ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО ЕЛЕКТРОТЕХНИКА - ВАРНА



**ДИПЛОМНА РАБОТА**

**Тема:**Разработка на уеб сайт за продаване на авточасти втора употреба „Parts4Cars”  
  
**Дипломант:** Пламен Христов Георгиев (12г клас)   
**Специалност:** Приложно програмиране  
  
**Професия:** Приложен програмист  
  
**Ръководител-консултант:** инж. Андрей Тодоров

ПГЕ Варна, 2022 г.

**Съдържание:**

Глава 1: Въведение

1. Цели и задачи на проекта
2. Очаквани резултати

Глава 2: Използвани технологии

1. C#
2. ASP.NET Core
3. ORM
4. MVC
5. SQL
6. HTML
7. CSS
8. JavaScript
9. Visual Studio

Глава 3: Проектиране

1. Изисквания към уеб приложението
2. Проектиране на графичния интерфейс
3. Структура на софтуерния пакет

Глава 4: Описание на програмните компоненти

1. Работа с данни
2. Изглед и дизайн на страниците
3. Заявки към базата данни

Глава 5: Ръководство за потребителя

1. Как се става администратор?
2. Използване на уеб приложението

Глава 6: Заключение и бъдещо развитие

Използвана литература

Приложения

**Глава 1: Въведение**

* 1. **Мотивация за създаването**

Днес IT технологиите са неизменна част от нашето ежедневие. Те са общодостъпни, лесни за употреба както от вкъщи, така и в движение. Благодарение на тях в днешно време много от задачите, които до скоро са се извършвали на хартия, в офис, на работа, могат лесно и удобно да се решат и чрез едно малко смарт устройство, побиращо се в някой от джобовете ни. Защо да не насочим и бизнеса си в тази посока? По този начин ще се разшири кръгозорът от клиенти и всички части ще бъдат достъпни от всяка точка на страната. А това значи по-голяма печалба.

* 1. **Цели и задачи на проекта**

Главната цел на този проект е разработката на едно цялостно уеб приложение, предназначено за продаване на авточасти втора употреба, даващо възможност на един бизнес да разшири обхвата си от клиенти. То трябва да бъде лесно за разбиране и удобно за употреба.

Задачите, които проектът налага са следните:

* Избор на подходяща среда за разработка – тя трябва да бъде масово използвана, за да бъде самото приложение лесно за поддръжка и бъдещо развитие и да е добре позната на разработчика, за да може той лесно да се справи с поставената му задача;
* Избор на технологии за дизайн на приложението;
* Проектиране и реализация на базата данни;
* Добре оформена структура на кода, за да бъде лесно четим, бързо разбираем и лесен за промяна;
* Защита от злонамерени потребители и софтуери

Разработката включва проектиране на самото приложението, изготвяне на дизайна му, разработване на логиката му, тестване и предаване към крайния потребител.

* 1. **Очаквани резултати**

Очакваните резултати са изготвеното приложение да бъде качествено, с пълна функционалност, добре изглеждащо, адаптивно и надеждно. Да успее да задоволи нуждите на крайния потребител, като осигури лесно боравене с него, бързо и удобно разглеждане, подбиране и поръчка на нужните за клиента авточасти.

Ползите, които се очакват за бизнеса са:

* бързо разширение;
* мигновено покачване на доходите;
* растеж на броя доволни клиенти;
* просперитет на бизнеса.

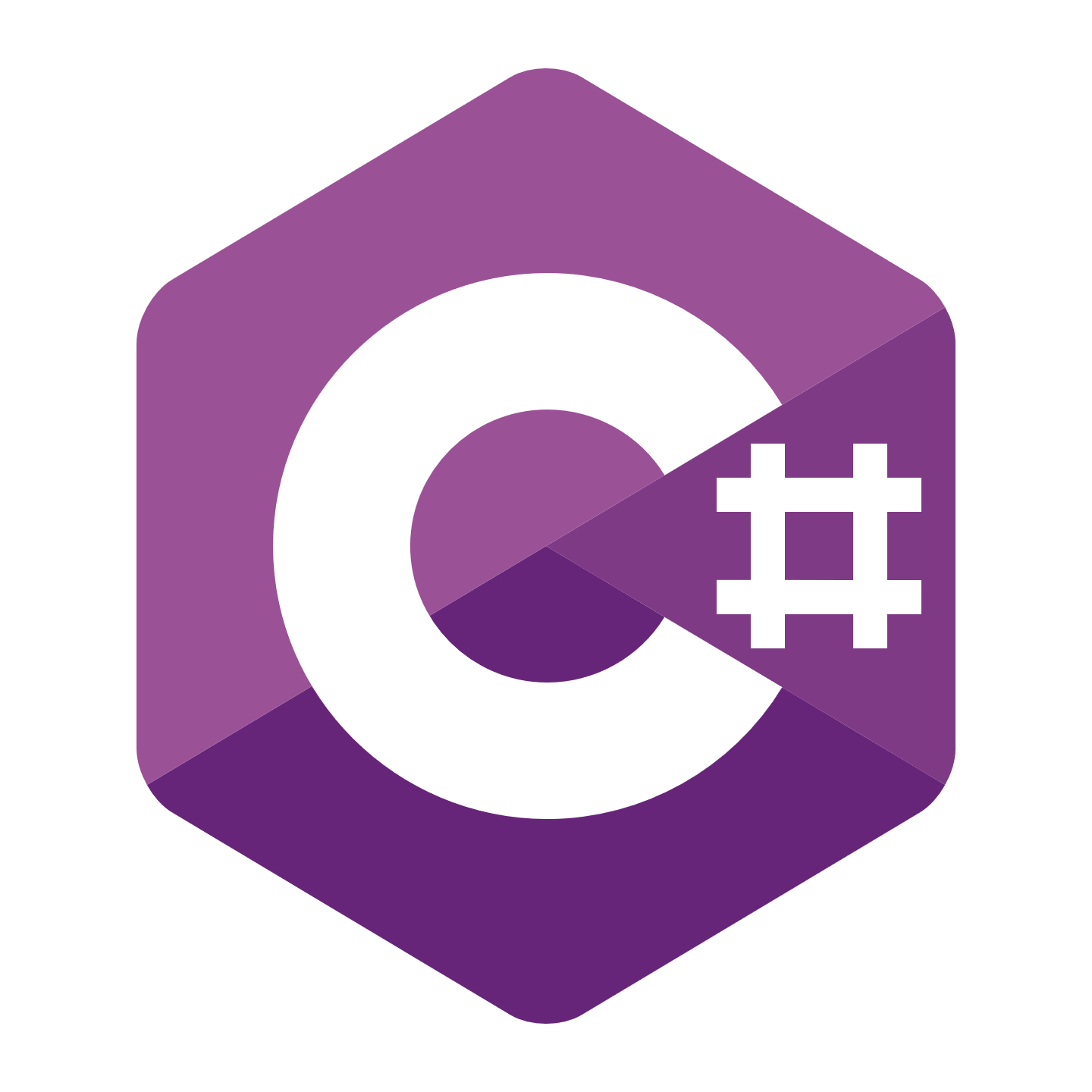
Ползите от страна на клиентите са:

* бързо снабдяване с нужните за ремонт на автомобила нискобюджетни, но здрави и качествени части;
* лесна поръчка отвсякъде.

**Глава 2: Използвани технологии**

При изготвянето на проекта имаме пълната свобода за избор на:

* среда за разработка;
* програмен език;
* методологии за разработка на софтуер;
* технологии за програмиране.
  1. **C#**

****

**Какво представлява езикът C#?**

В конкретния случай ще разгледаме програмния език C# (произнася се „Си Шарп“). Той е създаден от Microsoft през 2001 г., а последното обновление за него излезе на 15-ти ноември 2017 г. т.е. това е един актуален език. Може да се каже, че C# e Microsoft-ския 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. При работа с такъв език разработчикът няма нужда от настройване на връзките между детайлите на компютъра и нуждите на програмата т.е. той се концентрира върху самата програма, която създава.  
  
 При компютърните езици от ниско ниво разработчикът, обратно, трябва да „инструктира“ т.е. да настрои своята програма така, че да се знае с колко памет разполага, как е свързана с нея и т.н. тоест той настройва и връзката между компютъра и програмата.  
  
 Но да се върнем на езиците от високо програмно ниво и по-конкретно C#. След като разгледахме какво представлява в най-общи линии, логично изниква и следващият въпрос, а именно –

**Къде и за какво се използва?**

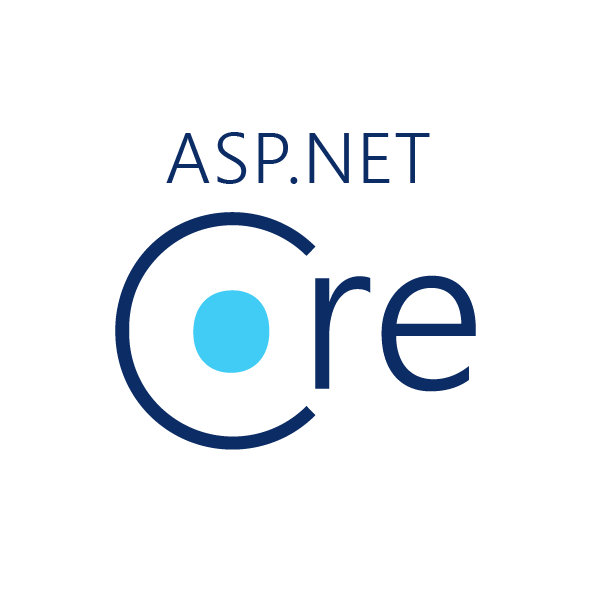
* **Десктоп приложения -** От Microsoft използват **C# във всички свои продукти** в това число всичките си десктоп приложения за Windows 8 и 10 т.е. **езикът става все по-актуален и широкоизползван**, което дава **повече шансове за реализация**. Това е така, тъй като езикът все пак е разработен като част от инициативата .Net (чете се „Дот нет“) – софтуерната рамка (framework) на Microsoft.
* **Уеб приложения -** ASP.NET е софтуерна рамка с отворен код, която също е част от Microsoft семейството .NET Framework. 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.
* **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.
* **Игри -** За много от вас може би това звучи несериозно, но все пак това също е една от възможностите за реализация със С#. 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.
  1. **MVC**

**Какво е MVC?**

****

MVC е съкратено от “Model-view-controller”. Представлява архитектурен шаблон при програмния дизайн. Той отговаря за разделянето на бизнес логиката на три взаимосвързани части: “Model”, “View” и “Controller”. Какво представляват те и какви са техните функции?  
  
 „Model“ – това представлява „ядрото“ на приложението ви. В него се записват всички данни, върху които искаме да работим;  
  
 „View“ – най-общо казано това е изходна част от софтуера, която визуализира наличните, обработени данни;  
  
 „Controller“ – това е тази част от библиотеката (library) или самия код, която извиква допълнителни методи върху събраните в модела данни и по този начин те могат да бъдат обработвани.

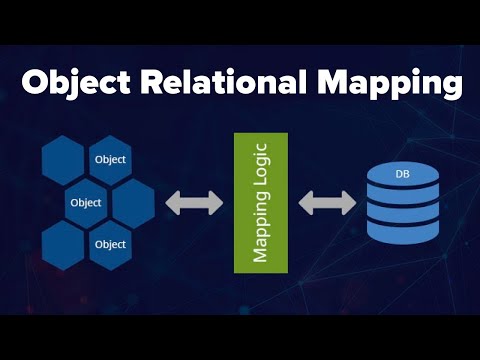
* 1. **ASP.NET Core**

****

**Какво е ASP.NET Core?**

ASP.NET Core е безплатна софтуерна рамка за уеб разработка, с отворен код. Също така тя се явява и следващата стъпка в еволюцията ASP.NET. Тя е разработена съвместно от Microsoft и общността, която е събрала през годините на своето развитие. ASP.NET Core е модуларна софтуерна рамка, която може да върви както на пълната .NET рамка, така и на крос-платформената .NET Core. Въпреки, че е нова софтуерна рамка, изградена върху нов web stack, тя има висока степен на съвместимост с ASP.NET MVC.  
  
 Спрямо своите предшественици ASP.NET Core поддържа нова функция – т.нар. „side by side versioning”. При нея различни приложения, които използват една и съща машина, могат да таргетират различни версии на ASP.NET Core, в зависимост от версиите (и нуждите) си. Това не е възможно с по-стари издания на ASP.NET.

* 1. **ORM**

****

**Какво е ORM(Object Relational Mapping)?**

ORM представлява техника за програмиране, която служи за трансформиране на данни, използвани в обектно-ориентирания програмен език или програма и релационната база данни (тип SQL) като персистенция. Това ще накара програмните стойности да създадат виртуална обектно-ориентирана база данни, която да съдържа данните, от които се нуждаем.

* 1. **SQL**

****

**Какво е SQL?**

**SQL**, или **Език за структурирани запитвания**, (на английски: ***S****tructured****Q****uery****L****anguage, SQL*) е популярен език за програмиране, предназначен за създаване, видоизменяне, извличане и обработване на данни от релационни системи за управление на бази данни. Стандартизиран е от ANSI и ISO.

*SQL* обикновено се произнася *ес-кю-ел* (*ess-cue-el*) (виж английската азбука) – или *сикуъл* (*sequel*). Имената на продукти, съдържащи *SQL*, имат всеки свое произношение, например: MySQL официално се произнася *май-ес-кю-ел* (*my ess cue el*); PostgreSQL – *постгрес* (*postgres*); и Microsoft SQL Server на български като *Майкрософт ескюел сървър*, а иначе като *Майкрософт сикуъл сървър* (*Microsoft-sequel-server*).

* 1. **HTML**

****

**HTML** (съкращение от термина на английски: ***H****yper****T****ext****M****arkup****L****anguage*, 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 е стандарт в интернет, а неговите стандарти се определят от международния консорциум W3C. Текущата версия на стандарта е HTML 5.0 (от 28 октомври 2014 г.), а предходната стабилна версия е HTML 4.1.

Описанието на документа става чрез специални елементи, наречени HTML елементи или техните маркери, които се състоят от тагове и съответстващите етикети (HTML tags) и ъглови скоби (като например елемента <html>). HTML елементите са основната градивна единица на кода, който изграждат уеб страниците. Чрез тях се форматира, графично оформя текста и неговите отделните части в рамките на една уеб страница, като например заглавия, цитати, текстови раздели, хипертекстови препратки и т.н. Най-често HTML елементите са групирани по двойки <h1> и </h1>.

В повечето случаи HTML кодът е написан в текстови редактори, с файлов формат .html, .htm, dhtml и се качва и хоства на сървъри, които са онлайн в интернет или са част от www мрежата. Тези .html файлове съдържат програмно на таговете на HTML и текстово съдържание със маркери и коментари – също инструкции за браузъра, за това какъв точно тип е .html страницата, а също за това как да се показва текстът, особено що се отнася до езиковите характеристики. За да се илюстрира как се включва текст в HTML код: <маркер> Някакъв текст. </край на маркера>. уеб браузърите са програмирани от своя страна така, в повечето случаи, макар че някои браузъри могат да имат съответно проблеми на версията, за да могат да прочетат HTML документите и да ги покажат на екрана като уеб страници. Браузърите не показват самите HTML тагове, освен ако не се отиде в менюто за да се направи това, така че те „интерпретират“ (тоест парсват) съдържанието на страницата като код и текст за да могат след работа на процесора да покажат желаното уеб-съдържание.

Основното предимство на HTML е, че уеб-страниците, които са го включват в кода си, могат да се разглеждат чрез показването им от браузъра на екрана на повечето устройства. Уебстраницата може да има дизайн, който дори изглежда с добър дизайн с помощта на CSS или „правилно оформен“ (например с помощта на C#), както върху монитора на персоналния компютър, но също и върху миниатюрния дисплей на пейджър или дисплея на мобилен телефон.

HTML може да прикрепя скриптове писани на езици като JavaScript, който е помощен за HTML, и това променя поведението на дадена уеб страница. Cascading Style Sheets (CSS) се използват, като това се прави за да се определя изгледа и оформлението на текста и други включени в страницата изображения и илюстриращи материали. World Wide Web Consortium (W3C) поддържа както HTML, така и CSS, и насърчава използването на CSS в HTML страниците още от 1997. Това допринася за разделяне съдържанието и структурата на уеб страниците от тяхното визуално представяне.

* 1. **CSS**

****

**CSS** (**Cascading Style Sheets**) е език за програмиране и също описание на уеб дизайн програмни стилове (език за стилови файлове, style sheet language) – използва се основно за описание на онлайн представянето на уеббазиран документ, който e 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.

**Начини за прилагане на стилове:**

- Външен източник на стил – линк

- Дефиниция в документа

- Дефиниция на конкретен ред в документа

**Концепция на селекторите:**

-\* - избор на всички елементи

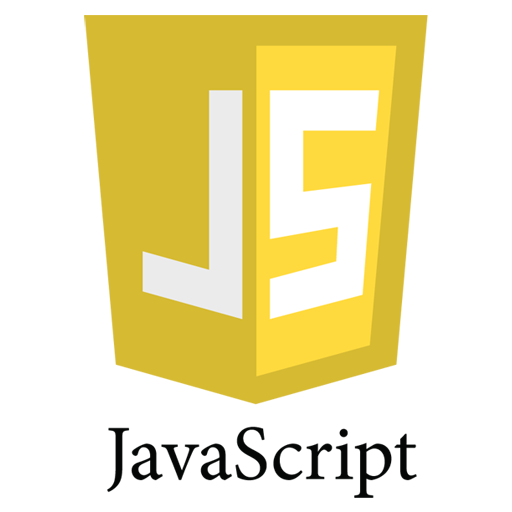
- . – избор на клас

- # - избор на елемент по ID

- Контекстуални селектори – комбинация от няколко селектора, като зададения ефект се проявява в зависимост от подредбата им.

- Групиране на селектори – няколко селектора могат да бъдат подредени като се отделят със запетаи и им се зададе еднакъв ефект, т.е. атрибути с еднакви стойности.

* 1. **JavaScript**

****

**Какво е JavaScript?**

JavaScript е програмен език, който позволява динамична промяна на поведението на браузъра в рамките на дадена HTML страницата. JavaScript се зарежда, интерпретира и изпълнява от уеб браузъра, който му осигурява достъп до Обектния модел на браузъра. JavaScript функции могат да се свържат със събития на страницата (например: движение/натискане на мишката, клавиатурата или елемент от страницата, и други потребителски действия). JavaScript е най-широко разпространеният език за програмиране в интернет. Прието е JavaScript програмите да се наричат скриптове.

**Какви са възможностите на JavaScript?**

JavaScript може да влияе на почти всяка част от браузъра. Браузъра изпълнява JavaScript кода в цикъла на събития т.е. като резултат от действия на потребителя или събития в браузъра (например document.onLoad).

**Основни задачи в повечето JavaScript приложения са:**

- Зареждане на данни чрез AJAX;

- Ефекти с изображения и HTML елементи: скриване/показване, пренареждане, влачене, слайд шоу, анимация и много други;

- Управление на прозорци и рамки;

- Разпознаване на възможностите на браузъра;

- Използване на камерата и микрофона;

- Създаване на 3D графики WebGL;

- По-добър и гъвкав потребителски интерфейс.

**Какво не може да се прави с помощта на JavaScript:**

- Не може да се записва информация на потребителския компютър или отдалечения сървър.

- Не може да се запазва информация директно в отдалечена база данни.

- Не може да се стартират локални приложения.

* 1. **Visual Studio**

****

**Какво е Visual Studio?**

**Microsoft Visual Studio** е мощна интегрирана среда за разработка (на английски: *integrated development environment*, *IDE*) на софтуерни приложения за Windows и за платформата .NET Framework. Използва се за разработка на конзолни и графични потребителски интерфейс приложения, както и Windows Forms или WPF приложения, уеб сайтове, уеб приложения и уеб услуги на всички поддържани платформи от Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework и Microsoft Silverlight.

Visual Studio предоставя мощна интегрирана среда за писане на код, компилиране, изпълнение, дебъгване (както за високо така и за машинно ниво), тестване на приложения, дизайн на потребителски интерфейс (форми, диалози, уеб страници, визуални контроли и други), моделиране на данни, моделиране на класове, изпълнение на тестове, пакетиране на приложения и стотици други функции. Могат да се добавят и плъгини, които повишават функционалността на почти всяко ниво – включително добавянето на поддръжка за source-control системи (като Subversion и Visual SourceSafe), добавяне на нови инструменти като редактори и визуални дизайнери за domain-specific languages или инструменти за други аспекти (като например: Team Foundation Server, Team Explorer).

Visual Studio поддържа различни езици за програмиране (например: C#, VB.NET, C/C++, F#, XML/XSLT, HTML/XHTML, JavaScript, CSS и други) и различни технологии за разработка на софтуер (Win32, COM, ASP.NET, ADO.NET Entity Framework, Windows Forms, WPF, Silverlight и още десетки други Windows и .NET технологии). Съществуват и отделни езикови версии на Visual Studio, които обаче предоставят по-ограничени услуги за потребителя: Microsoft Visual Basic, Visual J#, Visual C# и Visual C++.

**Глава 3: Проектиране**

* 1. **Изисквания към уеб приложението**

Регистрация. Вход/Изход.

Преглеждане на начална страница с всички налични обяви за авточасти.

Потребител

Филтриране на обявите по име, категория, марка и модел на автомобила.

Отваряне на нова страница с подробности за всяка отделна обява.

Администратор

Добавяне/Премахване на обява(страница) в/от количката.

Извършване на поръчка с избрани части.

Добавяне, редактиране, изтриване на обява.

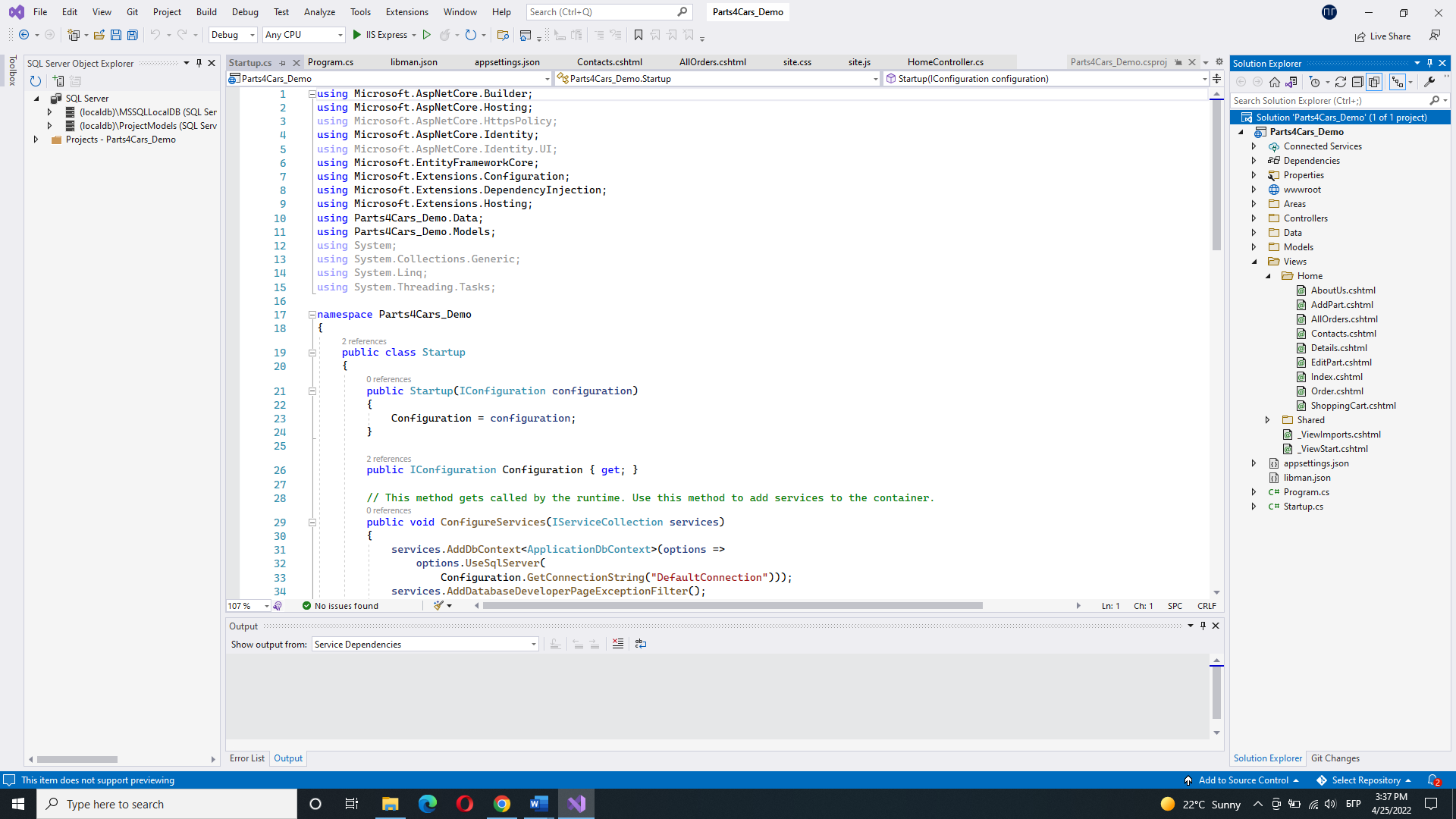
Преглед и изтриване на поръчки.

* Регистрация. Вход/Изход.
* Преглеждане на начална страница с всички налични обяви за авточасти.
* Филтриране на обявите по име, категория, марка и модел на автомобила.
* Отваряне на нова страница с подробности за всяка отделна обява.
* Добавяне/Премахване на обява(страница) в/от количката.
* Извършване на поръчка с избрани части.
* Добавяне, редактиране, изтриване на обява.
* Преглед и изтриване на поръчки.
  1. **Проектиране на графичния интерфейс**

След като изискванията на проекта са вече изяснени, е време да пристъпим към създаването на графичния интерфейс. Трябва да се създадат толкова страници, колкото са необходими за изпълнението на всяко едно от зададените изисквания. Трябва да се създаде и навигация, чрез която потребителят ще може да се прехвърля от страница на страница.

Когато тази стъпка е изпълнена, трябва да се премине към следващата – имплементация на логиката на приложението. Тоест, да се осъществи връзката между потребителския интерфейс и базата данни.

* 1. **Структура на софтуерния пакет**

****

* Connected Services – свързаните услуги (SQL Server Express LocalDB и др.);
* Dependencies – базови връзки;
* Properties – свойства на проекта;
* wwwroot – съдържа файлове за стилизация, javascript файлове и други;
* Areas – съдържа автоматично генерирани страници (за вход, за регистрация и т.н.);
* Controllers – съдържа класове, в които са създадени методите, чрез които се извикват различните уеб страници;
* Data – съдържа миграциите и клас, осъществяващ връзката към базата данни;
* Models – съдържа класове, които описват някой от таблиците в базата данни;
* Views – Съдържа основните страници на сайта и някои компоненти, показващи се при изпълнението на определено условие;
* аppsettings.json – базови настройки на приложението. Тук се оказва пътят към базата данни;
* libman.json – още базови настройки;
* Program.cs – главната програма на приложението, чрез която то се изгражда при стартиране;
* Startup.cs – клас, който настройва приложението по определен начин при стартиране.

**Глава 4: Описание на програмните компоненти**

**4.1 Работа с данни**

Базата данни се създава във вградения SQL сървър на Visual Studio. Таблиците в нея са създадени чрез Code-First принципа, което значи, че за всяка една е създаден обект(клас) в папка Models, който определя какви колони ще има и съответно техните ограничения. Създава се клас за връзката между базата данни и приложението в папката Data – ApplicationDbContext. След това се създава миграция в Package Manager Console(команда – ADD-MIGRATION <име на миграцията>), която автоматично създава таблиците в базата данни. Името на миграцията може да бъде произволно. Накрая се запазват промените пак в Package Manager Console(команда – UPDATE-DATABASE).

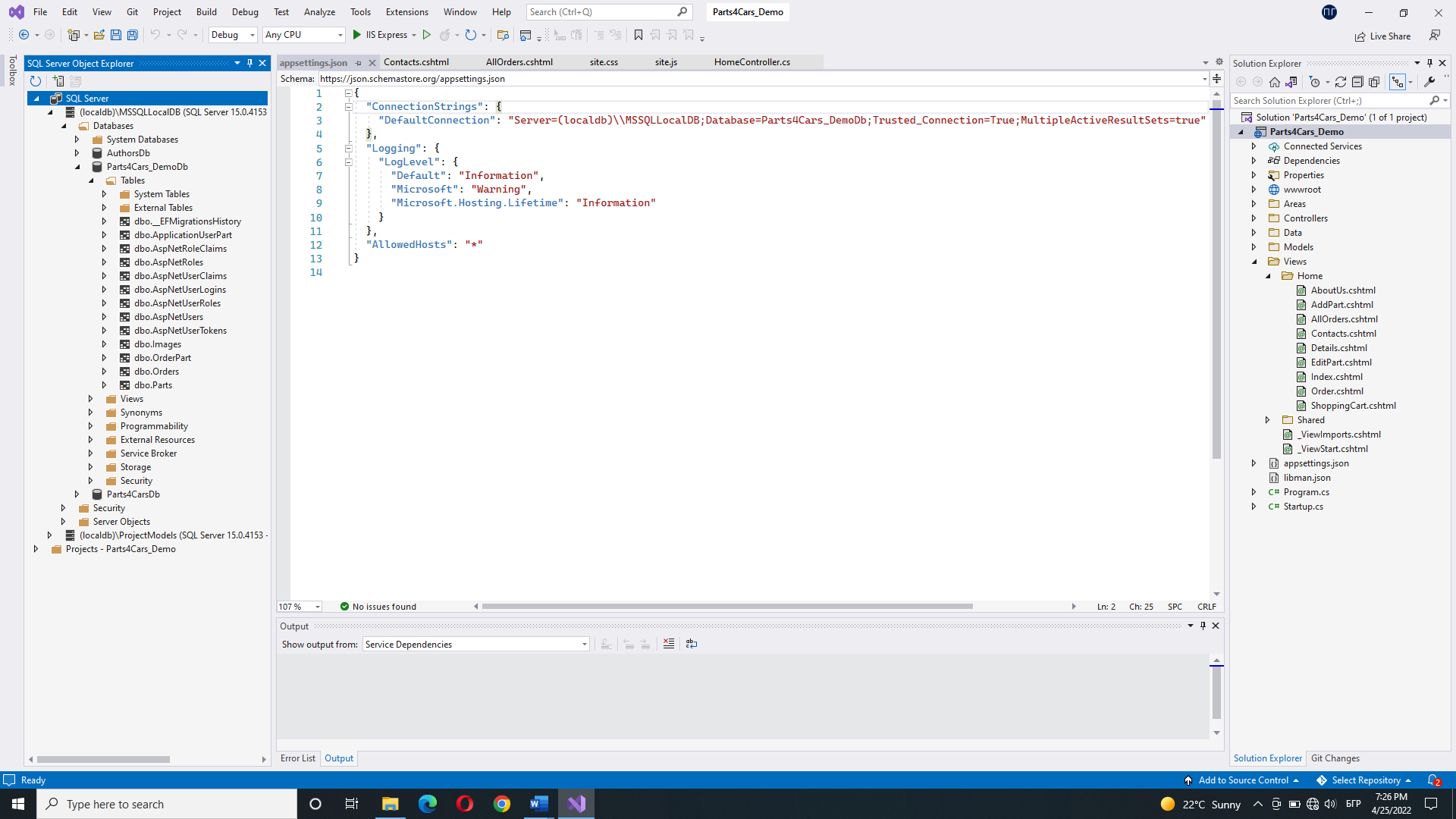
За връзка с нея се пише „символен низ за връзка (от английски - “connection string”), намиращ се във файла appsettings.json. В низа се вписват името на сървъра, името на базата данни, името на таблицата, дали връзката е сигурна и др.(в този ред):

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\MSSQLLocalDB;Database=Parts4Cars\_DemoDb;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

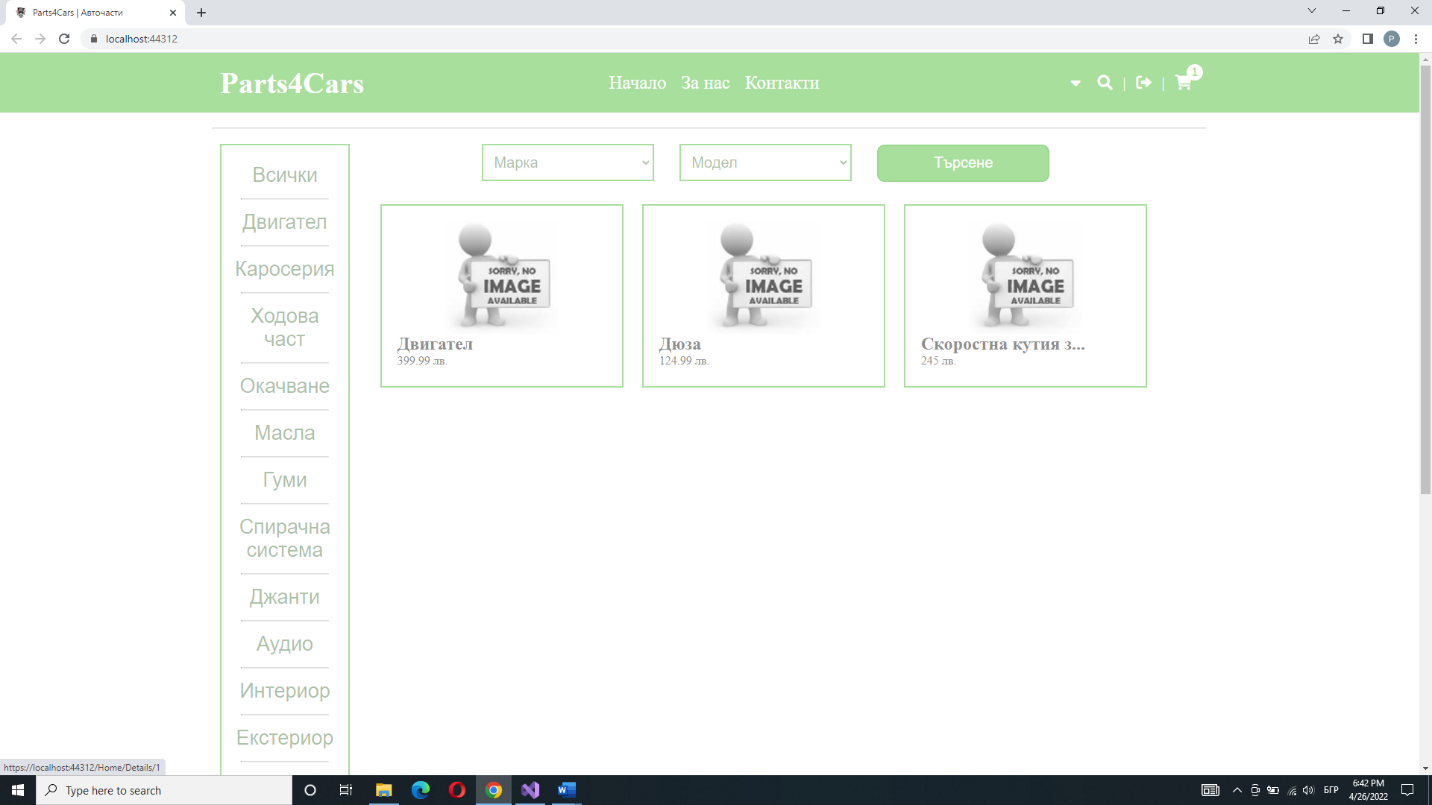
**Базата данни:**

****

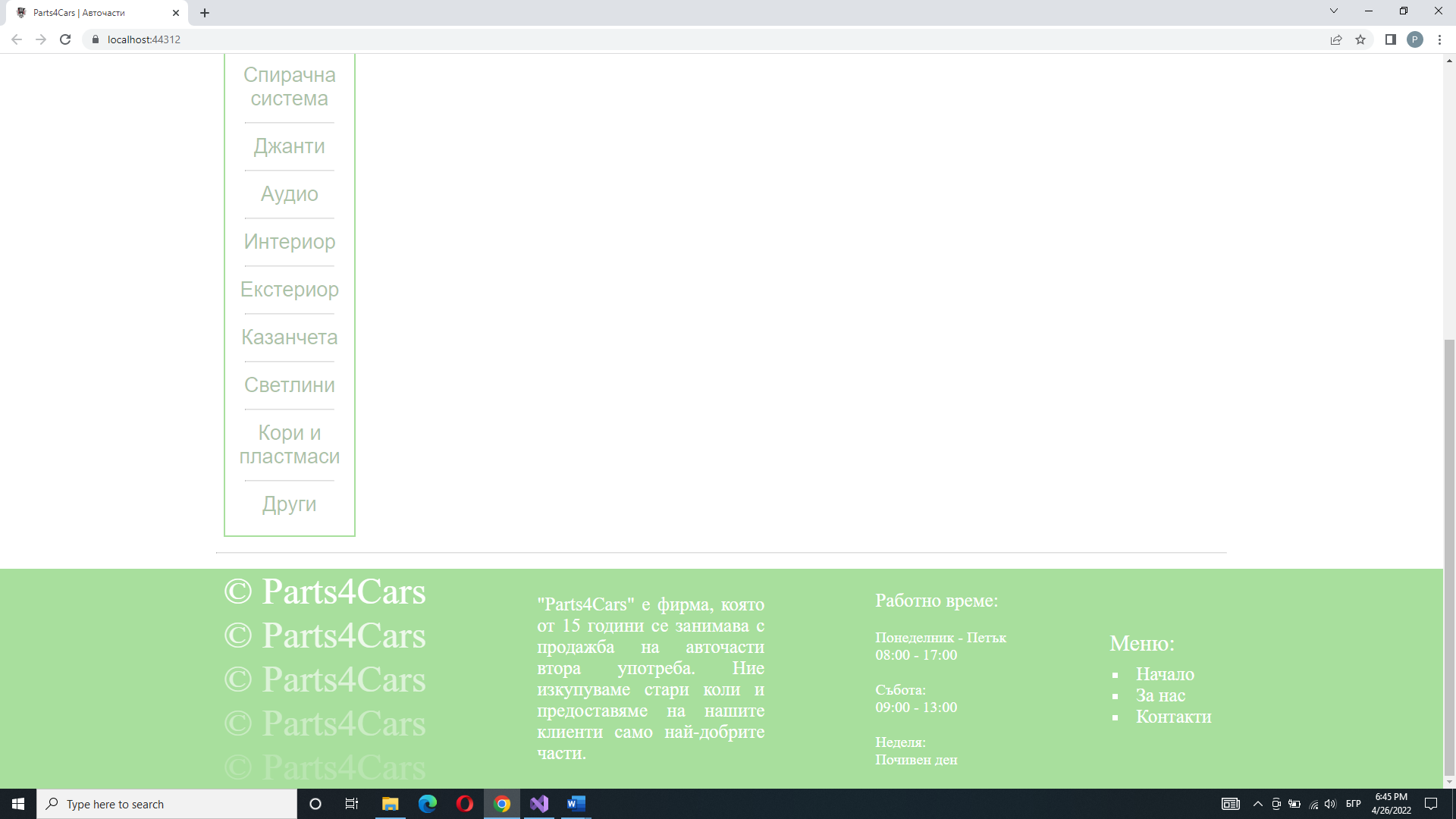
Това са таблиците, присъстващи в базата данни, която текущото приложение използва. Има таблица за части(Parts), таблица за снимки(Images), таблица за поръчки(Orders), таблица за потребители(AspNetUsers), таблица за ролите на потребителите(AspNetRoles), помощни таблици за връзките много към много(ApplicationUserPart, OrderPart, AspNetUserRoles), таблица за история на миграциите и други.

**4.2 Изглед и дизайн на страниците**

Всички страници на приложението са свързани помежду си, за да може потребителят по всяко време да достигне до желаната от него такава. Също така дизайнът на страниците е така оформен, че да може те да изглеждат еднакво добре както на лаптоп, така и на таблет и смартфон. Има два навигационни бара. Единият е в горния колонтитул, а другият е в долния. Този в горната част се скрива при по-малки устройства и остава само долният.

Горен навигационен бар:

Долен навигационен бар:

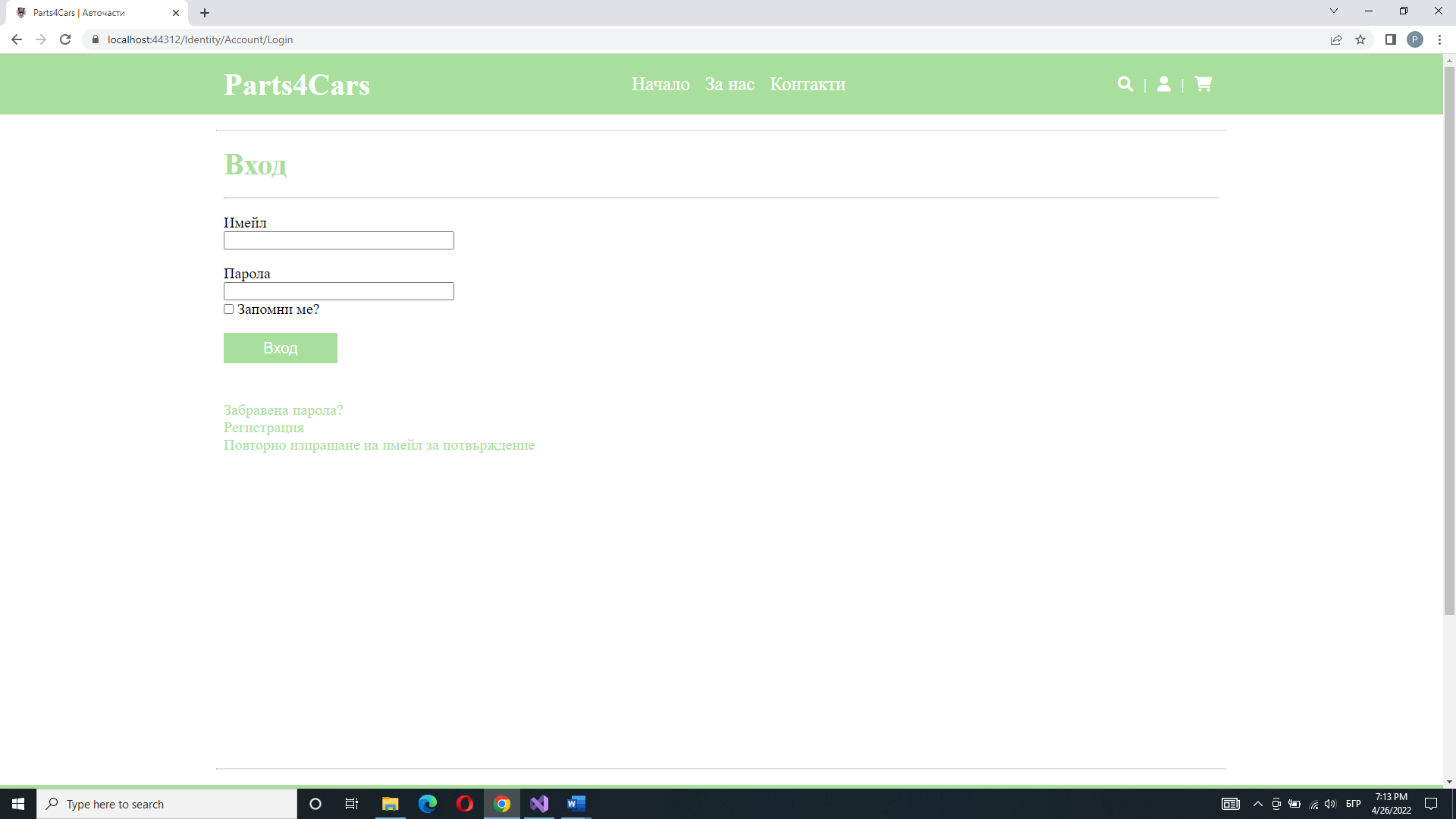


За да влезе в своя акаунт потребителят, той трябва да отиде на страницата за вход и да въведе своите имейл и парола. При натискане на бутона за вход се правят следните проверки:

* Попълнено ли е полето за имейл;
* Има ли наличен знак „@“ в имейла;
* Попълнено ли е полето за парола;

Ако всичко това е изпълнено, се изпраща заявка към програмата, за да се провери дали има съществуващ акаунт с тези данни за вход. Ако да – потребителят успешно влиза в своя акаунт и бива препратен към началната страница.

Форма за вход:

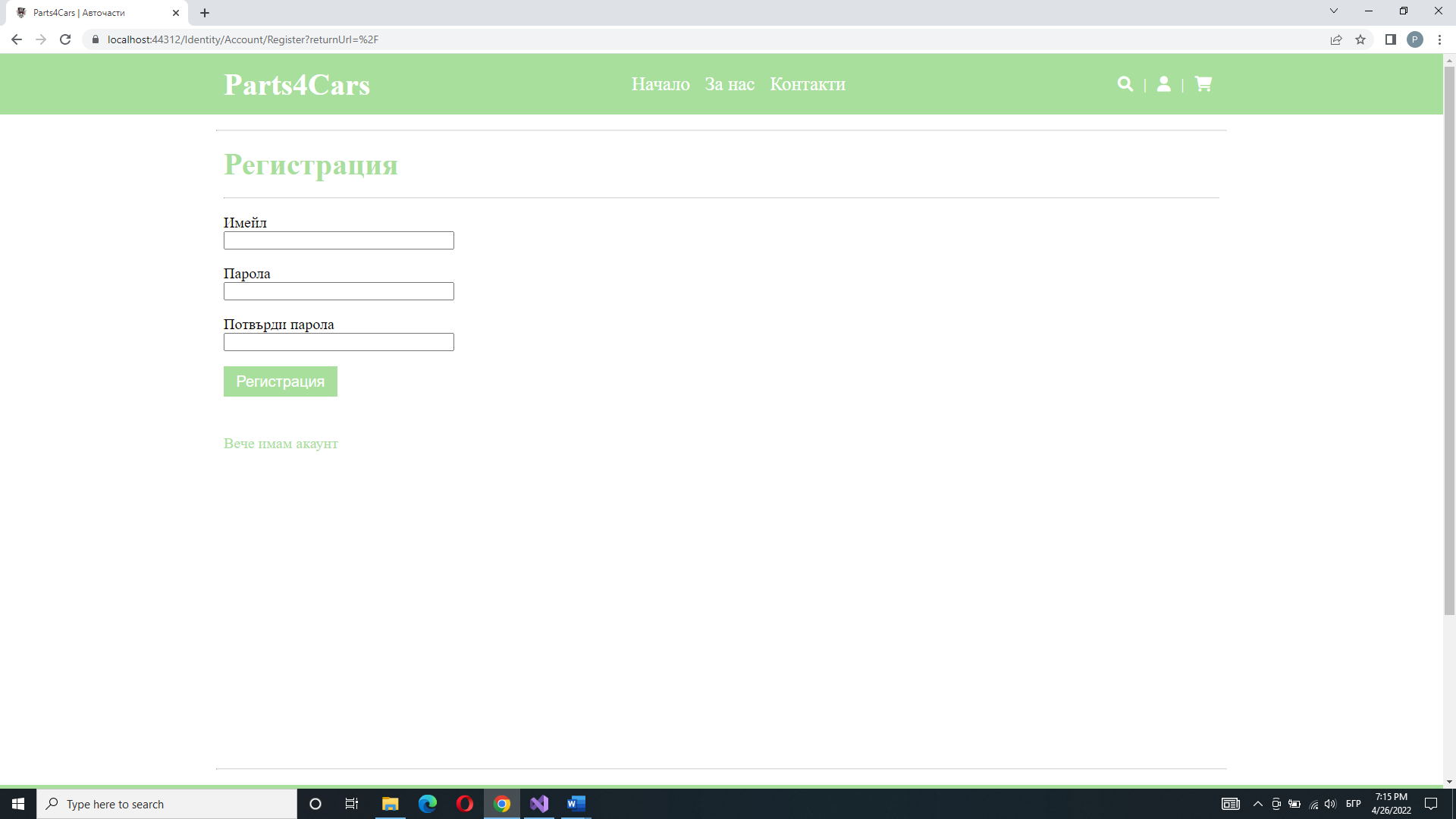


Ако потребителят е нов и все още няма акаунт на свое име, то той може да се регистрира като нов потребител. При опит за регистрация се извършват следните проверки:

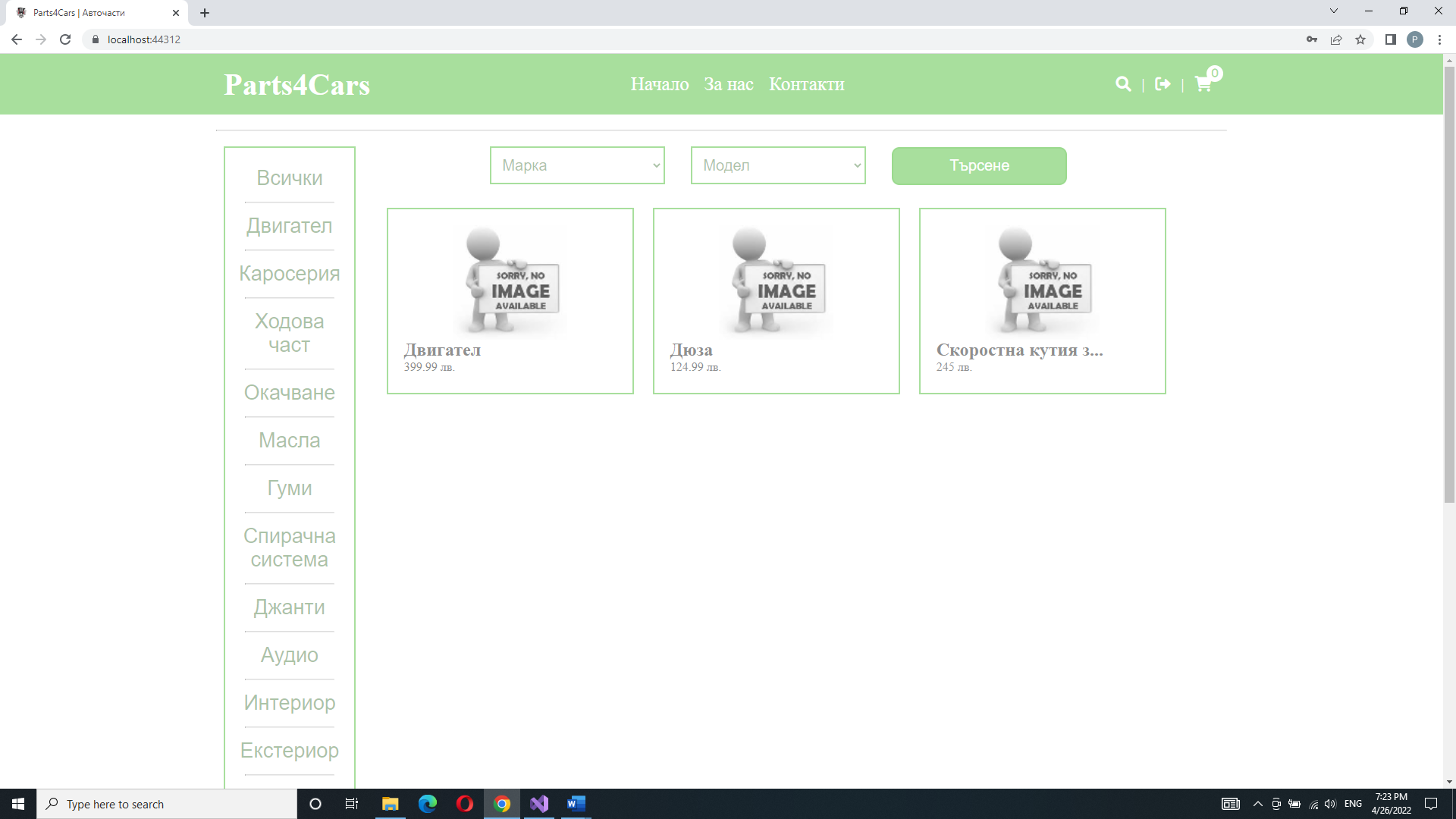
* Попълнено ли е полето за имейл;
* Има ли знак „@“ в имейла;
* Попълнено ли е полето за парола;
* Дължината на паролата между 6 и 100 символа ли е;
* Има ли налични поне една главна буква, една малка буква и една цифра;

Ако всички тези условия са изпълнени, то потребителят бива успешно регистриран и препратен към началната страница на сайта.

Форма за регистрация:



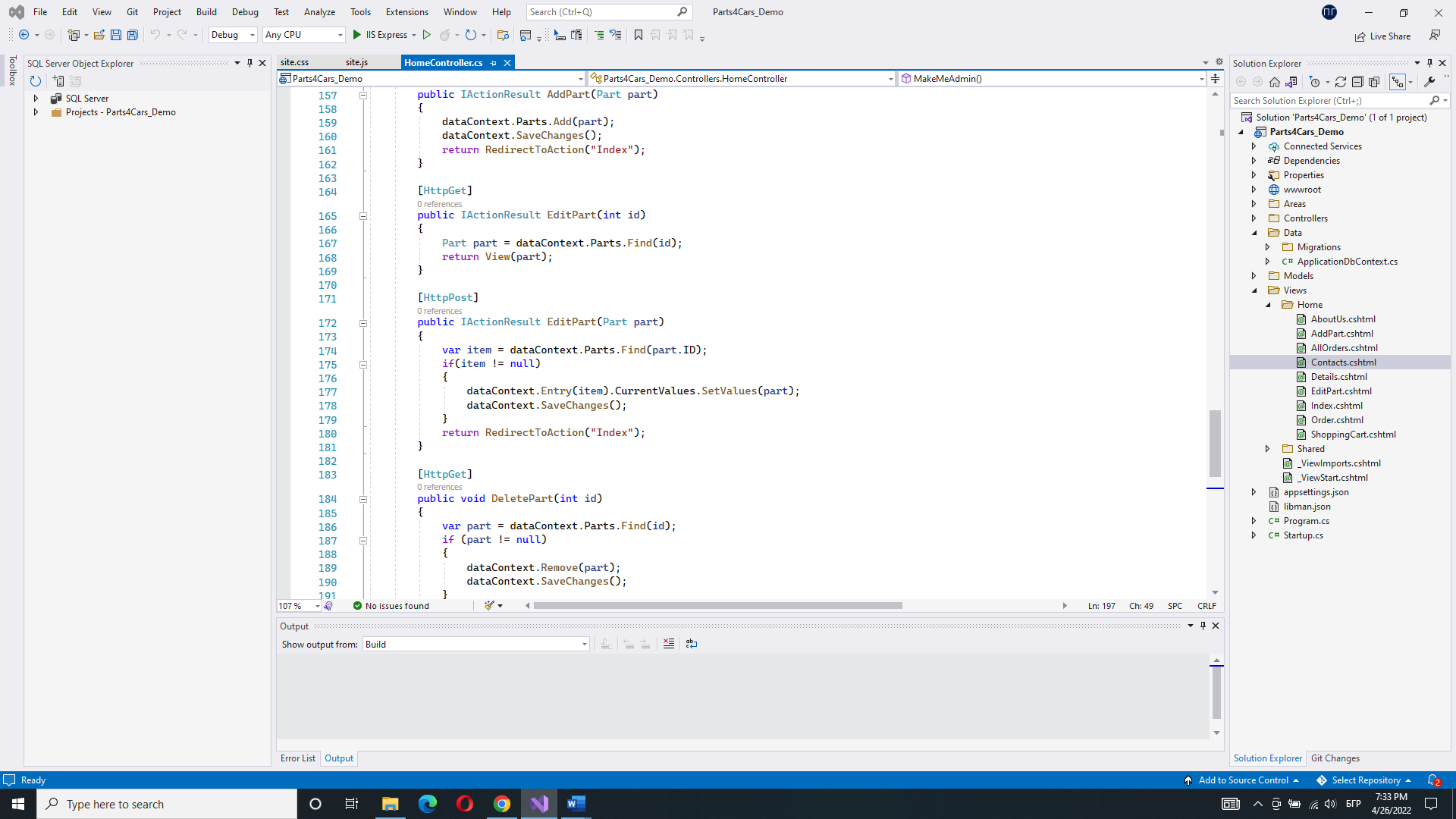
Начална страница на сайта:



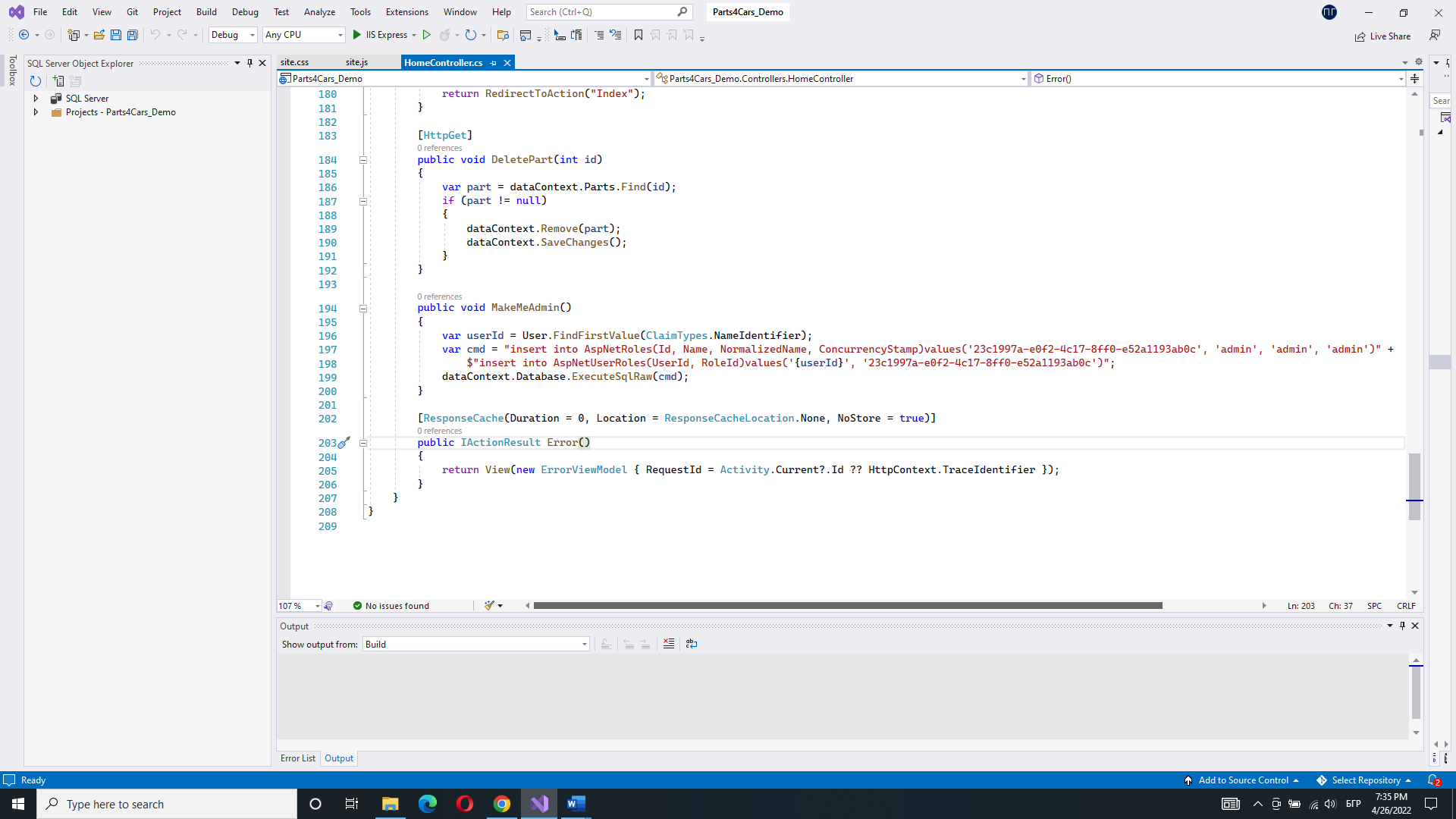
**4.3 Заявки към базата данни**

Заявките към базата данни се изпращат по два начина:

* Чрез ORM методи (**глава 2.3**);



* Чрез директна SQL заявка (**глава 2.5**);



И двата начина работят по един и същ начин. Чрез първия е по-лесно, по-бързо и по-удобно. Вторият обаче е по-комплексен и предоставя повече възможности.

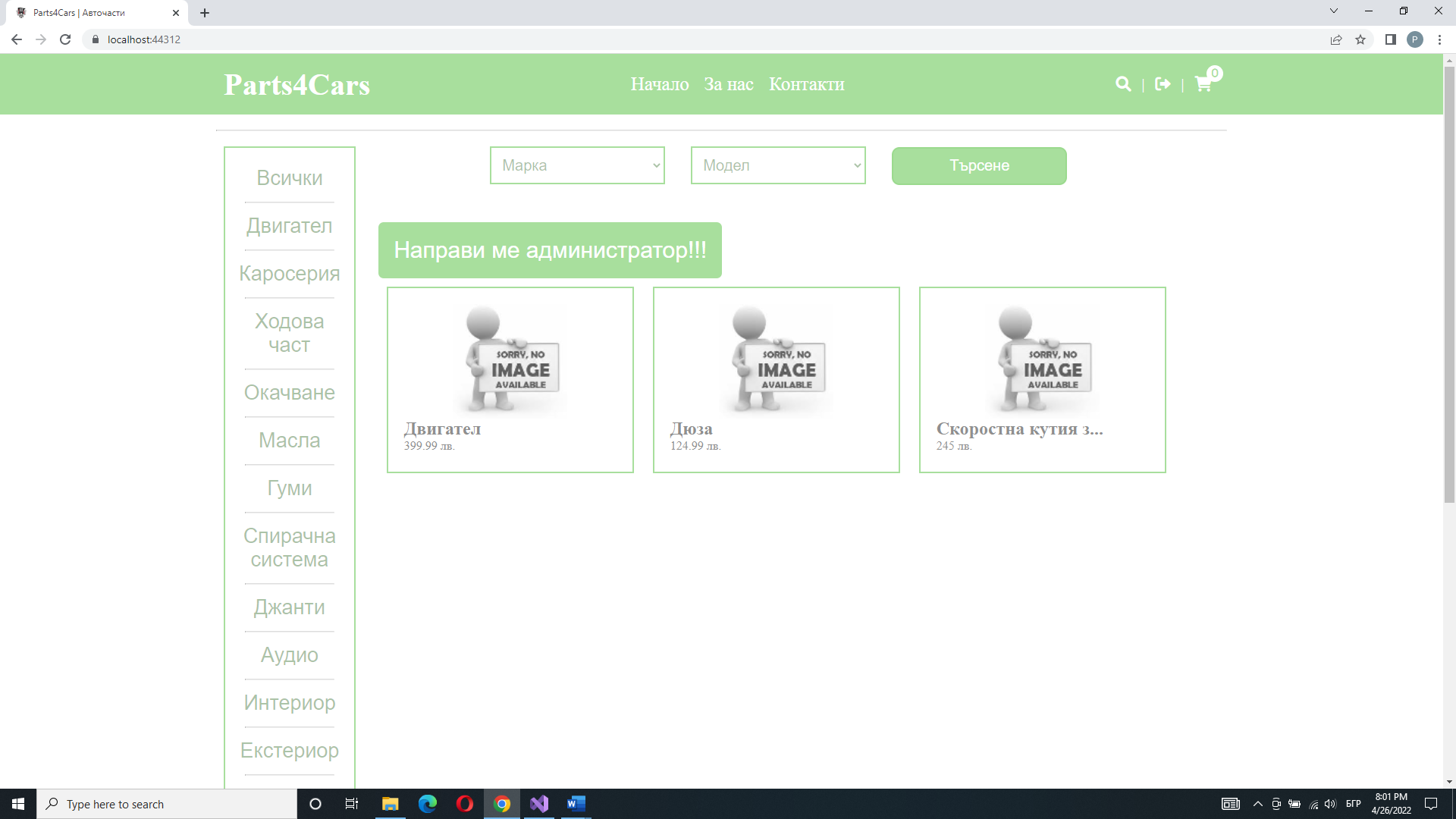
**Глава 5: Ръководство за потребителя**

**5.1 Как се става администратор>**

Администраторът може да бъде само един потребител. За да стане той такъв, трябва да са изпълнени следните две условия:

* Потребителят да притежава свой акаунт;
* Да няма записан потребител с роля администратор в базата данни.

Ако тези две условия са изпълнени, на началния екран се появява бутон с надпис: „Направи ме администратор!!!“.



Когато потребителят натисне този бутон, той автоматично се превръща в администратор и вече има достъп до всички страници и функции на сайта (**глава 3.1)**.

**5.2 Използване на уеб приложението**

При стартиране на уеб приложението, потребителят вижда началната страница, показана в **точка 4.2.** Чрез навигационните барове, той може да достъпи до всяка друга страница на приложението (**точка 4.2).** Това са:

* „Начало“;
* „За нас“;
* „Контакти“;
* детайли за дадена обява;
* „Вход“;
* „Регистрация“;
* „Количка“;
* страница за поръчка;
* „Добави обява“;
* страница за редактиране на дадена обява;
* страница за изтриване на дадена обява;
* „Поръчки“;
* и др.

**Глава 6: Заключение и бъдещо развитие**

Текущата дипломна работа има за цел да представи едно завършено уеб приложение, достъпно за всеки уеб потребител. Чрез него той може да разглежда и поръчва тези авточасти, които са му нужни.

Самият проект е така разработен, че да може лесно да се добавят нови функционалности в бъдеще. Частите му са така подредени, че да могат да бъдат достъпвани бързо.

В плана за развитие влизат следните идеи:

* Добавяне на снимки за всяка една обява;
* Добавяне на възможност за онлайн заплащане;
* Борса, за да могат потребителите да продават части помежду си, ако няма такава налична в сайта.

**Използвана литература:**

1. Dino Esposito - Programming ASP.NET Core
2. Jukie C. Meloni, Jennifer Kyrnin - HTML, CSS, and JavaScript All in One: Covering HTML5, CSS3, and ES6, Sams Teach Yourself
3. <https://stackoverflow.com/>
4. <https://www.youtube.com/>
5. <https://www.w3schools.com/>
6. https://docs.microsoft.com/
7. [https://softuni.bg/blog/](https://softuni.bg/blog/what-is-c-sharp-and-where-to-use-it)
8. [https://www.hwlibre.com/bg/](https://www.hwlibre.com/bg/orm-object-relational-mapping/)
9. https://bg.wikipedia.org/wiki/

**Приложения:**

**Главна HTML страница - \_Layout.cshtml:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Parts4Cars | Авточасти</title>

<**link** rel="shortcut icon" href="~/images/logo.png">

<**link** rel="stylesheet" type="text/css" href="~/css/site.css">

<**link** rel="stylesheet" type="text/css" href="~/lib/font-awesome/css/all.min.css">

<**script** src="~/js/site.js"></**script**>

</head>

<body>

<div class="header">

<**a** class="parts4cars-a" **asp-controller**="Home" **asp-action**="Index"><h1>Parts4Cars</h1></**a**>

<ul class="header-menu">

<li><**a** **asp-controller**="Home" **asp-action**="Index">Начало</**a**></li>

<li><**a** **asp-controller**="Home" **asp-action**="AboutUs">За нас</**a**></li>

<li><**a** **asp-controller**="Home" **asp-action**="Contacts">Контакти</**a**></li>

</ul>

<ul class="header-login-register">

<li><**partial** **name**="\_AdminMenuPartial" /></li>

<li><a href="#" class="search-button"><i class="fas fa-search"></i></a></li>

<li>|</li>

<li><**partial** **name**="\_LoginPartial" /></li>

<li>|</li>

<li><**partial** **name**="\_ShoppingCartPartial" /></li>

</ul>

</div>

<**form** class="search-bar">

<div class="search-input">

<input type="text" id="searchName" oninput="search\_name()" placeholder="Търсене"/>

</div>

</**form**>

</div>

<hr class="body-hr">

<div class="body">

@RenderBody()

</div>

<hr class="body-hr">

<div class="footer">

<div class="footer-logo-list">

<ul>

<li class="footer-logo-li1">&copy; Parts4Cars</li>

<li class="footer-logo-li2">&copy; Parts4Cars</li>

<li class="footer-logo-li3">&copy; Parts4Cars</li>

<li class="footer-logo-li4">&copy; Parts4Cars</li>

<li class="footer-logo-li5">&copy; Parts4Cars</li>

</ul>

</div>

<div class="footer-about-us">

<p>"Parts4Cars" е фирма, която от 15 години се занимава с продажба на авточасти втора употреба. Ние изкупуваме стари коли и предоставяме на нашите клиенти само най-добрите части.</p>

</div>

<div class="footer-worktime">

<ul>

<li class="footer-worktime-ul-li1">Работно време:</li>

<br>

<li>Понеделник - Петък</li>

<li>08:00 - 17:00</li>

<br>

<li>Събота:</li>

<li>09:00 - 13:00</li>

<br>

<li>Неделя:</li>

<li>Почивен ден</li>

</ul>

</div>

<div class="footer-menu">

<ul>

<p>Меню:</p>

<li><**a** **asp-controller**="Home" **asp-action**="Index">Начало</**a**></li>

<li><**a** **asp-controller**="Home" **asp-action**="AboutUs">За нас</**a**></li>

<li><**a** **asp-controller**="Home" **asp-action**="Contacts">Контакти</**a**></li>

</ul>

</div>

</div>

@await RenderSectionAsync("Scripts", required: false)

</body>

<script>

window.onload = partsInCart();

</script>

</html>

**Начална страница - Index.cshtml:**

@model ICollection<Part>

@{

string shortName = "";

}

<div class="body-left-menu">

<ul>

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="">Всички</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Двигател">Двигател</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Каросерия">Каросерия</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Ходова част">Ходова част</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Окачване">Окачване</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Масла">Масла</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Гуми">Гуми</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Спирачна система">Спирачна система</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Джанти">Джанти</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Аудио">Аудио</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Интериор">Интериор</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Екстериор">Екстериор</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Казанчета">Казанчета</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Светлини">Светлини</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Кори и пластмаси">Кори и пластмаси</button>

</li>

<hr class="body-left-menu-hr">

<li>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Други">Други</button>

</li>

</ul>

</div>

<div class="body-main">

<div class="category-dropdown">

<button onclick="CategoryDropdown()" class="category-dropbtn"><i class="fa-solid fa-bars"></i></button>

<div id="category-dropdown-content" class="category-dropdown-content">

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="">Всички</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Двигател">Двигател</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Каросерия">Каросерия</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Ходова част">Ходова част</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Окачване">Окачване</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Масла">Масла</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Гуми">Гуми</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Спирачна система">Спирачна система</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Джанти">Джанти</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Аудио">Аудио</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Интериор">Интериор</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Екстериор">Екстериор</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Казанчета">Казанчета</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Светлини">Светлини</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Кори и пластмаси">Кори и пластмаси</button>

<button class="body-left-menu-option" type="submit" onclick="search\_Category(this.value)" value="Други">Други</button>

</div>

</div>

<**form** class="body-main-search" **asp-controller**="Home" **asp-action**="Index" method="get">

<select id="searchCarMake" onchange="search\_makeAndModel()" name="SearchCarMake" class="car-brand">

<**option** **value**="">Марка</**option**>

<**option** **value**="Audi">Audi</**option**>

<**option** **value**="BMW">BMW</**option**>

<**option** **value**="Mercedes-Benz">Mercedes-Benz</**option**>

<**option** **value**="Honda">Honda</**option**>

</select>

<select id="searchCarModel" name="SearchCarModel" class="car-model">

<**option** **value**="">Модел</**option**>

</select>

<input type="submit" class="body-main-search-submit" value="Търсене">

</**form**>

<**partial** **name**="\_MakeMeAdminPartial" />

<div class="body-main-products">

@foreach (var part in Model)

{

<**a** **asp-controller**="Home" **asp-action**="Details" **asp-route-id**="@part.ID">

<div class="body-main-products-product">

<div class="body-main-products-product-img">

@if(part.Images == null)

{

<**img** src="~/images/noimage.jpg" alt="Picture">

}

else

{

<img src="@part.Images.ToList()[0].Url" alt="Picture">

}

</div>@if(part.Name.Length > 17)

{

char[] shortNameArray = new char[17];

for(int i=0; i < 17; i++)

{

shortNameArray[i] = part.Name[i];

}

shortName = new string(shortNameArray);

shortName += "...";

}

else

{

shortName = part.Name;

}

<h2 class="part-name">@shortName</h2>

<span>@part.Price лв.</span>

<p class="part-category">@part.Category</p>

</div>

</**a**>

}

</div>

</div>

**Малко CSS код:**

\* {

margin: 0;

padding: 0;

}

html, body {

width: 100%;

}

.header {

height: 80px;

background-color: rgba(153, 217, 140, 0.856);

align-items: center;

display: flex;

justify-content: space-between;

padding-left: 15.5%;

padding-right: 15.5%;

}

.header h1 {

color: rgb(255, 255, 255);

font-size: 40px;

}

.parts4cars-a{

text-decoration: none;

}

.header-menu {

display: flex;

}

.header-menu li {

font-size: 25px;

position: relative;

display: flex;

list-style-type: none;

color: rgb(255, 255, 255);

margin: auto;

}

.header-menu li a:hover {

color: rgba(48, 107, 36, 0.404);

transition: all ease 0.3s;

}

.header-menu li a {

text-decoration: none;

color: rgb(255, 255, 255);

padding: 10px;

}