**2 Проектирование программного продукта**

**2.1 Разработка модели данных. Формирование структуры проекта**

**2.1.1 Разработка модели данных**

Проектирование программного продукта следует начать с разработки модели данных. Для разработки сайта недостаточно знать лишь поставленные задачи и иметь исходные данные, но необходимо и описать функциональную структуру, отражающую его процессы.

Модель данных представляет собой множество структур данных, ограничений целостности и операций манипулирования данными. Моделью данных называется совокупность структур данных и операций их обработки.

Для того, чтобы определить важность сайта, определить его место и оценить отношения с другими сайтами необходимо составить его функциональное описание. Функциональное описание исходит из того, что всякий сайт выполняет некоторые функции: пассивно существует, обслуживает сайты более высокого порядка, служит средством для создания более совершенного сайта, обслуживает клиентов и т.п.

При анализе структуры сайта используется графическое описание, разновидностью которого является стандарт функционального моделирования IDEF0.

Методология IDEF0 является основой метода блочного моделирования. С ее помощью создаются модели широкого круга систем, состоящих из комбинации аппаратных, программных и людских ресурсов. Полученная модель представляет собой серию диаграмм.

Диаграмма представляет собой графическое представление данных линейными отрезками или геометрическими фигурами, позволяющее быстро оценить соотношение нескольких величин. Каждая диаграмма состоит из блоков и дуг, каждый блок представляет собой отдельное действие или функцию, а каждая дуга – объект, над которым в системе совершается действие. Описание IDEF0 модели построено в виде иерархической пирамиды, в вершине которой представляется общее описание сайта, а основание представляет собой множество более детальных описаний.

При построении модели в методологии IDEF0 используются три основных правила: построение модели начинается с описания системы в виде единого функционального блока на главной диаграмме, каждая диаграмма состоит из не менее трех и не более шести функциональных блоков, каждая диаграмма может содержать только элементы, входящие в функциональный блок, декомпозицией которой является данная диаграмма.

Рассмотрим построение функциональной модели сайта «DREAM CARS» (приложение Б).

В приложении Б (лист 1) изображена контекстная диаграмма, отображающая работу сайта, где выделены основные элементы программы. Входными данными являются данные пользователя и информация об автомобиле. Выходными данными сайта являются данные пользователя, договор о покупке авто и карточка авто.

К механизмам разрабатываемого приложения относятся пользователь, администратор, программное обеспечение. Пользователь может выступать в качестве клиента или администратора. Управляющими элементами являются ТЗ (приложение А) и законодательство Республики Беларусь.

В приложении Б (лист 2) изображена декомпозиция контекстной диаграммы. Из рисунка видно, что структуру сайта можно разделить на три основных блока: регистрация, работа с карточками, поиск.

Процесс «Регистрация, Авторизация» предназначен для организации разграничения прав пользователей. Входными данными процесса являются данные пользователя.

Процесс «Работа с карточками» предназначен для просмотра карточек автомобилей, размещенных в каталоге. Входными данными процесса работы с карточками являются информация об авто.

Процесс «Поиск» предназначен для поиска автомобилей в каталоге. Входными данными процесса являются критерии поиска.

В зависимости от типа учетной записи пользователь может использовать различные функциональные возможности. Для осуществления действий по добавлению автомобиля в каталог необходимо стать администратором.

Для того чтобы стать администратором необходимо войти под логином и паролем администратора системы.

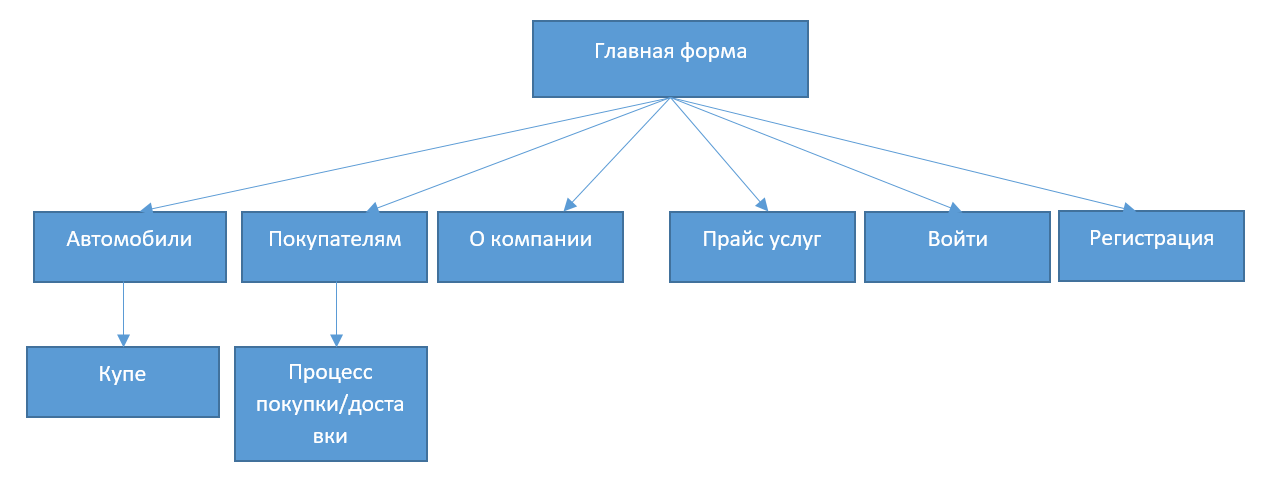
Администратор может осуществлять такие функции, как просмотр, редактирование, удаление, добавление информации на сайт.

В соответствии с моделью данных основными процессами сайта являются ознакомление с информацией, редактирование/удаление/добавление информации, просмотр информации об автомобилях, поиск, а также оставление заявки на страхование.

**2.1.2 Формирование структуры проекта**

В процессе выполнения учебного проекта необходимо создать сайт «DREAM CARS», который должен содержать всю необходимую информацию о автомобилях и функционал, описанный в приложении Г.

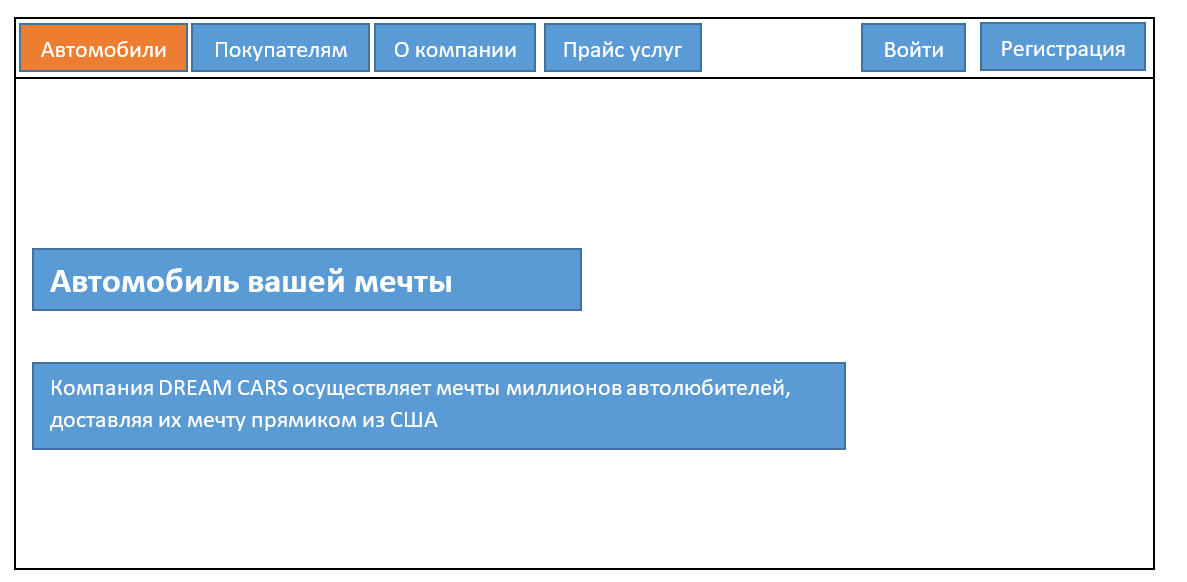
Формирование структуры проекта необходимо начинать с определения разделов меню, то есть, из каких страниц будет состоять программный продукт, и какие данные будет содержать каждая страница. Далее необходимо выбрать средства разработки и приступить к реализации.

****

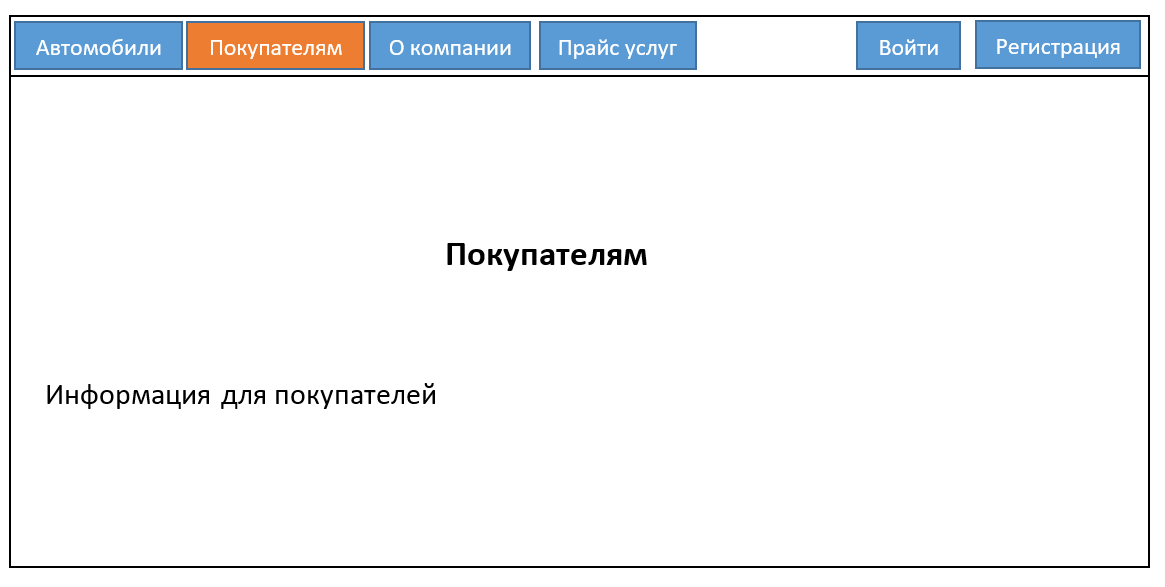
**Рисунок 2.1 – Карта сайта**

Карта сайта - это раздел, который позволит посетителю сориентироваться на сайте, уточнит, зачем нужен тот или иной раздел сайта, какую информацию посетитель там найдет. В общем случае карта представляет собой иерархическую модель структуры сайта. Для большинства веб-проектов наличие карты является необходимым условием успешной навигации по ним.

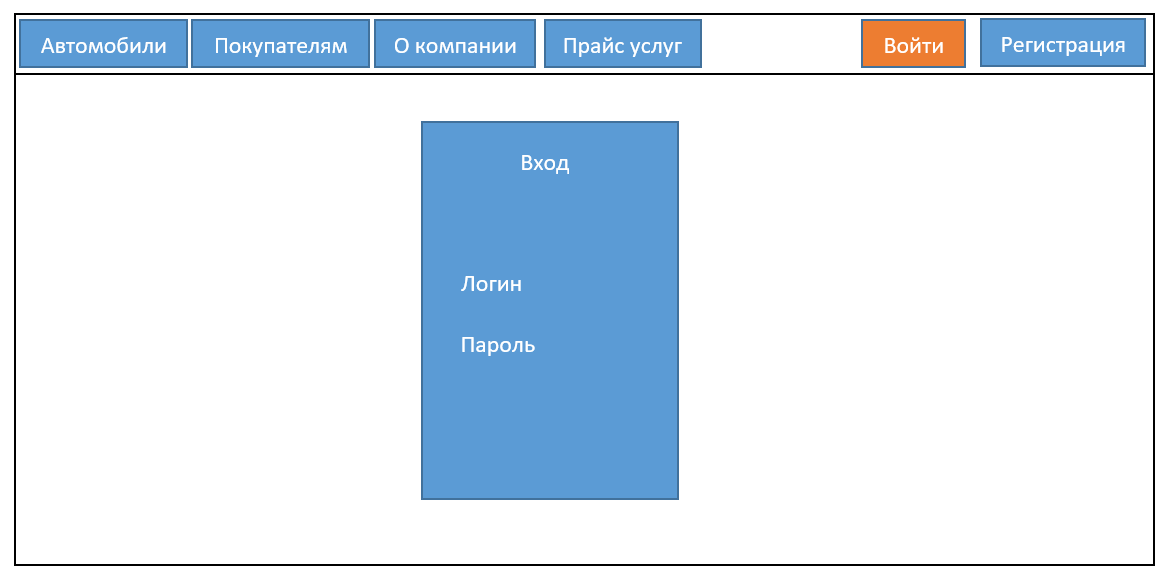
Общая структура страницы сайта представлена на рисунке 2.2. - 2.6.

****

**Рисунок 2.2- Общая структура страницы сайта**

****

**Рисунок 2.3- Общая структура страницы сайта**



**Рисунок 2.4- Общая структура страницы сайта**

Горизонтальное меню состоит из следующих разделов:

- раздел «Автомобили» осуществляет переход на главную страницу;

- раздел «Покупателям» открывает страницу с ссылками на статьи с информацией для покупателей;

- раздел «О компании» осуществляет переход на страницу с информацией о компании;

- раздел «Прайс услуг» осуществляет переход на страницу с информацией о ценах на услуги компании;

- раздел «Войти» осуществляет переход на страницу с панелью для входа пользователя в систему.

- раздел «Регистрация» осуществляет переход на страницу с панелью для регистрации нового пользователя в системе.

Пользователи могут просмотреть карточки авто, характеристики авто, найти авто, оставить заявку на страхование. Наиболее полно отражает представленную информацию диаграмма вариантов использования (приложение Д).

**2.2 Выбор инструментальных средств разработки**

Исходя из технического задания (приложение А), для реализации программного продукта были выбраны следующие средства разработки:

- HTML – язык гипертекстовой разметки документа,

- CSS – каскадные таблицы стилей, используется для форматирования отображаемых элементов на веб-странице,

- «Python»– язык программирования на стороне клиента,

- «Django» – это способ создания интерактивного веб-сайта.

**2.2.1 Язык гипертекстовой разметки HTML**

Для разметки содержимого веб-страниц применяется особый язык — HTML («HyperText Markup Language», язык гипертекстовой разметки). С помощью команд — тегов этого языка создают и абзацы текста, и заголовки, и врезки, и даже таблицы. Язык HTML интерпретируется браузерами и отображается в виде документа в удобной для человека форме.

Для создания HTML документов используют текстовые редакторы (например, «Блокнот»), текстовые процессоры («Word»), редакторы тегов HTML и визуальные HTML-редакторы. Чтобы создать HTML-страницу требуется, придерживаясь определённого стандарта и записать в текстовом файле HTML код и сохранить его на жёстком диске, изменив расширение на «.html» или «.htm».

Любой «HTML»-документ должен содержать следующие теги <html> </html>, <head> </head>, <body> </body>, <title> </title>. Порядок распо-ложения тегов в HTML документе представлен в листинге 2.1.

**Листинг 2.1 – Разметка веб-страницы**

<html>

<head>

<title>Название вашей страницы</title>

</head>

<body>

Тело вашего документа

</body>

</html>

Каждый тег заключен в угловые скобки – <…>. Внутри тега «head» располагается название HTML – документа, помимо этого тега внутри конструкции «head» могут располагаться другие строго определенные элементы HTML (у каждого свое правило ). Текст, заключенный в дескрип-торе «title» будет отображен в заголовке браузера. Элемент «body» (англ. – тело) это основной элемент страницы, дочерний от HTML, это тело страницы. Он содержит в себе большую часть элементов (тегов и контента) веб-страницы.

Несмотря на то, что HTML предоставляет массу возможностей и средств разметки, большинство из них статичны. Различные динамические эффекты и сервисы достигаются в связке HTML с CSS, «JavaScript», PHP, Python и другими инструментами веб-разработки.

**2.2.2 Каскадные таблицы стилей**

Для создания представления веб-страниц предназначена технология каскадных таблиц стилей («Cascading Style Sheets», CSS), или просто таблиц стилей.

Таблица стилей содержит набор правил (стилей), описывающих оформление самой веб-страницы и отдельных ее фрагментов. Эти правила определяют цвет текста и выравнивание абзаца, отступы между графическим изображением и обтекающим его текстом, наличие и параметры рамки у таблицы, цвет фона веб-страницы и многое другое.

Каждый стиль должен быть привязан к соответствующему элементу веб-страницы (или самой веб-странице). После привязки, описываемые выбранным стилем параметры начинают применяться к данному элементу.

Таблица стилей может храниться прямо в HTML-коде веб-страницы или в отдельном файле. Последний подход более соответствует концепции «Web 2.0»; она требует, чтобы содержимое и представление веб-страницы были разделены. Кроме того, отдельные стили можно поместить прямо в тег HTML, создающий элемент веб-страницы; такой подход используется сейчас довольно редко и, в основном, при экспериментах со стилями.

Таблицы стилей пишут на особом языке, который так и называется — «CSS». Стандарт, описывающий первую версию этого языка («CSS 1»), появился еще в 1996 году. В настоящее время широко поддерживается и применяется на практике стандарт «CSS 3». Обычный формат определения стиля CSS представлен в листинге 2.2.

**Листинг 2.2. – Формат определения стиля «CSS»**

<селектор>

{

<атрибут стиля 1>: <значение 1>;

<атрибут стиля 2>: <значение 2>;

. . .

<атрибут стиля n-1>: <значение n-1>;

<атрибут стиля n>: <значение n>

}

Основные правила создания стиля:

- определение стиля включает селектор и список атрибутов стиля с их значениями;

- селектор используется для привязки стиля к элементу веб-страницы, на который он должен распространять свое действие. Фактически селектор однозначно идентифицирует данный стиль;

- за селектором, через пробел, указывают список атрибутов стиля и их значений, заключенный в фигурные скобки;

- атрибут стиля представляет один из параметров элемента веб-страницы: цвет шрифта, выравнивание текста, величину отступа, толщину рамки и др. Значение атрибута стиля указывают после него через символ «:» (двоеточие). В некоторых случаях значение атрибута стиля заключают в кавычки;

- пары <атрибут стиля>:<значение> отделяют друг от друга символом «;» (точка с запятой);

- между последней парой <атрибут стиля>:<значение> и закрывающей фигурной скобкой символ «;» не ставят, иначе некоторые веб-обозреватели могут неправильно обработать определение стиля;

- определения различных стилей разделяют пробелами или переводами строк;

- внутри селекторов и имен стилей не должны присутствовать пробелы и переводы строки. Что касается пробелов и переводов строк, поставленных в других местах определения стиля, то веб-обозреватель их игнорирует. Поэтому мы можем форматировать CSS-код для удобства его чтения, как проделывали это с HTML-кодом.

**2.2.3 Язык программирования «Python»**

«Python» — Язык «Python» универсальный и на его основе можно писать приложения для различных целей. Создать сайт на «Python» вполне возможно, но сайт на питоне гораздо проще создается при использовании Джанго Питон. При помощи дополнительных библиотек можно создавать как приложения, так и веб сайты. К примеру, для создания игр на «Python» понадобиться библиотека «Pygame», а для создания сайтов – «Django».

**2.2.4 Фреймворк «Django»**

Писать сайты на Питон на самом деле можно и без использования сторонних библиотек или же фреймворков. В то же время, написание сайта на основе лишь «Python» – это долгий и сложный процесс. Чтобы облегчить дело были придуман отдельные фреймворки или, другими словами, готовые решения для ускорения процесса построения сайта.

Джанго создан на основе языка Питон. За счёт «Django» вы можете намного проще и быстрее создавать как легкие, так и сложные веб сайты. Преимущество Джанго перед другими фреймворками заключается в огромном наборе готового функционала, за счет которого можно быстро и просто создать систему регистрации на сайте, добавить форум на сайт, реализовать систему поиска по сайту или выполнить какое-то другое действие на сайте.

В Джанго используется схема MVC. Это схема позволяет удобно разбить файлы на одну из трех категорий: HTML шаблоны, файлы модели для работы с базой данных и файлы контроллеры для связи моделей и HTML шаблонов между собой.

**2.3 Алгоритмическое обеспечение проекта**

В ходе проектирования программного продукта исследована предметная область, собрана необходимая информация, определены цели и задачи проектирования веб-сайта. Для его реализации был использован язык веб программирования «Python», фреймворк «Django», «HTML 1.0», CSS.

В программном продукте были разработаны следующие модули: «Оформление меню», «Регистрация в системе».

**2.3.1 Модуль «Оформление меню»**

Данный модуль создает основную навигацию по сайту. В нём представлен такой вид меню, как горизонтальное с выпадающими подменю. С его помощью осуществляется простая и удобная навигация по страницам сайта.

**2.3.2 Модуль «Регистрация в системе»**

Регистрация в системе – это специальный модуль, который имеет поля ввода логина, пароля, подтверждение пароля, электронной почты и кнопку «Зарегистрироваться», служит для регистрации пользователя на сайт. С помощью этого модуля, пользователь может оставить заявку на страхование и просмотреть подробную информацию об автомобиле.

Модуль регистрации в системе реализован на языке программирования «Python» с использованием фреймворка «Django». В нем содержатся следующие поля:

- имя пользователя,

-пароль,

- подтверждение пароля,

- электронная почта.

## **2.4 Определение требований к техническим средствам**

Программное обеспечение клиентской части должно удовлетворять следующим требованиям:

- веб-браузер: «Internet Explorer 7.0», «Opera 9.5», «Chrome 2» и выше,

- включенная поддержка «Flash» и «cookies».

## **2.5 Хранение данных**

**2.5.1 Концептуальное моделирование предметной области**

Хранение информации – одна из важнейших функций сайта. Самым распространенным средством такого хранения являются базы данных. База данных (БД) – это файл специального формата, содержащий информацию, структурированную заданным образом. Для представления динамической информации на сайте необходимо разработать базу данных. База данных создается средствами «Django». База данных состоит из набора таблиц, таких как:

- «Автомобили в наличии» - отображает данные об автомобилях (марка и модель авто, краткое описание, полное описание, дата публикации карточки автомобиля, цена авто, изображения авто, тип кузова, двигатель, тип привода, год выпуска, объем двигателя, тип КПП, пробег) которые находятся в карточке авто;

- «Заявки на страхование» - отображает информацию пользователя, который оставил заявку на страхование.

Приступая к проектированию базы данных, необходимо сначала определить атрибуты (т.е. свойства сущностей), которые на основе анализа существующих между ними связей группируются в отношения, представляющие типы сущностей и связи между ними. Для наглядного представления сущностей и их связей рекомендуется просмотреть приложение Г.

Хорошо организованная структура базы данных позволит создать систему, поставляющую корректную информацию, которая может успешно использоваться для решения различных задач.

Для рассматриваемой задачи выделяют следующие сущности:

- заявки на страхование,

- пользователь / администратор

- автомобили.

Необходимо провести проектирование базы данных, результаты которого можно представить в виде трех таблиц (таблицы 2.1 – 2.3).

**Таблица 2.1 – Сведения к таблице «Автомобили»**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Назначение | Тип поля | Свойства поля |
| 1 | 2 | 3 | 4 |
| Марка и модель авто | Хранение названия марки и модели авто | Короткий текст | Обязательное поле |
| Краткое описание | Хранение электронной почты | Короткий текст | Обязательное поле |
| Полное описание | Хранение описания авто | Длинный текст | Обязательное поле |
| Цена Авто | Хранение цены авто | Короткий текст | Обязательное поле |
| Изображение 1 | Хранение изображения | Файл | Обязательное поле |
| Изображение 2 | Хранение изображения | Файл | Обязательное поле |
| Изображение 3 | Хранение изображения | Файл | Обязательное поле |
| Изображение 4 | Хранение изображения | Файл | Обязательное поле |
| Изображение 5 | Хранение изображения | Файл | Обязательное поле |
| Тип кузова | Хранение типа кузова | Короткий текст | Обязательное поле |
| Двигатель | Хранение типа двигателя | Короткий текст | Обязательное поле |
| Тип привода | Хранение типа привода | Короткий текст | Обязательное поле |
| Год выпуска | Хранение года выпуска | Короткий текст | Обязательное поле |

**Окончание таблицы 2.1**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| КПП | Хранение типа КПП | Короткий текст | Обязательное поле |
| Пробег | Хранение пробега автомобиля в километрах | Короткий текст | Обязательное поле |

Составлено автором

**Таблица 2.2 – Сведения к таблице «Заявки на страхование»**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Назначение | Тип поля | Свойства поля |
| ФИО | Хранение информации о фамилии имени и отчестве пользователя | Короткий текст | Обязательное поле |
| Номер телефона | Хранение информации о контактном телефоне пользователя | Короткий текст | Обязательное поле |
| Комментарий | Хранение комментария пользователя системы к заявке | Короткий текст | Обязательное поле |

Составлено автором

**Таблица 2.3 – Сведения к таблице «Пользователь / Администратор»**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Назначение | Тип поля | Свойства поля |
| 1 | 2 | 3 | 4 |
| Имя пользователя | Хранение логина пользователя | Короткий текст | Обязательное поле |
| Пароль | Хранение пароля пользователя (в зашифрованном виде) | Короткий текст | Обязательное поле |
| Имя | Хранение имени пользователя | Короткий текст | Необязательное поле |
| Фамилия | Хранение фамилии пользователя | Короткий текст | Необязательное поле |

**Окончание таблицы 2.3**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Адрес электронной почты | Хранение адреса электронной почты пользователя | Короткий текст | Обязательное поле |
| Права доступа | Хранение данных о статусе пользователя | Чек-бокс | Обязательное поле |

Составлено автором

Путем устранения дублирования спроектируем реляционную базу данных, в процессе нормализации которой в таблицах отсутствуют транзитивные зависимости и избыточность данных, а также обеспечивается защита целостности данных.