**П Р И Р О Д О- М А Т Е М А Т И Ч Е С К А Г И М Н А З И Я „Я Н Е С А Н Д А Н С К И“**



гр. Гоце Делчев, ул."Скопие" №4,

тел: директор: (0751) **60306**, секретар: **60308**, факс: **60308**,

e-mail**:** [**pmg.direktor@**](mailto:pmg.direktor@)**pmggd.bg**, web site: **https://pmggd.bg/**

**ДИПЛОМЕН ПРОЕКТ**

**Тема:** Уеб приложение за сервиз за леки автомобили „Ауто.бг“

На Костадин Иванов Картулев ученик/ученичка от XIIв клас

**професия код** 481020 „Системен програмист“

**специалност код** 4810201 „Системно програмиране“

**Ръководител/консултант:** Невсие И. Караахмед

Сесия: май-юни 2024г.

**Гоце Делчев**

**2024г.**

**Съдържание**

[1. Увод 3](#_Toc163546885)

[1.1. Цел на приложението 3](#_Toc163546886)

[1.2. Резултат от създаването на приложението 3](#_Toc163546887)

[2. Изложение 4](#_Toc163546888)

[2.1 Използвани технологии 4](#_Toc163546889)

[2.2 Архитектура 7](#_Toc163546890)

[2.3 Основна част 8](#_Toc163546891)

[2.3.1 Начална страница 8](#_Toc163546892)

[2.3.2 За Нас 9](#_Toc163546893)

[2.3.3 Количка 10](#_Toc163546894)

[2.3.4 Информация за поръчката 10](#_Toc163546895)

[2.3.5 Успешна поръчка 11](#_Toc163546896)

[2.3.6 Вход 12](#_Toc163546897)

[2.3.7 Регистрация 13](#_Toc163546898)

[2.3.8 Заявка за ремонт 13](#_Toc163546899)

[2.3.9 Вашите заявки 14](#_Toc163546900)

[2.3.10 Изход 15](#_Toc163546901)

[2.3.11 Детайли 15](#_Toc163546902)

[2.3.12 Обработка на заявки 16](#_Toc163546903)

[2.3.13 Редакция на заявка за автомобилно обслужване 17](#_Toc163546904)

[2.3.14 Публикуване на обява 18](#_Toc163546905)

[2.3.15 Редактиране на Обява 18](#_Toc163546906)

[2.3.16 Изтриване на Обява 19](#_Toc163546907)

[2.3.17 Потребители 20](#_Toc163546908)

[3. Заключение 20](#_Toc163546909)

[4. Използвана литература 21](#_Toc163546910)

[5. Приложения 22](#_Toc163546911)

[5.1 Стартиране на проекта 22](#_Toc163546912)

[5.2 Тестови данни на проекта 22](#_Toc163546913)

[5.3 База данни 22](#_Toc163546914)

[5.3.1 MongoDB 22](#_Toc163546915)

[5.3.2 Mongoose 24](#_Toc163546916)

# Увод

## 1.1. Цел на приложението

Дипломният проект представлява уеб приложение, което е за сервиз за леки автомобили. Приложението цели да предостави различни функционалности за покупка на части за коли на потребителите и да приема и обработва заявки за ремонт на техните автомобили в сервиза. Потребителите имат възможността да добавят избрана от тях част за автомобил в количката, да я премахват, да увеличават и намаляват броя ѝ. Потребителя също така има и страница където може да следи за прогреса на своите заявки за ремонт на автомобил, дали са чакащи, приети или са финиширани. Освен потребител има и още две роли -администратор и технически сътрудник. Техническият сътрудник може да обработва заявките за ремонт на автомобилите. Администраторът може да управлява потребителите, да създава нови обяви за части на коли, да редактира обявите, да ги изтрива и може да обработва заявките за ремонт на автомобилите.

## 1.2. Резултат от създаването на приложението

Резултатът е една централизирана, ефективна и удобна платформа, която осигурява лесен достъп до широк спектър от авточасти и услуги за ремонт, без да се налага клиентите да посещават физически места или да прекарват часове в търсене на подходящи части и услуги. Това значително улеснява процеса на поддръжка и ремонт на автомобили, като същевременно предлага следните предимства:

* Удобство: Потребителите могат лесно да намират и закупуват нужните части за своите автомобили от уюта на собствения си дом.
* Ефективност: Оптимизирането на процеса на обработка на заявките за ремонт намалява времето за изчакване и увеличава капацитета на обслужване на сервиза.
* Прозрачност: Възможността за следене на статуса на заявката за ремонт в реално време увеличава доверието и удовлетворението на клиентите.
* Управление на качеството: Въведените роли на администратор и технически сътрудник гарантират високо ниво на обслужване и възможност за бърза реакция при възникнали проблеми.

Като цяло, приложението не само ще оптимизира операциите на сервиза и ще подобри клиентското обслужване, но и ще спомогне за укрепване на връзката между сервиза и клиентите, като предложи иновативен, удобен и интегриран начин за управление на всички аспекти свързани с поддръжката на леки автомобили.

# Изложение

## 2.1 Използвани технологии

В разработката на уеб приложението за сервиз на леки автомобили са внимателно подбрани технологии, които да гарантират висока производителност, сигурност, и удобство за крайния потребител. Основните технологии включват:

* Visual Studio Code (VS Code): Visual Studio Code е лека, но мощна крос-платформена интегрирана среда за разработка(IDE), поддържана от Microsoft. Тя е предпочитана от разработчиците заради своята гъвкавост, поддръжка на множество програмни езици и вградени инструменти за отстраняване на грешки и управление на версии. VS Code поддържа разширения, които позволяват персонализиране и автоматизация на различни задачи, като по този начин се увеличава ефективността на разработка.



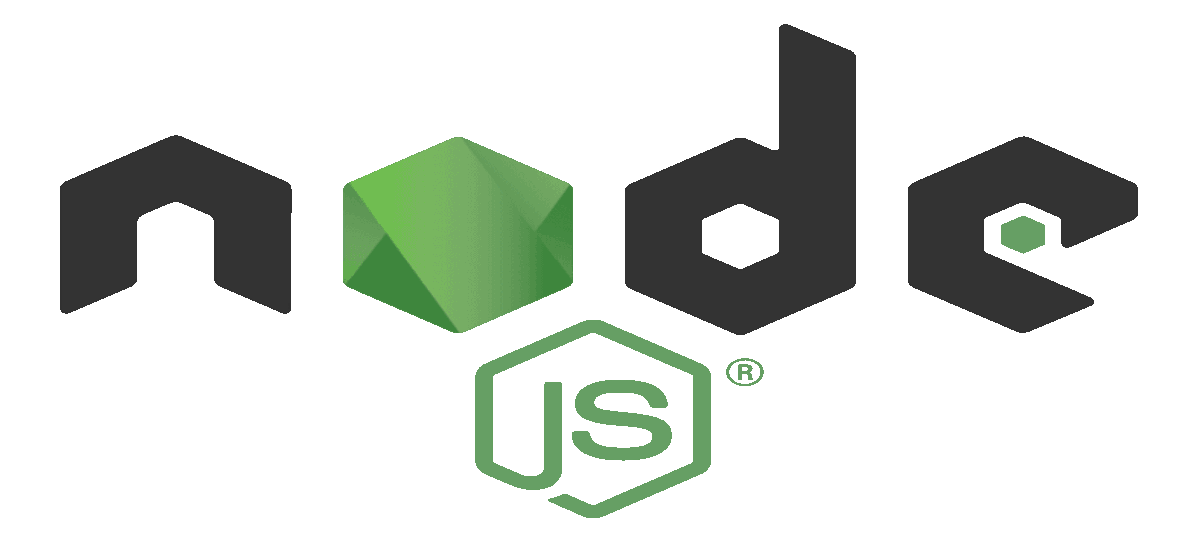
Фигура 1 - Visual Studio Code Иконка

* HTML (HyperText Markup Language): HTML е стандартният маркиращ език за създаване и структуриране на съдържание в интернет. В уеб приложението за сервиз на леки автомобили, HTML се използва за дефиниране на структурата на уеб страниците, включително текста, хипервръзките, изображенията и други елементи. HTML е ключов за създаването на семантична и достъпна структура на уеб приложението, което е основа за добра потребителска навигация и SEO практики.
* CSS (Cascading Style Sheets): CSS е езикът, използван за описване на визуалния стил и оформление на документи, написани на HTML или XML. В контекста на това уеб приложение, CSS се използва за задаване на цветовете, шрифтовете, разположението на елементите и други аспекти на визуалното представяне. Използването на CSS позволява на разработчиците да създават консистентен и привлекателен дизайн, който е адаптивен и отзивчив към различни устройства и размери на екрана.



Фигура 3 - CSS лого

Фигура 2 - HTML лого

* Node.js с Express.js: Node.js е JavaScript среда за изпълнение, която позволява изграждането на бързи и мащабируеми мрежови приложения. Node.js използва асинхронен, неблокиращ входно-изходен модел, който го прави ефективен за приложения с интензивен даннен трафик. Express.js е минималистичен уеб фреймуърк за Node.js, който предоставя редица мощни функции за уеб и мобилни приложения с лекота и гъвкавост, като например мидълуери, маршрутизация и шаблонизиране.

Фигура 5 - Express.js лого

Фигура 4 - Node.js лого

* Bcrypt: Bcrypt е библиотека за хеширане на пароли, предназначена да помогне в сигурното съхранение на пароли. В света на киберсигурността, хеширането на пароли е критична практика за защита на потребителската информация от неоторизиран достъп. Bcrypt автоматично включва сол (допълнителни случайни данни) към хеша, което прави резултата устойчив на атаки с радбойник и други методи за разкриване на пароли.
* Handlebars: Handlebars е шаблонен двигател за JavaScript, който позволява създаването на семантични шаблони с минимално писане на код. С Handlebars, разработчиците могат лесно да генерират HTML, който динамично се променя в зависимост от подадените данни. Това е идеално за проекти, където потребителският интерфейс трябва да отразява променящо се състояние на данни без страницата да се презарежда изцяло.

Фигура 7 - Handlebars лого



Фигура 6 - Bcrypt лого

* Еxpress-session: С помощта на express-session, разработчиците могат да управляват сесии в Node.js приложения, като по този начин се позволява запазването на потребителски данни (като например потребителски идентификатори и предпочитания) през различни заявки. Това е особено полезно за функционалности като потребителско влизане, където сесията трябва да бъде поддържана безопасно през цялото време на взаимодействие с приложението.
* MongoDB с Mongoose: MongoDB е NoSQL база данни, оптимизирана за голяма скорост, мащабируемост и гъвкавост. Тя е идеална за съхранение на JSON-подобни документи с динамични схеми, което прави лесно адаптирането към променящите се изисквания на приложението. Mongoose е библиотека за работа с MongoDB в Node.js приложения, която предоставя елегантен обектно-ориентиран интерфейс за моделиране на данните, улеснявайки така управлението на връзките между данните, валидацията и вграждането на бизнес логика.



Фигура 9 - Mongoose логo

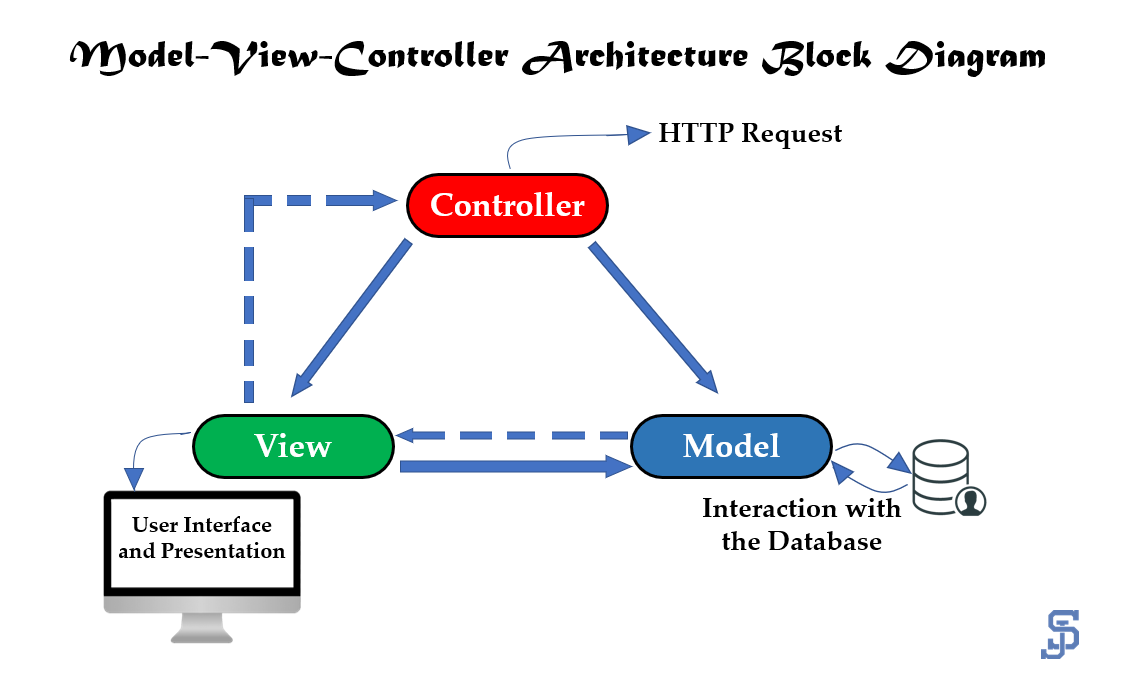
Фигура 8 - MongoDB лого

Тази комбинация от технологии осигурява разработчиците с мощен набор от инструменти за създаване на съвременни уеб приложения, които са бързи, сигурни и лесни за мащабиране.

## 2.2 Архитектура

За създаването на уеб приложението за сервиз на леки автомобили е избрана моделът на архитектурата Model-View-Controller (MVC), която допълнително подобрява структурата и управляемостта на проекта. MVC архитектурата разделя приложението на три основни компонента, които взаимодействат помежду си, но са логически разделени, което улеснява разработката, тестването и поддръжката. Ето как използваните технологии се вписват в MVC архитектурата:

* Model (Модел): В основата на модела стои MongoDB с библиотеката Mongoose, които заедно формират слоя на данните в приложението. Моделът отговаря за структурата на данните, тяхното съхранение, както и за логиката за достъп до базата данни. Моделите в Mongoose дефинират схемите за данни и предоставят мощен API за заявки към базата данни, което позволява лесно управление и манипулация на данни.
* View (Изглед): Изгледите са реализирани с помощта на Handlebars, който служи като шаблонен двигател за генериране на HTML. Изгледите отразяват представянето на данните към крайния потребител и се генерират в зависимост от текущото състояние на модела. Handlebars позволява създаването на динамични уеб страници, които автоматично се обновяват при промяна в данните, без да се налага презареждане на страницата.
* Controller (Контролер): Логиката на контролера е изградена с Node.js и Express.js, като тези технологии управляват потока на данни между моделите и изгледите. Контролерите обработват входящите заявки от потребителите, извършват необходимите операции над данните (чрез моделите) и връщат подходящ отговор (чрез изгледите). Express.js осигурява маршрутизация и средства за управление на сесии, което позволява лесно дефиниране на бизнес логиката и потребителските интеракции в приложението.



Фигура 10 - MVC принцип на работа диаграма

Добавянето на express-session и bcrypt към този набор от технологии подобрява функционалността и сигурността на приложението, като същевременно се спазва логиката на MVC архитектурата. Еxpress-session улеснява управлението на сесиите в контролерите, позволявайки лесно съхранение и достъп до потребителските данни през различните заявки. Bcrypt се използва в моделите за защитено съхранение на пароли, като по този начин се осигурява, че чувствителната потребителска информация е защитена съгласно най-добрите практики в индустрията.

## 2.3 Основна част

### 2.3.1 Начална страница

Началната страница представлява внимателно структурирано пространство, предназначено да даде на потребителя бърза ориентация и достъп до разнообразие от функционалности. Отляво на горната част на страницата е позиционирано зелено лого, което се откроява на тъмния фон и утвърждава бранда на приложението. Черната навигационна лента, разтегната през целия горен край, предлага бял текст за лесна четимост, с бутони за преминаване към различни секции на сайта, включително подробна информация за компанията, управление на количката, автомобилните части, както и опции за вход и регистрация на нови потребители.

Централното място под тази лента е предоставено на търсачката, която предлага филтриране по категории, свободно търсене и цена, давайки възможност за бързо и точно откриване на желаните продукти. Полетата за въвеждане са обозначени с интуитивни подсказки за текст, което води потребителя през процеса на търсене без усилия.

Под търсачката, страницата привлича вниманието с галерия от висококачествени изображения на автомобилни части, всяко с име на продукта и цена в български лева. Визуализациите са подредени в редици и всяки продукт е със собствен бутон "Детайли", оцветен в зелено за ненатрапчиво, но ясно акцентиране, като предлага допълнителна информация с един клик. Тази функционалност предполага, че потребителят може лесно да разгледа подробности за всяка част, което е от съществено значение за вземане на информирано решение за покупка.

Дизайнът на страницата е чист и модерен, със силно фокусирано внимание върху удобството на потребителя и лесната навигация. Цветовата палитра е сдържана, с акценти в зелено, което подчертава елементи на бранда и повишава усещането за една целенасочена и професионална среда. Всяка част от интерфейса е проектирана с мисъл за интуитивност, като се цели да се улесни потребителят не само в намирането на продукти, но и в разбирането на стойността, която сайтът предлага – бърз и надежден достъп до широка гама от автомобилни части.

Top of Form

### 2.3.2 За Нас

Страницата "За нас" е витрината, която разказва историята и представя мисията на компанията. Тя започва със заглавие, което ясно показва, че потребителите могат да научат повече за организацията зад уеб приложението. Текстът започва с топло "Добре дошли", което веднага създава чувство на приемственост и доверие. Представено е уверение, че компанията е надежден партньор в автомобилния свят и акцентира на своите стойности - качество на обслужване и внимание към клиента и неговия автомобил.

Следва раздел, който акцентира на "Нашият сервиз", където се обсъжда модерното оборудване на сервиза и широката гама от услуги, предлагани от квалифицирания екип. Текстът подчертава способността на компанията да се грижи за автомобилите на клиентите си, независимо дали става дума за рутинна проверка или специфичен ремонт. Използването на "най-добрите инструменти и технологии" подчертава ангажимента към качеството и ефективността.

Последният абзац е посветен на "Продажба на авточасти" и обяснява как компанията осигурява широк спектър от авточасти за различни марки и модели на автомобили. Тук се акцентира на конкурентните цени и гарантираното качество на продуктите, което осигурява на клиентите увереност, че поддръжката и ремонтите на техните автомобили се извършват с висококачествени части.

Общата цел на страницата "За нас" е да създаде прозрачна и уверена среда, където потребителите да чувстват, че техните автомобилни нужди са разбрани и че могат да разчитат на професионализма и опита на компанията. Тя играе ключова роля в изграждането на доверие и представянето на фирмата като авторитет в сектора, както и в подкрепа на репутацията на уеб приложението като цяло.

### 2.3.3 Количка

Страницата на количката предоставя подробен преглед на избраните от потребителя продукти, преди да преминат към завършване на покупката. Отляво на екрана, потребителят е представен с визуални миниатюри на продуктите, които са добавени в количката, в този случай автомобили, с ясно изложени имена на моделите и техните цени. Това дава възможност на потребителя да веднага визуално идентифицира избраните стоки и да прецени своя избор.

Всяка стока в количката съдържа интерактивни елементи за управление на количеството – бутони за увеличаване и намаляване на броя, което позволява лесно коригиране на поръчката, както и бутон "Изтрий", който дава възможност за бързо премахване на ненужни продукти от списъка. Това е важна функционалност, тъй като предоставя на клиентите гъвкавост във вземането на решения и управлението на покупките им.

Под списъка с продукти е изведена общата сума на поръчката, изписана с големи букви, което е удобно за бързо ориентиране относно общите разходи преди плащане. Това представяне помага на потребителите да проследяват своя бюджет и да вземат информирани решения за покупките си.

Дизайнът на страницата е съобразен с лесната употреба и нагледност, с акцент върху функционалността и ефективността. В долната част на страницата, голям и забележим бутон "Купи" призовава към действие и предоставя на потребителя възможност да премине към следващия етап на покупката, което е завършването на транзакцията. Този бутон е централен елемент на интерфейса и е създаден така, че да привлече вниманието и да улесни потребителя да завърши покупката със самочувствие и яснота.

### 2.3.4 Информация за поръчката

Страницата "Информация за поръчката" е следващата стъпка в процеса на покупка и е важна за събиране на данните, необходими за изпълнение на поръчката. Тук потребителят се среща с прост формуляр, който да попълни, включващ различни полета за въвеждане на лична информация.

На самата страница са изведени полета за "Пълно Име", "Имейл Адрес" и "Телефонен Номер", като всеки елемент е снабден с подходящо място за въвеждане на информацията. Така се осигурява, че компанията може да се свърже с клиента, както и да осигури точност при доставката на поръчките.

Освен това, клиентите могат да изберат метод на доставка между предложените опции като "Speedy" и "Econt", което дава гъвкавост и предпочитания според тяхните нужди или възможности. Това показва, че приложението взема предвид личните предпочитания и удобството на клиентите.

Следва поле за "Адрес", където потребителите вписват адреса за доставка, и поле за "Пощенски код", което е съществено за точността на куриерските услуги.

В самото дъно на формата, забележим бутон "Потвърди поръчката" стои в огромен размер и ярко зелен цвят, като изпъква на фона на страницата и кани потребителите да завършат процеса на поръчка.

Страницата е проектирана с цел предоставяне на бърза и лесна възможност за покупка, като намалява сложността на процеса и улеснява потребителя да продължи напред към финализиране на сделката. Чистият и прост дизайн подчертава функционалността и ергономията, като едновременно с това е създаден да минимизира всякакви потенциални грешки при въвеждане на данни от страна на клиента.

### 2.3.5 Успешна поръчка

След като информацията е въведена и процесът на покупка е продължен напред, потребителят стига до страница, която потвърждава успешното приемане на поръчката. Това е страница за потвърждение, която обикновено служи като завършек на процеса на пазаруване и е изключително важна за предоставянето на увереност и яснота на клиентите.

На страницата са представени поздравителни и благодарствени съобщения, които изразяват признание за избора на клиента и потвърждават завършването на транзакцията. "Вашата поръчка е успешно приета" е великолепно начало, което изпраща положително съобщение на клиента и в същото време подчертава успешността на операцията. "Благодарим ви, че избрахте нашите услуги" е израз на благодарност, който помага за изграждането на лоялност към марката.

Текстът продължава с обещание за бъдещо взаимодействие: "Ще се свържем с вас скоро, за да ви информираме за вашата поръчка." Това предоставя увереност на клиента, че той ще бъде държан в течение и че поръчката му се обработва внимателно.

В края на страницата е разположен бутон "Продължи с пазаруването", който дава възможност за бързо връщане към търговската част на сайта, ако клиентът желае да направи покупки или просто да продължи да разглежда.

Цялостният дизайн на страницата е създаден с мисъл за минимализъм и функционалност. Чистата и ненатоварена естетика помага на съобщението да изпъкне и дава чувство на завършеност на покупката.

### 2.3.6 Вход

На страницата за вход потребителите се сблъскват с интуитивен интерфейс, който изисква въвеждането на „Имейл“ и „Парола“. По отношение на валидацията, технологиите, използвани в разработването на приложението, играят ключова роля в осигуряването на сигурност и защита на потребителската информация.

При въвеждане на данни, Node.js със своя Express.js фреймуърк улеснява приемането на заявките от формуляра и взаимодействието със сървъра. Express.js може да се използва за създаване на мидълуери, които да валидират входящите данни, като проверяват дали имейл адресът отговаря на стандартния формат, и дали паролата е с подходяща дължина и съдържание.

За съхранението на паролите, bcrypt се използва за тяхното хеширане преди те да бъдат записани в базата данни MongoDB. При опит за вход в системата, въведената парола отново се хешира и се сравнява с хешираната версия, която е съхранена. Ако двете хеширани версии съвпадат, това означава, че въведената парола е валидна.

express-session може да се използва за управление на сесиите на потребителите след успешен вход. Това позволява запазването на потребителския статус (влезли или не) по време на навигацията им в сайта.

Mongoose, като ODM за MongoDB, помага в управлението на данните, включително проверката на потребителските данни при вход. Осигурява механизми за валидация на схемата на данните, което гарантира, че всички записи в базата данни отговарят на предварително дефинирани правила и формати.

Така страницата за вход, заедно с използваните технологии, осигурява надежден процес на аутентикация, който е основен за защитата на потребителската информация и сигурността на уеб приложението като цяло.

### 2.3.7 Регистрация

Страницата за регистрация предлага четири основни полета за въвеждане, необходими за създаване на нов потребителски профил в системата. Полетата са за „Имейл“, където новите потребители въвеждат своя имейл адрес, „Потребителско име“, където могат да изберат уникално име за акаунта си в приложението, и две полета за „Парола“ и „Повтори Парола“, които гарантират, че потребителят е въвел своята парола правилно.

По отношение на валидацията на регистрационните данни, използваните технологии в приложението играят важна роля. Node.js и Express.js могат да се използват за обработка на формата и за валидиране на входящите данни на сървъра. Може да има проверка за валиден имейл адресен формат, достатъчна сила и съответствие на паролата, както и за уникалност на потребителското име в базата данни.

Bcrypt се използва за хеширане на паролите преди тяхното съхранение в базата данни, осигурявайки така допълнителен слой защита на потребителската информация. След като всички полета бъдат попълнени и изпратени, Mongoose обработва данните и създава нов запис в MongoDB, като уверява, че всички потребителски данни отговарят на зададените схеми и валидационни правила.

Бутонът „Регистрация“ на страницата служи като крайния елемент, който потребителят натиска, за да изпрати своите данни и да създаде своя акаунт. Процесът на регистрация е важен не само за събиране на информация за нови клиенти, но и за установяване на сигурността и интегритета на потребителската информация в рамките на системата.

### 2.3.8 Заявка за ремонт

За потребители, които вече са влезли в своя профил, навигационната лента на сайта предоставя нови опции, отразяващи променения им статус. Вместо бутони за вход и регистрация, сега те виждат връзки към „ Заявка за ремонт“ и „Вашите заявки“, както и опцията „Изход“, което им позволява да напуснат своя профил безопасно. Това създава плавно и сигурно потребителско изживяване, позволявайки лесен достъп до специфични за тях ресурси и персонализирани действия.

Формата за „Заявка за ремонт“, която се появява на страницата, е предназначена да улесни клиентите при подаване на заявка за обслужване на техните автомобили. С полета за въвеждане на „Модел на автомобила“, „Година на производство“, „Телефонен номер“ и избор на „Вид на услугата“, формата е направена така, че да събира всичка необходима информация, за да може сервизът да планира и изпълни заявката ефективно. Полето за избор на услуга може да предложи различни опции на услугите, предлагани от сервиза, като например обща проверка, масло и филтри, специфични ремонти и т.н.

Бутонът „Изпрати заявка“ в долната част на формата е забележим и лесен за намиране, което насърчава потребителите да завършат своята заявка за услуга. Когато потребителят изпрати формата, данните могат да бъдат обработени със същите технологии, описани преди, включително Node.js, Express.js и MongoDB с Mongoose. Това включва валидация на данните от страна на сървъра и създаването на запис в базата данни, което започва процеса на обслужване на заявката от страна на техническия екип.

Целият процес е създаден да бъде интуитивен и удобен, като по този начин се улеснява взаимодействието между клиент и сервиз, и се осигурява висока степен на удовлетвореност на клиентите.

### 2.3.9 Вашите заявки

На страницата „Вашите заявки“ потребителите имат възможност да прегледат и управляват своите заявки за ремонт на автомобили, като са разделени в три категории: „В изчакване“, „Приети“ и „Завършени“. Ето кратко обяснение за всяка от тях:

1. В изчакване: Тук се показват всички заявки, които все още не са обработени от сервиза. Потребителят може да вижда кратки детайли за всяка заявка, като модел на автомобила и избраната услуга. Заявките в този раздел са предмет на преглед и потвърждение от страна на сервизния персонал.

2. Приети: В този раздел са заявките, които вече са приети за изпълнение. Те са потвърдени от сервизния персонал, и потребителят може да очаква, че когато откара колата си, тя ще бъде взета и ще се започне работа по тях. Описана е дата на която потребителя да докара колата си за ремонт в сервиза.

3. Завършени: Тук се извеждат заявките, които вече са изпълнени и автомобилът е готов за вземане от клиента.

Всяка категория има своя собствена зона в интерфейса, позволявайки потребителите да сменят между различните статуси на заявките си с един клик. Този тип организация помага за поддържането на чистотата и удобството на потребителския интерфейс, правейки управлението на заявките бързо и интуитивно.

### 2.3.10 Изход

Бутонът „Изход“ в навигационната лента е критичен за сигурността и удобството на потребителя. Когато потребителите натискат този бутон, се инициира процес на излизане от техния потребителски профил, което предотвратява неоторизиран достъп до техните лични данни и заявки.

Използвайки технологиите Node.js и Express.js, се създава маршрут, който обработва заявката за изход. Когато този маршрут се активира, express-session се използва за изчистване на текущата сесия на потребителя, като по този начин се премахват всички свързани със сесията данни, което включва потребителския идентификатор и друга информация, удостоверяваща, че потребителят е влязъл в системата.

След като сесията е изчистена, потребителят се пренасочва обратно към началната страница и се променя навигационното меню, което потвърждава, че са напуснали своя акаунт и че всяка последваща дейност ще изисква отново да влязат в системата. Този процес е важен за предотвратяване на случайно или намерено злоупотреба с потребителските акаунти и за поддържане на цялостната сигурност на уеб приложението.

### 2.3.11 Детайли

На страницата с детайли за продукт потребителите могат да намерят цялата необходима информация, за да направят информирано решение относно покупката си. Първо, те виждат висококачествена снимка на продукта, която им предоставя визуално представяне на това, което закупуват. След това има заглавие, което кратко описва продукта, заедно с подробно описание на основните му характеристики, като размери, материали и други важни детайли. Цената е ясно изписана, което позволява на потребителите да видят колко струва продукта.

Под цената се намират бутони за действия, като в зависимост от ролята си,дали потребителят е нормален потребител, технически сътрудник или админ, се показват различни бутони, които са - „Добави в количката“, „Редактиране”, „Изтриване“ , които дават възможност на потребителите да извършат желаното действие с продукта.

Тази страница е създадена така, че да предостави всички необходими детайли и информация, за да улесни потребителите в тяхното онлайн пазаруване. От изчерпателното описание до ясно изброените цени и бутоните за действия, всичко е проектирано с цел да предложи удобство и яснота на потребителите.

### 2.3.12 Обработка на заявки

На страницата "Обработка на заявки" техническият сътрудник има достъп до различни опции за управление на заявките за ремонт на автомобили в сервиза. Освен техническият сътрудник достъп до тази страница има и админът, който има достъп до всички страници, до които и техническият сътрудник има достъп, но предимно техническият сътрудник се занимава с обработката на заявки. В зависимост от статуса на всяка заявка, сътрудникът може да предприеме различни действия. Ето каква функционалност предлага всяка една от опциите:

**1. В изчакване:**

• Техническият сътрудник вижда всички заявки, които са предоставени от потребителите, но все още не са били обработени.

• Може да разгледа подробности за всяка заявка, като модел на автомобила и исканата услуга.

• Има възможност да приеме или откаже всяка заявка, взимайки решение въз основа на наличните ресурси и капацитет на сервиза.

**2. Приети:**

• Тук се показват заявките, които техническият сътрудник е приел за обработка.

• Той може да финализира заявката като натисне бутона „Завърши“

**3. Завършени:**

• В този раздел се показват заявките, които са успешно обработени и завършени.

• Техническият сътрудник може да види детайлите на заявките, включително телефонния номер на клиента, на когото принадлежи колата и да му звънне да си я вземе

Тези функционалности позволяват на техническия сътрудник ефективно да управлява процеса на обработка на заявките за ремонт на автомобили в сервиза, осигурявайки високо качество на обслужване и задоволство на клиентите.

### 2.3.13 Редакция на заявка за автомобилно обслужване

Когато техническият сътрудник натисне бутона "Редакция и приемане" от раздела "В изчакване", той бива пренасочен към страницата „Редакция на заявка за автомобилно обслужване“. На тази страница той има възможността да допълни само полето "Дата за докарване на автомобила" със съответната дата, на която потребителят трябва да докара автомобила си в сервиза.

Страницата за редактиране на заявка за автомобилно обслужване съдържа следните компоненти:

**1.Форма за редакция на заявка**: На страницата техническият сътрудник вижда форма за редакция на заявка, където се визуализират вече въведените данни от потребителя. Формата включва полета като модел на автомобила, година на производство, телефонен номер на клиента, вид услуга и дата за докарване на автомобила.

**2.Поле за дата на докарване на автомобила**: Под вече въведената информация техническият сътрудник вижда поле, което е предназначено за въвеждане на датата, на която потребителят трябва да докара автомобила си в сервиза. Той може да въведе съответната дата в това поле, за да може да се информира клиента.

**3.Бутон за запазване**: След като е допълнил полето с датата за докарване на автомобила, техническият сътрудник вижда бутон за запазване. Чрез натискане на този бутон въведената от него информация се запазва и заявката за обслужване на автомобила се обновява с дата за докарване.

Тази страница предоставя бърз и лесен начин за техническия сътрудник да обнови информацията за заявката за автомобилно обслужване и да осигури, че клиентът ще донесе автомобила си в срок в сервиза.

### 2.3.14 Публикуване на обява

Страницата "Публикуване на обява" представлява средство, чрез което администраторът може да създава нови обяви за части на автомобили, които ще бъдат достъпни за потребителите на платформата. Използва Node.js с Express.js за обработка на заявките към сървъра, а данните за обявите се съхраняват в MongoDB с Mongoose.

На страницата се предоставя форма или интерфейс, където администраторът може да въведе подробности за новата обява. Тези подробности включват снимка, цена, име, описание и категория на продукта.

Изображение на продукта - В полето за изображение се пише името на вече качено изображение във файловата система на приложението и по този начин винаги е достъпно това изображение и качеството му остава отлично.

След като администраторът завърши с попълването на информацията, той може да изпрати заявката за публикуване на обявата, която веднага става достъпна за потребителите на приложението.

Тази страница е от съществено значение за управлението на съдържанието на приложението от страна на администратора. Тя му предоставя интуитивен интерфейс, който му позволява лесно да добавя нови обяви и да актуализира съдържанието на платформата, като по този начин поддържа актуалност и привлекателност за потребителите.

### 2.3.15 Редактиране на Обява

Тази страница, която се достига от детайлите на съответния продукт и след натискане на бутона "Редактиране" само от администратора, представлява форма за редактиране на детайлите на конкретния продукт. Тя позволява на администратора да актуализира информацията, свързана с продукта, като например името, описанието, цената и изображението.

Страницата се състои от следните компоненти:

**1.Форма за редактиране**: На страницата се намира форма, в която се визуализират текущите данни за продукта. Тя включва полета за редактиране, като име, описание, цена и изображение, които могат да бъдат променяни.

**2.Изображение на продукта**: В полето за изображение се пише името на вече качено изображение във файловата система на приложението и по този начин винаги е достъпно това изображение и качеството му остава отлично.

**3.Бутон за запазване**: След като администраторът внесе желаните промени, той натиска бутона за запазване, който обновява информацията за продукта в базата данни и я визуализира на сайта. Това позволява на администратора да осъществи бързи и лесни промени в съдържанието на платформата.

Тази страница е от съществено значение за администратора, тъй като му дава възможност да поддържа актуалността и качеството на предлаганите продукти в платформата, като ги актуализира спрямо текущите изисквания и наличната информация.

### 2.3.16 Изтриване на Обява

Страницата, която се появява след натискане на бутона "Изтриване" от детайлите на съответния продукт, позволява на администратора да потвърди изтриването на съответния продукт. Това предоставя допълнително потвърждение, преди фактическото изтриване на информацията за продукта от базата данни на приложението.

Страницата се състои от следните компоненти:

**1.Форма за изтриване**: На страницата се намира форма, в която се визуализират текущите данни за продукта, който ще се изтрие при потвърждение. Тя включва полетата - име, описание, цена и изображение

**2.Бутон за изтриване**: Когато администраторът натисне бутона за изтриване, даденият продукт ще бъде окончателно изтрит от базата данни и няма да го има в приложението.

Страницата за потвърждение на изтриване е важен елемент от управлението на съдържанието на приложението, като предоставя на администратора възможност да предотврати случайно изтриване на информацията или да потвърди намерението си за изтриване на продукта.

### 2.3.17 Потребители

Страницата е за управление на потребителите в приложението и дава на администратора възможност да преглежда списъка с потребителите и да изтрива тези, които се смята, че трябва да бъдат премахнати от системата.

Страницата се състои от следните компоненти:

**1.Списък с потребители**: В главната част на страницата администраторът може да види списък с всички потребители на приложението. За всеки потребител се визуализира информация като потребителско име, имейл адрес и роля.

**2.Изтриване**: За всеки потребител има налична опция за изтриване на акаунта на потребителя, ако администраторът прецени за нужно. След като администраторът избере определен потребител и натисне този бутон, потребителят бива изтрит от системата.

Тази страница е от съществено значение за администрирането на потребителските профили в приложението и позволява на администратора да поддържа актуалността и реда в потребителската база данни.

# 3. Заключение

Проектът представлява комплексно уеб приложение, което е насочено към управлението на сервиз за леки автомобили. Целта на приложението е да предостави функционалности за потребителите за покупка на автомобилни части, както и за предаване и проследяване на заявки за ремонт на техните автомобили в сервиза. Системата има три различни роли - потребител, администратор и технически сътрудник, като всяка роля има своите уникални права и функционалности.

Приложението е разработено с внимателно подбраните технологии, използвани от много известни и обемни приложения в целия свят. Използваните технологии включват VS Code за разработка, Node.js с Express.js за сървърната страна, bcrypt за сигурност на паролите, Handlebars за шаблонизация, express-session за управление на сесиите и MongoDB с Mongoose за съхранение на данни. Тези технологии създават мощен набор от инструменти, които гарантират висока производителност, сигурност и удобство за крайните потребители.

Основните функционалности на приложението включват възможност за добавяне и управление на части в количката, закупуване на части, предаване на заявки за ремонт на автомобили, следене на прогреса на заявките, както и управление на потребителите, обявите за части и други от страна на администратора. Техническият сътрудник може да обработва заявките за ремонт на автомобили и да осъществява действия по тях.

В заключение, този дипломен проект е успешно реализиран и демонстрира високо ниво на владеене на съвременни уеб технологии. Той предлага решение, което отговаря на нуждите на потребителите и бизнеса, като предоставя сигурност, удобство и ефективност в управлението на автомобилните ремонти и закупуването на части

# 

# 4.Използвана литература

За разработването на проекта са използвани разнообразни източници за информация, включително онлайн сайтове, документация и ръководства на официалните уеб сайтове за съответните технологии. Ето основните използвани източници:

* Документация на Handlebars - <https://handlebarsjs.com/guide/>
* Документация на MongoDB - <https://www.mongodb.com/docs/>
* HTML - <https://en.wikipedia.org/wiki/HTML>
* CSS - <https://bg.wikipedia.org/wiki/CS/>
* JavaScript - <https://bg.wikipedia.org/wiki/JavaScript/>
* Stack Overflow - <https://stackoverflow.com/>
* W3School - <https://www.w3schools.com/>

# 5. Приложения

## 5.1 Стартиране на проекта

## 5.2 Тестови данни на проекта

## 5.3 База данни

### 5.3.1 MongoDB

MongoDB е документно ориентирана, NoSQL база данни, която се използва за съхранение и управление на големи обеми от структурирани, полу-структурирани и неструктурирани данни. Вместо релационната структура на традиционните релационни бази данни, MongoDB използва JSON-подобни документи с динамично създавани схеми.

Ето някои от основните характеристики на MongoDB базата данни:

**1.Документно ориентирана структура:**

MongoDB съхранява данни в документи, които се представят в JSON (JavaScript Object Notation) стил. Всяко документ се състои от набор от ключове (полета) и техните стойности.

**2.Гъвкавост на схемата:**

MongoDB позволява гъвкавост на схемата, което означава, че документите в една и съща колекция могат да имат различни структури. Не е необходимо предварително да дефинирате структурата на колекцията или да създадете схема, както е при релационните бази данни.

**3. Мащабируемост:**

MongoDB е проектирана да се мащабира хоризонтално на няколко сървъра, което ѝ позволява да обработва големи обеми данни и висока натовареност. MongoDB по подразбиране поддържа репликация и шардиране, които са две от ключовите стратегии за мащабируемост. Репликацията и шардингът в MongoDB не само играят важна роля в мащабируемостта на базата данни, но и подобряват надеждността и отказоустойчивостта на системата.

**• Репликация:**

-Репликацията позволява на данните да бъдат копирани и синхронизирани между няколко сървъра или реплики. Това създава няколко копия на основната база данни, наречени вторични реплики.

-Синхронизирането на данните между репликите осигурява наличност и отказоустойчивост на системата. В случай на отказ или повреда на хардуер, вторичните реплики могат да бъдат използвани за непрекъснат достъп до данните.

**• Шардинг:**

-Шардингът позволява разпределението на данните на множество машини или сървъри, наречени шардове, в зависимост от определен критерий като хеширане на ключа или диапазони на стойности.

-Разпределението на данните между шардовете позволява равномерно разпределение на трафика и натоварването, като се предотвратява прекомерното натоварване на единични сървъри.

-Шардингът подобрява производителността и мащабируемостта на базата данни, като позволява обработка на по-голям обем от данни и заявки.

Комбинирането на репликация и шардинг в MongoDB осигурява висока производителност, отказоустойчивост и мащабируемост на базата данни. Това позволява на MongoDB да обработва големи обеми данни и високи натоварвания на трафика в разпределена среда, като се осигурява ефективно управление на данните.

**4.Гъвкави операции за заявки:**

MongoDB предоставя богат набор от операции за заявки, които могат да бъдат използвани за извличане, манипулиране и агрегиране на данни. Заявките могат да бъдат изпълнявани върху цели колекции, документи, подмножества от данни и други.

**5.Индексиране:**

MongoDB поддържа индексиране, което улеснява бързото търсене и анализ на данни. Индексите могат да бъдат създавани върху единични или множество полета, което дава възможност за оптимизация на заявките.

**6.Вградена поддръжка на файлове:**

С GridFS, вградената система за съхранение на файлове на MongoDB, можете да съхранявате и управлявате големи файлове, като например изображения, видео файлове и документи.

**7.Атомарни транзакции:**

По-новите версии на MongoDB предоставят възможност за извършване на атомарни транзакции в рамките на един документ или в рамките на колекция. Това е важно за приложенията, които изискват консистентност на данните при извършване на множество операции.

Тези характеристики правят MongoDB подходящ избор за множество сценарии на приложение, където е необходима гъвкавост, мащабируемост и бърза обработка на данни.

### 5.3.2 Mongoose

Mongoose е библиотека за моделиране на обекти за MongoDB за Node.js среда. Тя предоставя средства за удобна работа с MongoDB чрез дефиниране на схеми за данните, моделиране на обекти и извършване на заявки.

Ето основните характеристики и функционалности на Mongoose:

**1.Дефиниране на схеми**: Mongoose позволява да дефинирате схеми за вашите данни, като определяте типовете на полетата, валидацията, индексите и други свойства на всяко поле.

**2.Създаване на модели**: След като сте дефинирали схемите си, можете да създадете модели, които да представляват документи в базата данни. Моделите са обекти, които ви позволяват да извършвате заявки и да манипулирате данните.

**3.CRUD операции**: Mongoose предоставя удобен начин за извършване на CRUD (Create, Read, Update, Delete) операции върху вашите данни. Това включва създаване на нови документи, търсене на съществуващи, актуализиране и изтриване на данни.

**4.Валидация на данни**: Мongoose предоставя възможност за дефиниране на валидация за вашите данни чрез задаване на правила за валидация за полетата в схемите.

**5.Връзка между документи**: Mongoose позволява да дефинирате връзки между различни документи чрез използване на референции или вградени документи.

**6.Middleware хука**: Mongoose поддържа middleware хука, които ви позволяват да извършвате определени действия преди или след определени събития като съхраняване, актуализиране или изтриване на документи.

**7.Поддръжка на транзакции**: От версия 5.0 нататък, Mongoose предоставя поддръжка на транзакции, което позволява извършването на няколко операции в рамките на една транзакция, като се гарантира консистентност на данните.

Mongoose е изключително полезен инструмент за разработка на приложения с Node.js, които използват MongoDB като база данни, като предоставя удобен начин за работа с данните и моделиране на обекти.