

МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

**ПРОФЕСИОНАЛНА ГИМНАЗИЯ “ГЕН. ВЛАДИМИР ЗАИМОВ” гр. СОПОТ**

4330 гр. Сопот, ул. ”Иван Вазов” №1, тел./факс: /03134/ 83-31, 83-32, e-mail: [pgzaimov@yahoo.com](mailto:pgzaimov@yahoo.com)

**ДИПЛОМЕН**

**ПРОЕКТ**

**Тема: РАЗРАБОТКА НА УЕБ САЙТ ЗА ПРОШУТЕРИЯ "Руски Свят"**

*Ученик: Николай Николаев Лалов*

***Професия:*** *код 481030 „Приложен програмист“*

***Специалност:*** *код 4810301 „Приложно програмиране“*

***Консултант:*** *……………………*

Сопот, 2023 г.

**СЪДЪРЖАНИЕ**

**УВОД**

В тази дипломна работа са описани неща за онлайн пазаруването и необходимостта от уеб сайтове за предлагане на стоки и услуги от производителя към потребителите. Имаме за задача да създадем уеб сайт за прошутерия, чрез който клиентите ще могат да разглеждат и поръчват предлаганите продукти

**Kратко описание на тема и областта на дипломната работа**

Прошутерия „Руски Свят“ се нуждае от уеб сайт, чрез който да увеличи и да популиаризира своята продукция като меса , прошута и сладка. В нашия район, има разнообразие от местни производители на телешко и свинско месо, на които трябва да се направи сайт. Направата на сайта, ще им допринесе много клиенти онлайн, които ще желаят да си закупят месните изделия и сладка.Клиентелата не е нужно да закупи даден продукт,а и да научи за тяхната история на пазара. Тя е с богато минало, което не е за изпускане.Историята накратко е отглеждането на животните им и тяхното гледане, правене и овкусяване на месото.

**Цели и задачи**

Настоящата дипломна работа има за цел да се проектира трислойно MVC WEB приложение, организиращо дейността на прошутерия „Руски свят“, което трябва да поддържа CRUD операции за артикулите, наличните им количества и продажбата им. Чрез създаването на уеб сайт за прошутерията ще разширим дейността ѝ, защото ще дадем възможност на клиентите, пазаруващи в интернет да прегледат и си закупят артикули от прошутерията.

***Задачите, които произтичат от целите са:***

* Разработка на уеб сайт, който трябва да разполага със следните публично достъпни секции „**Начало**“, „**Прошута**“, „**Кошница**“, „**Сладка**“ , „**Контакти**“,“**Обекти**“ , „**Услуги**“.
* Осигуряване на възможност за разглеждане на меса, асортименти, заявка за дегустация и да се поддържа информация за наличните количества от всеки артикул.
* Трябва да се осигури наличието на пазарска кошница, в която регистриран клиент да добавя избрани от него артикули в исканото количество, като по този начин формира поръчка. Пазарската кошница трябва да се визуализира на отделен екран. Добавените артикули от потребителя да са в списък с единична цена и количество. Поръчката има общата стойност и да може да се финализира с бутон „Поръчай“.
* В сайта да има три вида потребители в ролите: нерегистриран потребител, администратор и клиент.
* Сайтът трябва да може да се администрира с помощта на администраторско меню, което да позволи добавянето на нови артикули, промяна и изтриване на съществуващи такива и промяна на някоя от другите страници.
* Да се осигури възможност за това само регистрирани клиенти да могат да

поръчват стоки и/или заявяват услуга, като добавят избрани артикули в исканото количество, като по този начин формира поръчка.

* Дизайнът на сайта трябва да е адаптивен и да изглежда еднакво добре на различни устройства.

**ГЛАВА 1.**

**ПРОУЧВАНЕ ПО TEMATA**

* **Предпоставка за създаване на продукта**

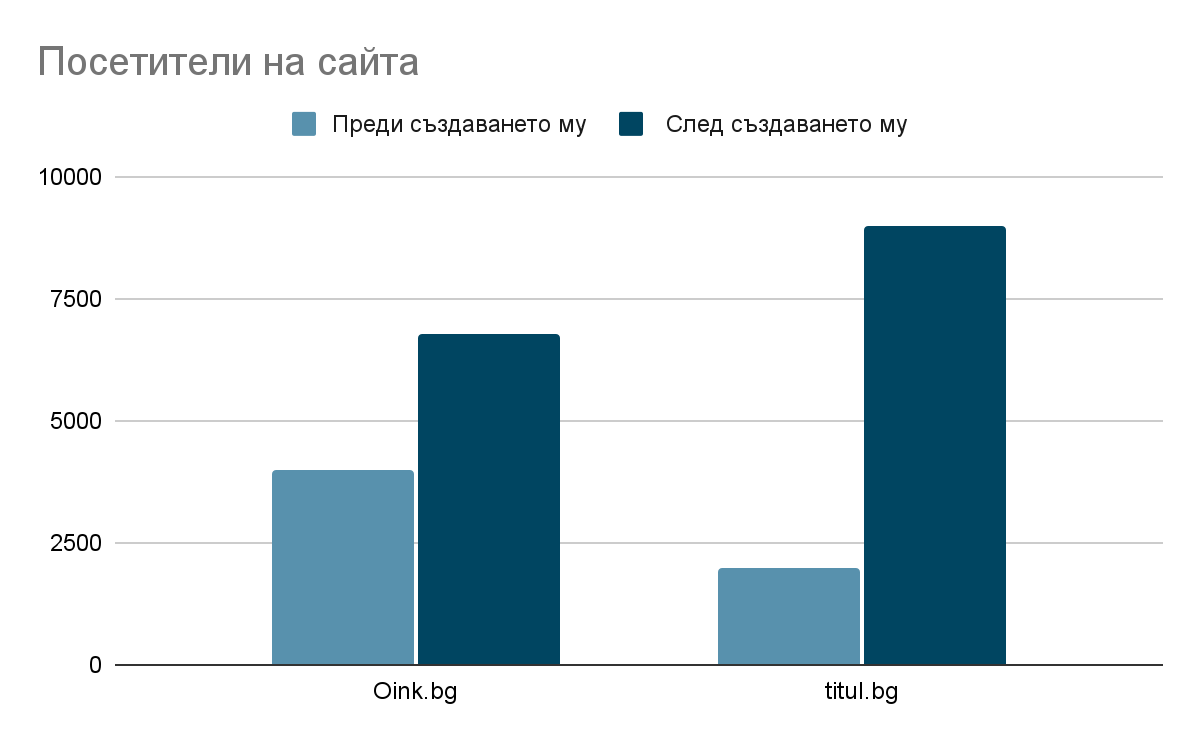
За популиаризирането на нашия сайт за прошутерията „Руски свят“ ,ще трябва детайлно и внимателно да се изготви сайт ,който ще е достъпен за всички клиенти .

Чрез сайта клиентите на прошутерията ще могат лесно да поръчват избрани от тях прошута и сладка в нужното количество с техните актуални цени, ще могат да разгледат и резервират услуги в прошутерията и също ще могат да се запознаят с нашата дейност.

Изработката на уеб сайт ще увеличи оборота на гореспоменатата прошутерия и интереса към закупуване на продукти като прошута или сладка онлайн.

По данни на българските медии фирмите ,които имат възможност да използват интернет са 98%, но само 32% от тях си имат така наречения сайт , който развива тяхната продаваемост и развитие, защото по този начин се развива от клиент на клиент и се получава така да има повече посетители на сайта и повече продажби.

* **Съществуващи решения и реализации**
* На диаграма едно са показани броя на посетителите преди и след създаването на сайтовете за изброените прошутерии.На диаграмата ясно можем да видим ,че след създаването на уеб сайт ,броят на посетителите значително нараства.Това доказва значителноста на сайтовете при бизнесите.Двете фирми Oink.bg и Titul.bg

****

**ГЛАВА 2.**

**ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА WEB БАЗИРАН….**

**2.1. Функционални изисквания към WEB базиран......... /**условието назадачата**/**

Да се реализира уеб приложение за онлайн пазаруване на стоки от прошутерия „Руски свят“.

Приложението трябва да поддържа следните функционалности:

o Добавяне на нов артикул: /с Администраторски функции/:

Каталожен номер

Име на артикул

Категория

Описание – кратък текст – описание на качеството, вкус, съставки

Снимка

Цена на артикул

Дата на вписване на артикула в системата

o Извличане на информация за един избран артикул

o Изтриване на артикул /с Администраторски функции/

o Промяна на спецификациите на даден артикул /с Администраторски функции/

o Извличане на всички артикули по избрана категория

o Да се направят роли за работа с приложението:

АДМИНИСТРАТОР

Клиент

o Регистрация на клиент:

Име

Фамилия

Електронна поща

Телефон за контакт

o Пазаруване /записване на артикули в пазарска кошница/:

Номер на клиент

Номер на артикул;

Количество стока /брой артикули/

Единична цена на артикул

Дата / час на направения избор;

Пазарската кошница трябва да може да се визуализира на отделен екран, като избраните от

клиента стоки са подредени в добре форматиран списъчен вид заедно с техните количества.

Да се пресмята и общата стойност на поръчката, която може да се финализира с бутон

„Финализирай поръчка“. Дизайнът е напълно свободен и авторски.

След финализиране на поръчка:

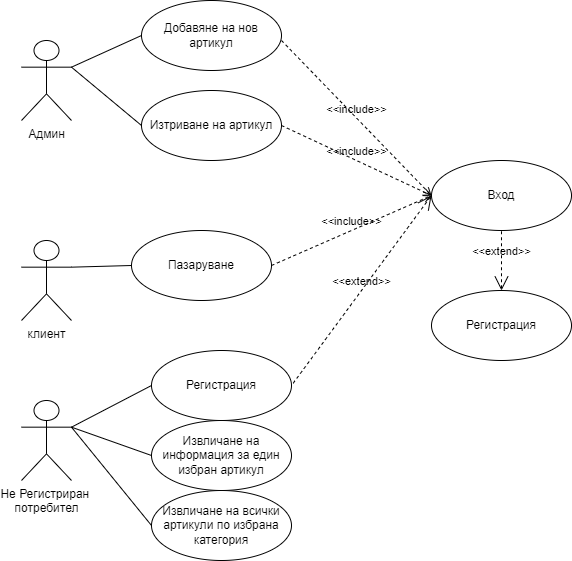
- Пазарската кошница се изпразва;

- Поръчката се записва в отделна таблица в БД на приложението /№ поръчка, списък

артикули с техните количества и цени, крайна сума за плащане, номер на клиента,

който е поръчал.

**2.2. Изготвяне на Use Cases Diagrams – описание на отделните на случаи на употреба**

****

**Фифура 1**

**Случай на употреба: Добавяне на нов артикул**

**Цел:Задаейства се, когато администратор иска да добави нов артикул**

**Категория:Основна**

**Успешни постусловия:Успешно добавяне на артикул**

**Неуспешни постусловия:Грешка при вход в системата**

**Актьори: Админ**

**Случай на употреба: Изтриване на нов артикул**

**Цел:Задаейства се , когато администратор иска да изтрива артикул**

**Категория:Основна**

**Успешни постусловия:Успешно изтриване на артикул**

**Неуспешни постусловия:Грешка при запис на промените**

**Актьори: Админ**

**Случай на употреба: Пазаруване**

**Цел:Задаейства се , когато клиента иска да добави нов артикул към кошницата**

**Категория:Основна**

**Успешни постусловия:Успешно добавяне на артикул в кошницата**

**Неуспешни постусловия:Грешка при добавяне на артикул в кошницата**

**Актьори: Клиент**

**Случай на употреба: Регистрация**

**Цел:Задаейства се , когато не регестрираниея потребиел иска да се регистрира**

**Категория:Основна**

**Успешни постусловия:Успешно регистриране**

**Неуспешни постусловия:Грешка при регистрацията**

**Актьори: Не регистриран потребител**

**Случай на употреба: Извличане на информация на артукул**

**Цел:Задаейства се , когато не регестрираниея потребиел иска да види**

**информацията**

**Категория:Основна**

**Успешни постусловия:Успешно извличане на информация**

**Неуспешни постусловия:Грешка при извличането**

**Актьори: Не регистриран потребител**

**Случай на употреба: Извличане на всички артикули**

**Цел:Задаейства се , когато не регестрираниея потребиел иска да извече всички**

**Категория:Основна**

**Успешни постусловия:Успешно извичане на всички артикули**

**Неуспешни постусловия:Грешка при извличането**

**Актьори: Не регистриран потребител**

**2.2.2. Случаи на употреба**

**Администратор:** Администраторът ще бъде един потребител, назначен от прошутерията, който ще може да:

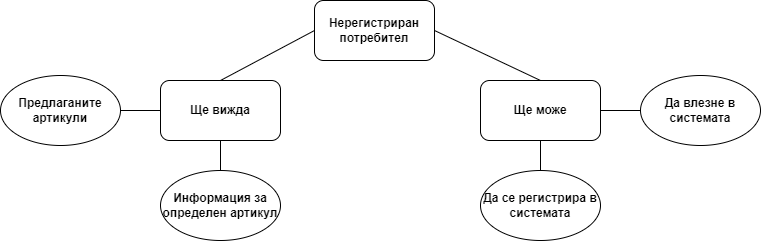
* Добавя нов артикул за продажба
* Изтрива вече добавен артикул за продажба
* Добавя промоции за артикулите във прошутерията

Картина, която съдържа текст, екранна снимка, Шрифт, дизайн

Описанието е генерирано автоматично

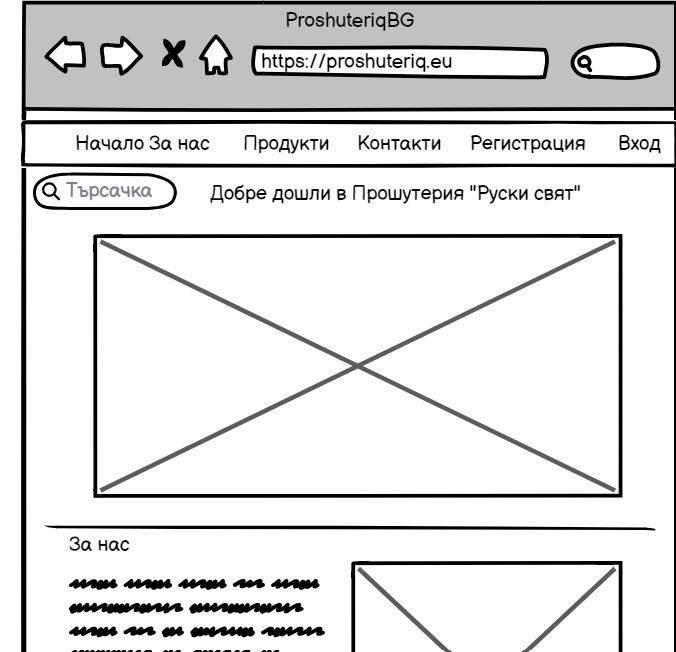
**Фигура 2**

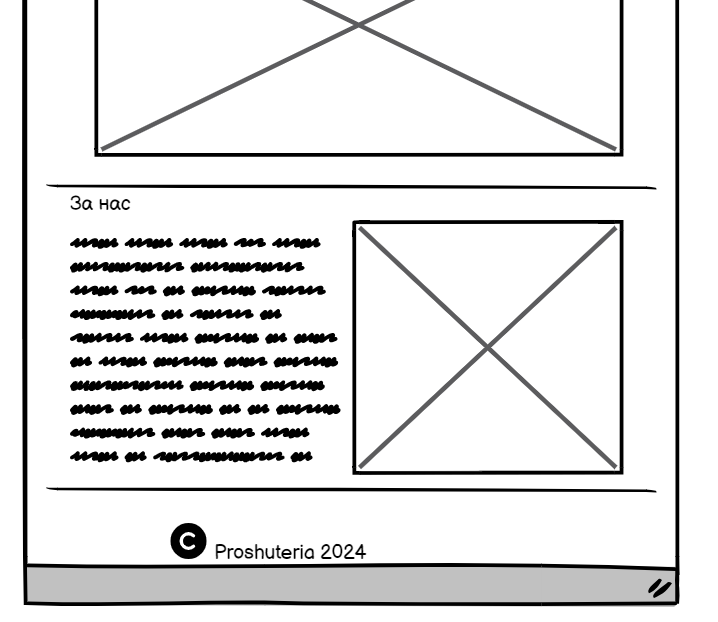
Не регистриран потребител: Това значи ,когато някой не се е регистирал и не може да се отчете като потребител и само може да разглежда без да поръчва месата, прошутата и сладката.

**** **Фигура 3**

**2.3. Прототип на потребителският интерфейс**

**2.3.1.НачаленЕкран**

****

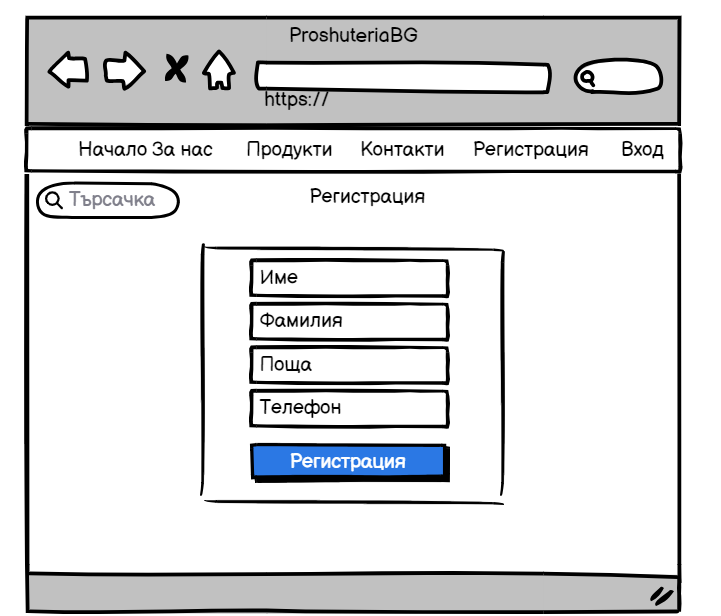
****

**Фигура 4**

**На фигура 4 е показано Home страницата на Сайта .**

Самата страница има меню, навигация, търсачка, текстово поле за информация ,снимка и други. Самото меню разполага със следните секции като: Начало, За нас, Продукти, контакти ,регистрация и вход.

**2.3.2.Регистрация**

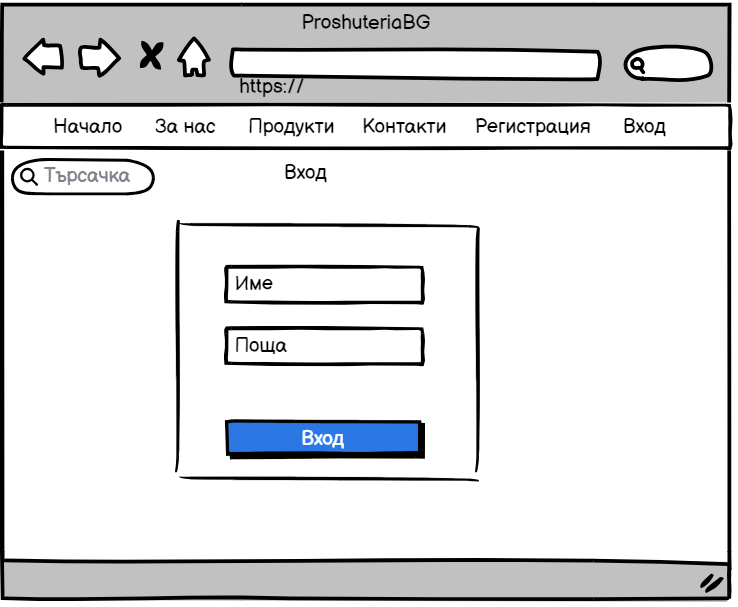
****

**Фифура 5**

На **фигура** 5 е показана страницата за регистрация на потребителя в системата .

Тук той въвежда своите данни за регистрация.Той трябва да си въведе своето име,фамилия, поща и телефон.

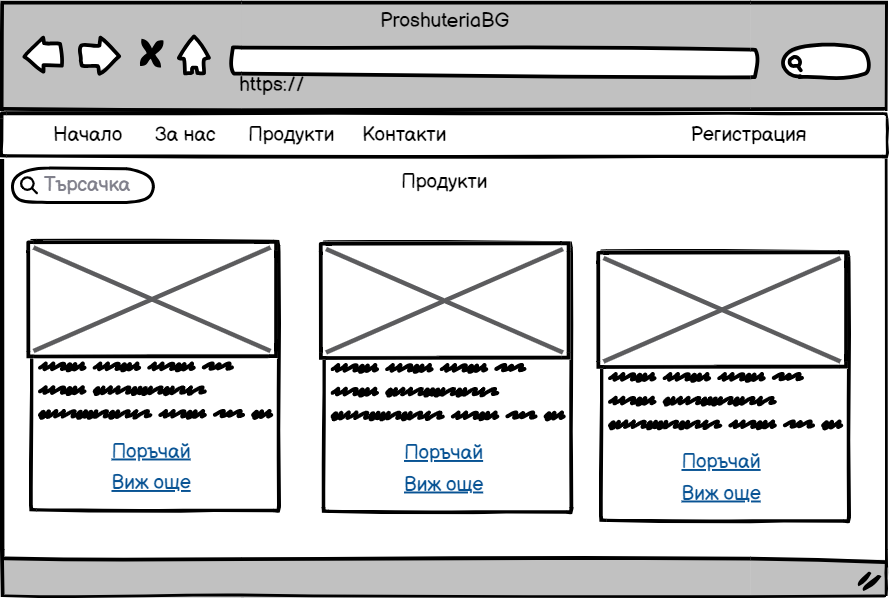
**2.3.3.Вход**

****

**Фигура 6**

На **фигура 6** е показана страницата за вход в системата . Тук потребителя ,ако вече е създал своя акаунт в сайта може да си влезе без проблем,ако няма ще му изисква регистрация и ще го върне на страницата за регистрация**.**

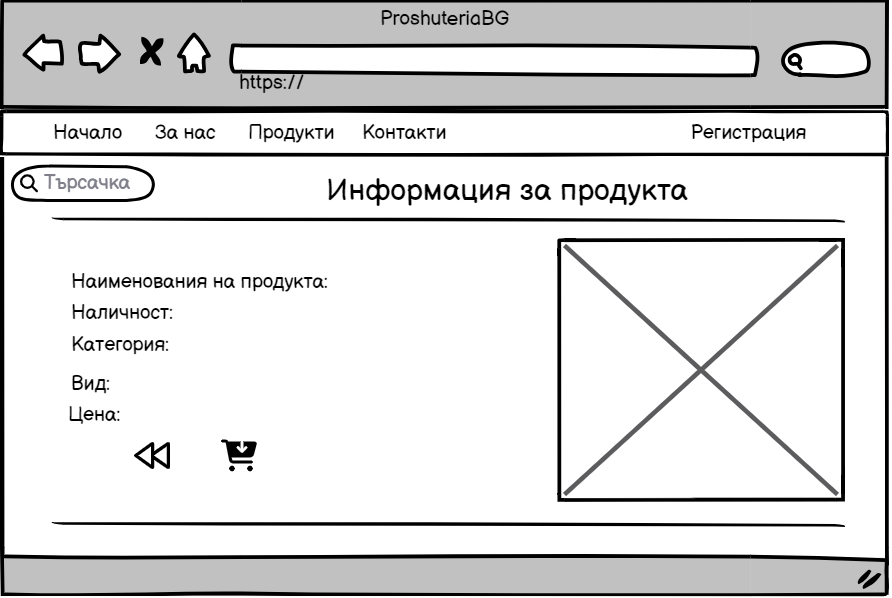
**2.3.4.Продукти**

****

**Фифура 7**

На **фигура 7** ще видите отделните продукти на прошутерия “Руски свят”.Тук потребителя може да поръча или да види подробна информация за продукта.

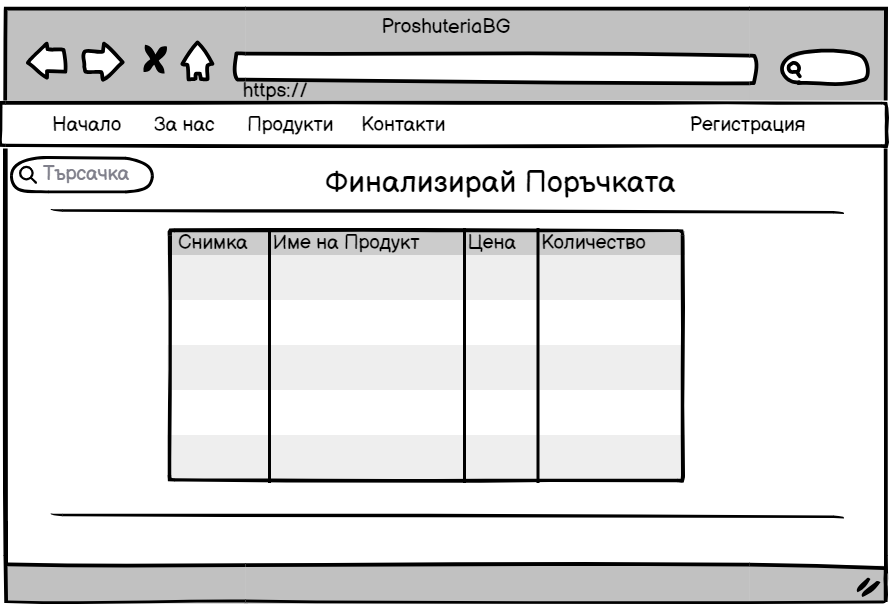
**2.3.5.Информация**

**Фигура 8**

Във **фигура 8** потребителя може да види подробна информация за даден продукт.Тук той също може да го поръча или да се върне назад към всички продукти.

**2.3.6.Финализация на поръчката (Кошница)**

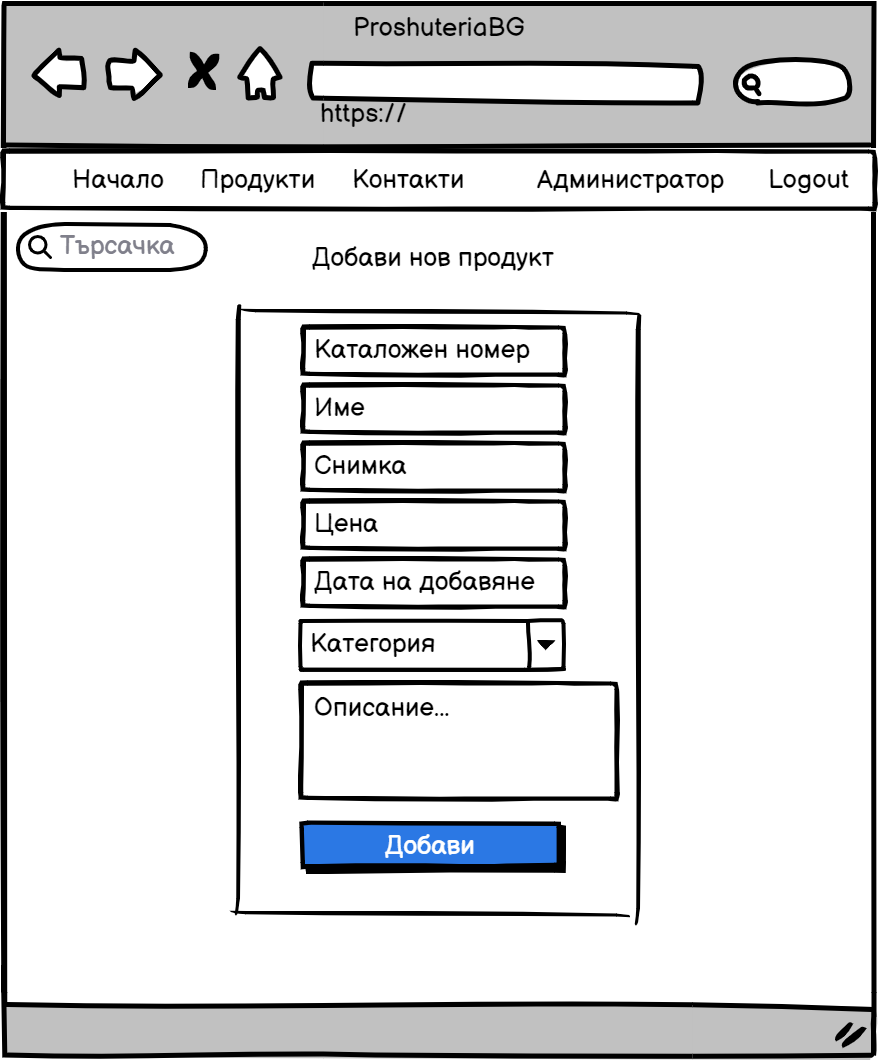
**Фигура 9**

****

На **фигура 9** ще може потребителя да финализира неговата поръчка ,като може да види нейната стойност , количество ,име на продукта и снимка .

**2.3.7 Добавяне на продукт само от администратора .**

**Фигура 10**

****

В **тази страница 10** администраторите на сайта могат да си добавят продукти като има категория ,дата,описание,име на продукта ,снимка и каталожния номер.

**2.4. Описание на избраната технология и софтуерните средства за разработка на приложението и основните алгоритми/**технологии: *ASP.Net, ASP.Net Core, SQL Server, средства: C#, HTML, CSS, JavaScript*, *Bootstrap***/**

**2.4.1. Изграждане на уеб сайтове с ASP.NET**

**Какво е ASP.NET?**

**ASP.NET представлява технология за разработка на мощни web приложения**

**базирани на .NET платформата на Майкрософт, предлагайки някои съществени**

**предимства в сравнение с предишните модели за разработка на web приложения.**

**Създаваните с ASP.NET приложения представляват код, който се извършва на един**

**уеб сървър. За разлика от така познатите ни съвременни, по-интелигентни**

**технологии, ASP.NET е в състояние да се възползва от предимствата на ранното**

**свързване, така нареченото just-in-time компилиране, оптимизация на кеширане (на**

**различни нива) и др. По този начин получавате висока производителност преди да**

**сте написали и ред програмен код. ASP.NET приложения могат да се разработват**

**дори на Notepad, но истинско повишаване на производителността се наблюдава при**

**използването на мощния инструментариум и средства за дизайн на Visual Studio.**

**Потребителския интерфейс се осъществява в WYSIWYG (What You See Is What You**

**Get) среда, като добавянето на контроли става посредством drag-and-drop. Налице е**

**опция за автоматично разгръщане (deployment) и още много интересни**

**характеристики.ASP.NET приложението има много характеристики които му**

**помагат да е с всяка изминала години стабилна платформа. Нека започнем с**

**мощността и гъвкавостта. Тази функция позволява на разработчеците да изберът и**

**да работят с един език за програмиране по техен избор, който трябва да бъде**

**подходящ за да може да се разработват проекти или приложения които са на .NET.**

**Следващата функционалност на ASP .NET приложението е лесното конфигуриране**

**на данни, тази функция облекчава изпълнението на всевъзможни задачи от прости**

**форми до пълната конфигурация на самият сайт. Благдарение на което ASP.NET**

**приложението позволява създаването на едни приложения, при които ясно може да**

**се види разграничаването на приложенчестата логика от презентационният код, като**

**обработката на събитията и създаването на един потребителси интерфейс. Нека да**

**продължим със следващата функция на ASP.NET приложението, която е много по**

**специална от другите изброени. Дадената архитектура на приложението позволява**

**на разработчеците да разширяват възможностите, които се предоставени от средата.**

**Девелапърът даже може дори да замени някои от субкомпонентнитте на даденото**

**приложение със разработените вече от него модули. ASP.NET е проектирано да бъде**

**мащабируемо, тоест с по-голям обем, като предоставя възможностите за**

**подобряването и обновяването на произжодителността на клъстерната или**

**многопроцесорната система. Тези процеси с внимателно следени и управлявани по**

**време на тяхното изпълнение, което позволява да бъде създаден нов процес ако**

**даденият се държи неадекватно, което води до положителната позиция към това**

**вашето приложение винаги да бъде на ваше разположение.**

**Предимства на ASP.NET:**

**-ASP.NET кодът е лесен за преглед и проверка. То позволява не само да се пише**

**много по-лесно, но също така улеснява девелапърите в процес на качествен**

**контрол.**

**-Подобрява мащаба, сигурността и надежността на приложенията, също така и**

**предлага по-качествената поддръжка на уеб браузърите**

**2.4.2 ASP.NET Core MVC**

**ASP.NET Core MVC предоставя следните функции за изграждане на уеб API и уеб**

**приложения:**

**- Използва модела на дизайна на View-Controller (MVC)**

**- Лек, с отворен код, тестируем, добър инструментариум**

**- RESTful услуги с ASP.NET Core Web API**

**- Вградена поддръжка за множество формати на данни**

**- Постигнат висококачествен архитектурен дизайн, оптимизирайки работата на**

**девелапърите**

**- Конвенция за конфигуриране**

**- Обвързването на модела автоматично картографира данни от HTTP заявки**

**- Проверка на модела с валидиране от страна на клиента и от страна на сървъра**

**заедно с тези ASP.NET Core MVC предоставя функции като:**

**-Routing**

**-Инжектиране на зависимостта – DI**

**-Силно типизирани изгледи с двигателя Razor**

**-Помощниците на маркери активират код от страна на сървъра в HTML**

**елементи**

**-Частични изгледи и преглед на компоненти**

**-Филтри, области, Middlewares**

**-Вградени функции за защита**

**-Идентичност с потребители, роли и външни доставчици**

**2.4.3 SQL Server**

**Microsoft SQL Server е система за управление на релационни бази данни,**

**разработена от Microsoft. В тази си роля (на система за управление) SQL Server има**

**основната функция да съхранява и извлича данни, по заявки на други софтуерни**

**приложения, които могат да вървят на същия или друг компютър в дадена мрежа (в**

**т.ч. и Интернет). MS SQL Server има над дузина различни издания, които са**

**подходящи за различни аудитории и различни натоварвания: от приложения за**

**отделни персонални компютри до масивни приложения работещи постояно в онлайн**

**режим и обхващащи огромно количество устройства. MS SQL Server се заражда**

**преди малко повече от 30 години, на 12-ти юни 1988 г. Microsoft сливат Ashton-Tate**

**и Sybase, за да създадат вариант на Sybase SQL Server за операционната система**

**OS/2, която по това време се разработва съвместно от IBM и Microsoft. Първото**

**издание на MS SQL Server излиза на следващата година и се превръща в първия**

**продукт на Microsoft, който е предназначен за бизнес пазарът на бази данни,**

**съревновавайки се с конкурентни продукти на компании като Oracle, IBM и по-късно**

**Sybase. В следващите няколко версии програмата върви в комплект с операционната**

**система OS/2, като едва версия SQL Server 6.0 е първата, която е изцяло интегрирана**

**за работа единствено с Windows NT, както и че системата се дистанцира по-осезаемо**

**от единия от първоизточниците си: Sybase. До 1994 г. MS SQL Server съдържа в**

**лицензионното споразумение три препратки за авторски права към Sybase, които**

**показват първоначалния произход на софтуера.Дистанцирането се дължи на**

**различия във вижданията на Sybase и Microsoft по отношение на качествата на**

**крайния продукт – различни визии, цялостен дизайн и пазарни схеми. Microsoft**

**успяват да спечелят пълни и изключителни права за всички версии на SQL Server,**

**които са написани за операционната система Windows. Това принуждава през 1996 г.**

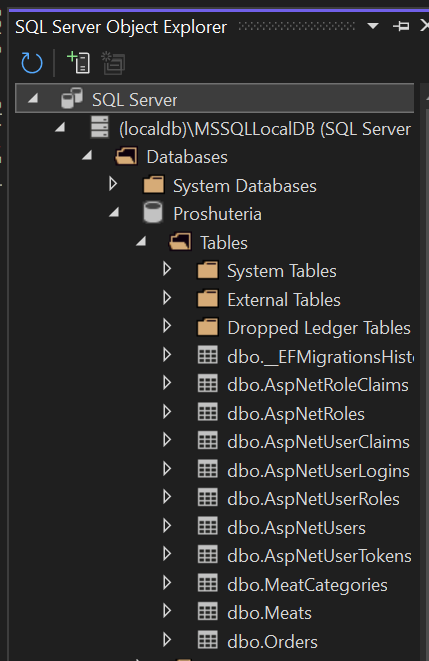
**Sybase да сменят името на своя продукт на Adaptive Server Enterprise, за да избегнат**

**объркване с продукта на Microsoft – MS SQL Server. Базите данни са стандартният**

**начин за използване на динамична информация в съвременните приложения,**

**правейки ги неизменна част от почти всяко едно приложение. Тук можем да видим**

**една SQL Server База данни:**

****

**Фигура 11 база данни**

**2.4.4 C#**

**C# се произнася като „Си Шарп“. Той е създаден от Microsoft през 2001 г., а**

**последното обновление за него излезе в края на 2021, т.е. това е един актуален език.**

**Може да се каже, че C# e Microsoft-ският отговор на програмния език Java, продукт**

**на компанията Sun Microsystems (понастоящем собственост на Oracle Corporation).C#**

**е обектно-ориентиран език, т.е. той изгражда системи посредством обекти, които**

**взаимодействат помежду си. Обектно-ориентираното програмиране се отличава от**

**традиционното програмиране, при което програмата е просто списък от инструкции,**

**които компютърът изпълнява.C# е програмен език от високо ниво. Това го прави**

**изключително достъпен за начинаещи програмисти. Като език от високо ниво, той е**

**по-лесно четим и разбираем, тъй като е по-близо до хората, отколкото до машините**

**(които, както вероятно знаеш, говорят само с нули и единици). Това дава**

**възможност на програмиста да се фокусира върху разработката, правейки процеса**

**по-лесен, бърз и разбираем.Освен това, C# разчита и на т.нар. garbage collector, който**

**се грижи автоматично за правилното управление на паметта като ресурс, който**

**използваш по време на разработване. Това дава изключително много свобода на**

**програмистите, освобождава ти време и премахва излишни затруднения в процеса на**

**работа.**

**Днес С# е един от най-използваните езици за програмиране. Той е език**

**за back-end разработка, т.е. се грижи за сървърната логика, a през**

**технологията Blazer вече опитва да се конкурира с JavaScript за целите на front-**

**end разработката. Ето какво ще можеш да създаваш, ако овладееш C#:**

**1. Десктоп приложения- От Microsoft използват C# във всички свои**

**продукти в това число всичките си десктоп приложения за Windows 8 и 10, т.е.**

**езикът става все по-актуален и широкоизползван, което дава повече шансове за**

**реализация.**

**2. Уеб приложения- ASP.NET е технологична рамка с отворен код, която**

**също е част от .NET. Бидейки част от едно и също семейство с програмния език C#,**

**това улеснява тяхното взаимодействие и дори го предполага. Това означава, че от**

**Microsoft се стремят да предразположат разработчиците да използват колкото се**

**може повече MS продукти в разработването на своите приложения и платформи.**

**Доброто взаимодействие между C# и уеб фреймуърка ASP.NET правят програмния**

**език подходящ и за уеб сайтове, платформи и уеб приложения.**

**3. Мобилни и cross-platform мобилни разработки**

**C# е подходящ също и за създаване на мобилни и междуплатформени**

**приложения. Visual Studio – IDE-то на Microsoft e съвместимо с Xamarin –**

**платформа за разработка на native Android, iOS и Windows приложения.**

**4. Игри**

**Разработката на игри със C# също е една от възможностите за реализация**

**с езика. Един от най-популярните гейм енджини, подходящ както за PC, така и за**

**конзолни и мобилни приложения, е Unity, който е микс от няколко езика от**

**семейството на „С“ – С, С++ и разбира се С#. С това далеч не се изчерпват игровите**

**енджини, писани частично или изцяло на C# и това също е една добра насока за**

**развитие.**

**2.4.5 HTML**

**HTML (съкращение от термина на английски: Hyper Text Markup Language,**

**произнасяно най-често като „ейч-ти-ем-ел“, в превод „език за маркиране на**

**хипертекст“) е основният маркиращ език за описание и дизайн на уеб страници.**

**HTML кодът казва на браузърите как да показват различни елементи на уеб**

**страница, като изображения, текстови полета, таблици и др. Има много тагове,**

**използвани за създаване на HTML страници; обаче има някои често срещани, които**

**ще виждате често. Най-основният етикет е . Този маркер съдържа мета-информация**

**за вашия уебсайт, включително неговото заглавие и друга важна информация, като**

**типа съдържание, което използва (текст/HTML). HTML е една от основните**

**технологии на подхода за създаване на съдържание в World Wide Web. Тя ви**

**позволява да проектирате и публикувате свои собствени уеб страници, за да ги**

**виждат другите, да ги разпространявате по интернет и да имат достъп до тях от**

**всеки уеб браузър. HTML документите са написани на специфичен HTML език,**

**който включва HTML елементи като параграфи, заглавия, списъци, таблици и т.н.**

**Използването на този език за маркиране става много лесно да създавате спретнати**

**изглеждащи документи с добавена семантична стойност към тях. По долу може**

**също така да прочетете малко по подробно неща свързани с този описателен език:**

**-HTML се използва за описване на структурата на даден документ използвайки**

**маркери – така наречените тагове**

**-HTML документите са съвкупност от маркиращи тагове**

**-Тези тагове описват съдържанието на документа и са изграждащите го**

**блокове**

**-Тези блокове се наричат HTML елементи**

**-HTML таговете се явяват един вид етикет на типа съдържание –**

**напр. heading, paragraph, table и т.н**

**-HTML документите съдържат само HTML тагове и чист текст**

**-HTML документите, също се наричат и уеб страници**

**-Браузърите не визуализират HTML таговете, а ги използват за показване на**

**съдържанието**

**HTML работи по структура, която ни помага да видим резултатите в една HTTP**

**страничка които сме написали като код на този описателен език. Също така той е**

**полезен за редактация на заглавия, параграфи, секции, таблици и много други**

**основни елементи на една уебаджийска страница. Когато някой кликне с курсора на**

**мишката на адресното поле и въведе един обикновен URL адрес, браузърът изпраща**

**GET заявка с която въведения адрес се взима и се изпраща на сървъра където е**

**качена страницата посредством HTTP(Hypertext Transport Protocol). Следователно**

**след това сървърът прочита заявката, която е испратена и изпраща POST заявка с**

**която се испраща и визуализира отговорът на сървърът към потребителя.**

**Тагове в HTML**

**Маркиращите HTML тагове, биват наричани по един прост начин, просто тагове.**

**Общо прието е да се наричат по този прост начин, поради простата причина, че в**

**тази сфера това са единствените тагове, които се изплозвта за маркиране.**

**HTML таговете са ключови думи (имена) оградени с ъглови скоби – &lt;html&gt;**

**HTML таговете повечето пъти се пишат по двойки**

**Във всяка двойка, първия таг е отварящия, а вторият затварящ. Втория таг се пише**

**точно като първият с разликата, че се добавя наклонена черта преди името му.**

**2.4.6 CSS – Cascading Style Sheet**

**CSS (Cascading Style Sheets) е език за описание на стилове (език за стилови**

**файлове, style sheet language) – използва се основно за описание на онлайн**

**представянето на уеббазиран документ, който написан на език за маркиране. Най-**

**често се използва допълнително към чистия HTML, но се прилага и върху XML**

**уебстраници и документи. Спецификацията на CSS официално се поддържа от**

**W3C.CSS още в началото на развитието на www започва да се добавя към**

**стандартния HTML с цел да бъдат разделени съдържанието и структурата на уеб**

**страниците отделно от тяхното визуално представяне. Преди стандартите за CSS,**

**установени от W3C през 1995 г., съдържанието на сайтовете и стила на техния**

**дизайн са писани в една и съща HTML страницата. В резултат на това HTML кодът**

**се превръща в сложен и нечетлив, а всяка промяна в проекта на даден сайт изисквала**

**корекцията да бъде нанасяна в целия сайт страница по страница. Използвайки CSS,**

**настройките за форматиране могат да бъдат поставени в един-единствен файл и**

**тогава промяната ще бъде отразена едновременно на всички страници, които**

**използват този CSS файл. CSS позволява да се определя как да изглеждат елементите**

**на една HTML страница – шрифтове, размери, цветове, фонове, и др. CSS кодът се**

**състои от последователност от стилови правила, всяко от които представлява**

**селектор, последван от свойства и стойности.**

**Съществуват няколко начина за прилагане на CSS стила:**

**1.Външен стил - Записът, който направихме в горния пример, се нарича външен**

**стил (External Style Sheet) и се използва когато трябва да се контролират**

**множество HTML документи, като нужните параметри се задават във външен**

**файл (file.css).**

**2.Вътрешен стил - Съществува и вътрешен за HTML документа стил (Internal**

**Style Sheet), който се използва за да се зададе вида на един отделен HTML**

**документ, като нужните свойства се задават със специалния таг &lt;style&gt; в секцията**

**на HTML страницата.**

**3.Вграден стил - Другият начин за налагане на стил е Inline Styles – вътрешни за**

**HTML таговете стилове. CSS стилът се разполага като атрибут директно в HTML**

**тага например .**

**Когато в един документ се използват и трите начина за налагане на CSS стил – с**

**най-висок приоритет е стилът на Inline Style (вътрешните за HTML таговете**

**стилове), след тях на по приоритет са Internal Style Sheet (стиловете от секцията**

**head на HTML документа) и последни по приоритет са External Style Sheet, т.е.**

**стиловете, декларирани във външен CSS файл.**

**2.4.7 JS(Java Script)**

**JavaScript е популярен език за програмиране, който се използва за създаване**

**на интерактивни и динамични уеб страници. Той е един от основните езици за**

**програмиране на уеб, заедно с HTML и CSS. JavaScript е изпълнен от страна на**

**браузъра на потребителя и се използва за създаване на ефектни елементи като**

**анимации, динамични форми, динамично генерирани списъци, изображения и много**

**други. Той също така се използва за извличане на данни от уеб страници и**

**комуникация с уеб сървъри. JavaScript е мултиплатформен език за програмиране и**

**може да се използва както за уеб, така и за мобилни приложения. Той има широк**

**кръг от библиотеки и фреймуърки, които помагат за ускоряване на процеса на**

**разработка на уеб приложения. JavaScript е един от най-широко използваните езици**

**за програмиране в света и се използва не само за уеб, но и за различни други**

**приложения като игри, мобилни приложения, десктоп приложения и други.**

**JavaScript е език с динамичен тип данни, който означава, че променливите могат да**

**съдържат различни типове данни, като числа, символни низове, булеви стойности,**

**обекти и други.**

**Един от най-големите предимства на JavaScript е, че може да се използва за**

**манипулиране на HTML документи и стилове, което позволява на програмистите да**

**създават динамични и интерактивни уеб страници, които да реагират на действията**

**на потребителя. JavaScript има много библиотеки и фреймуърки, като React, Angular**

**и Vue, които помагат на програмистите да създават по-бързо и ефективно уеб**

**приложения. Освен това, той се поддържа от всички основни браузъри като Chrome,**

**Firefox, Safari и Edge, което го прави много удобен за уеб разработка. В заключение,**

**JavaScript е мощен език за програмиране, който е важен инструмент за създаване на**

**уеб приложения и интерактивни уеб страници. JavaScript предоставя много**

**възможности за създаване на динамично генерирани уеб страници и уеб**

**приложения. Той се използва за създаване на ефектни елементи като слайдшоутове,**

**анимации и различни видове визуални ефекти. JavaScript също така има много**

**различни библиотеки и фреймуърки, като jQuery, Bootstrap, React, Angular и други,**

**които помагат на програмистите да разработват уеб приложения по-лесно и бързо.**

**JavaScript може да се използва и за обработка на данни и комуникация с уеб сървъри**

**чрез AJAX заявки. Това позволява на уеб приложенията да взаимодействат с бази**

**данни и други услуги на уеб сървъри, като например извличане на информация за**

**потребителите и изпращане на формуляри.**

**JavaScript е език, който се променя бързо и постоянно се актуализира с нови**

**функционалности и подобрения. Това означава, че програмистите трябва да следят**

**развитието на езика и да бъдат в крак с последните тенденции, за да могат да**

**създават най-ефективните и съвременни уеб приложения.**

**В заключение, JavaScript е един от най-широко използваните езици за**

**програмиране в света и предоставя много възможности за създаване на уеб**

**приложения и интерактивни уеб страници. Той е мощен език, който продължава да**

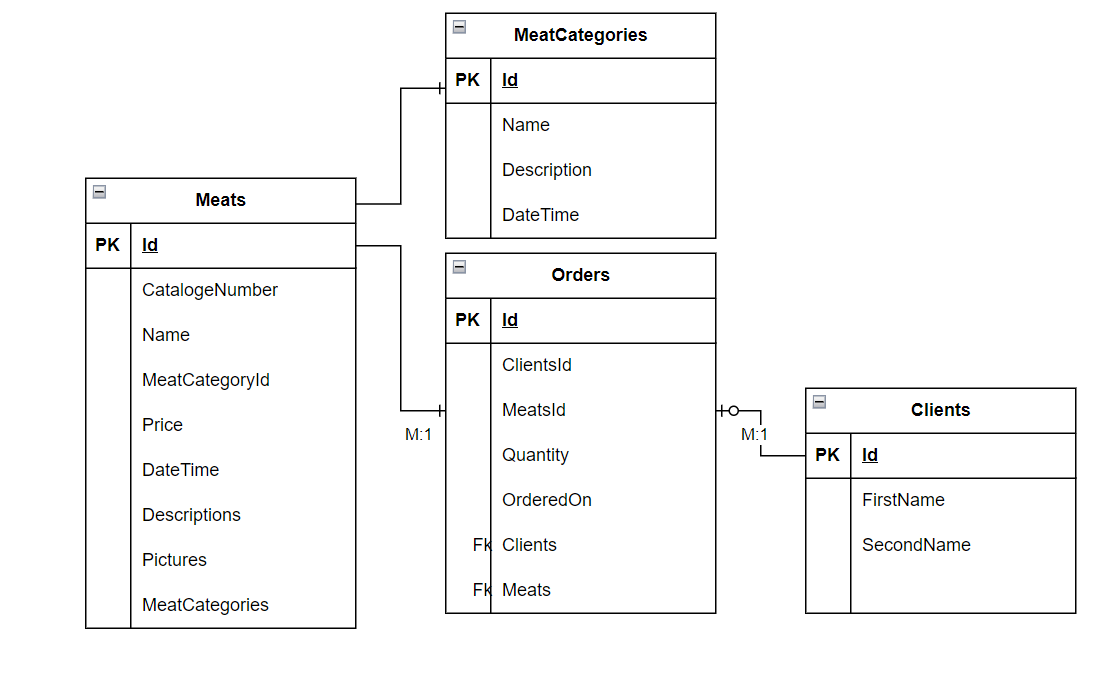
**се развива и да се подобрява с нови функционалности и инструменти.**

**ГЛАВА 3.**

**ПРОГРАМНА РЕАЛИЗАЦИЯ НА WEB БАЗИРАН ……………….**

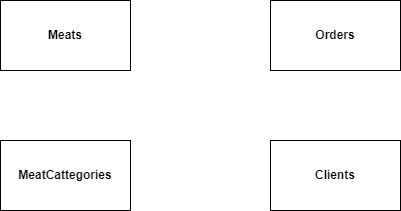
* **Структурата на базата от данни – Data Models**

*Освен E/R диаграмата, тук се описват: данните в таблицата, ключовите полета и релациите /по кои полета и за какво се прави дадената релация/*

***Фигура 12***

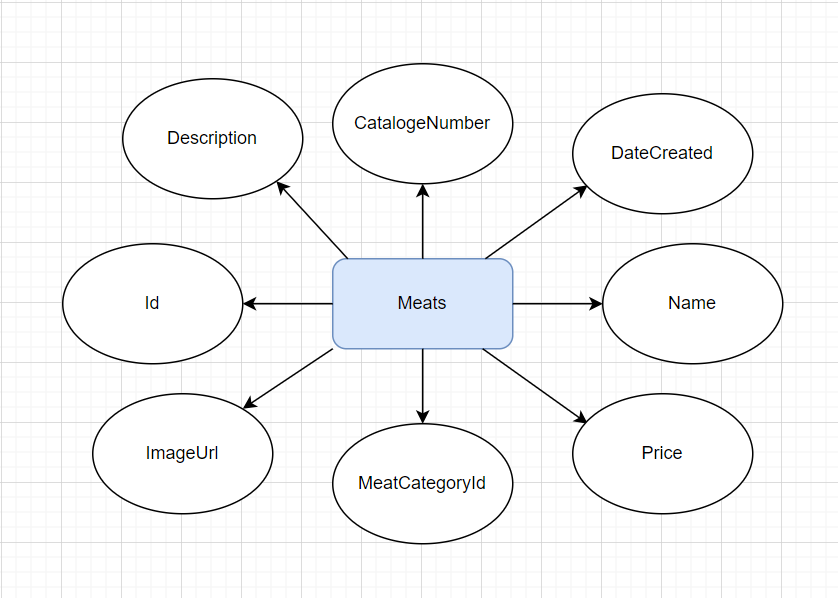
**3.1.2. Описание на обектите**

На **фигура 13** е показано, че E/R диаграмата се състои от 4 обекта, описващи предметната област на проекта. Това са:



***Фигура 13***

На **фигура 13** е показан обектът Meats(Меса).Това е основният обект, който реализира добавянето на ново месо. Той съдържа следните публично достъпни свойства: Id, Name, MeatscategoryId, CatalogeNumber, Description, Price, DateCreated и ImageUrl.

******

***Фигура 14***

**Свойства на обекта :**

**Id –** Това свойство присъства в почти всички обекти в базата от данни на нашето уеб приложение. Това е първичният ключ, по който се идентифицира всеки нов запис в базата и трябва да е от тип Int.

**Name –** Това свойство представлява наименованието в базата и трябва да е от тип string.

**MeatType–** Това свойство е от тип Int и представлява външен ключ към таблицата MeatTypes. Чрез това поле ще извлече вече готов запис за прошутото или сладкото.

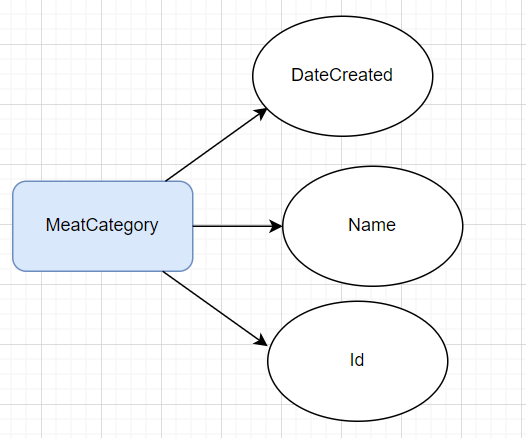
**MeatCategoryId –** Това свойство също представлява външен ключ към таблицата MeatCattegories Трябва да бъде от тип Int.

**Price –** Представлява цената, която ще се добавя. Трябва да бъде от тип Decimal, за да бъде десетично число.

**Description –** Описание , трябва да бъде String.

**DateCreated –** Това е дата на актуализация или добавяне в системата. Трябва да бъде от тип DateTime.

**ImgUrl –** Това визуализира снимката на даденото месо или сладко ,под формата на продукт в сайта.



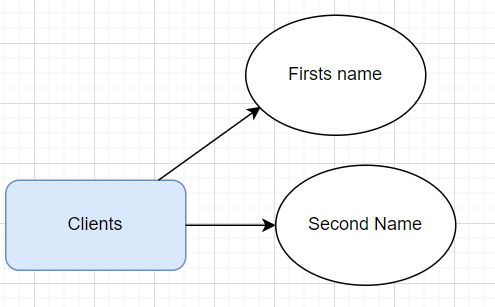
**Фигура 15**

Свойства на обекта:

**Id –** Това свойство присъства в почти всички обекти в базата от данни на нашето уеб приложение. Това е първичният ключ, по който се идентифицира всеки нов запис в базата и трябва да е от тип Int.

**Name –** Това свойство представлява наименованието в базата и трябва да е от тип string.

**DateCreated –** Това е дата на актуализация или добавяне в системата. Трябва да бъде от тип DateTime.

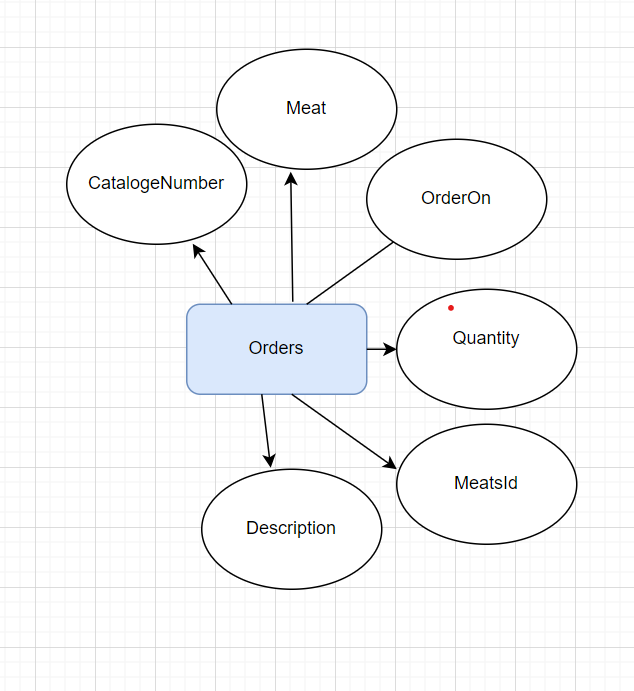
******

***Фигура 16***

***Свойства на обекта:***

**First Name –** Това свойство представлява наименованието в базата и трябва да е от тип string.

**Second Name –** Това свойство представлява наименованието в базата и трябва да е от тип string.



**Фигура 17**

**Обясанение**

**.**

**.**

**.**

**.**

**.**

**.**

**.**

**.**

**.**

**..**

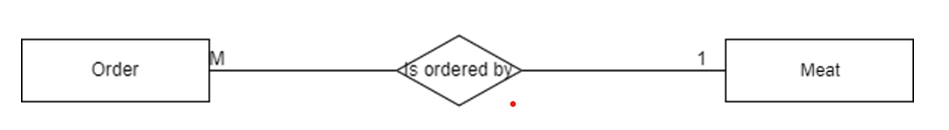
**.**

**3.1.4. Описание на връзките между обектите в базата от данни**

**3.1.4.1** *Един клиент поръчва повече от един път*

**

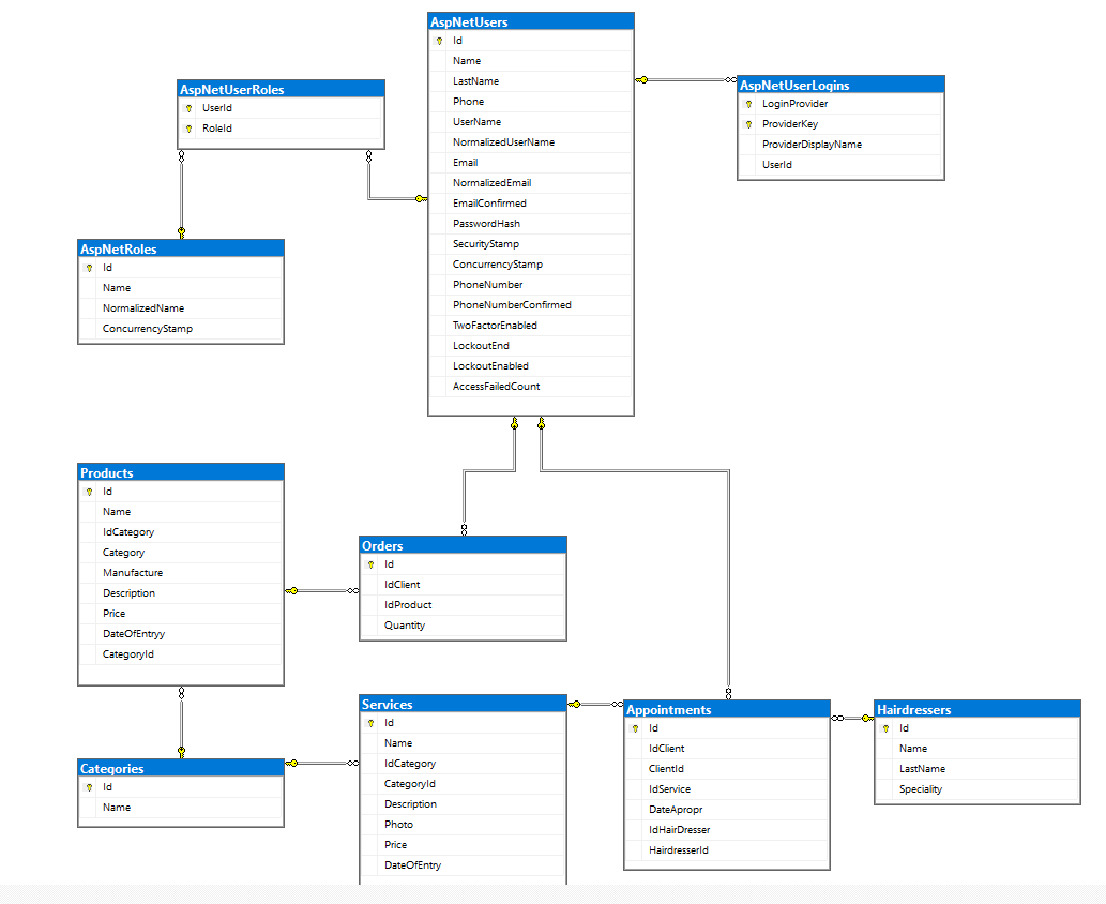
*Един продукт мможе да се поръчва много пъти*

**

*Една категория се повтаря за много продукти*

**

**E/R диаграма на проекта след като са създадени и реализирани класовете в ASP.NET приложението, което е показано на фигура 18**



**Фигура 18**

На **фигура 18** е показана диаграма на базата от данни на нашето ASP.NET Core MVC приложение, която е създадена по технологията ORM – Code – First. Класовете, които описват отделните таблици в базата се намират в папката Data от структурата на проекта. Класът ApplicationDbContext.cs описва структурата на базата от данни, като клиентът е описан в класа Customer.cs като наследник на таблицата AspNetUser от IdentitySystem. Пакетът EntityFramework представлява пакет от класове и инструменти за работа с бази от данни от SQL Server със средствата на езика C#. Той поддържа всички SQL заявки, съобразени с релациите в базата данни. Таблиците, от точка 3.1.3. са описани в посочената диаграма.

* **Структура на MVC приложението – *Controllers*, *Views*, *ViewModels*, *Routs –*** *описание на стартирането и комуникация между Models – Views – Controllers.*

**>>>** *Тук се копира код от конкретния /Вашия/ проект, като се правят обяснения по кода и LINQ изразите. Кодът се копира ( а не скрийншот!!!) и се поставя в каре… като примера по-долу.*

**>>> За какво да пишем:**

* **Как работи схемата Routes Engine**
* **Как работи шаблона на Controller с GET и POST заявки към Actions**
* **Описание на CRUD операциите- особености на отделните ACTIONS, параметри и валидиране на моделите, получени от View, както и зареждане на модела, който се подава към**

*Примерно: обяснете всеки фрагмент от следния код…*

* **Работа с View моделите (където е необходимо). Разлика с използването на ViewBag и/или ViewData, които се използват във Вашия проект (може да се опише разликата между двата обекта)**
* **Описание на Razor Pages и Razor Views (TagHelpers и HtmlHelpers)**
* **Описание и конфигурация на Services (ако е приложимо)**

* **Авторизация и оторизация на потребителите на приложението –** *описание на функционалността на отделните потребители според ролите в приложението и как е реализирано това в проекта /според описания дизайн в т. 2.3./ - кой потребител до какви функции има достъп, от какво меню се осъществява този достъп и как това е реализирано в кода /на Controller, от View и т.н. /*

**ЗАКЛЮЧЕНИЕ**

*Общение на реализирания проект, предимства и недостатъци, теденции за бъдещи разработка и развитие.*

*До каква степен е постигнато планираното за разработване приложение.*

**ИЗПОЛЗВАНА ЛИТЕРАТУРА**

Примери: /трябва да се следва точно следния формат/

1. Наков С., Колев В. и колектив, Въведение в със C#, София, 2015, ISBN 978-954-400-527-6

2. Колисниченко Д., HTML 5 & CSS 3 - практическо програмиране за начинаещи. Трето издание, Асеневци, София, 2017, ISBN 9786197356113

3. Pro ASP.NET Core 3: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages 8th ed. Edition, Apress, New York, 2020, ISBN 1484254392

4. Learn ASP.NET Core 3: Develop modern web applications with ASP.NET Core 3, Visual Studio 2019, and Azure, 2nd Edition Paperback, Packt, Birmingham, UK, 2019, ISBN 1789610133

Електронни ресурси

5. <https://www.w3schools.com/>,

6. <https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-3.1>; <http://www.introprogramming.info/intro-csharp-book/>

7. <https://to6esko.github.io/> - Български превод на Eloquent JavaScript