# Дипломна работа

на тема

**Уеб приложение за покупко-продажба на автомобили**

Дипломант: **Александър Александров Свиларов**

ученик от 12А клас в

ЧПГДН "СофтУни Светлина"

Научен ръководител: **Калоян Костадинов**

Рецензент: **XXX XXX XXX**

Дата: **12 май 2023**

Сесия: май-юни 2023 г.

София, 2023 г.

*Указания:*

* *Вашата дипломна работа трябва да има обем от* ***минимум 30 страници****.*
* *При попълването на документа да се използва стил „****Normal****“, в който е настроен шрифта и размера, така, че да бъдат изпълнени критериите за форматиране.*
* *След попълването на основната информация, всички указания (текстовете в сиво) да се премахнат.*
* *За формулирането на отделните точки в съдържанието използвайте стиловете „****Heading 1****“ (за основни дялове), „****Heading 2****“ (за отделните точки в дяловете), „****Heading 3****“ (за отделните подточки във всяка точка на дяловете). За описателния текст в точките използвайте стил „****Normal****“. За кода да се използва стил „****Code****“.*
* *Всяка* ***фигура****,* ***картинка*** *и* ***таблица*** *да бъдат означени и номерирани. Използвайте пестеливо разполагане на текста спрямо фигурите/картинките, така, че при разпечатването да не се хаби излишен брой листи.*
* *Спазвайте изискванията за ползване и* ***цитиране*** *на чужди разработки (авторско право).*
* *След като приключите с работата по всички точки и заличите ненужните текстове, направете обновяване на* ***Съдържанието****, така, че всички нововъведени точки и подточки да придобият местоположение спрямо страниците*
* *В меню File 🡪 Info въведете* ***автора на документа****, така, че при проверка от страна на рецензента аворството да бъде потвърдено недвусмислено.*
* *Запишете* ***файла с подходящо име****: кратко заглавие + име на дипломанта + класа, примерно:* ***Polivna-sistema-Stefan-Kirov-12B-klas.docx****.*

# Съдържание

[Дипломна работа 1](#_30j0zll)

[Съдържание 3](#_1fob9te)

[Увод 4](#_3znysh7)

[Проблем 4](#_2et92p0)

[Цели на дипломния проект 4](#_tyjcwt)

[Задачи, произтичащи от целите 4](#_3dy6vkm)

[Глава 1. Проучване (смени заглавието) 6](#_1t3h5sf)

[Глава 2. Задание (смени заглавието) 7](#_4d34og8)

[Глава 3. Проектиране и имплементация (смени заглавието) 8](#_2s8eyo1)

[Архитектура на системата 8](#_17dp8vu)

[Имплементация на системата 8](#_3rdcrjn)

[Тестове 9](#_26in1rg)

[Внедряване (deployment) 10](#_lnxbz9)

[Глава 4. Ръководство за потребителя (смени заглавието) 11](#_35nkun2)

[Заключение 12](#_44sinio)

[Информационни източници 13](#_2jxsxqh)

[Приложение 1 / 2 / … 14](#_z337ya)

[Рецензия на дипломен проект 15](#_3j2qqm3)

# 

# Увод

DeltaCar е уеб приложение, което има за цел да улесни процеса на покупко-продажба на автомобили. То предоставя на потребителите възможност да намерят подходящия автомобил за тях, да сравняват различни марки и модели и да закупят автомобила онлайн.

Целта на проекта е да създаде едно лесно за използване приложение, което да предоставя на потребителите добра и удобна услуга. То ще бъде достъпно за всички, които желаят да купят или продадат автомобил, и ще съдържа полезна информация, която ще ги насочи към правилния избор.

Приложението ще предлага разнообразни филтри и опции за търсене, които да помогнат на потребителите да намерят автомобила, който отговаря на техните нужди и предпочитания. След като потребителите изберат автомобила, който искат да закупят, те ще могат да се свържат директно с продавача, за да се договорят за цената и да уточнят всички детайли по сделката. Например, потребителите могат да търсят автомобили по марка, модел, цена, година на производство, тип на двигателя, тип на горивната система, пробег и други параметри. Тези филтри помагат на потребителите да се фокусират върху тези автомобили, които отговарят на техните нужди. Системата за търсене на DeltaCar се отличава също и с възможността да се търсят автомобили по ключови думи, което е особено полезно за тези потребители, които търсят автомобил с даден проблем.

DeltaCar ще предлага и много други полезни функции за потребителите. Те ще могат да преглеждат описания на автомобилите, да разглеждат снимки и детайлни характеристики. Приложението ще предлага и интегрирани услуги за проверка на автомобили, които да помогнат на потребителите да се уверят, че сделката, която сключват, е сигурна и безопасна.

Комбинацията от множеството параметри за търсене и гъвкавостта на системата правят DeltaCar по-добро от другите сайтове за покупко-продажба на автомобили. Това приложение предоставя на потребителите иновативен начин за намиране на идеалния автомобил за тях.

## Проблем

## DeltaCar решава проблема на хората, които търсят удобен и бърз начин за покупка на автомобил. Сайтът ще предоставя богат избор от автомобили от различни марки и модели, с различни характеристики и цени.

## Проблемът, който сайтът решава, е, че традиционните начини за покупка на автомобили - от дилър или частен продавач, могат да бъдат трудни и времеемки за хората. При това, те може да не могат да намерят точно това, което търсят. DeltaCar предлага удобство и лесно търсене на автомобили в едно място, с информация за модела, състоянието и цената на всяко едно предложение. Освен това, сайтът осигурява безопасност и сигурност на покупката, като се грижи за всички необходими документи и проверки на автомобилите.

## Този проблем е актуален за милиони хора, които търсят нов или употребяван автомобил в света. Сайтът DeltaCar предоставя решение на този проблем за хора, които търсят лесен и бърз начин да закупят автомобил, както и за тези, които искат да продадат своя автомобил. Освен това, сайтът е полезен за хора, които търсят конкретен модел или марка автомобил, който може да бъде трудно да се открие в традиционните магазини или от частни продавачи.

## Цели на дипломния проект

Целите на дипломния проект DeltaCar са да разработи уеб базирана платформа за продажба на автомобили, която да предоставя на потребителите бърз и лесен начин за търсене и закупуване на автомобили, както и за продажба на автомобили.

Специфичните цели на проекта включват следното:

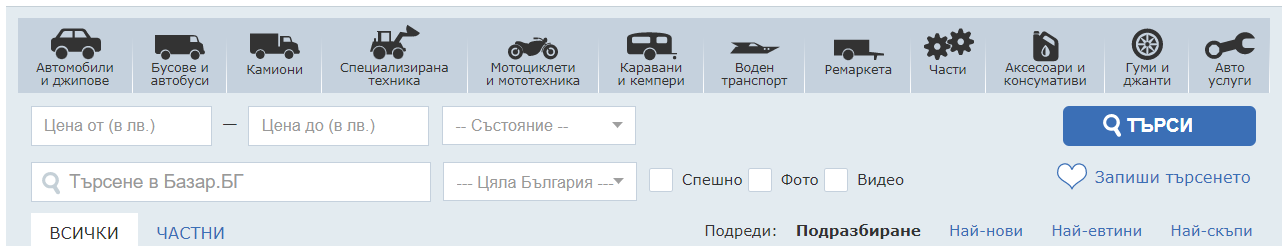
* Разработване на потребителски интерфейс, който да е удобен, лесен за използване и да предоставя ясна информация за всеки автомобил, който е наличен за продажба.
* Интегриране на търсачка за автомобили, която да позволява на потребителите да търсят автомобили по марка, модел, година на производство, цена и други характеристики.
* Интегриране на система за онлайн плащания, която да позволява на потребителите да закупуват автомобили директно през уебсайта. ***(Планирано)***
* Разработване на система за управление на продажбите, която да позволява на администраторите на сайта да управляват автомобилите, които са налични за продажба, както и да проследяват статуса на всяка продажба. ***(Планирано)***
* Интегриране на система за комуникация между потребителите и администраторите на сайта, която да позволява на потребителите да получават бърз и ефективен отговор на въпроси и да получават помощ при нужда. ***(Планирано)***

# Проучване

Закупуването или продажбата на автомобил може да бъде една времеемка и не винаги лесна задача. За да има успех продавачът, той трябва да притежава автомобил в добро състояние, привлекателен външен вид, добро описание, честно отношение към клиентите и тн. Преди съществуването на интернет пространството подобни сделки са се извършвали най-често в автокъщи – човек отива там, говори с човек, отговорен за продажбите, обяснява какво търси, изискванията си и предпочитанията си, след което бива насочен към няколко автомобила, с които автокъщата разполага и той избира този, който му допада най-много.

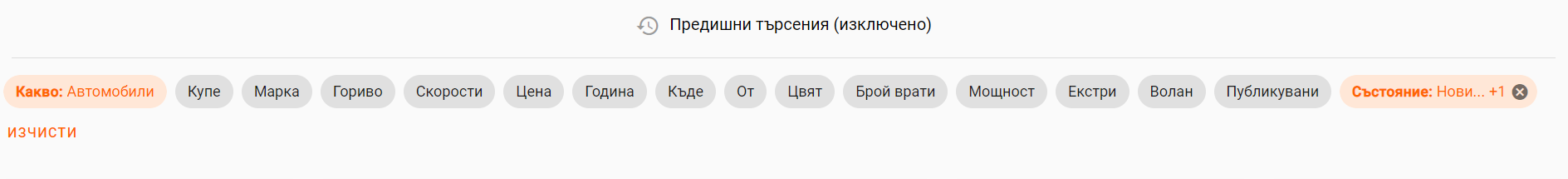
Един от основните проблеми на този начин за покупка на автомобил е времето, което отнема, без гарантиран резутат. Винаги може да се случи така, че автокъщата да не разполага с автомобил, който да покрива изискванията на даден клиент – тогава целият този процес започва отначало. След 2-3 различни срещи денят е загубен, а купувачът все още няма желаният автомобил. С развитието на интернет, подобни проблеми бързо намират своите решения – сферата на автомобилите не е изключение. В днешно време най-популярният начин за покупка на автомобил е онлайн.

В България имаме няколко основни сайта за тази цел – Mobile.bg, Cars.bg, Auto.bg, както и Bazar.bg и Olx.bg. Всеки един от тези сайтове има своите плюсове и минуси. В сайтове като Bazar.bg търсачката е силно ограничена по ключови думи, локация, състояние и цена:



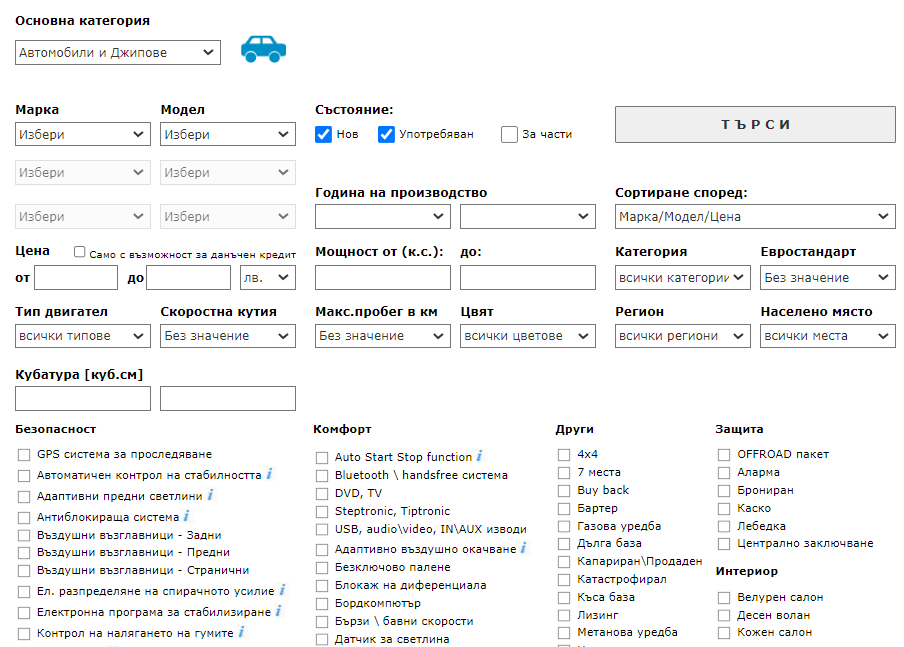
Фигура 1: Търсачката в Bazar.bg

В Auto.bg наблюдаваме подобна ситуация – ограничена търсачка, този път с малко повече опции:



Фигура 2: Търсачката в Auto.bg

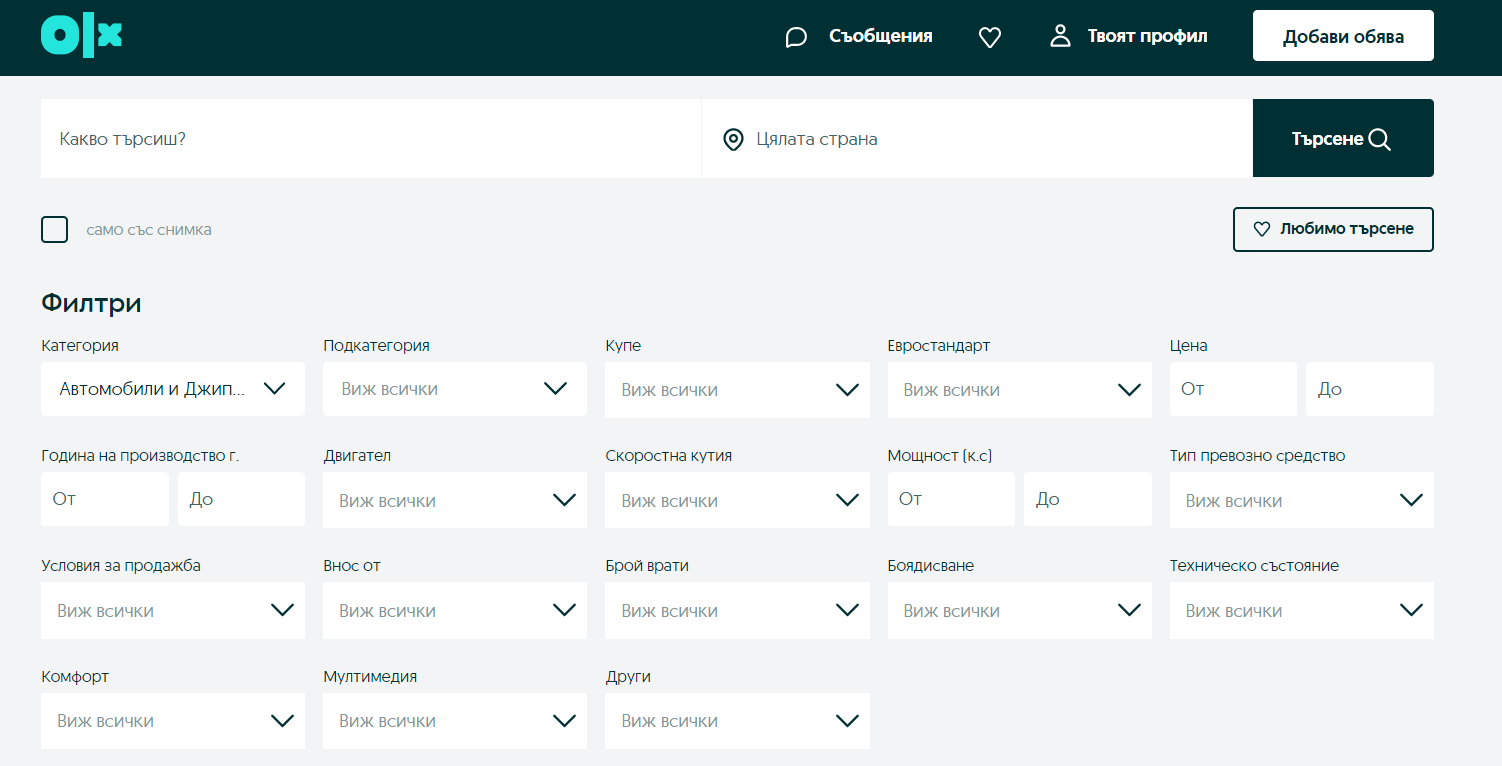
В сайтове като Mobile.bg ще забележим, че търсачката е сравнително по-обогатена, имайки редица опции, от които да изберем:



Фигура 3: Търсачката в Mobile.bg

Mobile.bg разполага с много добре разработена търсачка, която включва почти всичко, от което се нуждае един клиент. Единственото, което липсва, е търсенето по ключови думи – една важна подробност, която често бива забравена. Има хора, които търсят коли със специфичен проблем – например има механици, които търсят кола за „гарнитура на глава“, която да оправят и продадат за повече пари. DeltaCar се отличава от другите сайтове за покупко-продажба на автомобили благодарение на своята по-добре разработена система за търсене, която обединява различните видове търсачки, показани по-горе. Потребителите имат достъп до множество параметри, които могат да използват, за да намерят точно този автомобил, който отговаря на техните нужди и предпочитания. Функцията за за търсене по ключови думи е много важна част от системата за търсене, с която разполага DeltaCar.

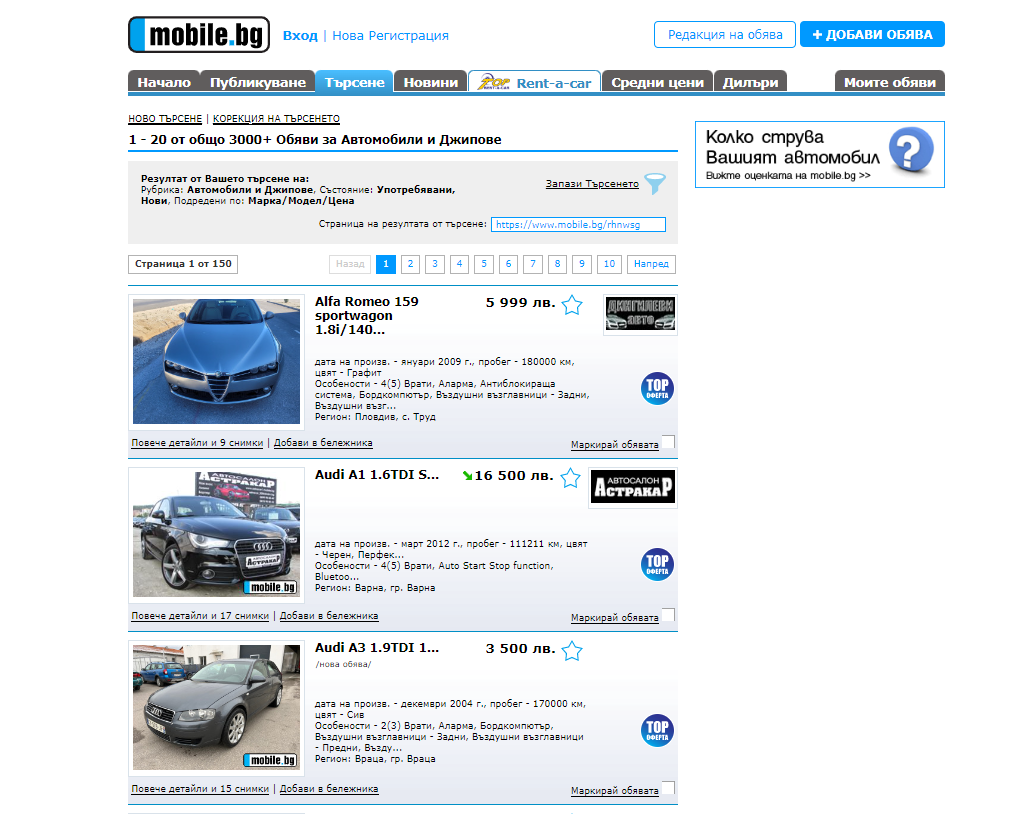
От всички подобни сайтове изненадващо най-добра търсачка има Olx.bg, като единственият проблем там, е че сайтът не е стриктно предназначен за покупко-продажба на автомобили.



Фигура 4: Tърсачката в Olx.bg

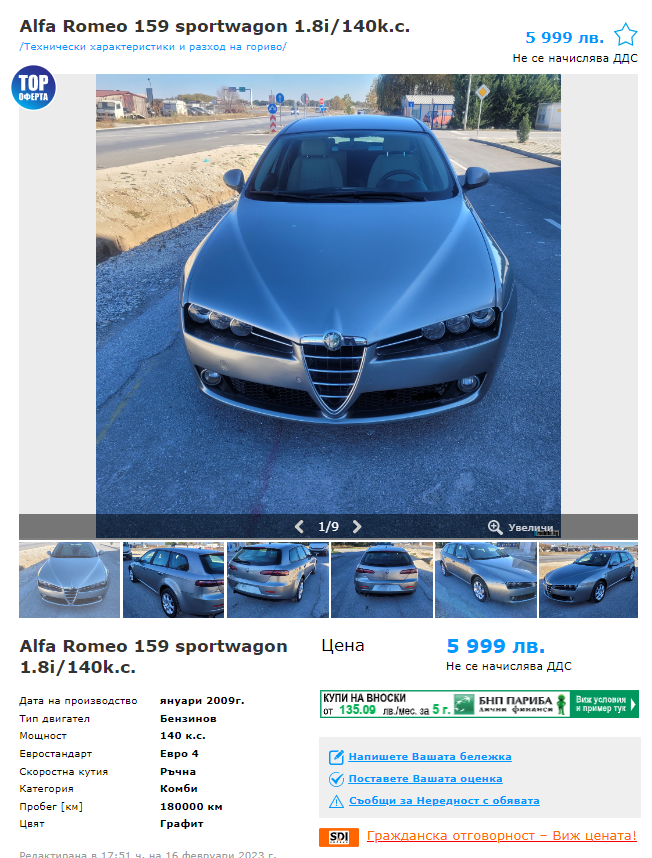
Нека разгледаме и дизайна на различните уебсайтове, които предлагат услуга за покупко-продажба на автомобили.

В Mobile.bg дизайнът е семпъл, но вече е леко остарял. Със сигурност не е в крак с тенденциите в уеб дизайна, но пък е лесен за разбиране и потребителите не се затрудняват да отгатнат как да навигират из сайта.

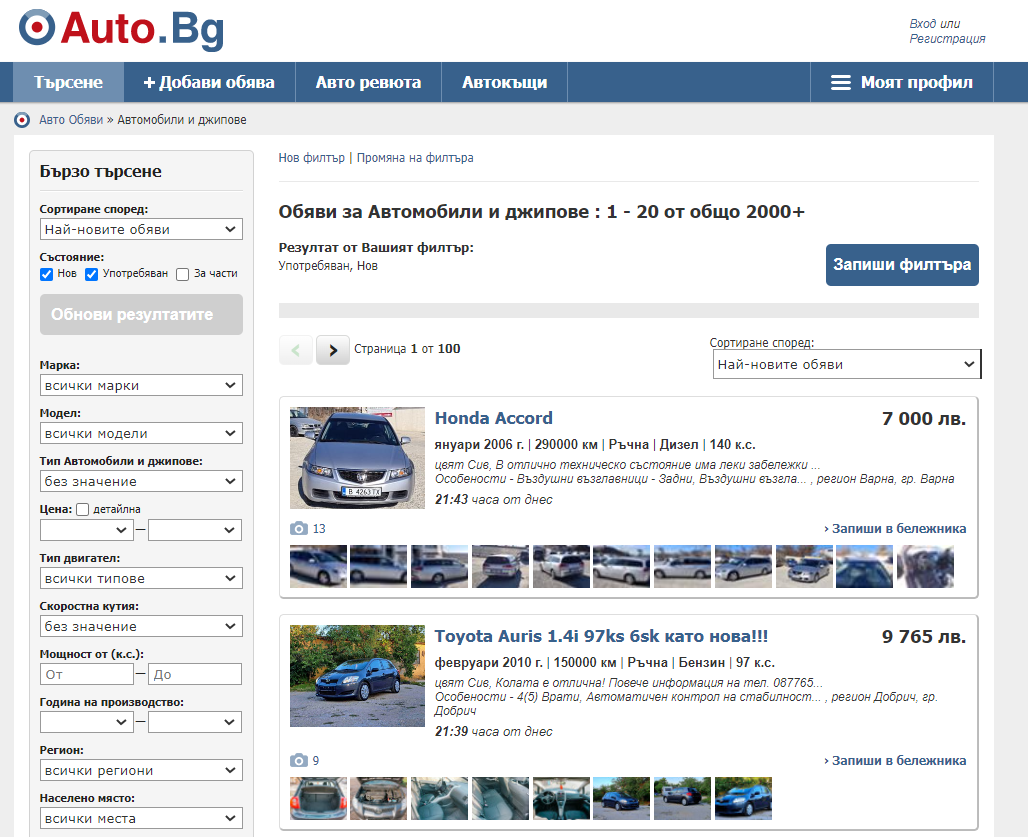


Фигура 5: Дизайн в Mobile.bg

Дизайнът в самата страница, където се намира обявата, е добре структуриран, като е предтавена най-важната за потребителя информация:

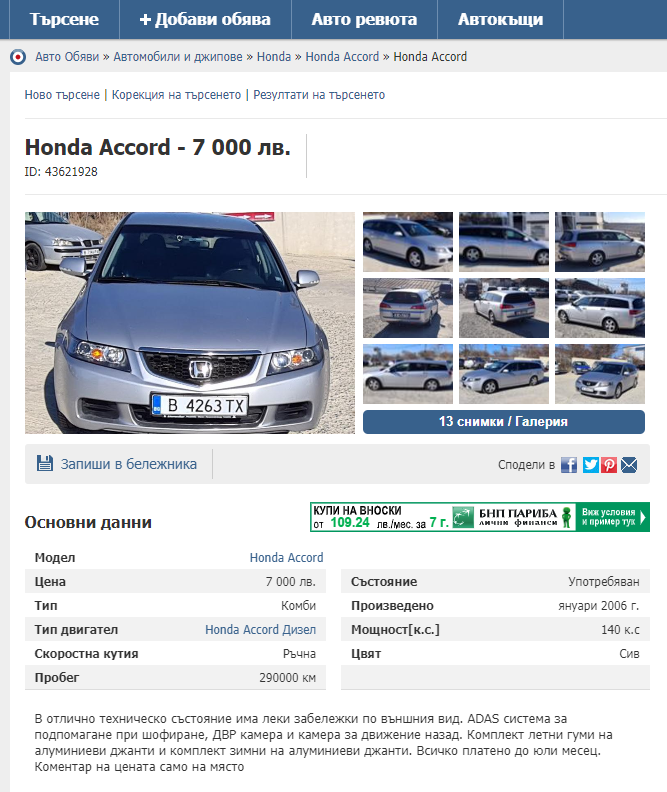


Фигура 6: Дизайн на обява в Mobile.bg

Дизайнът в Аuto.bg е подобен, но тук виждаме подобрение в Navigation Bar-a, както и опцията за Бързо Търсене, която е полезна.

Фигура 7: Дизайн в Auto.bg

Подобен дизайн наблюдаваме и в страницата на обявата:



Фигура 8: Дизайн на обява в Auto.bg

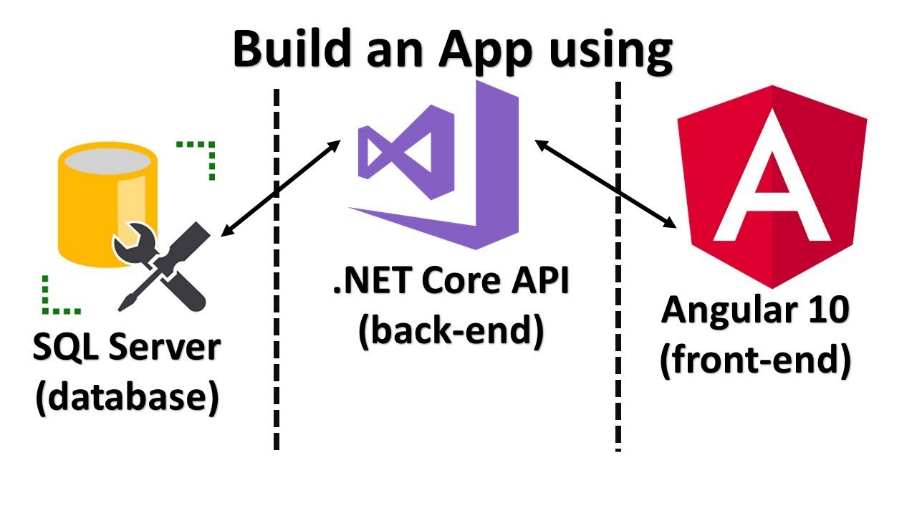
# Глава 2. Използвани технологии

За да създадем уеб приложението, ще използваме различни технологии за backend-a и frontend-a. За backend-a ще използваме C# и Entity Framework, а за frontend-a – Angular (с Typescript).

# 2.1 Backend

Както казах вече, backend-a ще бъде написан на C# и Entity Framework. Този framework ни позволява да използваме т.нар ORM (object-relational mapping), който ще ни помогне да свържем лесно базата данни с обектно ориентирания език C#.

Преди да обясним подробно какво е ORM, нека ви покажа как ще бъде построено уеб приложението:



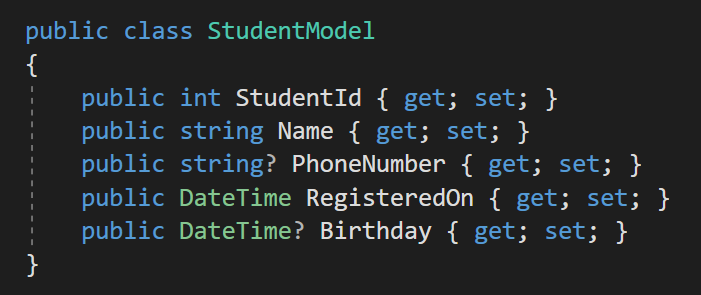
Фигура 9: Структурата на уеб приложението (технологии)

# 2.1.1 Какво е ORM и как функционира в Entity Framework?

ORM е начин за конвертиране на данни между релационна база данни и обектно-ориентиран език за програмиране, създавайки база данни с виртуални обекти, която може да се използва от обетно-ориентирания език за програмиране.

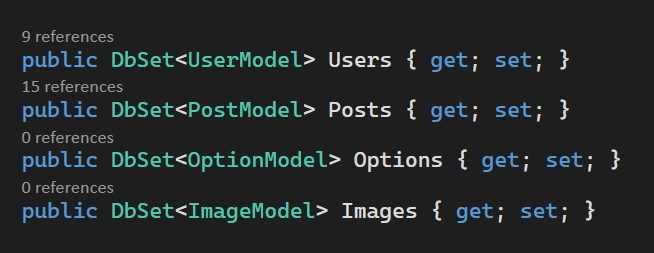
В Entity Framework това се случва чрез помощта на модели и миграции - процес на автоматично създаване и актуализиране на релационната база данни на основата на промените в моделите. Това означава, че EF Core може да проследява промените в модела на данните и да създава, актуализира или премахва таблиците и колоните в базата данни, когато е необходимо.

Един модел в Entity Framework представлява клас, който има различни property-та. Те могат да бъдат различни типове, като Entity Framework сам ще ги превърне в типове, които SQL Server да разбере – например тип ***string*** ще бъде превърнат в тип ***nvarchar***, тип ***DateTime*** ще бъде превърнат в ***datetime2*** и така нататък:



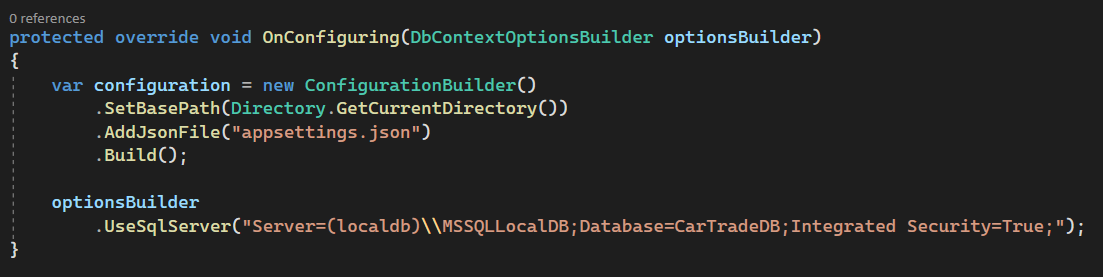
Фигура 10: Модел в Entity Framework

Тези модели можем да превърнем в таблици, използвайки т.нар ***DbSet***, където задаваме и желаното име на таблицата:



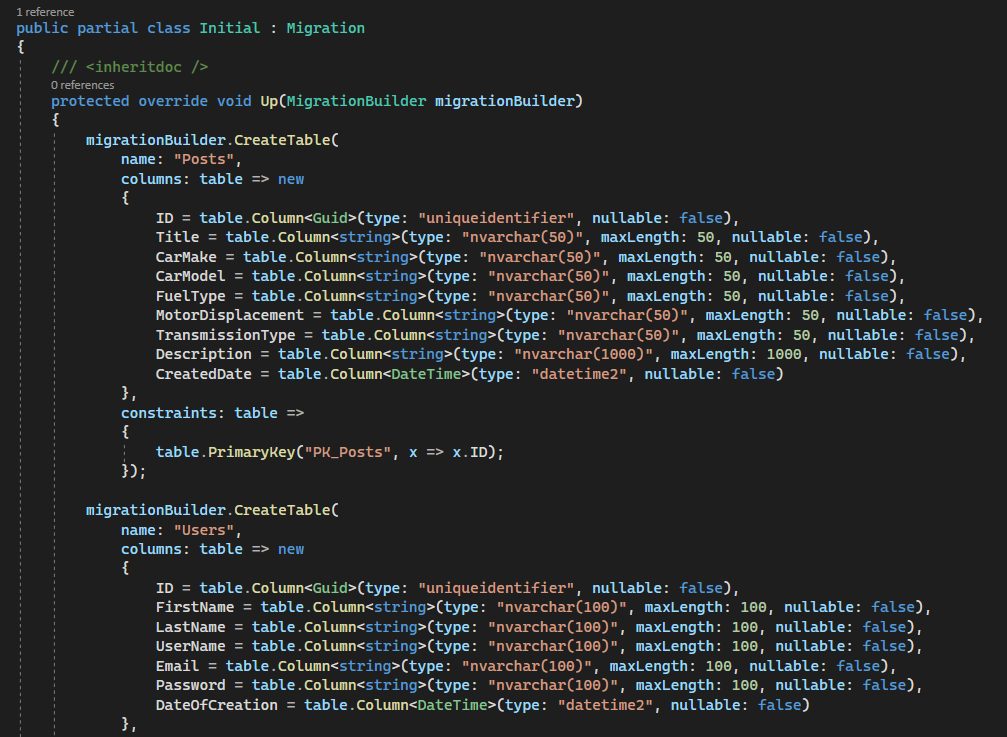
Фигура 11: Използване на DbSet в Entity Framework

Разбира се, трябва да имаме и конфигурация, коятно да посочи на Entity Framework как да намери / къде да създаде базата данни, която искаме да създаваме, използваме или модифицираме. Това можем да постигнем чрез метода **OnConfiguring**:



Фигура 12: Метод OnConfiguring в Entity Framework

След като имаме модел и сме решили, че той ще бъде таблица, задавайки го като DbSet, трябва да направим миграция. Миграцията в Entity Framework се извършва с помощта на инструмента за миграции на Entity Framework, който позволява на разработчиците да генерират т.нар миграционни скриптове и да ги приложат върху базата данни. Този процес включва изграждане на миграционна история, която записва всички промени в модела на данните, като например добавяне или премахване на таблица или колона. Когато се генерира нова миграция, инструмента за миграции автоматично генерира необходимите SQL команди за актуализиране на базата данни.



Фигура 13: Миграционен скрипт в Entity Framewok

# Глава 3. Проектиране и имплементация (смени заглавието)

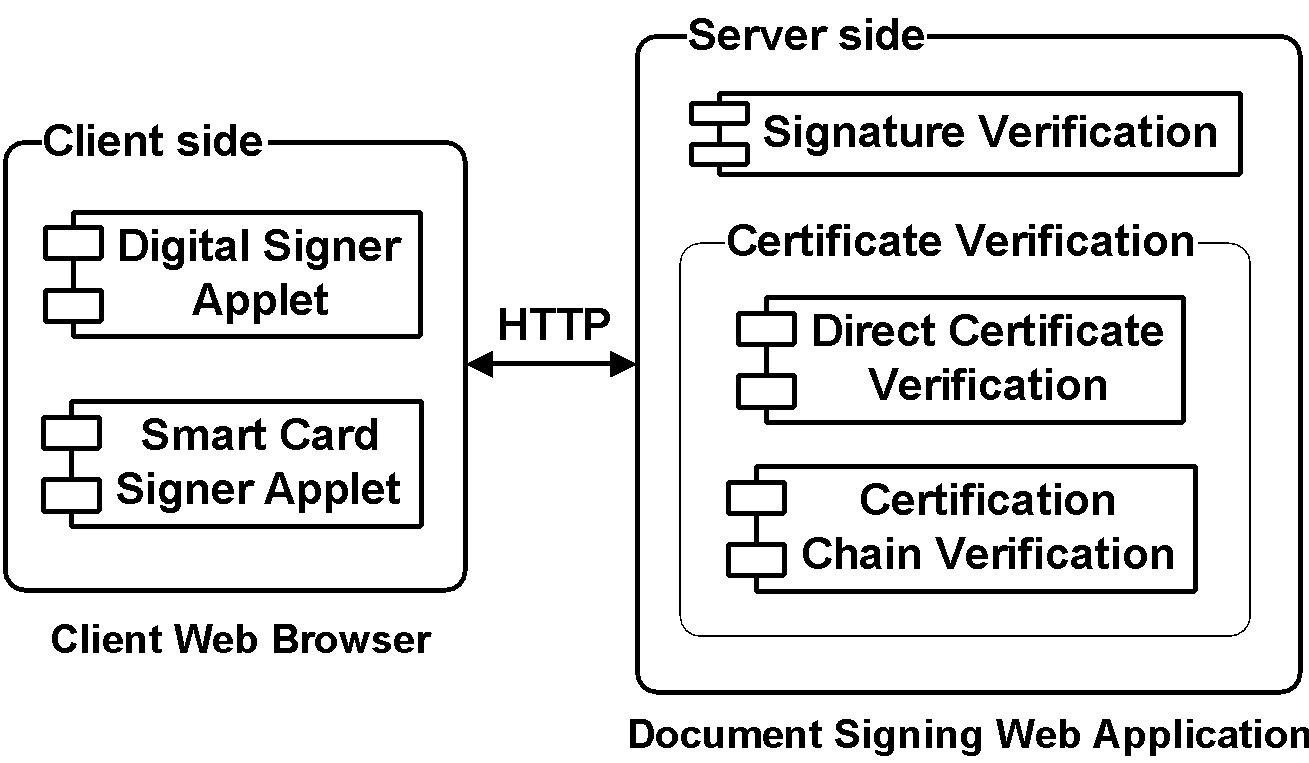
***Същинската част*** *на дипломната с най-голям обем. Тя включва описание на начина на* ***реализация*** *на софтуерния проект, включващ софтуерна* ***архитектура*** *(компоненти на системата, слоеве, предназначение и начин на комуникация между тях + диаграма), диаграма на* ***базата данни*** *(ако се използва база данни),* ***клас диаграма*** *(ако е удачно да се направи), използвани* ***алгоритми*** *(ако има нещо по-специфично), ключови фрагменти от* ***сорс кода*** *(ако има нещо по-специфично) и други* ***ключови моменти от реализацията на проекта*** *(максимум* ***12-15 страници****).*

## Архитектура на системата

*Тази подчаст описва* ***архитектурата и дизайна*** *на софтуерния продукт / система. Това може да включва диаграми, схема на базата данни и др.*

*Пример:*

*Системата e базирана на класическата клиент-сървър архитектура, реализирана чрез уеб браузър и уеб приложение (фигура 3-1):*



*Фигура 3-1. Архитектура на системата за подписване на документи в уеб среда*

## Имплементация на системата

*Описва се* ***имплементацията на системата*** *– основни компоненти (примерно сървърна част, mobile app, Web app), по-важни класове, слоеве (примерно data layer, UI layer), използвани външни библиотеки и фреймуърци.*

*Пример:*

*Подписването на документ в уеб браузъра с цифров сертификат от PFX файл се реализира чрез Java аплет “****DigitalSignerApplet****”. Той се състои основно от класа DigitalSignerApplet, който реализира основната му функционалност*

|  |
| --- |
| **DigitalSignerApplet.java** |
| **import java.applet.Applet;**  **import java.awt.\*;**  **…**  **public class DigitalSignerApplet extends Applet {**  **…**  **}** |

*Този клас имплементира следните стъпки: …*

*…*

*Подписването на документ в уеб браузъра със смарт карта се реализира чрез Java аплет “****SmartCardSignerApplet****”. Той работи по следния начин …*

*…*

*Обработката на подписания документ от страна на сървъра и верификацията на положения цифров подпис се извършва от специализирано уеб приложение. То е изградено по J2EE стандартите за уеб приложения с MVC framework “Struts” и работи по следния начин: …*

## Тестове

*Включва описание на* ***проведените тестове*** *(****ръчни****,* ***автоматизирани****), тяхната имплементация и резултати. Описва се и построената* ***CI система*** *(ако има такава).*

*Пример:*

*От страна на клиента системата успешно тествахме с:*

* *Java Plug-In 1.4 (за аплета с PKCS#12 хранилище) и Java Plug-In 1.5 (за аплета със смарт карта)*
* *уеб браузъри Internet Explorer 5.0, Internet Explorer 6.0, Mozilla 1.3, Mozilla 1.7, Firefox 1.0, Netscape Communicator 4.5 и Netscape 6.1 върху Windows 98, Windows 2000, Windows XP и Windows 2003 Server*
* *Mozilla 1.3 върху Red Hat Linux 8.0 (с графична среда KDE 3.0)*
* *смарт карта Utimaco Safeware с PKCS#11 драйвери Utimaco SafeGuard Smartcard Provider и карточетец ACR38*
* *цифрови сертификати от VeriSign, Thawte, GlobalSign и StampIT*

*Върху всички посочени платформи и уеб браузъри системата работи коректно и според очакванията...*

*Системата беше разгледана и тествана и от специалисти от Асоциация за информационна сигурност (ISECA), както и от експерти, работещи в областта на удостоверителните услуги от „Информационно обслужване” АД (StampIT). Изказаното мнение…*

## Внедряване (deployment)

*Представя се* ***начинът на достъпване на системата****. Тя може да бъде качена в Интернет, да се хоства в облачна среда като Azure или да се изтегля като инсталатор от публичен уеб сайт. Предоставя се* ***линк за достъп*** *и/или* ***инструкции*** *за достъп до приложението.*

*Пример:*

*За да изпълним приложението DocumentSigningDemoWebApp.war е необходимо да го инсталираме (deployment) на някой J2EE сървър или сървър за Java уеб приложения (Servlet container).*

*Ако използваме сървъра за J2EE уеб приложения Apache Tomcat, е достатъчно до копираме файла DocumentSigningDemoWebApp.war в директория %TOMCAT\_HOME%/webapps7 и да стартираме Tomcat. След това (при стандартна инсталация и конфигурация на Tomcat) приложението е достъпно от адрес: http://localhost:8080/DocumentSigningDemoWebApp/ (фигура 3-8):*

Картина, която съдържа текст

Описанието е генерирано автоматично

*Фигура 3-8. Уеб приложение за подписване на документи в уеб среда и проверка на подписи*

# Глава 4. Ръководство за потребителя (смени заглавието)

*Съдържа* ***ръководство на потребителя: как се инсталира, конфигурира и изпълнява системата****. Трябва да се опише програмния продукт от изискванията към хардуера, през инсталацията, до начално конфигуриране и описание на ключовите екрани и как се работи с тях (когато това не е очевидно), заедно с подходящи screenshots (максимум* ***4-6 страници****).*

*Пример:*

*Системата е достъпна като сорс код от следното GitHub хранилище:* [*https://github.com/nakov/NakovDocumentSigner*](https://github.com/nakov/NakovDocumentSigner)

*…*

*За да работи правилно аплетът, е необходимо той да бъде подписан. Можем да използваме следния скрипт за да си генерираме саморъчно-подписан сертификат, с който да подпишем след това аплета:*

|  |
| --- |
| **generate-certificate.bat**  **del SmartCardSignerApplet.jks**  **keytool -genkey -alias signFiles -keystore SmartCardSignerApplet.jks -keypass !secret -dname "CN=Your Company" -storepass !secret**  **pause** |

*…*

# Заключение

*Съдържа* ***обобщение на постиженията*** *в дипломния проект и дипломната работа и* ***възможностите за усъвършенстването*** *и обогатяването им (максимум* ***1-2 страници****).*

*Пример:*

*Системата* [*NakovDocumentSigner*](http://www.nakov.com/documents-signing/) *е работещ пример, който илюстрира един платформено независим подход за използване на цифрови подписи от Java-базирани уеб приложения. Тя решава проблемите, които …*

*Ще разгледаме най-важните проблеми и недостатъци на системата* [*NakovDocumentSigner*](http://www.nakov.com/documents-signing/)*: (изброяваме ги)…*

# Информационни източници

*Включва* ***цитираната*** *и* ***използвана*** *в записката на дипломния проект* ***литература****. Започва на отделна страница от основния текст.*

*Примери:*

*1. Николов А., Програмиране на С++, Техника, София, 1998.*

*2. John A., Main Principles of C++ Programming, International Journal of Programming, Vol. 35, No 5, May 2001, pp. 112-183.*

# Приложение 1 / 2 / …

*Съдържат* ***документация****, която не е намерила място в текста поради ограниченията в обема й или за по-добра прегледност подредба. В текста трябва да има* ***препратка*** *към всички приложения.*

*Пример:*

*В* ***" Приложение 1: Тестване, оценка и усъвършенстване"*** *е направена критична оценка на реализираната система…*

*…*

***Приложение 1: Тестване, оценка и усъвършенстване***

*Една от основните идеи, залегнали при проектирането и имплементирането на системата за цифрово подписване на документи в уеб среда, е тя да бъде платформено независима и да…*

***Експериментално тестване на системата***

*По време на тестовете на системата NakovDocumentSigner използвахме следния софтуер oт страна на сървъра:*

* *Windows XP SP2*
* *Java Runtime Environment версия 1.4.2*

# Рецензия на дипломен проект

|  |  |  |  |
| --- | --- | --- | --- |
| **Тема на дипломния проект** |  | | |
| **Ученик** |  | | |
| **Клас** |  | | |
| **Професия** |  | | |
| **Специалност** |  | | |
| **Ръководител- консултант** |  | | |
| **Рецензент** |  | | |
| **Критерии за допускане до защита на дипломен проект** | | **Да** | **Не** |
| Съответствие на съдържанието и точките от заданието | |  |  |
| Съответствие между тема и съдържание | |  |  |
| Спазване на препоръчителния обем на дипломния проект | |  |  |
| Спазване на изискванията за оформление на дипломния проект | |  |  |
| Готовност за защита на дипломния проект | |  |  |
| Силни страни на дипломния проект |  | | |
| Допуснати основни слабости |  | | |
| Въпроси и препоръки към дипломния проект |  | | |

**ЗАКЛЮЧЕНИЕ:**

Качествата на дипломния проект дават основание ученикът/ ученичката ................................................... да бъде допуснат/а до защита пред членовете на комисията за подготовка, провеждане и оценяване на изпит чрез защита на дипломен проект- част по теория на професията.

.........05.2023г. Рецензент:................................

град София