает возможность гибко выбирать подходящий язык для реализации бизнес-логики приложения.
* **Интеграция с Google Services:** Android Studio обеспечивает удобную интеграцию с различными сервисами Google (например, Google Firebase, Google Analytics), что может быть полезно для аналитики и улучшения взаимодействия с пользователями.

**2. XAMPP**

**XAMPP** — это комплект программного обеспечения, включающий сервер Apache, систему управления базами данных MySQL (или MariaDB), интерпретатор PHP и другие компоненты, необходимые для разработки и тестирования веб-приложений.

**Обоснование выбора:**

* **Локальная разработка серверной части:** XAMPP позволяет разрабатывать и тестировать серверную часть приложения на локальном сервере без необходимости установки сложных конфигураций. Это удобно на этапе разработки, так как позволяет быстро тестировать и отлаживать серверные запросы и API.
* **Скорость развертывания:** С помощью XAMPP можно быстро развернуть среду для разработки серверной части на локальной машине, что ускоряет процесс разработки.
* **Простота использования:** XAMPP является простым в установке и настройке, что делает его идеальным инструментом для начинающих и небольших проектов.

**3. Java**

**Java** — это объектно-ориентированный язык программирования, который используется для разработки Android-приложений. Он является основным языком для мобильной разработки на платформе Android и поддерживается официально.

**Обоснование выбора:**

* **Поддержка Android:** Java является одним из основных языков для разработки приложений на Android, и она имеет большую документацию и сообщество, что помогает решать возникающие проблемы.
* **Производительность и стабильность:** Java известна своей стабильностью и эффективностью в работе с большим объемом данных, что важно для серверной логики приложения, например, для обработки информации о товарах и заказах.
* **Большая база библиотек:** Java предоставляет обширные библиотеки для работы с сетью, базами данных, а также с пользовательскими интерфейсами, что упрощает процесс разработки.

**4. PHP**

**PHP** — это серверный язык программирования, который широко используется для разработки динамических веб-сайтов и серверных приложений. Он активно используется для разработки API и взаимодействия с базой данных.

**Обоснование выбора:**

* **Легкость в разработке веб-сервисов:** PHP идеально подходит для создания RESTful API, которые будут использоваться мобильным приложением для взаимодействия с сервером. Он прост в настройке и интеграции с базой данных MySQL.
* **Широкая поддержка и документация:** PHP имеет большое сообщество и многочисленные фреймворки (например, Laravel), что позволяет быстро реализовать функционал для взаимодействия с базой данных и обработки запросов от клиента.
* **Совместимость с MySQL:** PHP отлично интегрируется с MySQL, что позволяет эффективно обрабатывать запросы к базе данных и хранить информацию о товарах, пользователях и заказах.

**5. MySQL**

**MySQL** — это одна из самых популярных реляционных систем управления базами данных (СУБД), которая используется для хранения и управления данными в различных приложениях.

**Обоснование выбора:**

* **Надежность и производительность:** MySQL обладает высокой производительностью при работе с большими объемами данных, что делает ее идеальной для хранения информации о товарах, заказах и пользователях в приложении.
* **Широкая распространенность и поддержка:** MySQL имеет большую популярность и поддержку, что позволяет быстро решать возникающие вопросы, а также использовать различные инструменты для работы с данными (например, phpMyAdmin).
* **Совместимость с PHP и XAMPP:** MySQL прекрасно интегрируется с PHP и XAMPP, что позволяет создавать эффективную серверную часть для мобильного приложения и обеспечивать быстрые запросы к базе данных.

**6. Visual Studio Code**

**Visual Studio Code (VS Code)** — это легкий и мощный текстовый редактор, который используется для разработки кода на различных языках программирования. Он поддерживает плагины и расширения для работы с HTML, CSS, JavaScript, PHP, а также с другими языками.

**Обоснование выбора:**

* **Многофункциональность:** VS Code — это универсальный редактор, который можно использовать для разработки как серверной части (на PHP), так и для работы с фронтендом (HTML, CSS, JavaScript).
* **Поддержка расширений:** В VS Code есть множество расширений для работы с базами данных, интеграции с Git, а также для работы с различными фреймворками и библиотеками.
* **Простота и скорость:** Этот редактор очень быстрый и удобный, что позволяет эффективно работать над проектами, не тратя время на сложные настройки.

Использование **Android Studio**, **XAMPP**, **Java**, **PHP**, **MySQL** и **Visual Studio Code** для разработки мобильного приложения магазина автозапчастей обеспечит оптимальные условия для создания высококачественного, масштабируемого и безопасного приложения. Каждый из выбранных инструментов и технологий играет ключевую роль в реализации всех функциональных задач приложения, от работы с базой данных и серверной логикой до создания интерфейса для удобной работы пользователей с каталогом товаров и процессом оформления заказов.

## 2.3. Краткое руководство пользователя

При запуске приложения, пользователь попадает на окно авторизации рис. 1. Чтобы попасть в каталог, нужно ввести логин и пароль.

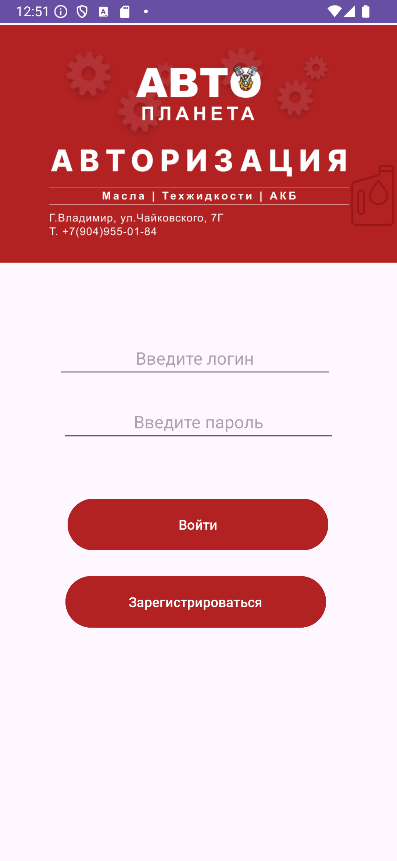


Рис. 1 - Авторизация

Если аккаунт существует, то пользователь попадает в каталог. Если аккаунта нет, то требуется пройти регистрацию Рис. 2.

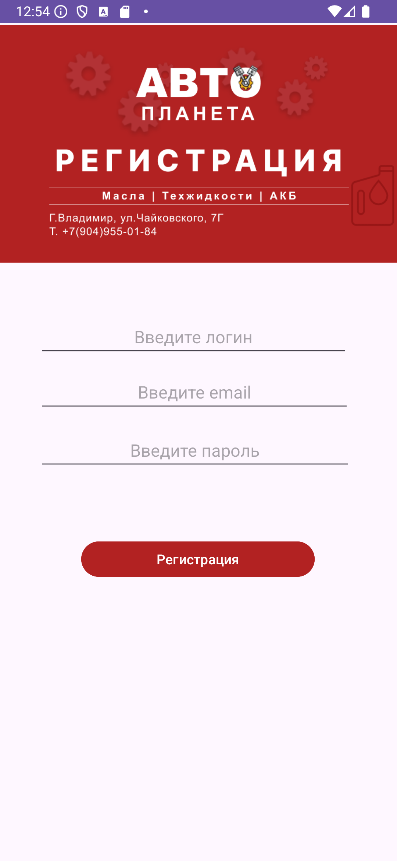


Рис. 2 – Регистрация

После авторизации пользователь попадает в каталог с товарами Рис. 3. 

Рис. 3 – Каталог товаров

В каталоге пользователь может увидеть все имеющиеся в наличии запчасти, с фотографиями и ценами. При нажатии на интересующий товар, пользователю открывается карточка товара, где так же можно увидеть фотографию товара, его цену и описание.



Рис. 4 – Карточка товара

Для добавления товара в корзину, пользователю нужно нажать кнопку «Добавить в корзину». Это можно сделать, как из каталога, так и из карточки товара. Когда пользователь выбрал нужные запчасти и добавил их в корзину, с помощью меню навигации, расположенного внизу экрана, пользователь может перейти в корзину для дальнейшего оформления заказа.

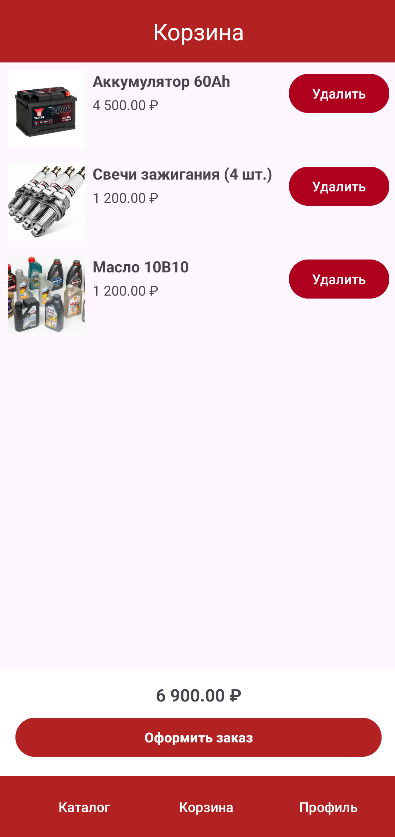


Рис. 5 – Корзина

В корзине находятся все товары, которые пользователь добавил из каталога. Снизу над кнопкой «Оформить заказ» показана сумма всех товаров в корзине. Если пользователь передумал насчет какого-то из товаров, то с помощью кнопки «Удалить», пользователь может убрать товар из корзины.

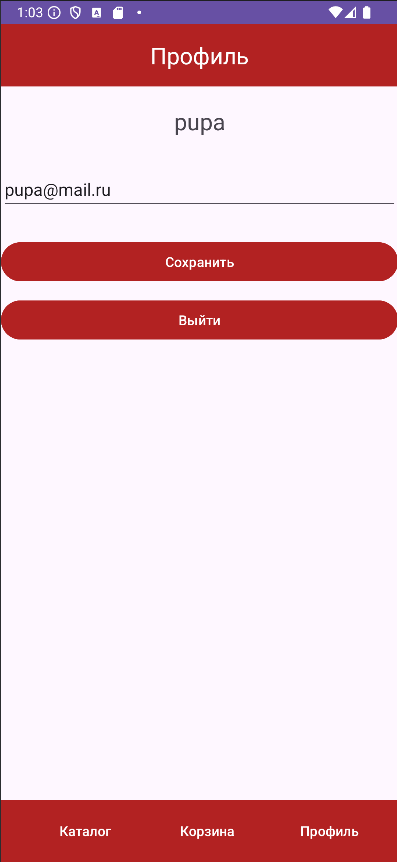


Рис. 6 – Профиль

Так же в меню навигации пользователь может перейти в профиль. В этом окне пользователь может посмотреть свой логин, изменить E-Mail адрес и выйти из аккаунта.

Для добавления, изменения, удаления товаров, требуется авторизоваться в админ-аккаунт. Логин admin, пароль admin. После авторизации в админ-аккаунт, перед пользователем открывается страница, в которой можно взаимодействовать с наполнением каталога.

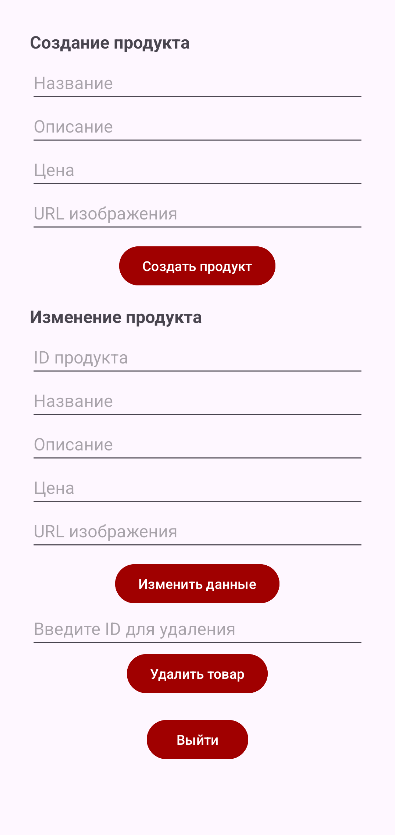


Рис. 7 – Админ-аккаунт

# **ЗАКЛЮЧЕНИЕ**

В процессе разработки мобильного приложения для магазина автозапчастей были решены ключевые задачи, связанные с созданием функционального и удобного интерфейса для пользователей, а также эффективной серверной частью, обеспечивающей обработку данных и выполнение операций. В ходе работы был выбран оптимальный набор технологий и инструментов, соответствующий требованиям проекта и обеспечивающий его успешную реализацию.

Для разработки клиентской части приложения была использована **Android Studio**, что позволило создать современное и удобное приложение для платформы Android с использованием возможностей, предоставляемых этой IDE. В качестве языка программирования для мобильного приложения был выбран **Java**, который является основным для разработки на Android, что гарантирует стабильную и производительную работу приложения.

Для разработки серверной части приложения был выбран **XAMPP**, который обеспечил простоту настройки локального сервера для тестирования и разработки, а также **PHP** для создания API, которое обеспечивало взаимодействие между клиентской частью и базой данных. **MySQL** был использован для хранения и обработки данных о товарах и пользователях, что гарантирует надежность и высокую производительность работы с базой данных.

Одним из ключевых аспектов разработки было создание функционала авторизации и регистрации, а также каталога товаров с детализированными карточками и возможностью добавления товаров в корзину. Это сделало приложение удобным и интуитивно понятным для пользователей, обеспечив высокую степень удовлетворенности.

Использование **клиент-серверной архитектуры** позволило разделить логику приложения на клиентскую и серверную части, что упростило процесс разработки и тестирования, а также обеспечило масштабируемость и безопасность.

В результате проведенной работы было разработано мобильное приложение, которое отвечает всем требованиям функциональности, безопасности и удобства использования. Полученные результаты демонстрируют успешность выбранной архитектуры и технологий, а также их эффективность для реализации проекта.

В дальнейшем возможно расширение функционала приложения, включая добавление новых возможностей, таких как интеграция с платежными системами, улучшенная система рекомендаций и дополнительные способы фильтрации товаров. Это позволит повысить конкурентоспособность приложения и удовлетворенность пользователей.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Горовой, М. А., Разработка мобильных приложений для Android. / М. А. Горовой, В. Н. Лобанов — М.: Эксмо, 2018. — 384 с.
2. Хейккинен, М. Разработка Android-приложений. Профессиональный подход. / М. Хейккинен — СПб.: Питер, 2017. — 416 с.
3. Шахид, М. Android. Разработка приложений для Android. Полное руководство. / М. Шахид — М.: Вильямс, 2016. — 720 с.
4. Резников, А. В. PHP и MySQL. Разработка веб-приложений. / А. В. Резников — М.: Диалектика, 2017. — 320 с.
5. Козлов, С. В. Основы работы с базами данных. MySQL для начинающих. / С. В. Козлов — СПб.: БХВ-Петербург, 2019. — 256 с.