# **ГЛАВА 2 РАЗРАБОТКА ДОКУМЕНТАЦИИ**

В соответствии требованиями ГОСТ Р ИСО/МЭК 14764–2002 с будут разработаны: «технические требования (спецификации), руководства программиста по сопровождению, руководства пользователя и руководства по вводу в действие (инсталляции)»[10].

## **2.1 Руководство пользователя**

### **2.1.1 Введение**

Этот раздел руководства пользователя предоставляет обзор системы "Выбор автомобиля по каталогу", включая ее область применения, краткое описание возможностей, требуемый уровень подготовки пользователей и список эксплуатационной документации, с которой необходимо ознакомиться перед началом работы. Основная цель раздела — помочь пользователям понять, для каких задач предназначена система, каковы ее ключевые возможности, и какие знания требуются для эффективного использования.

1. Область применения. Система "Выбор автомобиля по каталогу" применяется для подбора нового или подержанного автомобиля по множеству критериев, таких как марка, модель, год выпуска, цена, пробег, тип кузова, цвет, тип топлива и другие технические параметры. Она предназначена для частных лиц, автодилеров, компаний по аренде автомобилей и других организаций, которым необходим доступ к базе данных транспортных средств для принятия решений о покупке, продаже или аренде.

2. Краткое описание возможностей. Основные возможности системы включают:

* Поиск и фильтрация автомобилей позволяет находить автомобили по заданным критериям, комбинируя различные фильтры для получения наиболее релевантных результатов.
* Сравнение моделей дает возможность сравнивать до нескольких моделей по ключевым параметрам для более информированного выбора.
* Детальные описания предоставляет полную информацию о выбранных автомобилях, включая фотографии, технические характеристики и историю обслуживания.
* Публикация объявлений пользователи могут улучшить свой аккаунт до статуса продавца и самостоятельно размещать объявления о продаже как новых, так и подержанных автомобилей.
* Уведомления и подписки уведомляет о новых предложениях, соответствующих сохраненным запросам пользователя.
* Аналитические инструменты. Продавцы могут отслеживать активность своих объявлений, анализировать спрос и использовать другие инструменты аналитики.

3. Уровень подготовки пользователя. Пользователь должен иметь базовые навыки работы с интернет-браузерами и операционными системами Windows или macOS. Для эффективного использования системы "Выбор автомобиля по каталогу" желательно понимание основных принципов работы с базами данных и фильтрации. Продавцам рекомендуется знание основ управления объявлениями и анализа статистики.

4. Перечень эксплуатационной документации. Перед началом работы пользователю рекомендуется ознакомиться со следующими документами:

* Общее описание системы. Документ содержит полную информацию о назначении, структуре и функциональности системы.
* Техническое руководство по установке и настройке. Руководство описывает требования к системе, процесс установки клиентских приложений и порядок проверки работоспособности.
* Руководство продавца. Документ предоставляет подробные инструкции для пользователей со статусом продавца по размещению и управлению объявлениями.

Таким образом, введение предоставляет общий контекст системы, определяя ее задачи, возможности и требования к уровню подготовки пользователя. Это позволяет начать использование системы с четким пониманием функциональности и назначения.

### **2.1.2 Назначение и условия применения**

Система "Выбор автомобиля по каталогу" предназначена для автоматизации процесса поиска, сравнения и выбора автомобилей по различным параметрам. Она служит инструментом для покупателей, позволяя находить подходящие модели по индивидуальным предпочтениям, и для продавцов, предлагая удобные возможности по публикации объявлений о продаже автомобилей. Назначение системы — упростить коммуникацию между продавцами и покупателями автомобилей, предоставляя удобную платформу для анализа и обмена информацией.

1. Назначение:

* Автоматизация поиска. Система предоставляет расширенные возможности по поиску автомобилей на основе гибких фильтров и критериев. Это позволяет покупателям легко найти нужные модели среди огромного количества предложений.
* Сравнение и анализ: Пользователи могут сравнивать выбранные модели автомобилей по различным характеристикам, что помогает принимать более взвешенные решения.
* Управление объявлениями. Продавцы получают полный контроль над своими объявлениями, могут публиковать, редактировать и отслеживать статистику по каждому объявлению.
* Уведомления и аналитика. Пользователи могут подписываться на уведомления о новых предложениях и анализировать рыночные тренды с помощью встроенных инструментов аналитики.

1. Условия применения:

* Техническая среда. Для работы с системой требуется доступ к сети интернет и наличие любого современного интернет-браузера (Google Chrome, Firefox, Safari, Edge). Мобильное приложение доступно на платформах Android и iOS. Поддерживаются операционные системы Windows и macOS.
* Технические требования. Для корректного функционирования система должна работать на компьютерах с оперативной памятью не менее 4 ГБ и свободным местом на диске не менее 2 ГБ. Мобильное приложение требует не менее 1 ГБ свободной памяти и текущую версию ОС Android или iOS.
* Уровень доступа. Работа с системой доступна зарегистрированным пользователям, имеющим личный аккаунт в системе. Продавцам необходимо повысить уровень доступа для возможности управления объявлениями.
* Безопасность данных. Система требует от пользователей соблюдения конфиденциальности учетных данных и не рекомендует передавать их третьим лицам. Вся персональная информация обрабатывается в соответствии с политикой конфиденциальности.

В качестве дополнительных материалов в данном разделе может быть представлена таблица с минимальными техническими требованиями к системе или диаграмма с основными функциями и возможностями, чтобы облегчить понимание функционала и условий использования системы "Выбор автомобиля по каталогу".

### **2.1.3 Подготовка к работе**

В разделе "Подготовка к работе" описывается порядок установки и настройки системы "Выбор автомобиля по каталогу", а также проверка её работоспособности перед началом полноценной эксплуатации. Данный раздел включает информацию об установке клиентской части и базовых настройках, необходимых для эффективного функционирования системы.

1. Состав дистрибутива и необходимые компоненты. Для корректной работы системы требуются:

* Веб-браузер: Современный интернет-браузер с поддержкой HTML5 и JavaScript (например, Google Chrome, Mozilla Firefox, Safari, Microsoft Edge).
* Мобильное приложение. Для доступа через мобильные устройства необходимо установить приложение "Выбор автомобиля по каталогу" из официальных магазинов Google Play или App Store.
* Аккаунт пользователя: Зарегистрированный аккаунт в системе, который предоставляет доступ к функционалу.

2. Порядок установки и настройки.

* Веб-браузер. Убедитесь, что браузер обновлен до последней версии. Перейдите на сайт системы и выполните регистрацию или вход с существующими учетными данными.
* Мобильное приложение. Установите приложение из магазина Google Play или App Store, найдя его по названию. После установки откройте приложение и войдите с существующим аккаунтом или создайте новый.
* Настройка аккаунта. После входа в систему необходимо заполнить основные данные профиля (имя, контактную информацию, интересующие марки и модели) для персонализации поиска и подписок на уведомления.

3. Проверка работоспособности:

* Веб-сайт. Зайдите на сайт системы и выполните вход с учетными данными. Проверьте доступность функций поиска, фильтрации, сохранения результатов в избранное и отправки сообщений продавцам.
* Мобильное приложение. Откройте приложение и выполните вход. Проверьте работу функций поиска, сравнения автомобилей, управления избранным и подписок на уведомления.

4. Настройка уведомлений:

* Уведомления в браузере. Включите уведомления на сайте, чтобы своевременно получать сообщения о новых предложениях.
* Уведомления в мобильном приложении. Включите уведомления в настройках приложения для мгновенного оповещения о новых автомобилях и сообщениях от продавцов.

При возникновении проблем с установкой или настройкой системы рекомендуется обратиться в службу технической поддержки, контактные данные которой указаны на сайте и в мобильном приложении.

### **2.1.4 Описание операций**

В данном разделе описываются все основные операции, которые могут выполнять пользователи в системе "Выбор автомобиля по каталогу". Каждая операция связана с определенной функцией или задачей, которая реализуется через интерфейс системы. В описании приведены требования к каждой операции, порядок ее выполнения и особенности работы.

1. Выполняемые функции и задачи:

|  |  |  |
| --- | --- | --- |
| Функция | Задачи | Описание |
| Поиск и фильтрация | Поиск по ключевым словам | Позволяет искать автомобили по определенным ключевым словам, таким как марка, модель или год. |
| Фильтрация по параметрам | Позволяет применять фильтры по цене, пробегу, типу кузова и другим техническим характеристикам. |
| Сравнение автомобилей | Публикация новых объявлений | Дает возможность продавцам создавать и публиковать новые объявления о продаже автомобилей. |
| Редактирование объявлений | Позволяет изменять или обновлять существующие объявления. |
| Управление объявлениями | Добавление в избранное | Сохраняет понравившиеся автомобили для быстрого доступа в будущем. |
| Удаление из избранного | Удаляет автомобили из списка избранного. |
| Управление избранным | Публикация новых объявлений | Дает возможность продавцам создавать и публиковать новые объявления о продаже автомобилей. |
| Редактирование объявлений | Позволяет изменять или обновлять существующие объявления. |

2. Описание операций обработки данных:

Операция: Поиск автомобиля

* Условия выполнения. Пользователь должен быть зарегистрирован и авторизован в системе.
* Основные действия:

1. Перейдите к форме поиска, используя меню или кнопку на главной странице.

2. Введите ключевые слова или установите фильтры по интересующим параметрам.

3. Нажмите кнопку "Поиск".

4. Просмотрите результаты и при необходимости уточните критерии.

Операция: Сравнение автомобилей

* Условия выполнения. Необходимо добавить, как минимум, две модели для сравнения.
* Основные действия:

1. Добавьте автомобили в список сравнения через кнопку в карточке модели.

2. Перейдите в раздел сравнения из меню или уведомления.

3. Просмотрите таблицу сравнения по основным характеристикам.

Операция: Публикация объявления

* Условия выполнения: Пользователь должен иметь статус продавца.
* Основные действия:

1. Перейдите в раздел "Мои объявления".

2. Нажмите "Создать объявление" и заполните форму с данными об автомобиле.

3. Загрузите фотографии и укажите контактную информацию.

4. Опубликуйте объявление.

Операция: Управление избранным

* Условия выполнения: Пользователь должен быть зарегистрирован и авторизован.
* Основные действия:

1. В карточке автомобиля нажмите на значок "Добавить в избранное".

2. Перейдите в раздел избранного для просмотра сохраненных автомобилей.

3. Удалите ненужные модели, используя соответствующую кнопку.

### **2.1.5 Аварийные ситуации**

В данном разделе описаны возможные ошибки и нештатные ситуации, которые могут возникнуть при использовании системы "Выбор автомобиля по каталогу", а также предложены действия по их устранению. Классификация ошибок разделена на несколько групп в зависимости от природы проблемы. Приведенная таблица содержит информацию об обнаружении, диагностике и решении наиболее распространенных ошибок.

|  |  |  |  |
| --- | --- | --- | --- |
| Класс ошибки | Ошибка | Описание проблемы | Требуемые действия |
| Сетевые ошибки | Сервер недоступен | Сервер временно недоступен или не отвечает | Проверьте соединение с интернетом, повторите попытку позже. При сохранении проблемы обратитесь в техподдержку. |
| Сбой локальной сети | Проблема с подключением к локальной сети | Перезагрузите устройство и проверьте подключение к сети. Если проблема сохраняется, обратитесь к сетевому администратору. |
| Ошибки авторизации | Неверные учетные данные | Логин или пароль введены неверно | Проверьте правильность ввода логина и пароля. Если проблема не устранена, выполните сброс пароля или обратитесь в техподдержку. |
| Заблокированный аккаунт | Аккаунт заблокирован после нескольких неудачных попыток входа | Дождитесь автоматической разблокировки или свяжитесь с техподдержкой. |
| Ошибки доступа | Отсутствие доступа к разделу | Пользователь не имеет прав на доступ к данному разделу | Проверьте свои права доступа и, при необходимости, свяжитесь с администратором или технической поддержкой. |
| Системные ошибки | Ошибка базы данных | Сбой при взаимодействии с базой данных | Перезапустите приложение и попробуйте снова. Если проблема сохраняется, обратитесь в техподдержку. |
| Превышено время ожидания ответа сервера | Сервер не отвечает из-за высокой нагрузки | Подождите несколько минут и повторите запрос. |
| Ошибки клиентской части | Неправильное отображение страницы | Сбой в работе браузера или устаревшая версия | Перезапустите браузер, обновите его до последней версии или попробуйте другой браузер. |
| Сбой установки мобильного приложения | Недостаточно памяти на устройстве или устаревшая версия ОС | Освободите память, обновите ОС и попробуйте переустановить приложение. |

Для более специфических случаев, не описанных в таблице, рекомендуется обратиться в службу технической поддержки, предоставив подробную информацию об ошибке и обстоятельствах её возникновения.

## **2.2 Руководство программиста по сопровождению**

### **2.2.1 Структура программного обеспечения АС**

Структура программного обеспечения "Выбор автомобиля по каталогу" разрабатывалась с учетом требований к гибкости, масштабируемости и удобству пользователя. Основными компонентами системы являются клиентская часть, серверная часть, база данных, а также коммуникационный интерфейс.

1. Клиентская часть. Клиентская часть программного обеспечения разделена на два основных приложения:

* Веб-сайт на React предназначен для работы на стационарных компьютерах и мобильных устройствах через веб-браузер. Использование React позволяет создать интерактивный и отзывчивый пользовательский интерфейс.
* Кроссплатформенное мобильное приложение на Dart/Flutter обеспечивает работу на различных мобильных операционных системах, включая Android и iOS. Это приложение позволяет пользователям взаимодействовать с системой напрямую через мобильные устройства.

Оба приложения используют общую структуру навигации и дизайн, что обеспечивает консистентный пользовательский опыт.

2. Серверная часть. Разработана на языке программирования Go (Golang), что обеспечивает высокую производительность и масштабируемость серверных операций. Сервер обрабатывает запросы от клиентских приложений, выполняет логику приложений и взаимодействует с базой данных.

3. База данных. Используется PostgreSQL, реляционная СУБД, которая хранит все данные системы, включая информацию о пользователях, автомобилях и транзакциях. ER-диаграмма базы данных демонстрирует структуру и взаимосвязи между таблицами, что облегчает понимание и разработку базы данных [6].

4. API и коммуникационный интерфейс. Взаимодействие между клиентскими приложениями и сервером происходит через gRPC, что позволяет эффективно управлять запросами и ответами, гарантируя быстродействие и безопасность обмена данными. Описание gRPC API предоставлено в виде proto-файла, который включает определения всех доступных сервисов и операций.

5. Диаграммы:

* Диаграмма взаимодействия клиентов, сервера и БД иллюстрирует поток данных и коммуникации между различными компонентами системы, позволяя понять архитектурные решения.
* Диаграмма экранов клиента показывает навигацию и взаимодействие пользовательского интерфейса. На этой диаграмме отображается, какие кнопки и действия ведут на другие страницы, что помогает в разработке и понимании пользовательского опыта.

Эта структура программного обеспечения не только обеспечивает эффективное функционирование системы "Выбор автомобиля по каталогу", но и облегчает ее дальнейшее масштабирование и адаптацию к изменяющимся требованиям пользователей и технологий.

### **2.2.2 Функции частей программного обеспечения АС**

В этом разделе описаны назначение и функции каждой из основных частей программного обеспечения системы "Выбор автомобиля по каталогу". Каждый компонент играет важную роль в обеспечении общей функциональности и эффективности работы системы.

1. Клиентская часть:

Веб-сайт на React. Предназначен для обеспечения пользователям доступа к основным функциям системы через веб-браузер. Основные функции включают:

* Регистрация и авторизация. Пользователи могут создавать учетные записи, входить в систему и управлять своими профилями.
* Поиск и фильтрация автомобилей позволяет искать автомобили по заданным критериям, таким как марка, модель, год выпуска, тип кузова и другие параметры.
* Сравнение автомобилей. Сравнение нескольких моделей по ключевым характеристикам, чтобы помочь пользователям принять более взвешенное решение.
* Управление объявлениями. Для продавцов — создание, редактирование и удаление объявлений о продаже автомобилей.
* Избранное и подписки. Сохранение понравившихся автомобилей в избранное и подписка на уведомления о новых предложениях.

Мобильное приложение на Dart/Flutter. Кроссплатформенное мобильное приложение предоставляет доступ к тем же функциям, что и веб-сайт, но в более оптимизированном для мобильных устройств формате. Основные функции включают:

* Регистрация и авторизация: Создание учетных записей и вход в систему через мобильное устройство.
* Поиск и сравнение: Аналогичные функции по поиску, фильтрации и сравнению автомобилей.
* Избранное: Сохранение автомобилей в избранное и возможность подписки на уведомления.

2. Серверная часть:

* Управление пользователями: Регистрация и авторизация пользователей, управление профилями, правами доступа и персонализацией.
* Управление объявлениями: Прием, хранение и публикация объявлений о продаже автомобилей, а также предоставление аналитических инструментов для отслеживания активности.
* Рекомендации и уведомления: Анализ поведения пользователей для персонализации предложений, а также отправка уведомлений о новых предложениях и сообщениях от продавцов.

3. База данных PostgreSQL:

* Хранилище данных: Сохранение и управление всей информацией о пользователях, автомобилях, объявлениях, транзакциях и взаимодействиях.
* Оптимизация запросов: Индексация данных и выполнение сложных запросов для быстрого поиска и фильтрации автомобилей.

Изображение выглядит как текст, диаграмма, снимок экрана, План

Автоматически созданное описание

Рисунок 1 – ER диаграмма базы данных

4. API через gRPC:

* Интерфейс взаимодействия позволяет клиентским приложениям безопасно взаимодействовать с серверной частью и базой данных для выполнения всех основных функций, включая авторизацию, управление объявлениями, поиск и сравнение [7].

Каждая из частей программного обеспечения обеспечивает свой набор функций, что вместе создает комплексную и эффективную систему для поиска и сравнения автомобилей в каталоге.

### **2.2.3 Методы и средства разработки программного обеспечения АС**

В процессе разработки программного обеспечения системы "Выбор автомобиля по каталогу" использовался комплекс современных методов программирования и инструментов, обеспечивающих высокую производительность, надежность и эффективность разработки. Для каждой части системы применялись специализированные инструменты и технологии.

1. Клиентская часть:

Веб-сайт на React:

* Метод программирования. Компонентный подход, основанный на принципах функционального программирования, для обеспечения модульности и повторного использования кода.
* Инструменты разработки. React (JavaScript-библиотека для разработки интерфейсов), Webpack (сборка и оптимизация кода), Redux (управление состоянием приложения).
* Применение. Разработка пользовательского интерфейса веб-сайта с использованием адаптивного дизайна, обеспечивающего отзывчивость на всех устройствах.

Мобильное приложение на Dart/Flutter:

* Метод программирования: Виджетный подход, при котором каждый элемент интерфейса представлен как отдельный виджет.
* Инструменты разработки: Flutter SDK (разработка кроссплатформенных приложений), Dart (язык программирования).
* Применение. Разработка кроссплатформенного мобильного приложения с единым кодом для Android и iOS, позволяющего ускорить процесс разработки и облегчить поддержку.

2. Серверная часть на Golang:

* Метод программирования: Микросервисная архитектура, основанная на разделении логики по независимым сервисам.
* Инструменты разработки: Go (язык программирования), gRPC (межпроцессное взаимодействие), Docker (контейнеризация), Swagger (документирование API).
* Применение: Реализация микросервисной архитектуры с использованием Go, которая обеспечивает высокую производительность и надежность сервера.

3. База данных PostgreSQL:

* Метод программирования: ER-моделирование для оптимального проектирования базы данных и ее структуры.
* Инструменты разработки: PostgreSQL (реляционная база данных), pgAdmin (графический инструмент управления БД), SQL (язык запросов).
* Применение: Проектирование схемы базы данных с учетом оптимальной индексации и реляционных связей между таблицами для ускорения обработки запросов.

4. API через gRPC:

* Метод программирования: Протокол удаленного вызова процедур (RPC) для эффективного взаимодействия между клиентской и серверной частями.
* Инструменты разработки: Protocol Buffers (формат сериализации), gRPC (протокол RPC).
* Применение: Разработка proto-файлов для определения сервисов и методов взаимодействия между компонентами системы, а также генерация кода клиента и сервера.

Использование этих методов и средств разработки позволило создать надежную и масштабируемую систему "Выбор автомобиля по каталогу" с гибким интерфейсом и эффективной серверной частью, обеспечивающей быструю обработку данных.

## **Руководство по вводу в действие**

### **2.3.1 Операционная система**

Для развертывания и запуска серверной части системы "Выбор автомобиля по каталогу" была выбрана операционная система Ubuntu Server 20.04 LTS. Это стабильная версия популярного дистрибутива Linux с долгосрочной поддержкой (LTS), которая предоставляет надежную и безопасную среду для развертывания серверных приложений. Основные характеристики и преимущества этой операционной системы приведены ниже.

1. Наименование и обозначение**:**

* Операционная система: Ubuntu Server
* Версия: 20.04 LTS

2. Краткая характеристика:

* Поддержка архитектуры. Операционная система поддерживает архитектуры x86-64 и ARM, что делает ее универсальной для развертывания на различных типах серверов [5].
* Безопасность. Включает встроенные механизмы безопасности (например, AppArmor) и регулярные обновления безопасности на протяжении долгосрочного периода поддержки.
* Пакетный менеджер использует пакетный менеджер APT, позволяющий легко управлять установкой и обновлением программного обеспечения.
* Контейнеризация. Встроенная поддержка Docker и других технологий контейнеризации делает Ubuntu Server оптимальной для разворачивания микросервисных архитектур.
* Совместимость. Широкий набор предустановленных пакетов и поддержка популярного программного обеспечения упрощают интеграцию и развертывание.

3. Обоснование выбора:

* Надежность. Ubuntu Server 20.04 LTS предоставляет долгосрочную поддержку с регулярными обновлениями безопасности, что гарантирует стабильную работу серверных приложений [5].
* Масштабируемость. Операционная система оптимизирована для работы с микросервисной архитектурой и контейнеризацией, что идеально подходит для серверной части системы "Выбор автомобиля по каталогу".
* Совместимость. Поддержка новейших версий Golang, PostgreSQL и gRPC обеспечивает надежное взаимодействие серверных компонентов с остальной системой.

4. Источники информации:

* Документация. Подробное описание версии Ubuntu Server 20.04 LTS и руководство по установке доступны в официальной документации на сайте ubuntu.com.

Таким образом, выбор Ubuntu Server 20.04 LTS обеспечивает надежную и оптимальную платформу для развертывания серверной части системы, позволяя обеспечить долгосрочную поддержку, безопасность и высокую производительность.

### **2.3.2 Переменные окружения и конфигурационные файлы**

Для корректного функционирования серверной части системы "Выбор автомобиля по каталогу" необходимо настроить переменные окружения и конфигурационные файлы. Эти настройки обеспечивают конфиденциальность, безопасность и правильную работу различных компонентов системы. Рассмотрим основные аспекты настройки переменных окружения и конфигурационных файлов.

Переменные окружения используются для хранения чувствительных данных, таких как пароли и ключи API, а также для настройки параметров системы, которые могут изменяться в зависимости от среды развертывания (разработка, тестирование, продуктивная среда). Основные переменные окружения для нашей системы включают:

1. DATABASE\_URL: строка подключения к базе данных PostgreSQL. Формат строки подключения: postgres://user:password@host:port/database.

3. GRPC\_SERVER\_ADDRESS: адрес gRPC сервера.

4. API\_KEY: ключ для доступа к внешним API (например, сервисы поиска автомобилей).

Конфигурационные файлы используются для хранения настроек, которые не должны изменяться во время работы приложения. Эти файлы позволяют легко управлять параметрами приложения и обеспечивают возможность быстрого изменения конфигурации при необходимости.

Основные конфигурационные файлы для нашей системы включают:

1. config.yaml: основной конфигурационный файл, содержащий настройки базы данных, сервера и прочих параметров.
2. docker-compose.yml: файл для настройки контейнеров Docker, обеспечивающий запуск всех необходимых сервисов (например, базы данных, серверных компонентов) в изолированных контейнерах.

Для управления переменными окружения рекомендуется использовать утилиты, такие как dotenv или системы управления секретами (например, Vault). Это позволяет хранить переменные окружения в файлах .env, которые затем загружаются при запуске приложения:

1. Создайте файл .env в корневой директории проекта.
2. Загрузите переменные окружения из файла .env при запуске приложения.

Использование переменных окружения и конфигурационных файлов обеспечивает гибкость и безопасность при развертывании и эксплуатации серверной части системы "Выбор автомобиля по каталогу".

Для обеспечения удобства пользователей также разработано кроссплатформенное мобильное приложение на Dart и Flutter. Это приложение предоставляет доступ к функционалу системы "Выбор автомобиля по каталогу" на мобильных устройствах.

1. Установка Flutter SDK. Следуйте инструкциям на официальном сайте Flutter для установки Flutter SDK.

2. Настройка проекта.

3. Добавление зависимостей. В файле pubspec.yaml укажите необходимые пакеты и зависимости.

4. Настройка API и переменных окружения: Создайте файл .env для хранения переменных окружения, таких как URL API.

5. Запуск приложения. Для запуска приложения на устройстве или эмуляторе используйте команду.

6. Сборка приложения для публикации. Для сборки APK (Android) или IPA (iOS) файлов используйте команды.

### **2.3.3 Режимы работы**

Для обеспечения гибкости и адаптивности системы "Выбор автомобиля по каталогу" предусмотрены различные режимы работы как для серверной, так и для клиентской части приложения. Управление режимами работы осуществляется с помощью флагов, которые указываются при запуске приложений. Это позволяет конфигурировать и изменять поведение приложений в зависимости от текущих задач и среды развертывания.

Серверное приложение системы реализовано на языке Golang и поддерживает несколько режимов работы, которые задаются с помощью флагов. Основные режимы включают:

1. Development (Режим разработки). В этом режиме сервер запускается с расширенными возможностями для отладки и тестирования. Этот режим включает детализированные логи и горячую перезагрузку при изменении кода.

2. Production (Продакшен режим). В продакшен режиме сервер работает с оптимизированными настройками для обеспечения максимальной производительности и безопасности. Логи ограничены только важными сообщениями об ошибках.

3. Testing (Тестовый режим). Этот режим предназначен для запуска автоматических тестов и интеграционных проверок. Сервер может использовать отдельную тестовую базу данных и конфигурации.

Клиентская часть системы, разработанная на React, также поддерживает несколько режимов работы, которые задаются с помощью флагов при запуске. Основные режимы включают:

1. Development (Режим разработки). В этом режиме клиентское приложение запускается с расширенными возможностями для отладки, включая source maps и подробные логи. Также доступен режим горячей перезагрузки для быстрого применения изменений.

2. Production (Продакшен режим). В продакшен режиме клиентское приложение собирается и оптимизируется для обеспечения высокой производительности и минимального времени загрузки. Все исходные карты и логи отключены.

3. Testing (Тестовый режим). Этот режим используется для выполнения автоматических тестов клиентского приложения. Приложение запускается в специальной среде с тестовыми данными и конфигурациями.

Использование флагов для управления режимами работы позволяет:

1. Повышение гибкости. Легко переключаться между различными режимами работы без необходимости изменения исходного кода или конфигурационных файлов.

2. Обеспечение безопасности. В продакшен режиме отключаются ненужные логи и отладочные сообщения, что повышает безопасность системы.

3. Удобство тестирования: Тестовый режим позволяет изолировать тестовую среду и использовать отдельные конфигурации и базы данных, что упрощает проведение автоматизированных тестов.

Внедрение и использование различных режимов работы обеспечивают эффективное и безопасное функционирование системы [2, c. 127] "Выбор автомобиля по каталогу" в различных условиях эксплуатации.

## **Технические требования**

### **2.4.1 Введение**

Документ содержит технические требования к автоматизированной системе выбора автомобиля по каталогу. Система состоит из клиентской части (веб-сайт на React и кроссплатформенное мобильное приложение на Dart/Flutter), серверной части на Golang, взаимодействие между клиентом и сервером осуществляется через gRPC, а база данных использует PostgreSQL.

### **2.4.2 Функциональные требования**

|  |  |
| --- | --- |
| Функциональные требования | Описание |
| Регистрация и авторизация пользователей | Пользователь должен иметь возможность зарегистрироваться, предоставив email и пароль. Авторизация должна быть реализована с использованием JWT. |
| Просмотр каталога автомобилей | Пользователь должен иметь возможность просматривать каталог автомобилей. Каталог должен поддерживать фильтрацию по марке, модели, году выпуска, цене и другим параметрам. |
| Поиск автомобилей | Пользователь должен иметь возможность искать автомобили по ключевым словам. |
| Детальная информация об автомобиле | Пользователь должен иметь возможность просматривать детальную информацию о выбранном автомобиле, включая описание, технические характеристики, фотографии и отзывы. |
| Сравнение автомобилей | Пользователь должен иметь возможность добавлять автомобили в список для сравнения и просматривать их характеристики бок о бок. |
| Добавление отзывов и оценок | Авторизованные пользователи должны иметь возможность оставлять отзывы и оценки для автомобилей. |
| Избранное | Пользователь должен иметь возможность добавлять автомобили в список избранного для быстрого доступа в будущем. |

### **2.4.3 Нефункциональные требования**

|  |  |
| --- | --- |
| Нефункциональные требования | Описание |
| Производительность | Система должна обрабатывать до 1000 запросов в секунду при пиковых нагрузках. Время отклика сервера не должно превышать 200 миллисекунд. |
| Безопасность | Данные пользователей должны быть защищены с использованием SSL/TLS для всех сетевых взаимодействий. Пароли должны храниться в зашифрованном виде с использованием алгоритмов шифрования, соответствующих текущим стандартам безопасности. |
| Надежность | Система должна обеспечивать 99.9% доступность. База данных должна поддерживать резервное копирование и восстановление. |
| Удобство использования | Интерфейс должен быть интуитивно понятным и поддерживать адаптивный дизайн для различных устройств. |
| Масштабируемость | Архитектура системы должна поддерживать горизонтальное и вертикальное масштабирование. |

### **2.4.4 Интрфейсные требования**

|  |  |
| --- | --- |
| Интерфейсные требования | Описание |
| Пользовательский интерфейс (UI) | Веб-сайт должен быть реализован на React и поддерживать основные браузеры (Chrome, Firefox, Safari, Edge). Мобильное приложение должно быть реализовано на Dart/Flutter и поддерживать платформы iOS и Android. |
| API | Клиент и сервер должны взаимодействовать через gRPC API. API должно быть документировано с использованием протоколов gRPC и содержать описание всех методов и типов данных. |
| База данных | Серверная часть должна взаимодействовать с базой данных PostgreSQL. База данных должна быть структурирована с учетом требований к производительности и масштабируемости. |

### **2.4.5 Ограничения и допущения**

• Команда разработки имеет доступ к исходному коду всех компонентов системы.

### **2.4.6 Требования к данным**

• Данные автомобилей должны включать марку, модель, год выпуска, цену, технические характеристики, фотографии и отзывы.

• Данные пользователей должны включать email, пароль (в зашифрованном виде), и список избранного.

### **2.4.7 Юридические требования**

• Система должна соответствовать требованиям GDPR для защиты данных пользователей.

• Все компоненты системы должны использовать лицензированные технологии и библиотеки.

### **2.4.8 Критерии приемки**

• Система считается принятой, если она соответствует всем функциональным и нефункциональным требованиям, а также успешно проходит все тесты (функциональные, нагрузочные, безопасность).

• Проведение демонстрации функциональности перед заказчиком.

• Подготовка и передача всей необходимой документации для сопровождения.

Эти технические требования (спецификации) представляют собой основу для тестирования и сопровождения автоматизированной системы выбора автомобиля по каталогу, обеспечивая ясное и полное описание ожидаемого поведения системы и критериев ее качества.

**Выводы по главе**

Вторая глава документации посвящена разработке документации для системы "Выбор автомобиля по каталогу". Она начинается с руководства пользователя, где описываются область применения системы, её основные возможности и уровень подготовки пользователей. Пользователи могут эффективно искать, сравнивать автомобили и управлять объявлениями, а также получать уведомления и использовать аналитические инструменты.

Далее рассматривается структура программного обеспечения, включающая клиентскую часть (веб-сайт на React и мобильное приложение на Flutter), серверную часть на Golang и базу данных PostgreSQL. Описаны функции каждой части системы, такие как регистрация и авторизация пользователей, поиск и сравнение автомобилей, управление объявлениями и избранным.

Глава также охватывает методы и средства разработки, подчеркивая использование современных технологий для создания гибкой и масштабируемой системы. Например, веб-сайт разработан с использованием React, а мобильное приложение - с использованием Flutter.

Раздел по развертыванию и запуску включает выбор операционной системы Ubuntu Server 20.04 LTS для серверной части, настройку переменных окружения и конфигурационных файлов, а также описание различных режимов работы системы (разработка, продакшн, тестирование).

Последняя часть представляет из себя спецификацию, описывающую функциональные, нефункциональные, интерфейсные и юридические требования, ограничения, допущения, требования к данным и критерии приемки.

Таким образом, вторая глава предоставляет всесторонние руководства по поддержке, настройке и эксплуатации системы "Выбор автомобиля по каталогу", обеспечивая её эффективное сопровождение