Министерство образования Иркутской области

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

КП.09.02.07.22.211.25 ПЗ

ИНФОРМАЦИОННАЯ СИСТЕМА

«Обслуживание автомобилей»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (П.Р. Юносов) |

Оглавление

[ВВЕДЕНИЕ 3](#_Toc117439958)

[1.Описание предметной области ИС 6](#_Toc117439959)

[2. Анализ инструментальных средств разработки, используемых при реализации ИС 7](#_Toc117439960)

[2.1 Обоснование выбора программного продукта 9](#_Toc117439961)

[3 Техническое задание 12](#_Toc117439962)

[4.Проектирование Ис 13](#_Toc117439963)

[4.1 Структурная схема ИС 13](#_Toc117439964)

Введение

Темой курсового проекта является информационная система «Обслуживание автомобилей».

Автотранспорт является очень популярным средство передвижения человека. Транспортное средство – это техническое устройство, главной задачей которого является перевозка людей и грузов. Автотранспорт может быть поврежден и для этого необходимы автосервисы, которые позволяют исправить существующие проблемы автотранспорта.

При работе с автотранспортом главное – это определение и поиск необходимых материалов для исправления повреждений автомобиля, а также создание отчетов о проделанных работах. Во многих автомастерских есть проблема c ПО или вообще его отсутствия. Из-за этого появляется множество проблем: организация рабочего процесса, работа с клиентами и отчетами.

Но не каждая мастерская пользуется электронными таблицами и подобным ПО. Разрабатываемая информационная система поможет оптимизировать работу автомастерских.

Цель: разработать информационную систему «Обслуживание автомобилей».

Задачи:

1. Провести предпроектное исследование.
2. Составить техническое задание на разработку программного продукта в соответствии с ГОСТ.
3. Провести проектирование программного продукта.
4. Реализовать программный продукт.
5. Разработать документы для программного продукта.
6. Рассчитать стоимость разработки и внедрения программного продукта.

Информационная система позволяет организовать работу, сборку отчетов о проделанных работах с транспортом. Поможет накопить клиентскую базу, вести учет работы сотрудников и учёт затраченных материалов на сборку. Создатель автосервиса всегда должен отслеживать работу своего сервиса.

1. Описание предметной области ИС

Предметной областью курсового проекта является информационная система «Обслуживание автомобилей» (далее ИС).

Автосервис — услуга, которая пользуется большой популярностью у автомобилистов. Автомобиль требует тщательного обслуживания и своевременного устранения технических неполадок. Производятся все манипуляции по техобслуживанию обычно на станции технического обслуживания (СТО).

Автотранспорт — это вид транспорта, осуществляющий перевозку грузов и пассажиров по безрельсовым путям с использованием колёсного движителя.

Ремонт — комплекс операций по восстановлению исправного состояния колесного транспортного средства (его составных частей, систем).

Станция техобслуживания автомобилей (СТО) — это предприятие автомобильного сервиса, на котором оказываются различные виды услуг от диагностики до ремонта автомобильного транспорта с целью дальнейшей его эксплуатации.

Все обращающиеся в автосервис клиенты, желающие оформить заказ на оказание услуг, должны заполнить форму регистрации, в которой необходимо указать свои ФИО, марку автомобиля, номерной знак и ФИО владельца машины, контактный телефон.

Таким образом, в функционирование автосервиса входит:

1. регистрация клиентов;
2. информация о доступных услугах;
3. справочная информация о сделанном заказе;
4. выполнение заказа;
5. отчет о проделанных работах и расчет стоимости предоставленных услуг.
6. Анализ инструментальных средств разработки, используемых при реализации ИС

Для разработки ИС можно использовать следующие программные продукты: SQL, Django, Python, JavaScript, Microsoft Visual Studio Сode, Microsoft Visio, SQLite, DataGrip, SQLiteStudio, Draw.io, JetBrains Pycharm, Atom, Fleet, Node.js, PHP.

SQL — «Язык структурированных запросов» декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Django — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования МVС.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

JavaScript — это полноценный динамический язык программирования, который применяется к НТМL документу, и может обеспечить динамическую интерактивность на веб-сайтах.

Visual Studio Code (VS Code) — текстовый редактор, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

Microsoft Visio — векторный графический редактор, редактор диаграмм и блок-схем для Windows.

SQLite — это встраиваемая кроссплатформенная БД, которая поддерживает достаточно полный набор команд SQL и доступна в исходных кодах.

DataGrip — это инструмент от компании JetBrains для управления различными базами данных. Можно сказать, что это расширенный аналог PhpMyAdmin, который может работать не только с MySQL или MariaDB, но и с PostgreSQL и другими базами данных.

SQLiteStudio — представляет собой программу, которая предназначена для создания и редактирования баз данных. При помощи данного софта можно легко создать таблицы, функция format имеет разнообразные инструменты, позволяющие быстро форматировать выбранную базу данных.

Draw.io — это бесплатное кроссплатформенное программное обеспечение для рисования графиков с открытым исходным кодом, разработанное на HTML5 и JavaScript.

PyCharm — это интегрированная среда разработки (IDE), используемая в компьютерном программировании, специально для языка программирования Python.

Atom — бесплатный текстовый редактор с открытым исходным кодом для Linux, macOS, Windows с поддержкой плагинов, написанных на JavaScript, и встраиваемых под управлением Git.

Fleet — легковесный текстовый редактор для тех случаев, когда нужно быстро посмотреть и отредактировать код. При этом Fleet легко трансформируется в полноценную IDE: движок анализа кода работает независимо от редактора.

NodeJS — программная платформа, основанная на движке V8, превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API, написанный на C++, подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода.

PHP (рекурсивный акроним словосочетания PHP: Hypertext Preprocessor) — это распространенный язык программирования общего назначения с открытым исходным кодом. PHP сконструирован специально для ведения Web-разработок и его код может внедряться непосредственно в HTML.

* 1. Обоснование выбора программного продукта

Для разработки ИС было решено использовать Visual Studio Code, DataGrip и SQLite.

Для выбора инструментов разработки были произведены сравнения, которые показаны в таблице 1, таблице 2, таблице 3, и таблице 4.

Таблица 1 — Сравнение IDE для разработки программного продукта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название IDE | Visual studio code | JetBrains Pycharm | Atom | Fleet |
| Стоимость | Свободное ПО | Свободное ПО | Свободное ПО | Community edition |
| Автосохранение | + | + | + | + |
| Автодополнение | + | + | + | + |
| Интеграция с системами управления версиями (git) | + | -  (возможно добавить плагин) | + | + |
| Поиск по функции в коде | + | - | + | + |
| Встроенные языки программирования | возможно добавить плагинов (python html css и т.д) | - Для чистой разработки на Python | C, C#, Java, HTML, JavaScript, PHP, Python, Ruby | Python, Java, Rust, C#, Php, Go, и т.д. |

Таким образом, из представленных IDE можно выделить несколько инструментов, способных облегчить и ускорить разработку программного обеспечения, а именно Visual studio code, JetBrains Pycharm, Atom, Fleet так как программный продукт будет реализован с помощью web - технологий. Исходя из того, что для разработки необходим язык программирования python, то в качестве IDE был выбран Visual studio code. Иза своей модульности и большей поддержки плагинов и языков программирования.

Таблица 2 – Сравнение языков программирования для разработки программного продукта

| Название языка программирования | Node.js | PHP | Python |
| --- | --- | --- | --- |
| Простота обучаемости | + | + | + |
| Читабельность | + | + | + |
| Динамическая типизация | + | + | + |
| Скорость разработки | + | – | + |
| Интеграция баз данных | – | + | + |

Вывод, из представленных языков программирования можно выделить Python, так как именно этот язык более интересный и перспективный в разработке веб – приложения.

Таблица 3 — Сравнение СУБД для разработки программного продукта.

|  |  |  |  |
| --- | --- | --- | --- |
| Название СУБД | Oracle | MySQL | SQLite |
| Язык программирования | С, С++, Java, Ruby, Objective | Delphi, С, С++, Java, Perl, РНР, Pvthon, Ruby, SQL | РНР, Python, Ruby и др. |
| Скорость обработки | Быстрая | Быстрая | Медленная |
| Операционная система Windows | Поддерживается | Поддерживается | Поддерживается |
| Исходный код | Закрытый | Открытый | Открытый |
| Лицензия | Коммерческая | Свободная | Свободная |

В результате сравнения, из представленных СУБД можно выделить SQLite, так как оно имеет свободную лицензию и подходит для создания web – приложения. Для SQLite не требует отдельного сервера и это является огромным плюсом для небольшого проекта.

1. Техническое задание

В начале разработки создавалась техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 19.

Согласно ГОСТ 19 техническое задание должно включать следующие разделы:

Введение.

1. Основание для разработки.

2. Назначение разработки.

3 Требования к программе или программному изделию.

3.1. Требования к функциональным характеристикам.

3.2. Требования к надежности.

3.3. Требования к составу и параметрам технических средств.

3.4. Требования к информационной и программной совместимости.

3.5. Требования к транспортировке и хранению.

4. Требования к программной документации.

5. Технико-экономические показатели.

6. Состав и содержание работ по созданию системы.

Техническое задание на разработку информационной системы представлено в Приложение А.

1. Проектирование ИС
   1. Структурная схема ИС

Проектирование ИС началось с построения диаграммы вариантов использования. На рисунке 1 представлена диаграмма прецедентов Uses CASE. Она содержит 3 актёра, которые могут выполнять суммарно 5 функций, часть из которых может делать несколько актёров, а часть – только определённый актёр.

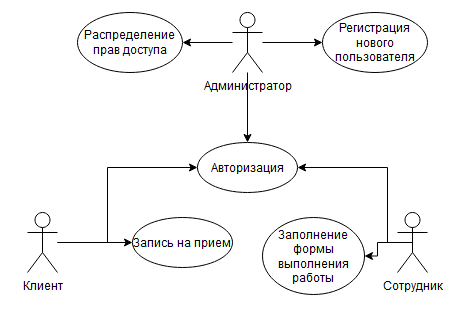


Рисунок 1 – диаграмма прецендентов.

На рисунке 2 представлена диаграмма деятельности. Она содержит 3 роли.

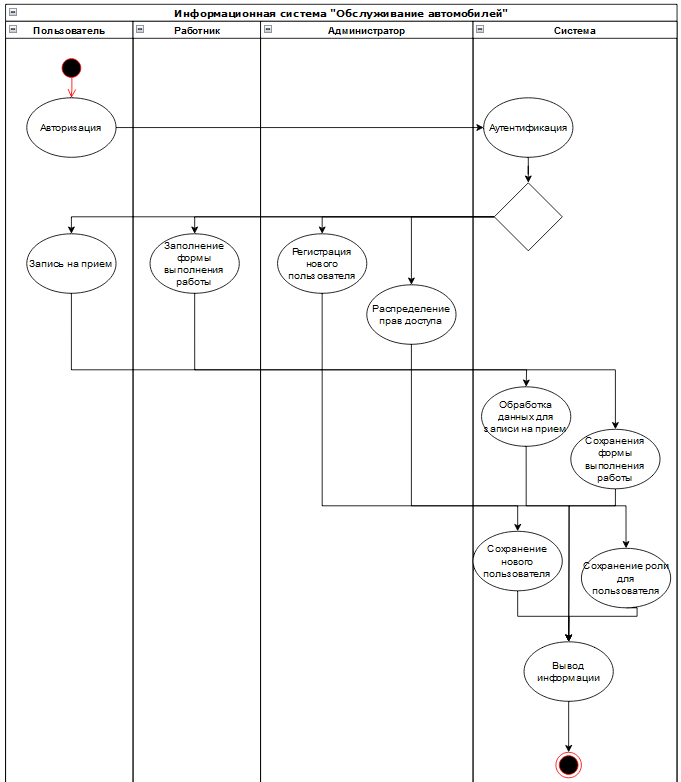


Рисунок 2 – диаграмма деятельности.

Процесс начинается с пользователя, потому что чтобы использовать функционал ИС необходимо пройти авторизацию. Авторизация вызывает процесс аутентификации пользователя в системе, определяя права доступа пользователя к различным модулям ИС.

Права доступа «Пользователь» открывают модуль «Запись на прием». Данный модуль позволяет обеспечить запись потенциального клиента на диагностику, где оказываются услуги обслуживания автомобилей.

Права доступа «Работник» открывают модуль «Заполнение формы выполнения работы». Данный модуль позволяет подтвердить выполнение работы сотрудником.

Права доступа «Администратор» открывает несколько модулей «Регистрация нового пользователя» и «Распределение прав доступа». «Регистрация нового пользователя» позволяет добавить в ИС нового пользователя с права доступа, назначенные администратором. «Распределение прав доступа» позволяет администратору назначать права доступа пользователям.

ИС обрабатывает работу модулей и сохраняет обработанные данные в БД.

На рисунке 3 изображена диаграмма развёртывания, которая изображает аппаратные компоненты системы и взаимодействие между ними.

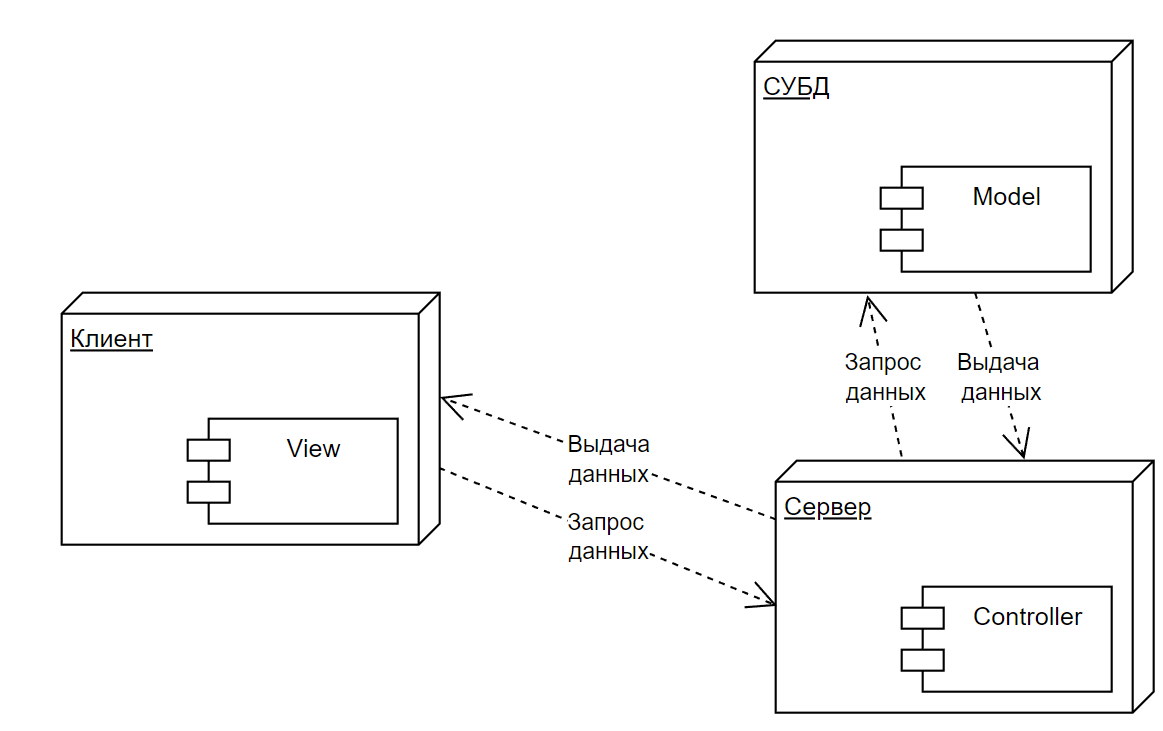


Рисунок 3 – диаграмма развёртывания.

Диаграммы использования позволяют понять, как будут происходить разграничения прав доступа различных пользователей. Позволяют выделить необходимые процессы для достижения этих разграничений.

* 1. Функциональная схема ИС

Функциональные схемы необходимы для определения деятельности ИС. Выделить основные функции, которые ИС выполняет и их подфункции.

Контекстная диаграмма – это модель, представляющая систему как набор иерархических действий, в которой каждое действие преобразует некоторый объект или набор объектов. Выделяются входные и выходные данные, управление и пользователи. Контекстная диаграмма ИС представлена на рисунке 4.

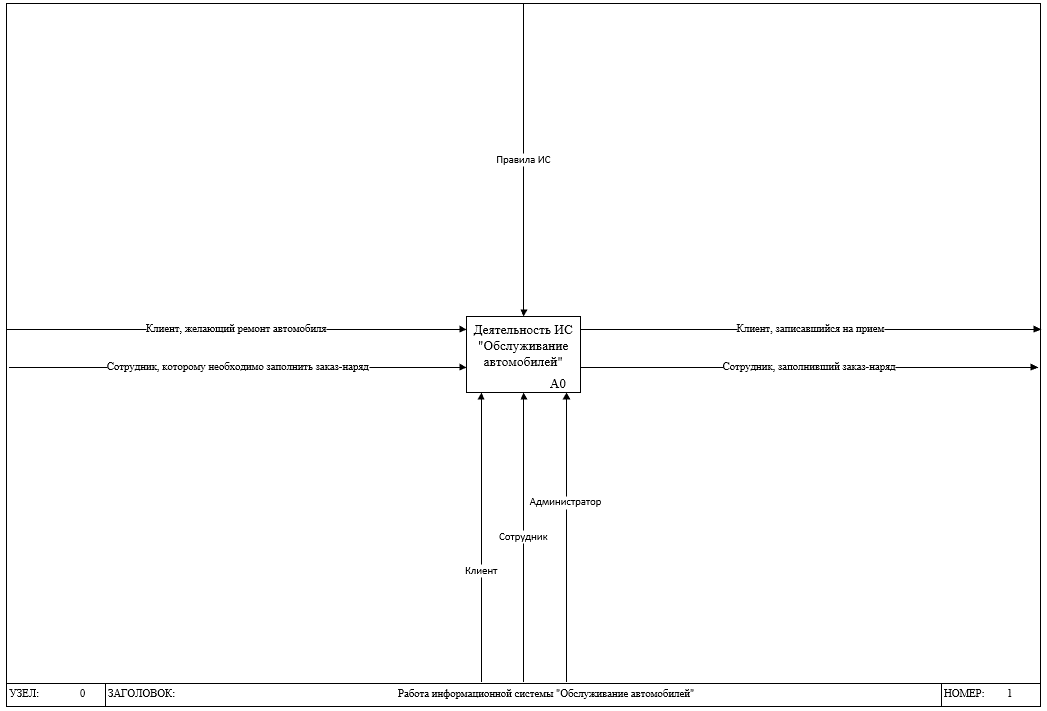


Рисунок 4 – контекстная диаграмма.

Для демонстрации деталей в процессе необходимо спроектировать диаграмму декомпозиции.

Диаграмма декомпозиции – это функциональная схема, описывающая контекстную диаграмму.

На рисунке 5 представлена диаграмма декомпозиции ИС, которая расписывает функции ИС. На диаграмме раскрываются основные функции пользователей, а именно: авторизация, действия администратора, действия сотрудника, действия клиента.

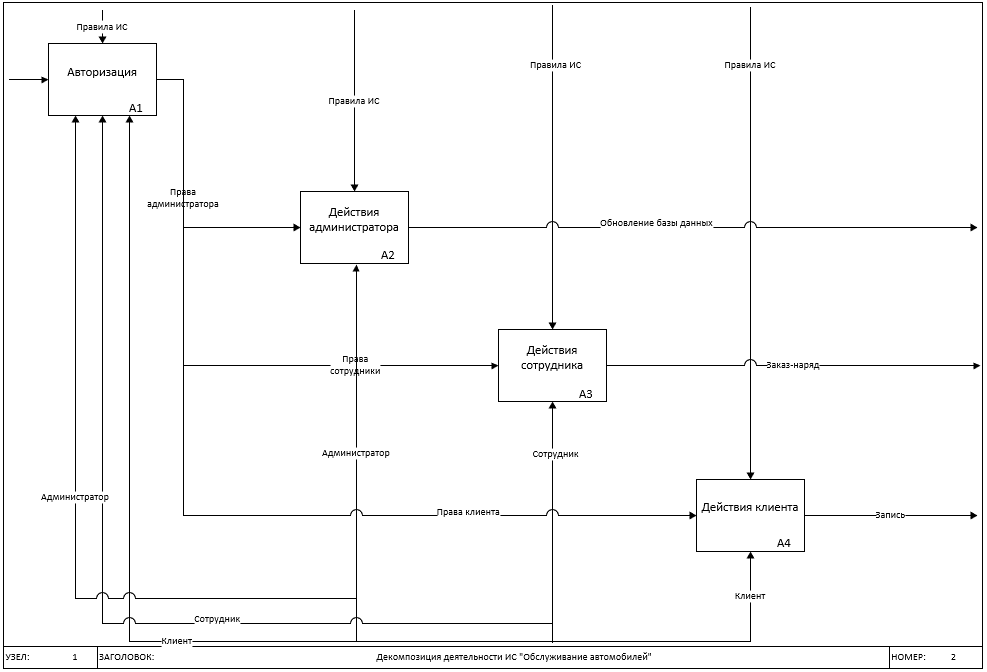


Рисунок 5 – диаграмма декомпозиции.

На рисунке 6 представлена декомпозиция функции «Действия администратора». Функция содержит 2 отдельных подфункции: «Распределение прав доступа», «Регистрация нового пользователя». Диаграмма декомпозиции также содержит входные данные, выходные данные, управление механизмом и пользователя.

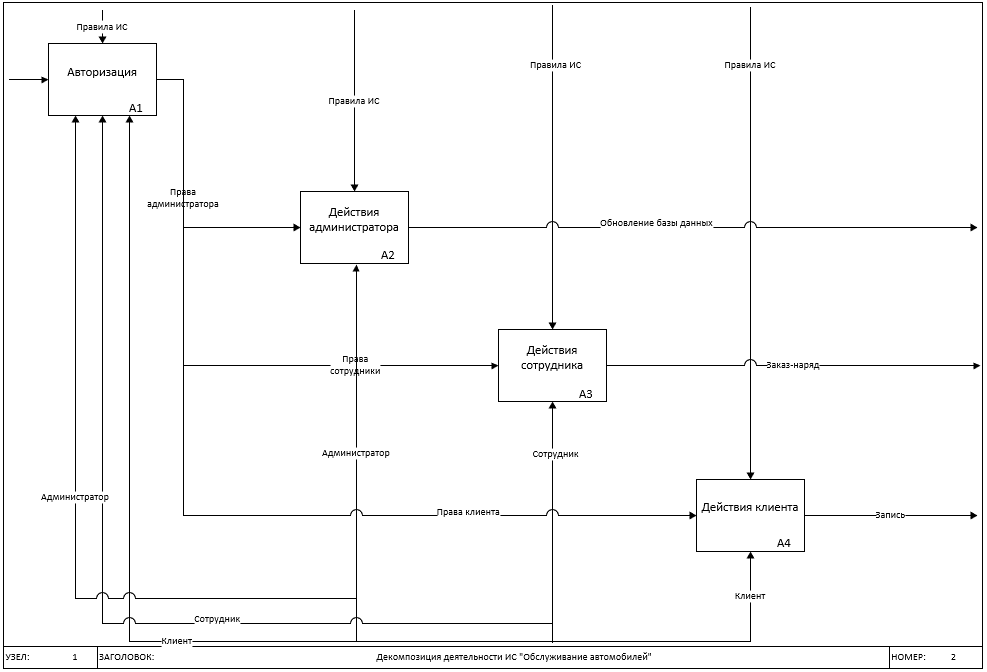


Рисунок 6 – диаграмма декомпозиции функции «Действия администратора».

Рисунок 7 представляет диаграмму декомпозиции функции «Действия сотрудника». На ней изображены 2 подфункции: «Заполнение формы выполнения работы» и «Сохранение данных».

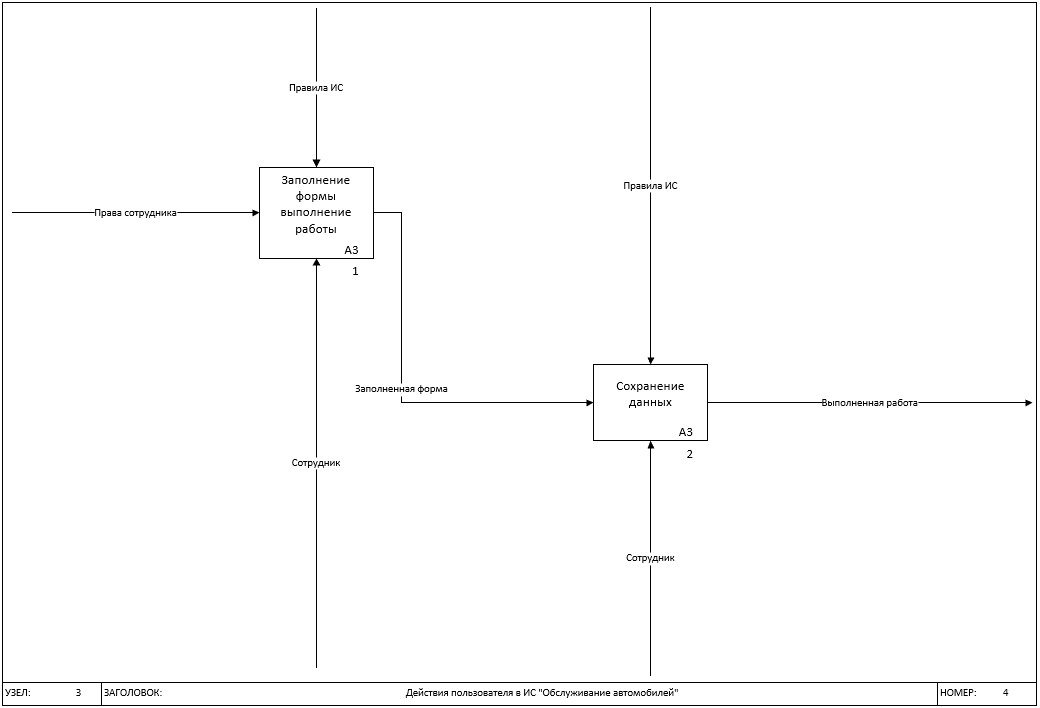


Рисунок 7 – диаграмма декомпозиции функции «Действия сотрудника».

Рисунок 8 представляет диаграмму декомпозиции функции «Действия клиента». На ней изображены 2 подфункции: «Запись на прием» и «Сохранение данных».

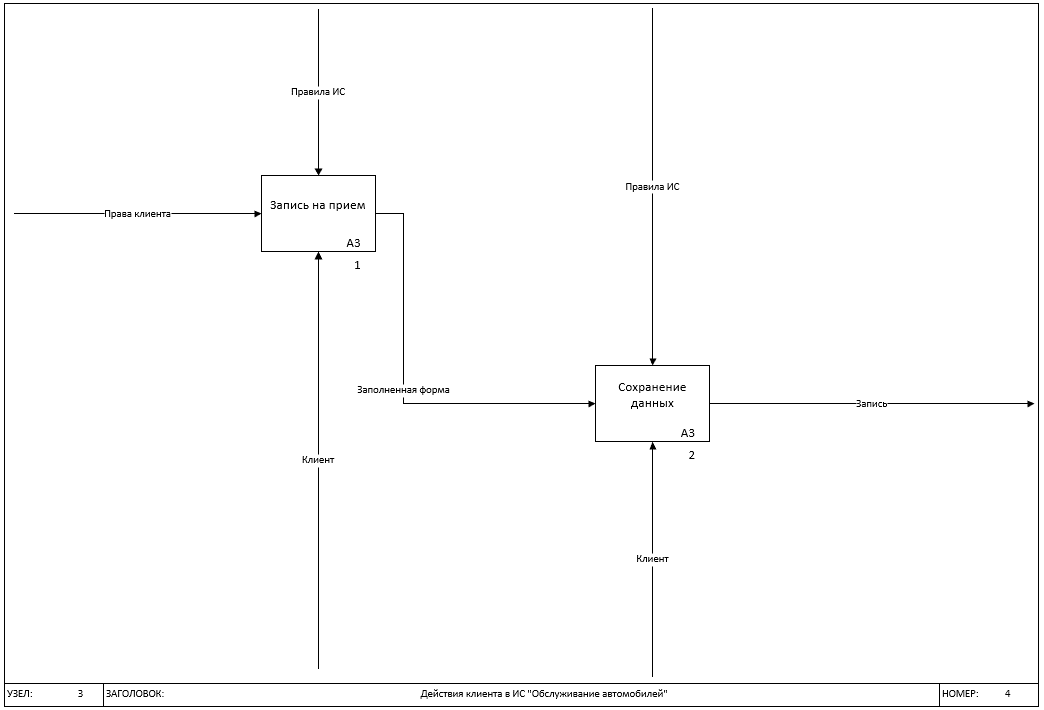


Рисунок 8 – диаграмма декомпозиции функции «Действия клиента».

На рисунке 9 показана диаграмма потоков данных, которая описывает передачу данных между разными функция и объектами ИС.

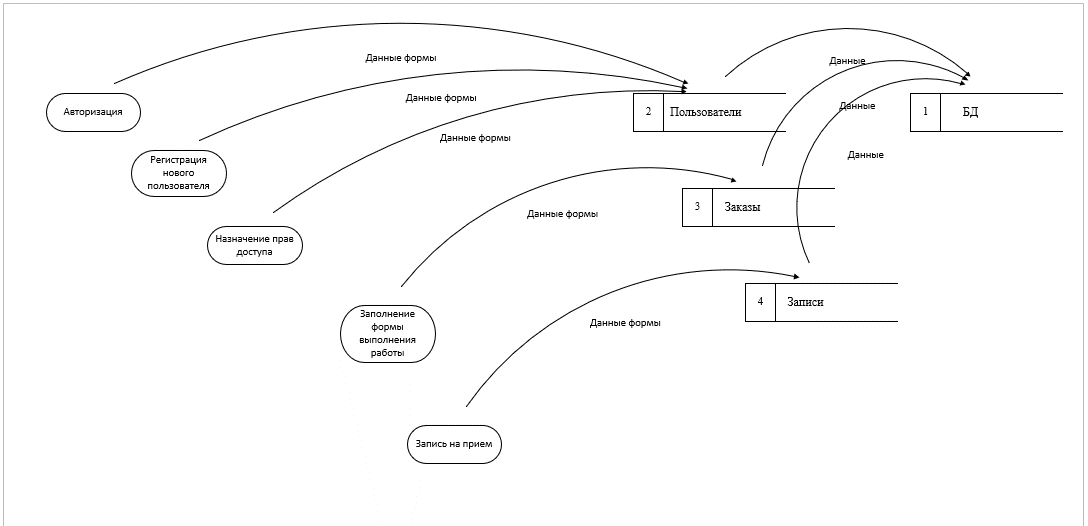


Рисунок 9 – диаграмма потоков данных.

Подробное функциональное проектирование позволяет глубже исследовать проектную область и цель проекта, что может позволить сэкономить время на разработке информационной системы, а также позволит избежать переписывания кода.

* 1. Проектирование базы данных

Проектирование базы данных начинается с концептуального проектирование базы данных.

Концептуальное проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных. Концептуальное проектирование демонстрирует инфологическую модель данных. Для наглядности инфологическая модель была разделена на логические модули и предоставлена в 3 рисунках. На рисунке 10 представлена логический модуль «Автомобили»

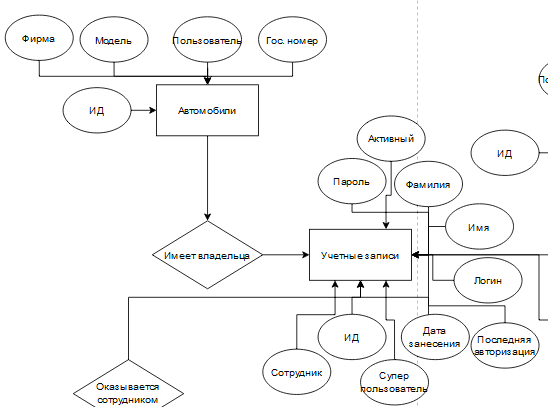


Рисунок 10 – часть инфологической модели, связанная с автомобилями.

В неё входит 3 объекта: фирмы, модели и автомобили. Автомобили связаны с клиентом, у которого имеется учетная запись в ИС.

На рисунке 11 представлена логический модуль «Права доступа».

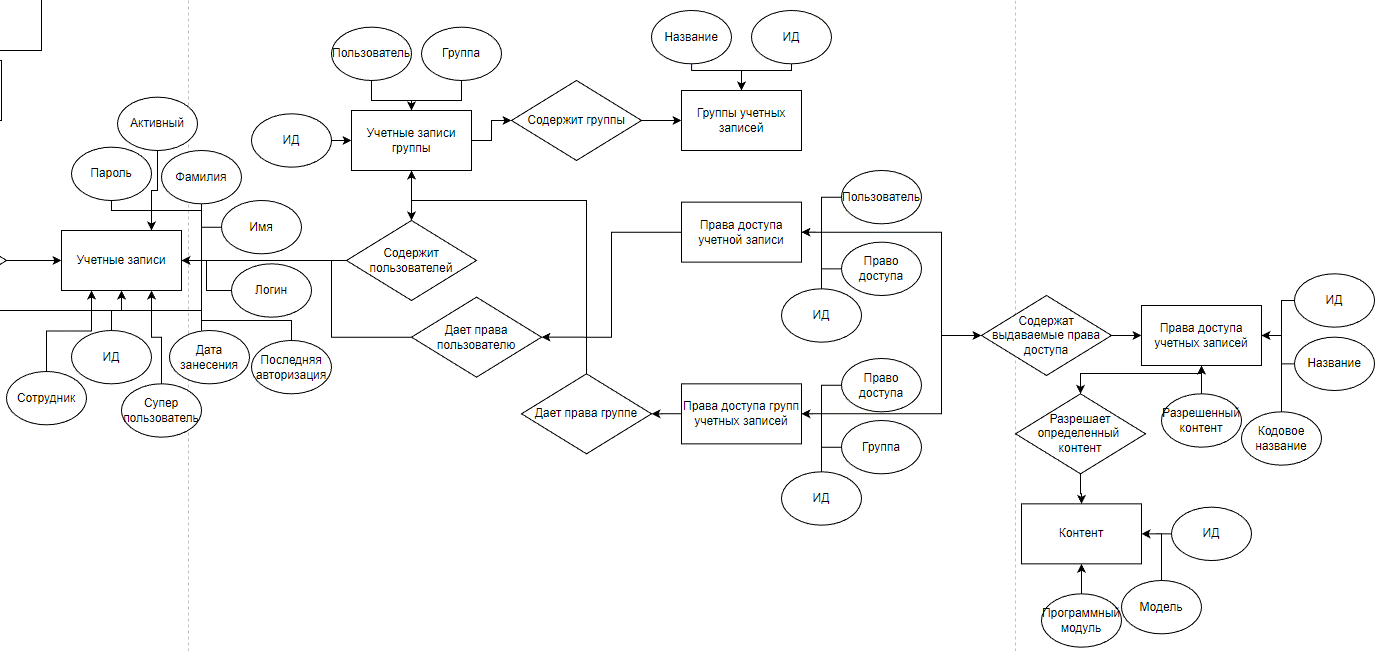


Рисунок 11 – часть инфологической модели, связанная с правами доступа.

В неё входит 5 объектов: группы, права доступа, контент, права доступа группы, права доступа учетной записи и учетные записи группы.

На рисунке 12 представлена логический модуль «Заказы».

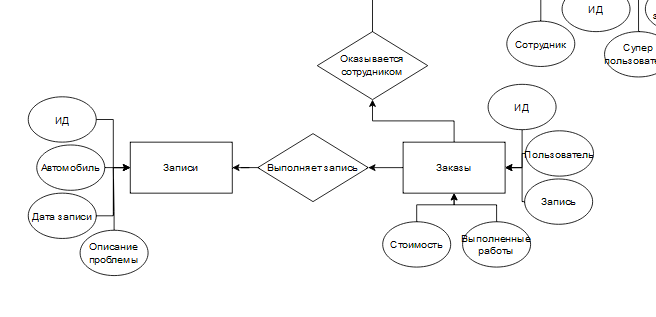


Рисунок 12 – часть инфологической модели, связанной с заказами.

В неё входит 6 объектов: записи, заказы, материалы для заказов, материалы, инструменты для заказов, инструменты.

Далее происходит преобразование концептуальной модели в логическую модель, по формальным правилам. Таким образом, логическое (даталогическое) проектирование — создание схемы базы данных на основе конкретной модели данных, например, реляционной модели данных.

Даталогическая модель данных отображает сущности и связи между ними. На рисунке 13, 14, 15 представлена даталогическая модель ИС, в которой изображено 16 сущностей. Для наглядности даталогическая модель была разделена на логические модули и предоставлена в 3 рисунках.

На рисунке 13 изображены 7 сущностей: auth\_group\_permissions, auth\_group, auth\_permission, django\_content\_type, auth\_user\_user\_permissions, auth\_user\_groups, auth\_user.

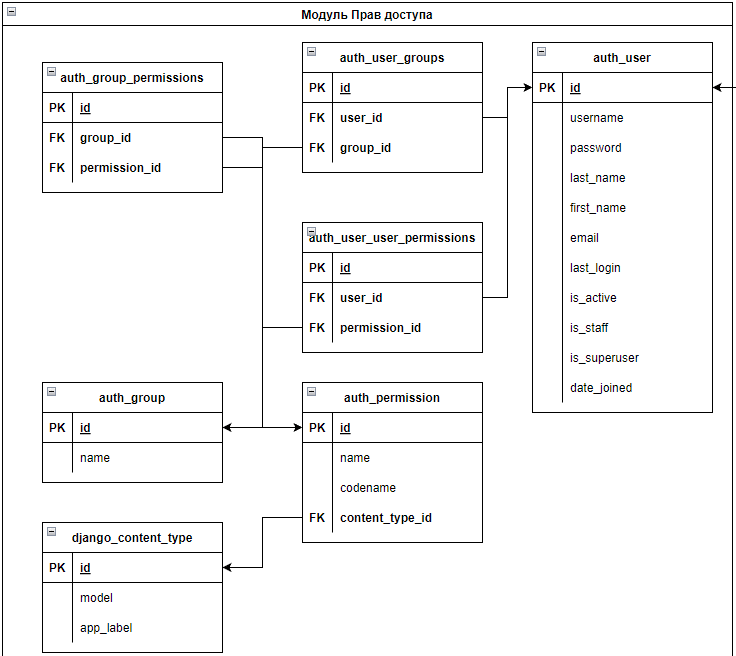


Рисунок 13 – часть даталогической модели, связанной с правами доступа.

На рисунке 14 изображена 1 сущность: cars.

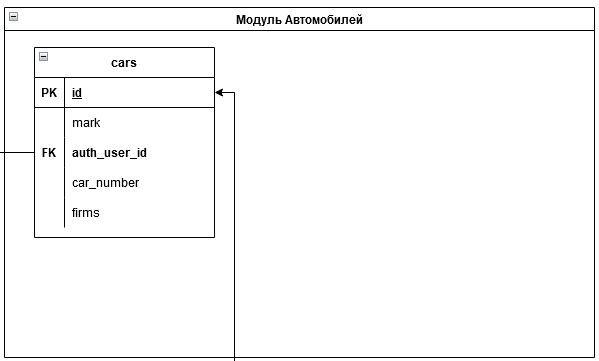


Рисунок 14 – часть даталогической модели, связанная с автомобилями.

На рисунке 15 изображены 2 сущности: orders, records.

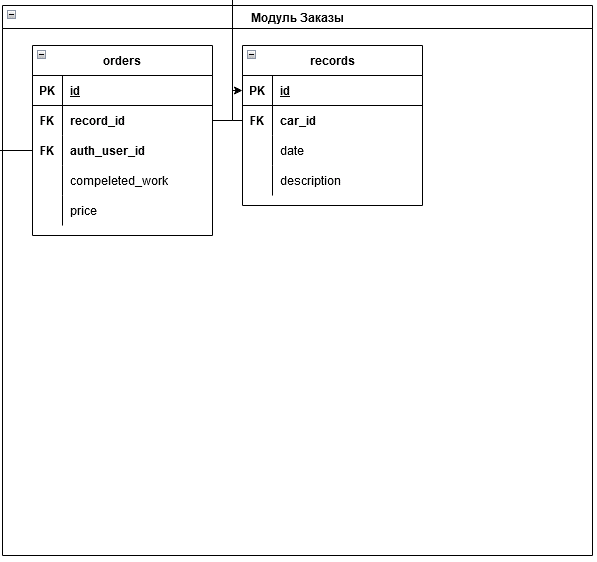


Рисунок 15 – часть даталогической модели, связанной с заказами.

На рисунке 16 представлена ER-диаграмма базы данных. Она содержит 10 таблицы информационной системы, 3 таблицы для фреймворка Django и 2 таблицы базы данных SQLite. Таблицы Django хранят выполненные миграции и сессии пользователей. Таблицы SQLite хранит настройки для базы данных.

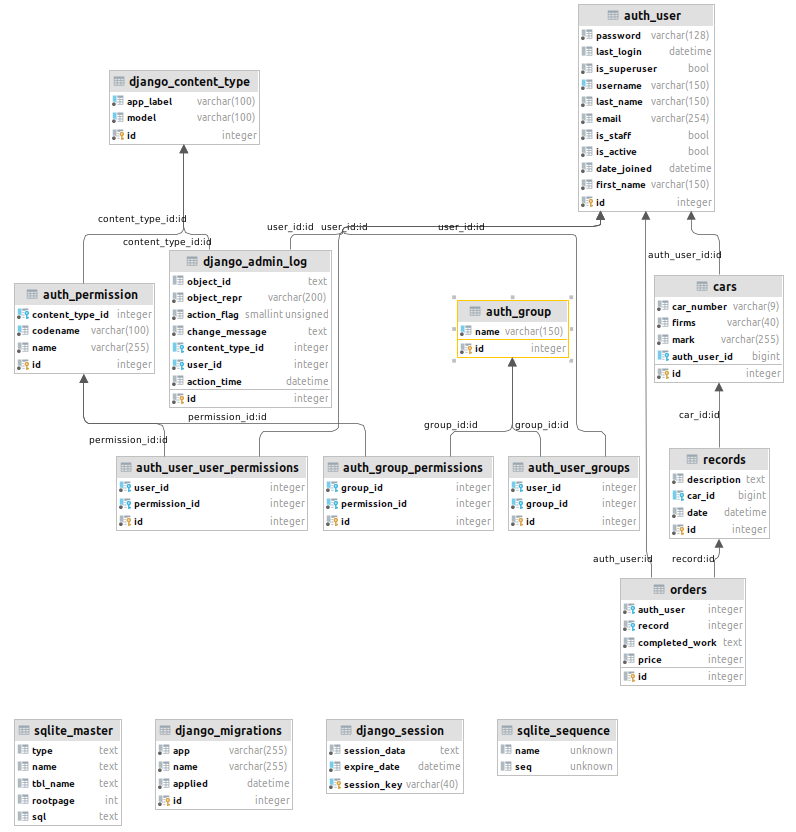


Рисунок 16 – ER-модель.

Полный перечень таблиц базы данных ИС представлен в таблице 4.

Таблица 4 — Таблицы ER-модели.

|  |  |
| --- | --- |
| Таблица | Описание |
| cars | Таблица автомобилей. |
| records | Таблица записей на прием. |
| orders | Таблица заказов. |
| auth\_user | Таблица учетных записей. |
| auth\_group | Таблица групп. |
| auth\_user\_groups | Таблица учетных записей группы. |
| auth\_group\_permissions | Таблица прав доступа группы. |
| auth\_permission | Таблица прав доступа. |
| auth\_user\_user\_permissions | Таблица прав доступа учетной записи. |
| django\_admin\_log | Таблица логирования от Django. |
| django\_content\_type | Таблица модулей ИС от Django. |
| sqlite\_sequence | Таблица конфигураций БД SQLite. |
| django\_migrations | Таблица выполненных миграций от Django. |
| django\_session | Таблица сессий от Django. |

БД приведена к 1 начальное форме (далее – НФ), т.к. все поля, принимаемые больше одного значения, декомпозированы. Также БД имеет 2 и 3 НФ, т.к. каждый не ключевой атрибут приведен к неприводимости, и каждый не ключевой атрибут не транзитивно (непосредственно) зависит от первичного ключа.

* 1. Проектирование интерфейса

Для разработки пользовательского интерфейса был выбран инструмент draw.io – браузерный инструмент для создания диаграмм, блок-схем и прочего.

В результате проектирование интерфейса будущей информационной системы были спроектированы прототипы семи страниц: главная страница, страница авторизации, страница регистрации, страница панели администратора, страница добавление автомобиля клиентом, страница записи клиентом, страница формы выполнения работы.

На рисунке 17 изображено проектирование главной странице. На ней пользователя информируют о достижениях компании.

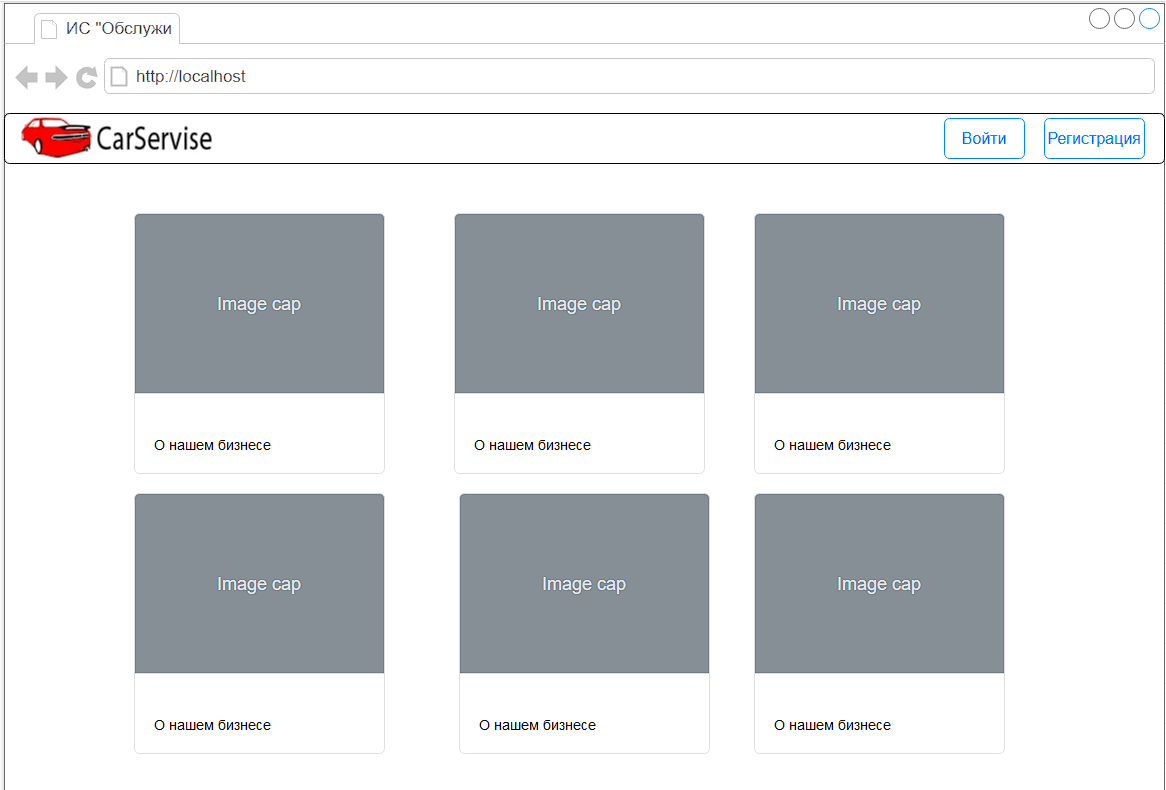


Рисунок 17 – проектирование главной страницы.

На рисунке 18 изображено проектирование страницы регистрации. На ней пользователь имеет право зарегистрироваться в информационной системе.

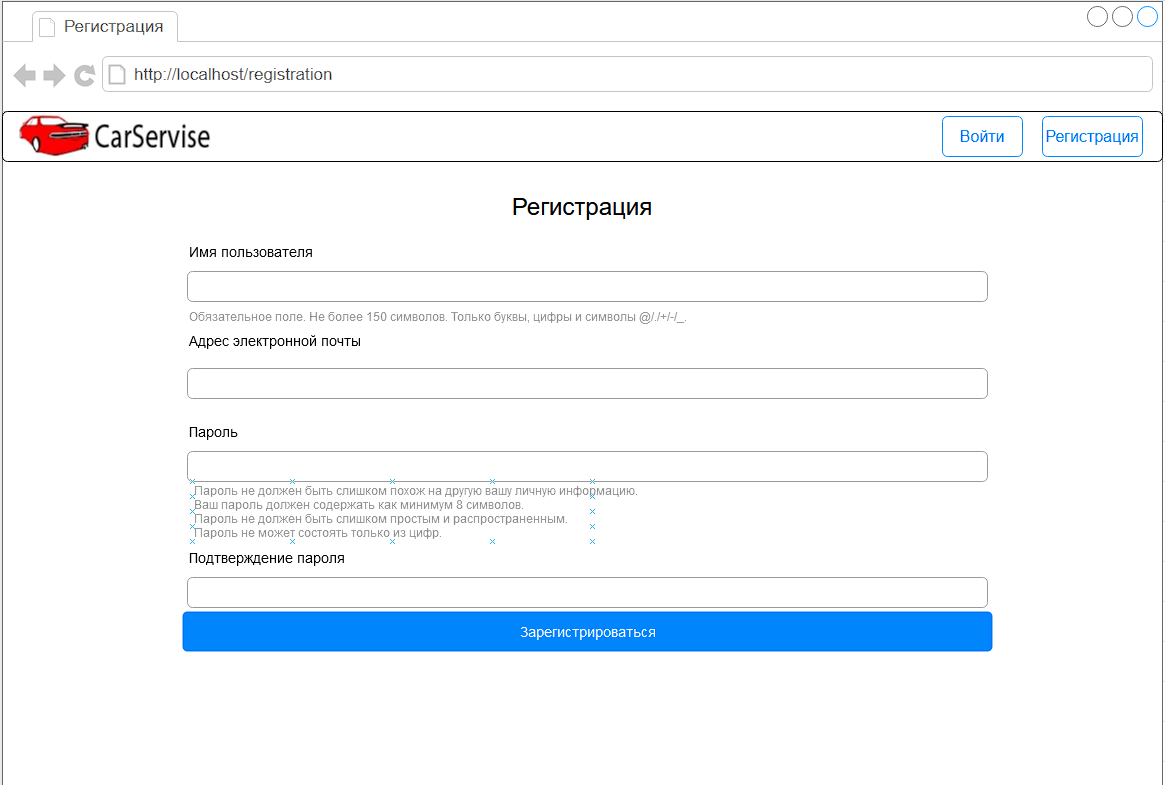


Рисунок 18 – проектирование страницы регистрации.

На рисунке 19 изображено проектирование страницы входа. На ней пользователь имеет право авторизироваться в информационной системе.

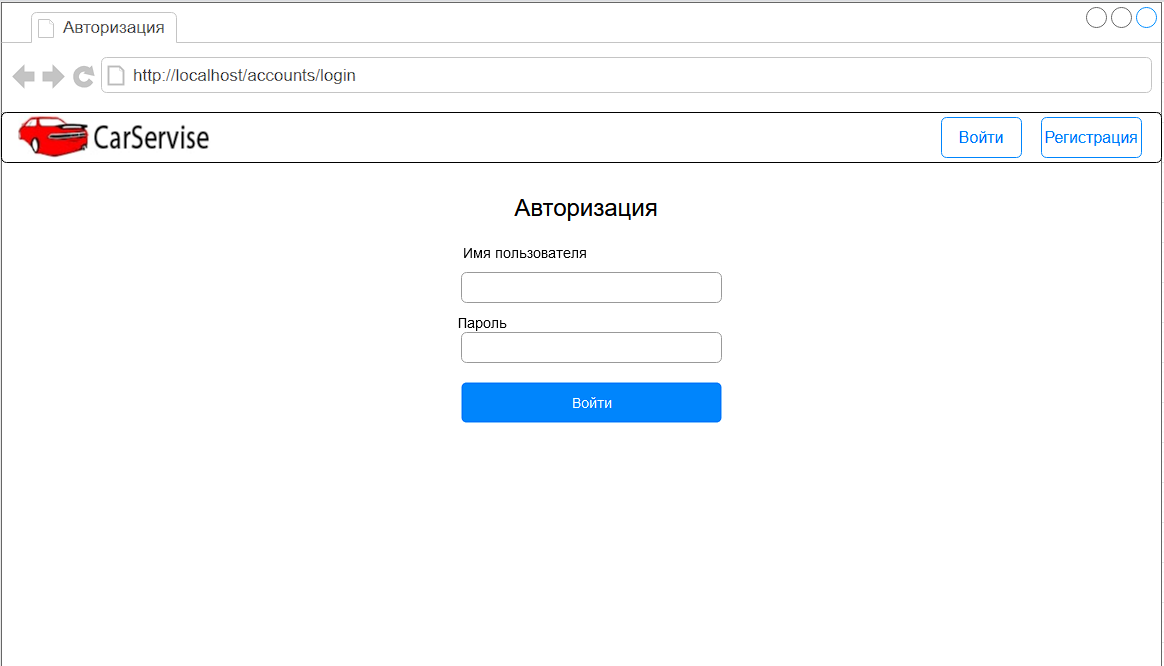


Рисунок 19 – проектирование страницы авторизации.

На рисунке 20 изображено проектирование страницы панели администратора. На ней администратор имеет право изменять данные информационной системы.

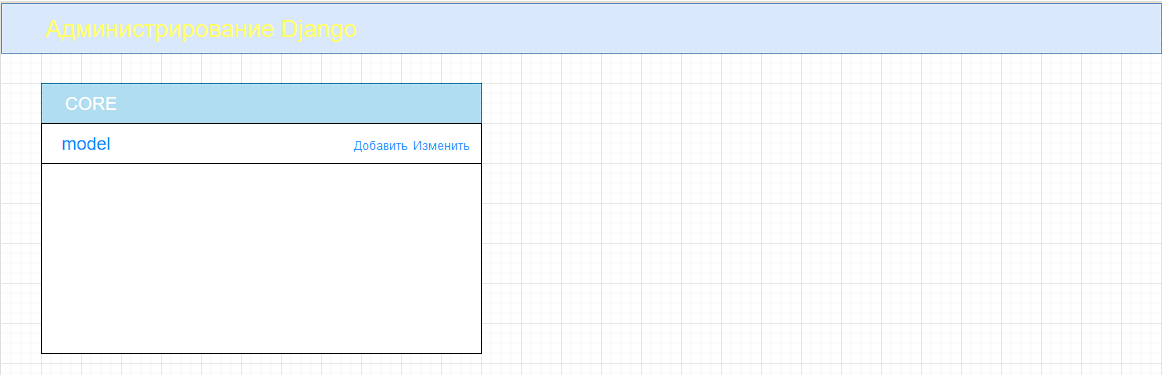


Рисунок 20 – проектирование страницы панели администратора.

На рисунке 21 изображено проектирование страницы выполнения заказа. На ней сотрудник заполняет форму о том какие работы проведены при выполнении заказа по записи и в какую стоимость обошлись работы.

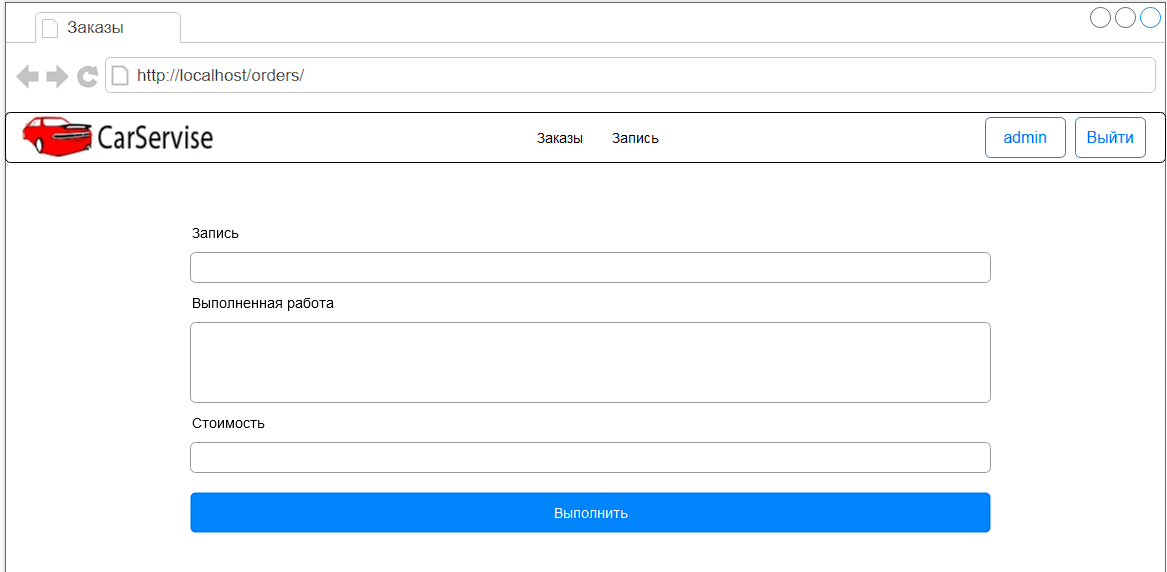


Рисунок 21 – проектирование страницы выполнения заказа.

На рисунке 22 изображено проектирование страницы добавления автомобиля. На ней клиент имеет право добавить данные своего автомобиля для информационной системы.

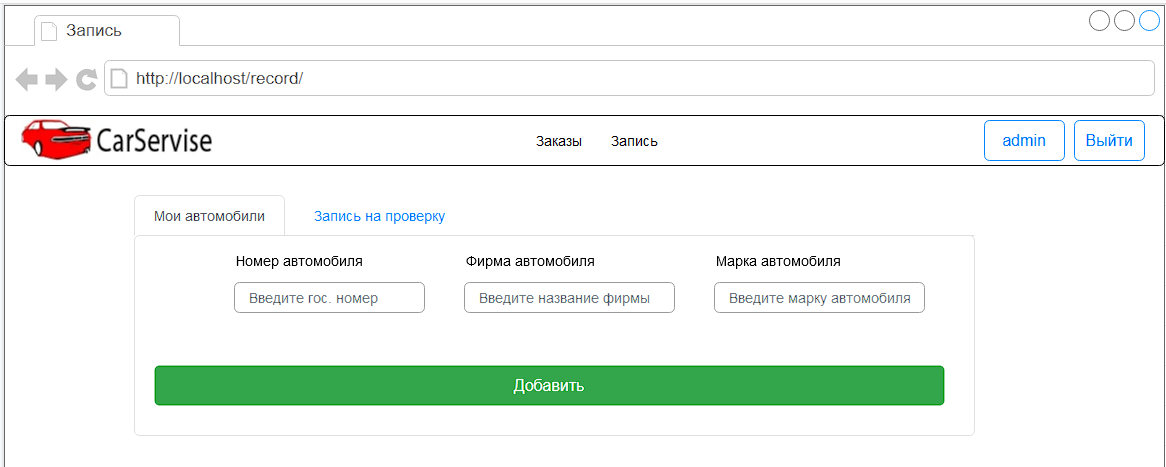


Рисунок 22 – проектирование страницы добавления автомобиля.

На рисунке 23 изображено проектирование страницы записи. На ней клиент имеет право записать свой автомобиль на осмотр и указать имеющиеся проблемы.

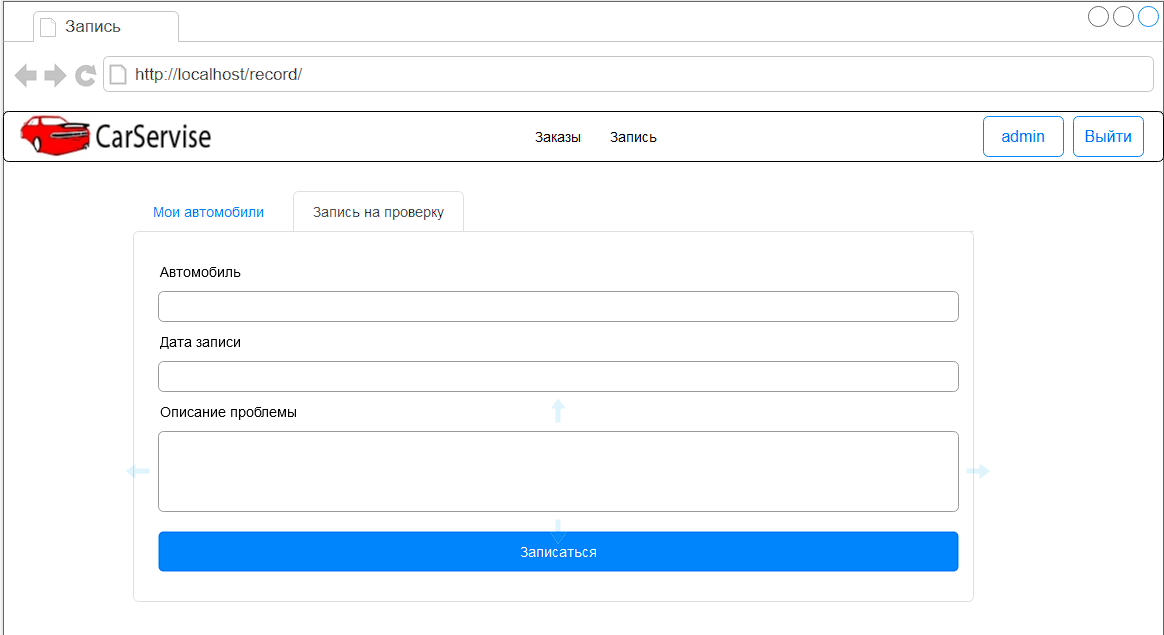


Рисунок 23 – проектирование страницы записи.

Благодаря созданию прототипов, разработка непосредственно программного продукта будет значительно облегчена, за счёт наглядных примеров будущих страниц ИС.

1. Разработка ИС
   1. Разработка интерфейса ИС

В информационной системе «Обслуживание автомобилей» разработка интерфейса происходит на стороне сервера. Серверная часть информационной системы разработана при помощи фреймворка для языка программирования python – Django. Django использует шаблонизатор Ninja, который позволяет в файлах с html-разметкой использовать python-вставки.

Всего было разработано 5 шаблонов:

1. general – главный шаблон, который содержит в себе подключение css-стилей, js-скриптов, а также меню. Часть кода шаблона представлен на рисунке 24.1 и внешний вид на рисунке 24.2.

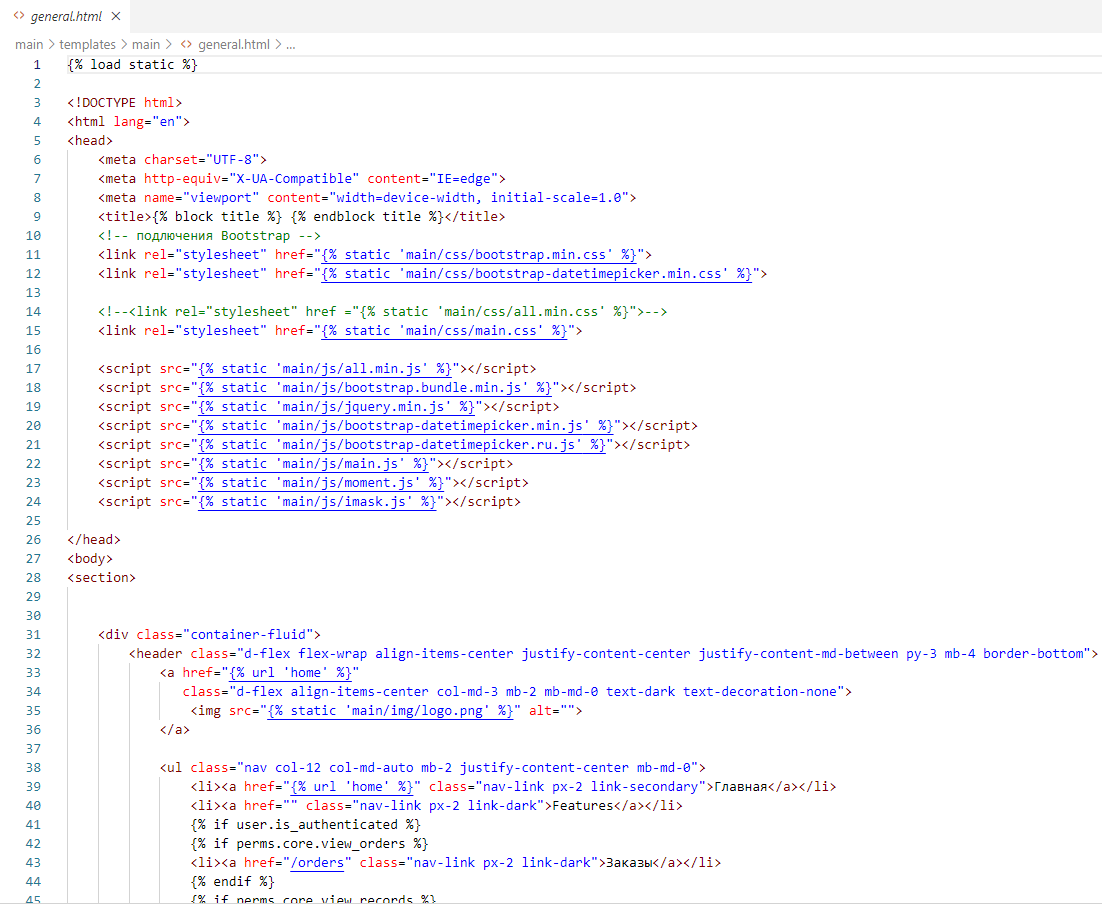


Рисунок 24.1 – часть кода шаблона “general.html”.

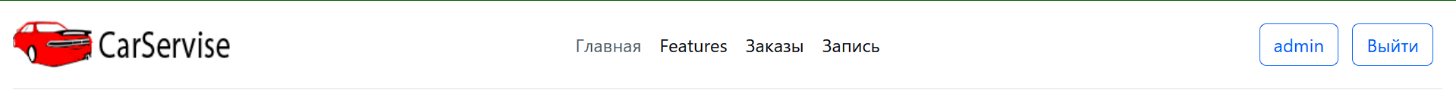


Рисунок 24.2 – внешний вид шаблона “general.html”.

1. login – интерфейс страницы входа. В ней располагается форма для входа в информационную систему. Часть кода шаблона представлен на рисунке 25.1 и внешний вид на рисунке 25.2.

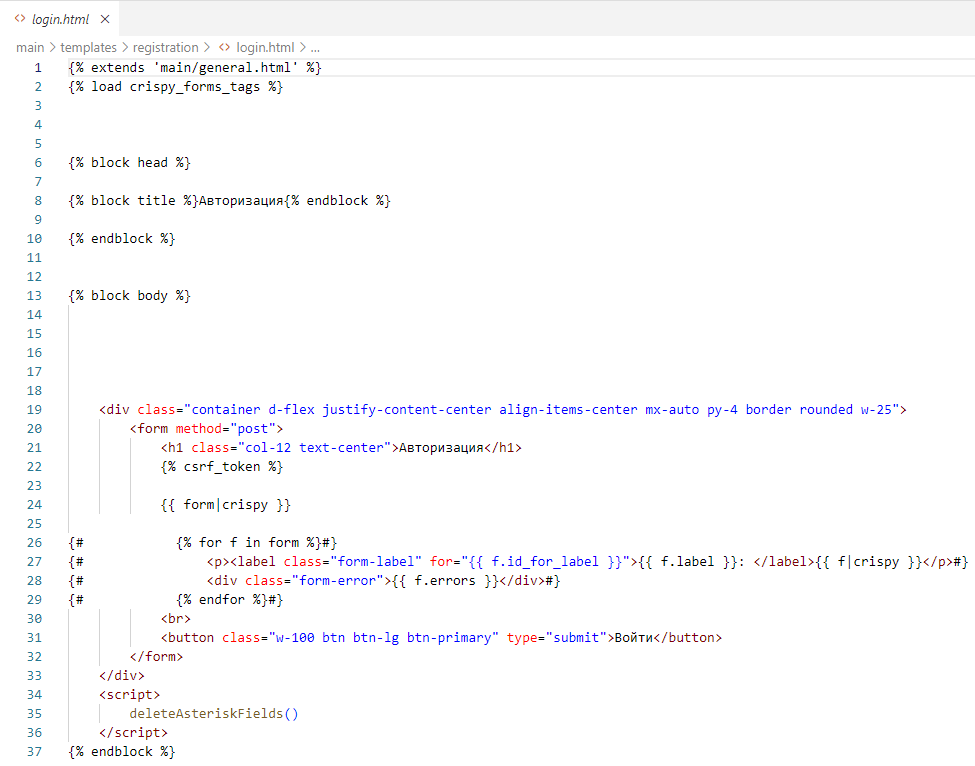


Рисунок 25.1 – часть кода шаблона “login.html”.

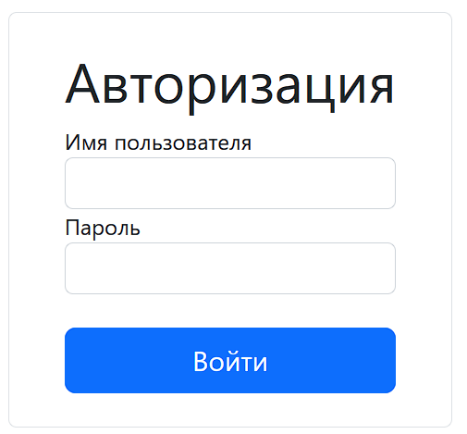


Рисунок 25.2 – внешний вид шаблона “login.html”.

1. registration – интерфейс страницы регистрация. В ней располагается форма для регистрации пользователя (клиент) в информационной системе. Часть кода шаблона представлен на рисунке 26.1 и внешний вид на рисунке 26.2.

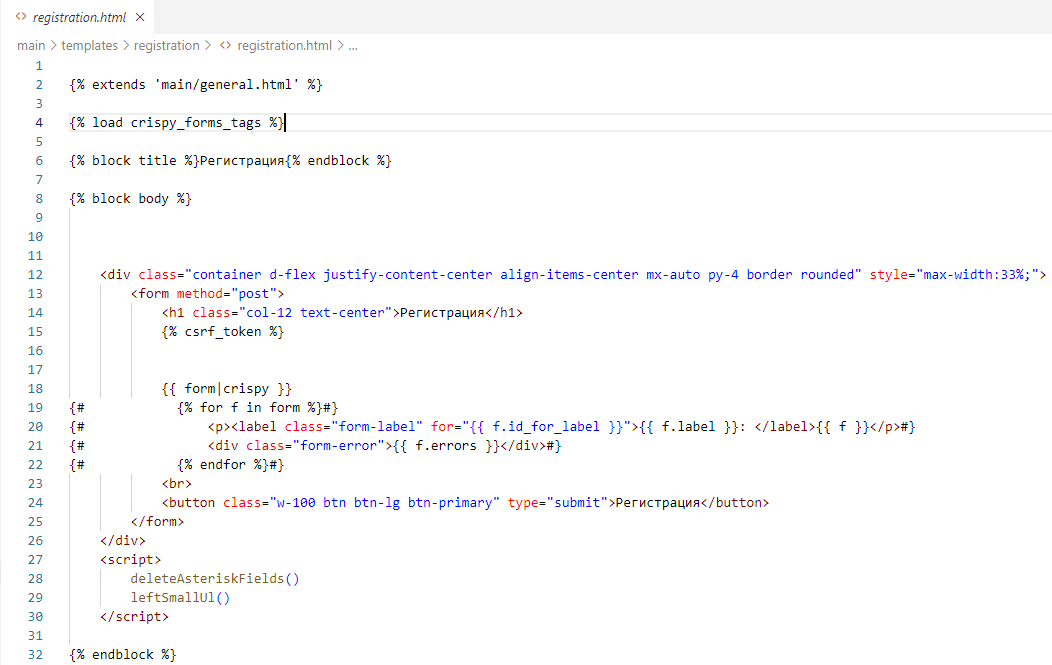


Рисунок 26.1 – часть кода шаблона “registration.html”.

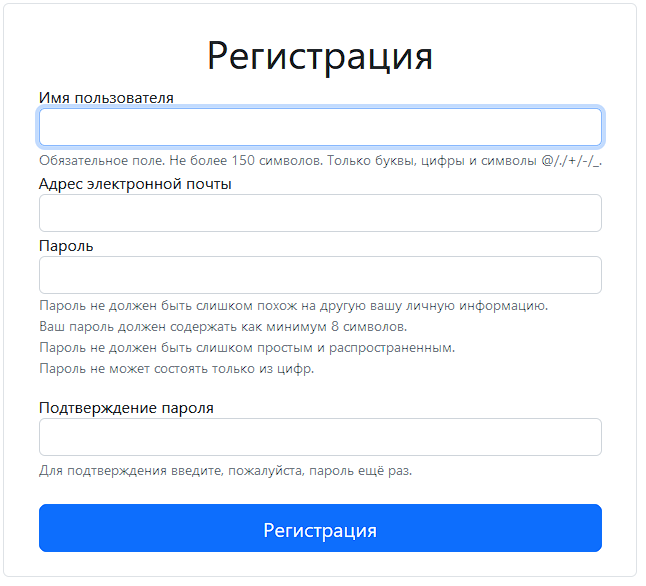


Рисунок 26.2 – внешний вид шаблона “registration.html”.

1. record – интерфейс страницы запись. В ней отображаются формы для добавления автомобиля пользователем и записи на осмотр. Часть кода шаблона представлен на рисунке 27.1 и внешний вид на рисунке 27.2.

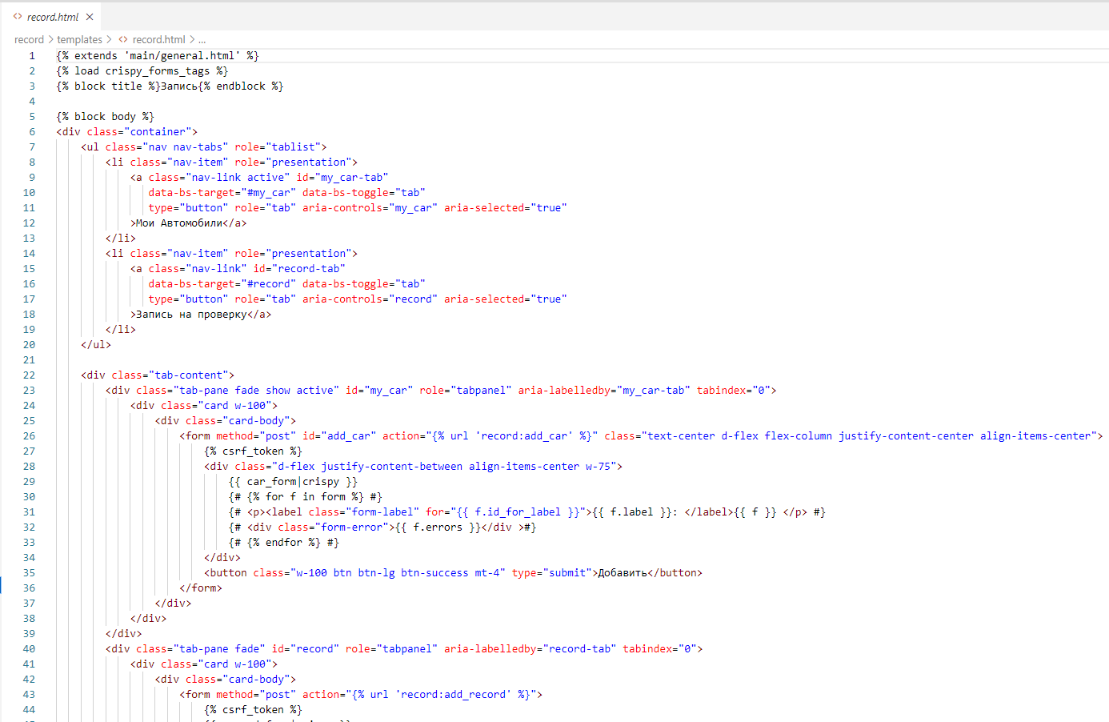


Рисунок 27.1 – часть кода шаблона “record.html”.

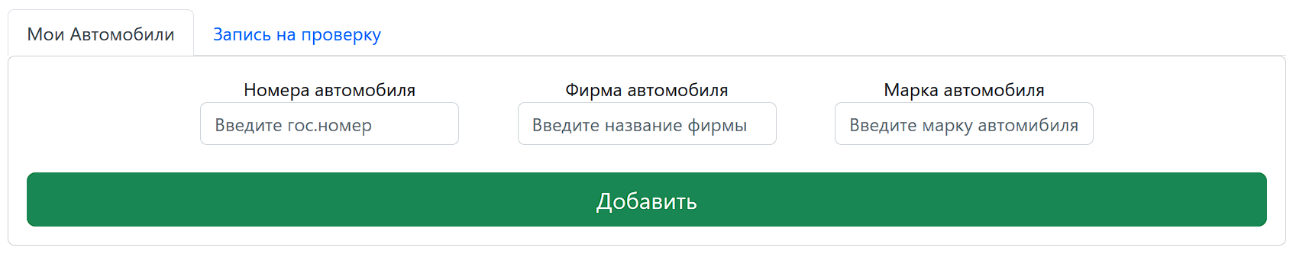


Рисунок 27.2 – внешний вид шаблона “record.html”.

1. orders – интерфейс страницы заказы. В ней отображается форма выполнения заказа для сотрудника.

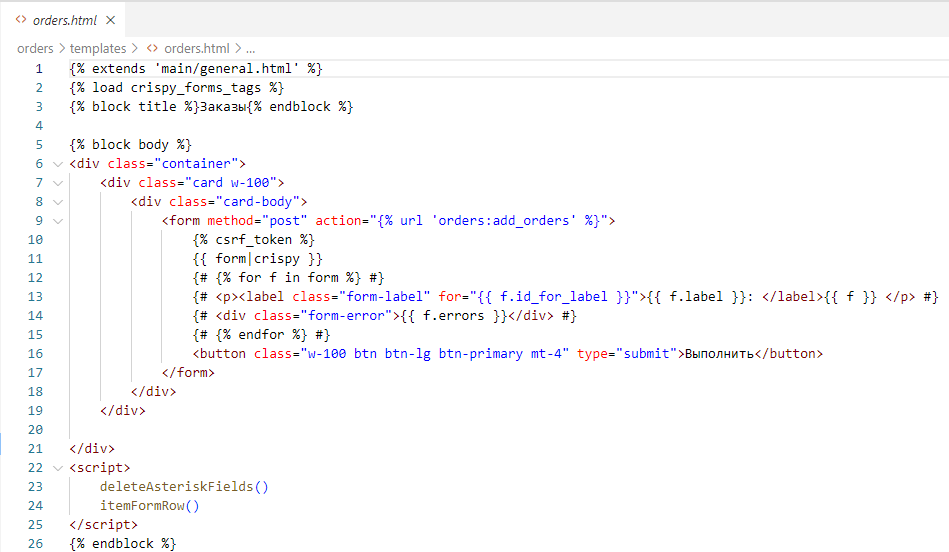


Рисунок 28.1 – часть кода шаблона “orders.html”.

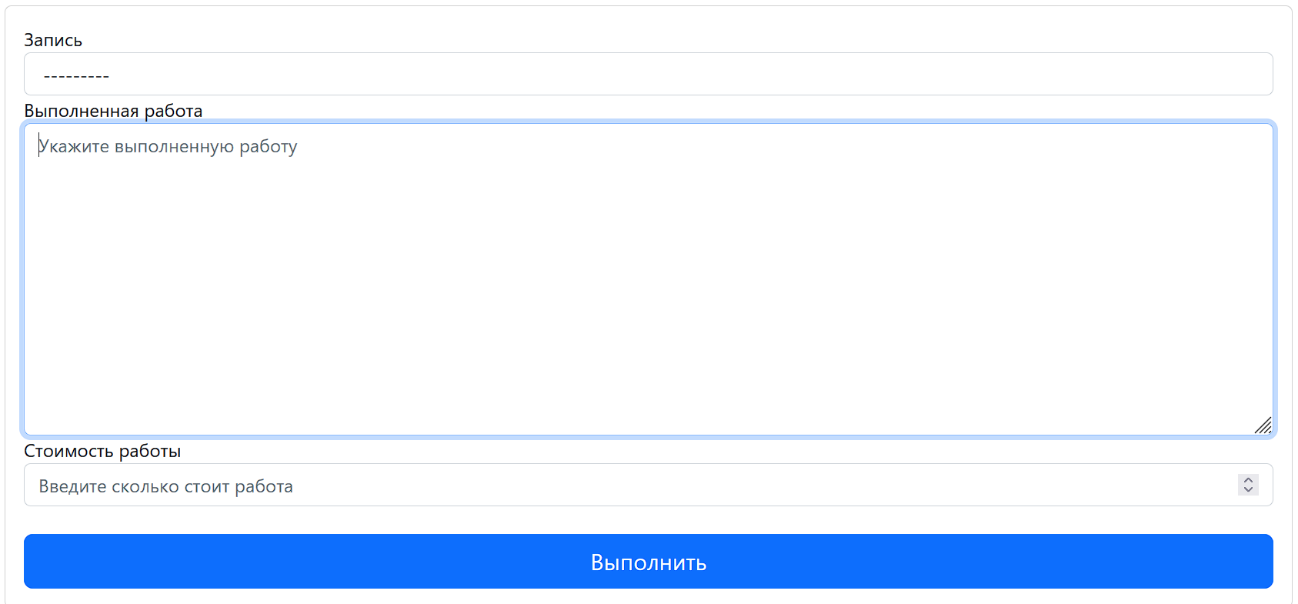


Рисунок 28.2 – внешний вид шаблона “orders.html”.

* 1. Разработка базы данных ИС

В качестве базы данных для реализации программного продукта была выбрана база данных SQLite. Всего в базе данных располагается 16 таблиц (ER-модель представлена на рисунке 16).

Среди них: таблица автомобилей, таблица записей на прием, таблица заказов, таблица учетных записей, таблица групп, таблица учетных записей группы, таблица прав доступа группы, таблица прав доступа, таблица прав доступа учетной записи, таблица логирования от Django, таблица модулей ИС от Django, таблица конфигураций БД SQLite, таблица выполненных миграций от Django, таблица сессий от Django.

Все таблицы спроектированы так, чтобы избежать избыточности, при этом достичь максимальной скорости работы, БД нормализирована, и приведена к третьей нормальной форме, таблицы в которой связываются друг с другом посредством внешних ключей.

* 1. Разработка ИС

Подключение к базе данных SQLite осуществляется в конфигурационном файле settings.py (рисунок 29).



Рисунок 29 – конфигурация для подключения к базе данных

Для сохранения данных в базу данных использовались разработанные модели и формы к ним. На рисунке 30.1 представлена модель Records, которая позволяет манипулировать данными таблицы records.

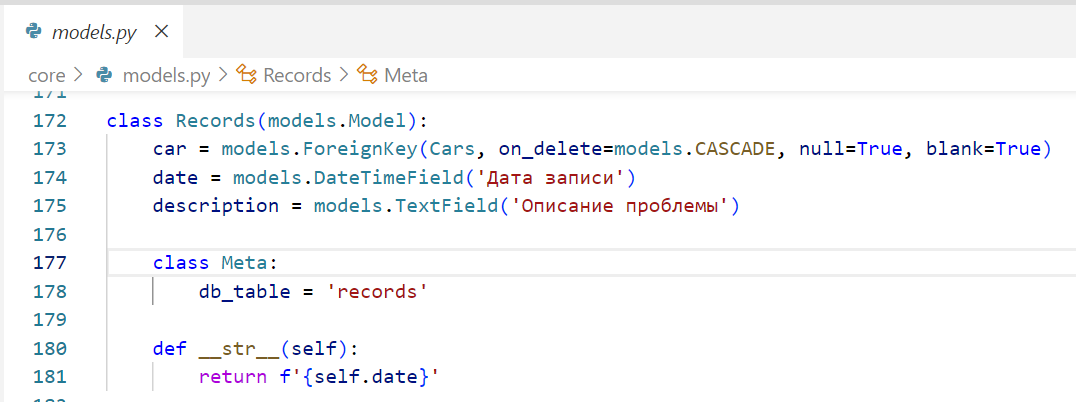


Рисунок 30.1 – модель Records

На рисунке 30.2 представлена форма RecordForm, которая добавляет данные в таблицу records.

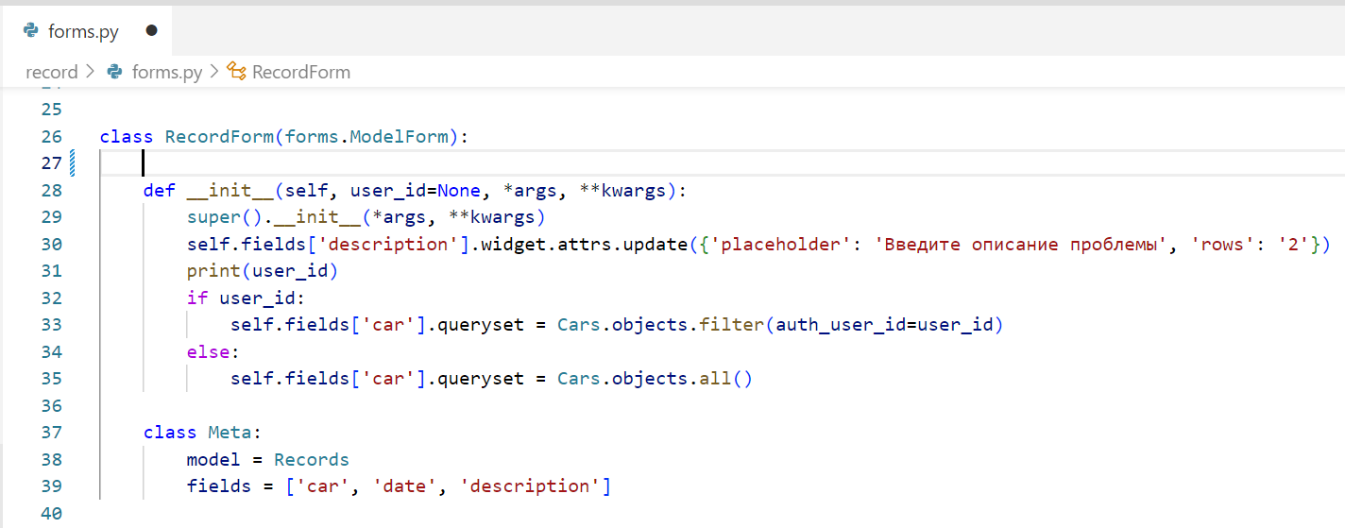


Рисунок 30.2 – форма RecordForm

На рисунке 30.3 представлено представление RecordView, которое отображает данные в интерфейсы.

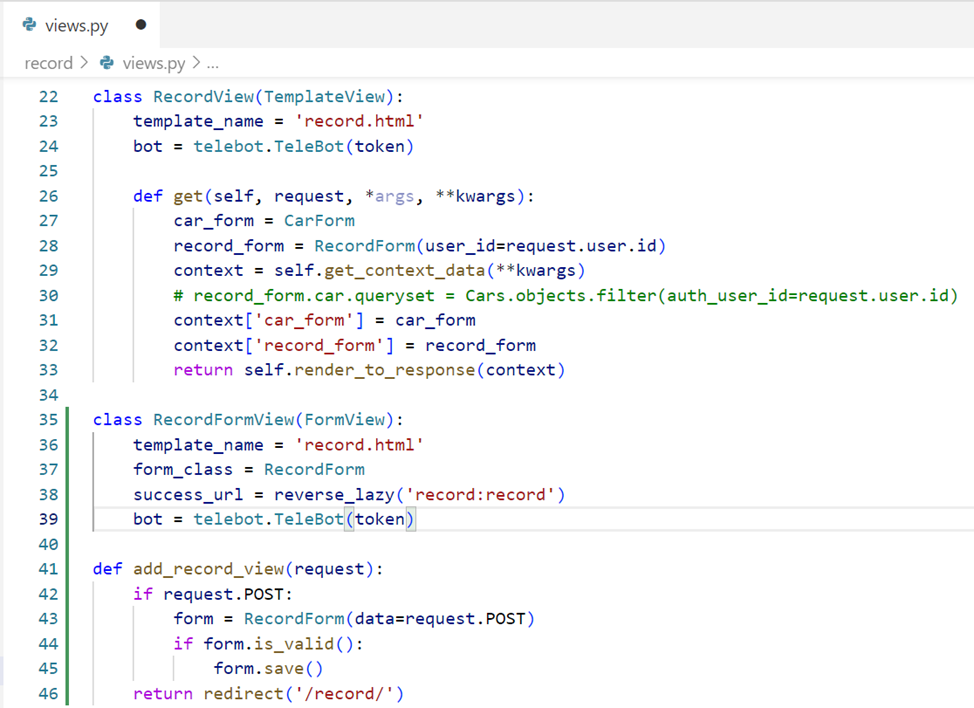


Рисунок 30.3 – представление RecordView

На рисунке 30.3 представлено представление RecordView, которое отображает данные в интерфейсы.

1. Разработка ИС
   1. Разработка интерфейса ИС