**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 НСКИ„**



гр. Гоце Делчев, ул."Скопие" №4,

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

e-mail: **pmg.direktor@pmggd.bg**, web site: **http://www.pmggd.bg**



**ДИПЛОМЕН ПРОЕКТ**

**Тема: Уеб Сайт на фирма за недвижими имоти „Домко“**

На Симона Николаева Огнянова, ученичка от XIIв клас

**професия код** 481020 „Системен програмист“

**специалност код** 4810201 „Системно програмиране“

**Ръководител/консултант:** Надя Воденичарова

Сесия: май-юни 2024г.

**Гоце Делчев**

**2024 г.**

Съдържание

[**Увод 3**](#_heading=h.gjdgxs)

[**1. Същност на проекта: 3**](#_heading=h.30j0zll)

[**2. Цели на проекта: 3**](#_heading=h.1fob9te)

[**3. Постигнати резултати: 3**](#_heading=h.3znysh7)

[**3.1 Панелът за потребителя включва: 3**](#_heading=h.2et92p0)

[**3.2 Панелът за администратора включва: 4**](#_heading=h.tyjcwt)

[**4. Технологии: 6**](#_heading=h.3dy6vkm)

[**Основна част 10**](#_heading=h.1t3h5sf)

[**1. Цел на проекта 10**](#_heading=h.sch96zz0vrbi)

[**2. Задачи за реализация: 11**](#_heading=h.2s8eyo1)

[**3. Етапи на реализация 11**](#_heading=h.17dp8vu)

[**3.1 Иницииране: 11**](#_heading=h.3rdcrjn)

[**3.2 Планиране: 12**](#_heading=h.26in1rg)

[**3.3 Изпълнение: 13**](#_heading=h.lnxbz9)

[**3.4 Приключване: 28**](#_heading=h.35nkun2)

[**Заключение 30**](#_heading=h.1ksv4uv)

[**Използвана литература 31**](#_heading=h.44sinio)

[**Приложения 32**](#_heading=h.2jxsxqh)

[**Приложение 1 32**](#_heading=h.fh4x49r6m8et)

[**Приложение 2 32**](#_heading=h.jlkudz75rb2r)

[**Приложение 3 32**](#_heading=h.mfvjvr22nkkt)

# Увод

## **1. Същност на проекта:**

Проектът представлява сайт на фирма за недвижими имоти и представя информация за фирмата и услугите, които предлага - наеми и продажби на имоти. Името на фирмата е “Домко”.

## **2. Цели на проекта:**

* Проектът представлява сайт за управление на фирма за недвижими имоти. Целта е да се улесни работата на фирмата и да се придаде по-голяма популярност. Улеснява работата на собствениците и дава възможност на потребителите да намират по-лесно желаните от тях имоти.
* Сайтът съдържа филтри по няколко критерия, които правят лесно намирането на подходящ за потребителите имот, според техните специфични изисквания, да запазват час за оглед и да оставят коментар към администратора, който да направи комуникацията по-лесна.

## **3. Постигнати резултати:**

Изграден е сайт, който съдържа два отделни панела - панел за администратор и панел за потребител. Панелът за потребителя предоставя възможност за използване на услугите на фирмата, достъп до информация за фирмата и форма за обратна връзка от клиентите към администраторите на сайта. Панелът за администратора позволява добавяне, редактиране и изтриване на имоти под наем и имоти за продажби, управление на резервациите и съобщенията от потребителите, както и създаване на други администратори.

### *3.1 Панелът за потребителя включва:*

* Начална страница - съдържа компонент, който представя базово сайта и неговите услуги.
* Страница ”Наеми” - в първата си част страницата съдържа форма, която дава възможност на потребителите да въведат критерии и да филтрират имотите спрямо своите изисквания. След това под формата на отделни компоненти, които приличат на карти се визуализират отделните имоти. В тези карти има и бутон "Галерия", който води към по-подробна информация за избрания имот.
* Страница “Продажби” - в първата си част страницата съдържа форма, която дава възможност на потребителите да въведат критерии и да филтрират имотите спрямо своите изисквания. След това под формата на отделни компоненти, които приличат на карти се визуализират отделните имоти. В тези карти има и бутон "Галерия", който води към по-подробна информация за избрания имот.
* Страница “За нас” - в първата си част съдържа базова информация за историята на фирмата и нейното развитие в годините. В следващата част се визуализира информация за екипа на фирмата.
* Страница “Контакти” - в първата си част съдържа компоненти с адреса, телефона и имейла на фирмата, които позволяват на клиентите да се свържат с лицата, отговарящи за комуникацията. Във втората част има форма, чрез която потребителите могат да изпращат обратна връзка към администраторите и да комуникират с тях. В дясната страна на страницата има географска карта, която показва точния адрес на офисите.
* Страница “Влизане и Регистрация” - визуализират се в навигационния бар само ако потребителят не е влязъл в своя профил. "Влизане" съдържа полета, в които да се въведат потребителско име, парола и бутон, който да направи проверка дали данните се съдържат в базата данни. "Регистрация" съдържа полета, в които да се въведат потребителско име, имейл, парола, повторение на паролата и бутон, който да изпрати данните в базата дании и те да се запазят в нея.
* Страница “Количка” - съдържа два компонента. Единият е компонентът за имота, който е добавен в нея, а в другия се пази общата цена за огледи и бутон "Завърши поръчка", който води към страницата за завършване на поръчката.
* Страница “Завърши поръчка” - съдържа полета, в които да се въведат името и фамилията на човека, който ще отиде на огледа, датата и часа на огледа, и бутон, който да изпрати данните в базата данни.
* Страница “Изход” - потребителят излиза от профила си. Това става чрез изтриване на сесията му.

### *3.2 Панелът за администратора включва:*

* Начална страница - банер, който представя частта на администратора.
* Страница “Наеми” - в първата си част съдържа текст/икона, която води към добавяне на нов имот. Във втората част са визуализирани създадените имоти под наем, подобно на визуализацията за клиентите. Разликата е, че при администратора бутонът "Галерия" е заменен с бутоните "Редактиране" и "Изтриване". "Редактиране" води към форма за промяна на имота, а "Изтриване" изтрива избрания имот.
* Страница “Продажби” - в първата си част съдържа текст/икона, която води към добавяне на нов имот. Във втората част са визуализирани създадените имоти за продажба. Страницата изглежда и работи като страницата за имоти под наем, като разликата е в заявките. Ако страницата "Наеми" прави заявка към модела Rents в базите данни, то страницата "Продажби" прави заявка към модела "Sales".
* Страници за редактиране и добавяне на имоти - съдържат еднакви форми, но с различни функции. Заявката за добавяне на имот просто изпраща въведените данни в базата данни. Заявката за редактиране намира имота по id и го обновява спрямо въведените данни.
* Страница “За нас” - съдържа форма, в която да бъде въведен тектът, който ще се визуализира за потребителите. Въвежда се желаното заглавие на страницата, историята на фирмата и услугите, предлагани от нея. Бутонът "Обнови" променя страницата спрямо въведения текст.
* Страница “Контакти” - в първата си част съдържа форма, в която да бъде въведен тектът, който ще се визуализира за потребителите. Въвежда се желаното заглавие на страницата, адрес, телефон и имейл на фирмата. Бутонът "Обнови" променя страницата спрямо въведения текст. В долната част има линк, който води администратора към съобщенията за обратна връзка, оставени от потребителите.
* Страница “Съобщения от потребителите” - визуализира имената на всички потребители, оставили съобщение за обратна връзка. Когато бъде натиснато името на клиента страницата води към съобщението, оставено от него. Когато бъде натиснат знакът за отметка, това означава, че администраторът маркира съобщението като прочетено и то се изтрива от базите данни.
* Страница “Резервации” - подобно на "Съобщения от потребители" визуализира имената на клиентите, направили резервация. Когато бъде натиснато името на клиента се появява информация за резервацията - типът на запазения имот, адресът му, желаният час и общата цена, която клиентът трябва да предостави за огледите. Когато бъде натиснат знакът за отметка, това означава, че администраторът маркира резервацията като изпълнена и тя се изтрива от базите данни.
* Страница “Изход” - администраторът излиза от профила си. Това става чрез изтриване на сесията му.

## **4. Технологии:**

* ***Visual Studio Code*** - средата, в която разработих своя проект. Избрах да работя с него по няколко причини - има интуитивен интерфейс, който позволява лесно редактиране на кода, поддържа големи популярни програмни езици като JavaScript, Python, Java, C#, HTML, CSS и други, включва вграден терминал, който позволява да се изпълняват команди директно от редактора, има интегрирана поддръжка с Git, което ми помогна да работя по проекта си от различни устройства.
* ***MongoDB Atlas*** - предлага уеб-базирано управление на MongoDB инстанции, което прави конфигурирането и мониторинга на базите данни лесно и достъпно от всяко място, което ми даде възможност да работя от различни устройства, а не локално, само едно. Платформата автоматично мащабира база данни в реално време, в зависимост от натоварването и нуждите на приложението. Това гарантира висока производителност и наличност дори при големи обеми на данните. MongoDB Atlas предоставя множество възможности за сигурност, включително управление на достъпа. Платформата интегрира с други облачни услуги като AWS, Google Cloud и Azure, позволявайки да се изграждат пълноценни приложения в облачната среда с минимални усилия.
* ***Node.js*** - позволява използване на JavaScript за програмиране на сървърната страна, което води до еднообразие в разработката и лесно преминаване между клиентския и сървърния код. Известен с асинхронния си модел на програмиране, който използва неблокиращ вход/изход (non-blocking I/O). Това позволява на сървърните приложения да обработват много заявки ефективно и да използват по-малко ресурси. Node.js разполага с голяма и активна общност, която разработва разнообразни модули (пакети), достъпни чрез npm (Node Package Manager). Тези модули могат да се използват за добавяне на различни функционалности към приложенията, като например работа с бази данни, обработка на изображения, създаване на API и други. Благодарение на асинхронния си модел и ефективността на изпълнение, Node.js е изключително подходящ за скалируеми приложения, които трябва да се справят с голям брой едновременни заявки и натоварване.
* ***Express*** - минималистична уеб рамка (framework) за сървърно програмиране в JavaScript, която е базирана на платформата Node.js. Тя улеснява създаването на уеб приложения и API-та чрез предоставяне на набор от инструменти и функции за обработка на HTTP заявки и отговори, управление на приложението, работа със сесии, аутентикация, мидълуеъри и други. Express предоставя прост и удобен начин за дефиниране на URL маршрути и тяхната обработка. Маршрутите могат да бъдат настроени за обработка на HTTP заявки от методи като GET, POST, PUT и DELETE. Middleware е основна концепция в Express, която позволява на приложенията да изпълняват последователен ред от функции преди или след обработката на маршрутизацията. Това ми позволи да добавя функционалности като влизане, аутентикация, авторизация и други към приложението си. Express предоставя лесни за използване методи за обработка на HTTP заявки и генериране на HTTP отговори. Това включва възможността за четене на данни от тялото на заявката, изпращане на статус кодове и заглавия, както и изпращане на отговори с различни формати като HTML, JSON и други. Express е много гъвкав и лесно се интегрира с различни библиотеки и модули от общността. Това ми даде възможност да използвам допълнителни функционалности и да разширявам възможностите на своите приложения. Express поддържа различни шаблонни двигатели като Pug (преди известен като Jade), Handlebars, EJS и други, които ми дадоха възможност да генерирам HTML страници динамично и лесно. Поради своята гъвкавост и модулна архитектура, Express ми помогна да създам, своя уеб сайт.
* ***Express-Handlebars*** - шаблонен двигател за Node.js, който позволява лесно създаване на HTML шаблони с динамично вграждане на данни. Той е вдъхновен от Handlebars.js и е специално проектиран за използване с Express.js. Избрах го защото е: синтактично опростен - лесен за разбиране и писане. Базира се на стандартен HTML, като позволява вграждане на JavaScript изрази за изчисляване на стойности или за обхождане на данни; Лесно се интегрира с Express.js - той е специално създаден за интеграция с Express.js, което ми позволи да го настроя лесно в моето Express приложение; Той е мощен и гъвкав - предлага богат набор от функционалности, включително условни оператори, цикли, вграждани шаблони и други. Това го прави подходящ за създаване на различни видове шаблони, от прости HTML страници до сложни уеб приложения; Поддържа частични шаблони - позволи ми да създам частични шаблони за често използвани елементи на страницата като хедъри, футери или навигационни менюта и да ги включа лесно в други шаблони; Позволява достъпност до данни - лесно успях да се подам данни към шаблоните от Express.js приложението, което ги прави идеални за визуализиране на динамични съдържания, включително данни от база данни или външни API-та; Дава възможност за поддръжка на разширения и помощни функции - позволява лесно дефиниране на собствени помощни функции и разширения, което ми позволи да разширя функционалността на шаблоните си според моите нужди.
* ***Bcrypt*** - алгоритъм за криптиране на пароли, който е разработен с цел повишаване на сигурността на потребителските пароли. Той използва изпитан метод на криптиране, наречен "криптография със сол" (salted cryptography), за да затрудни атаките на пароли, включително атаки срещу хеширането на пароли. Избрах да работя с него, защото е известен с високата си степен на сигурност. Той използва т. нар. сол (случайна стойност), която се добавя към паролата преди да се хешира. Това усложнява атаките срещу таблиците с хеширани пароли (rainbow table attacks) и атаките срещу пароли с общо използвани хешове; Итеративен е - изграден е от алгоритъм, който ми позволи да задавам броя итерации за използване при хеширането на паролата. По-голямото количество итерации прави атаките на базата на пароли по-трудни и по-бавни; Съвместимост - Bcrypt е наличен за много различни програмни езици и платформи, което го прави лесен за интеграция в различни системи и работи много добре с JavaScript и Node.js; Актуалност - Bcrypt е проектиран да се справя със съвременните методи за атака и криптоанализ. Това го прави подходящ за използване дори във високорискови среди, където сигурността е критична.
* ***Express-session*** - middleware за Node.js, който се използва във framework-а Express за управление на сесии в уеб приложенията и сайтовете. Сесиите са механизъм за запазване на състоянието на потребителите между различните заявки към сървъра и представляват важен инструмент за управление на сесиите на потребителите и управление на сигурността. Защо го избрах: дава възможност за сесионно управление - предоставя удобен начин за създаване и управление на сесии за потребителите на уеб приложението. Това включва създаване на нова сесия, запазване на данни в сесията и изтриване на сесии; Гъвкавост - много гъвкав и ми позволи да конфигурирам различни опции, като например методите за съхранение на сесии, времето на изтичане на сесията, допълнителни настройки за сигурност и други; Сигурност - при правилната конфигурация, express-session може да помогне за предотвратяване на атаки срещу сесиите, като например атаки със сесионни куки, фиксиране на сесии, CSRF (между-сайтово препращане на код) и други; Лесна интеграция - лесно за интегриране в приложения, които използват Express.js, което прави сесионното управление достъпно и удобно за моя проект.
* ***Mongoose*** - библиотека за моделиране на обекти за MongoDB и Node.js. Тя предоставя прост и елегантен начин за работа с MongoDB бази от данни, като добавя допълнителни функционалности и улеснява комуникацията между сайта и базата от данни. Избрах Mongoose, защото има много предимства пред другите библиотеки от този тип: 1. Mongoose ми позволи да дефинирам структура и схема за моите данни в MongoDB Atlas. Това включва дефиниране на модели, които отразяват различните типове данни, валидации, подразбирателни стойности и други характеристики на данните. 2. Валидацията на данни предлага вградена валидация на данни, която позволява да се гарантира, че данните, които влизат в базата от данни, отговарят на определени критерии или правила. 3. Удобен е за работа с базата данни - предоставя удобен API за извършване на различни операции с базата данни като създаване, четене, актуализиране и изтриване на документи. Този API улеснява и оптимизира комуникацията между сайта и MongoDB Atlas; Поддръжа middleware - предоставя middleware функционалност, която позволява да се изпълняват определени действия преди или след извършването на определени операции върху данните. Това включва вградени възможности за валидация, обработка на данни при изпращане и достъпване, и други; 4. Притежава вградени типове на данни - предлага множество вградени типове данни, които могат да бъдат използвани за дефиниране на полета в моделите на базата данни, като например текстови низове, числа, масиви, вложени обекти и други.
* ***Paint.Net*** - Използвах я, за да изчертая логото на проекта. Има интуитивен и лесен за използване потребителски интерфейс, който прави редактирането на изображенията достъпно дори за начинаещи потребители. Менюта, панели и инструменти са организирани логично и предлагат достъп до различните функционалности на програмата. С помощта на Paint.NET успях да извърша базови редакционни операции като обръщане, изрязване, мащабиране и изчертаване на обекти в логото. Paint.NET включва множество вградени филтри и ефекти, които приложих към моето изображение за създаването на ефективен дизайн. Тези филтри и ефекти включват размекване, острие, наситеност, цветова корекция и други. Въпреки че е по-основен, Paint.NET включва и поддръжка на слоеве, която ми позволи да работя с различни елементи на изображението като независими едни от други. Paint.NET поддържа различни формати на файлове, включително JPEG, PNG, BMP, GIF и други. Това ми позволи да импортирам и експортирам моето лого в различни формати според конкретните ми нужди (като png, за да го вмъкна в сайта и като pdn, за да бъде запазено в работен вариант). Избрах програмата, защото е лесна за използване и предлага големи възможности.

# **Основна част**

## **1. Цел на проекта**

Целта на дипломния проект е да се демонстрират умения и познания за работа с подходящи технологии за изработка на качествен и изцяло изграден уеб сайт. Част от технологиите, които използвах бяха малко или напълно непознати за мен първоначално, което изискваше търсене и проучване от различни източници. Това е и втората цел на проекта - да ни научи да мислим кои технологии са най-подходящи за реализацията му, как да приложим нови идеи и да се научим да не поставяме граници на логическото ни мислене. Моят дипломен проект представлява уеб сайт на фирма за недвижими имоти "Домко". Проблемът е реален, може да се използва в реална среда, от реални фирми. Това именно е следващата цел на този проект - да ме сблъска с решаване на реални проблеми в реална среда.

Като паралелка с професионална насоченост ние трябва да бъдем максимално подготвени за професионална кариера. Необходимо е да можем да изграждаме изцяло готови продукти, да ги демонстрираме и да ги защитаваме пред аудитория. Това е и четвъртата цел на проекта - да сме подготвени за удовлетворяване желанията на клиентите като преминем през изготвяне на документации, защита и получим оценка от рецензенти.

Не на последно място - целта на проекта е да ни научи как да защитаваме качествено нашите продукти. Защото това е едно от най-важните неща при разработка на проект в професионалната кариера. Трябва да можем да убедим лицата, които са готови да платят за него, че продуктът ни е по-добър от този на другите и че няма да претърпят загуби, ако инвестират в него.

## **2. Задачи за реализация:**

За да бъде изграден проектът е необходимо да бъдат реализирани следните функционалности:

* Сайтът да предлага възможност за разглеждане на различни дейности и услуги, предлагани от фирмата. Да предлага филтриране на информация по един или няколко критерия.
* Сайтът да съдържа количка за електронна търговия.
* Сайтът трябва да разполага със секции: "Начало", "За нас", "Наеми", "Продажби", "Галерия", "Контакти" и др.
* Трябва да притежава авторско лого.
* Да се управлява от администраторско меню, което да позволява актуализиране и сортиране на информацията.
* Потребителят да може да разглежда и да заявява услуги и имоти под наем или за продажба.
* Дизайнът трябва да бъде адаптивен и да изглежда еднакво добре на различни устройства.

## **3. Етапи на реализация**

### *3.1 Иницииране:*

* *Дефиниране на целите и изискванията на проекта:*
* Допитване до хора, които се занимават с този бизнес, за да разберете точните изисквания за уеб сайта.
* Идентифициране на основните функционалности и характеристики, които трябва да включва сайтът.
* *Анализ на конкуренцията и пазара:*
* Изследване на уеб сайтове на конкурентите за недвижими имоти, за да се разбере какви функционалности се предлагат на пазара.
* Изучаване на потребителските предпочитания и тенденции в областта на уеб дизайна за недвижими имоти.
* *Съставяне на план на проекта:*
* Определяне на сроковете за различните етапи на проекта.
* Разпределение на ресурсите и времето, необходими за всяка фаза.
* *Създаване на дизайн и информационна архитектура:*
* Създаване на прототипен дизайн на уеб сайта, включително структура на страниците, навигация и основни елементи на интерфейса.
* Подбор на цветове, шрифтове и изображения, които ще подчертаят бранда на фирмата за недвижими имоти.
* *Разработка на функционалността:*
* Избор на подходяща уеб технология и платформа за разработка на сайта.
* Имплементиране на основните функционалности, като търсене на имоти, филтриране, карта за локализация и др.
* *Тестване и отстраняване на грешки:*
* Извършване на тестове за функционалност и съвместимост на уеб сайта с различни устройства и браузъри.
* Идентифициране и отстраняване на всякакви грешки или неизправности.
* *Подготовка за пускане в експлоатация:*
* Оптимизиране на уеб сайта за търсачки (SEO) за по-добра видимост в онлайн търсачките.
* Подготовка на документация за администриране и поддръжка на уеб сайта.
* *Обучение и поддръжка:*
* Предоставяне на техническа поддръжка и подновяване на съдържанието според нуждите на клиента.

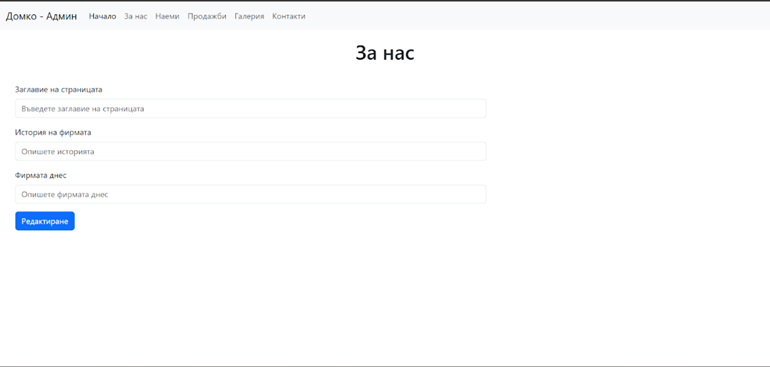
### *3.2 Планиране:*

* Събиране на подходяща информация за подобен тип проекти - какво биха желали клиентите, как биха искали да се използва и какво оценяват потребителите. Обмислят се технологиите, които ще се използват и се избират най-подходящите, за да може сайтът да работи максимално бързо и ефективно.
* Създаване на архитектурата на сайта - създават се шаблони и файлове за всяка страница от приложението. Файловете се разделят в папки, които трябва да бъдат максимално ясни и добре подредени.
* Създаване на функционалност на приложението - изгражда се базово html съдържание на страниците, оформят се схемите на базите данни, създават се заявки към тях, подреждат се архитектурно в специално създадени папки.
* Създаване на дизайна - оформят се всички компоненти от страниците - бутони, текстови полета, изображения, цветове, секции и др., за да може да се използва максимално лесно и да бъде достъпен за всеки потребител и за всяко устройство.
* Създаване на авторско лого - оформяне на малко изображение, което да дава ясна информация за името на фирмата и целите, но и да не е прекалено натрупано.

### *3.3 Изпълнение:*

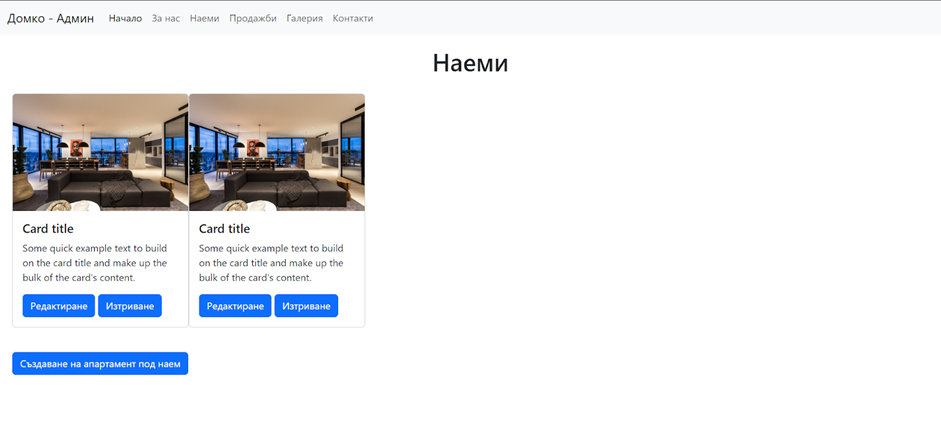
* Събрах основна информация и изисквания за сайтове от типа.
* Избирах подходящи технологии.
* Създадох план за работа и разпределяне на задачите по дни.
* Създадох архитектурата на сайта - оформих файлове за различните страници, реализирах ги в шаблони, осъществих тяхното функционално стартиране.
* Добавих съдържание в страницата “За нас | Администратор” - базова форма, чрез която се редактира страницата на потребителя динамично. (фиг. 1)

фиг. 1



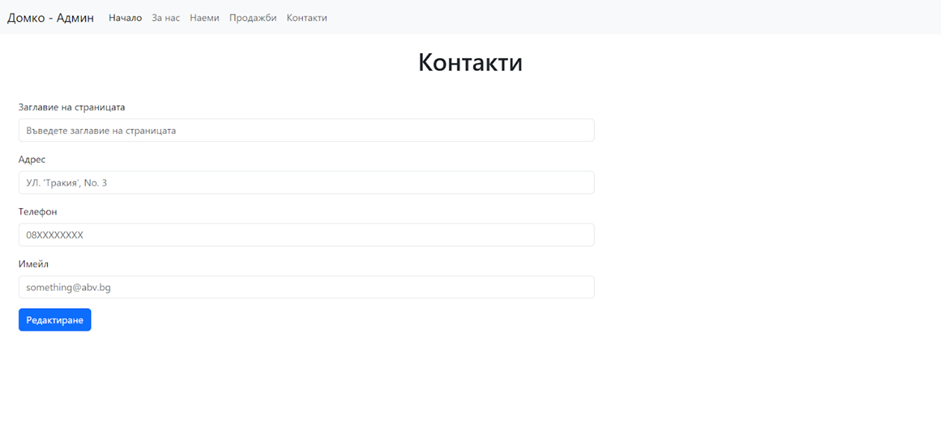
* Добавх съдържание в “Наеми | Администратор” - под формата на карти се визуализират съществуващите имоти. Картите съдържат бутони “изтрий” и “редактирай”, които по-късно ще водят към страници за редактиране и изтриване на конкретния имот. Създадох бутон “Създай имот под наем”, който ще води към страницата за създаване на имоти под наем. (фиг. 2)

фиг. 2



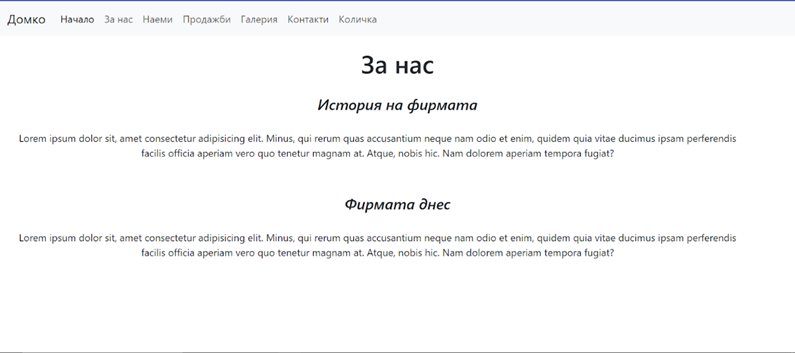
* Добавих съдържание в “Продажби | Администратор” - под формата на карти се визуализират съществуващите имоти. Картите съдържат бутони “изтрий” и “редактирай”, които по-късно ще променя да водят към страници за редактиране и изтриване на конкретния имот. Създаден е бутон “Създай имот за продажба”, който ще води към страницата за създаване на имоти за продажба
* Премахнах “Галерия” от навигационния бар за администратора - необходимо е, защото администраторът ще създава изображенията към самите карти и ще ги вижда в момента.
* Добавих съдържание в “Контакти | Администратор” - базова форма, чрез която се редактира страницата на потребителя динамично. (фиг. 3)

фиг. 3



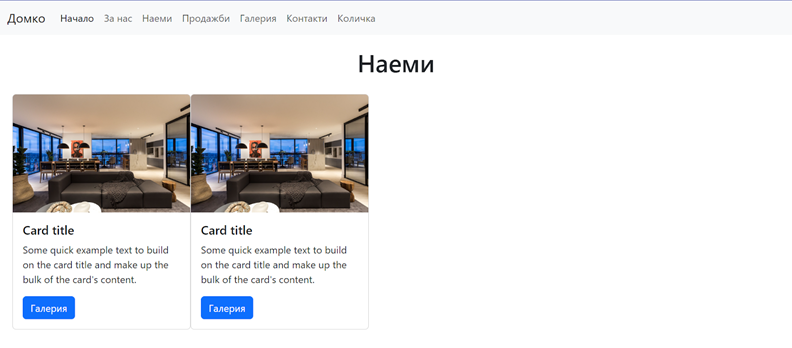
* Добавих съдържание в “За нас | Домко” - добавих секции и параграфи в страницата за потребителя, които по-късно ще зареждат информацията, създадена от администратора. (фиг. 4)

фиг. 4



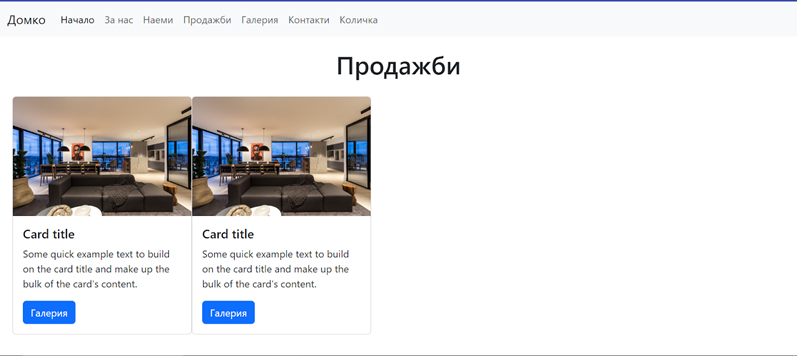
* Добавих съдържание в “Наеми | Домко” - под формата на карти се визуализират съществуващите имоти. Картите съдържат бутон “Галерия”, който по-късно ще води към галерията на конкретния имот. (фиг. 5)

фиг. 5



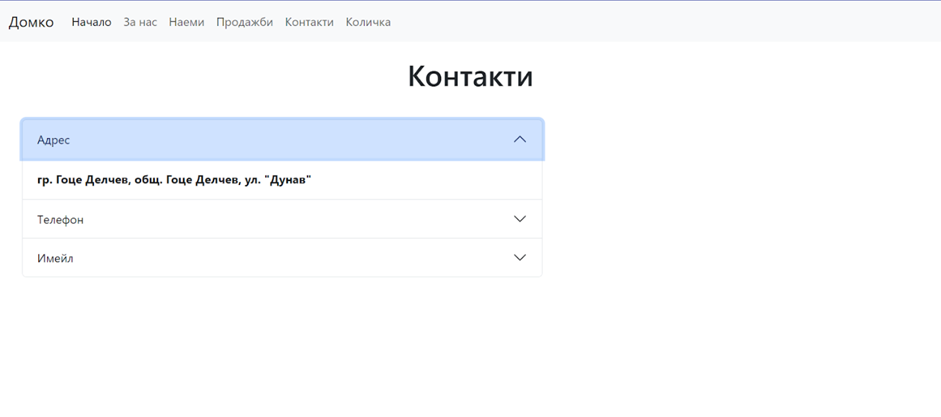
* Добавих съдържание в “Продажби | Домко” - под формата на карти се визуализират съществуващите имоти. Картите съдържат бутон “Галерия”, който по-късно ще води към галерията на конкретния имот. (фиг. 6)

фиг. 6



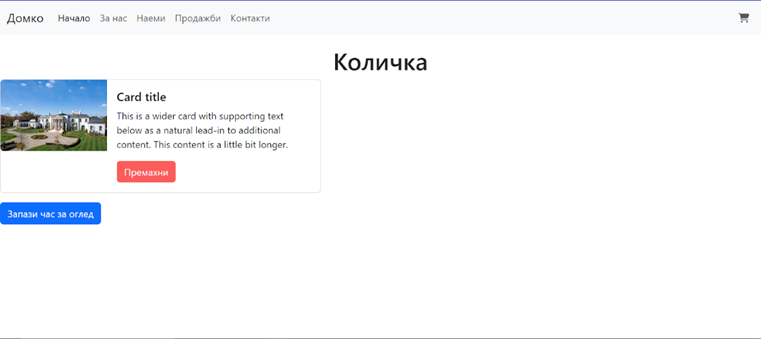
* Премахнах “Галерия” от навигационния бар за потребителя - необходимо е, защото самите карти на имотите ще имат бутон “Галерия”, който ще води към повече изображения на имотите и по-конкретна информация за тях.
* Добавих съдържание в “Контакти | Домко” - създадох компонент, който съдържа информация за контактите на фирмата. По-късно информацията ще бъде създавана от администратор и ще се зарежда от база данни. (фиг. 7)

фиг. 7



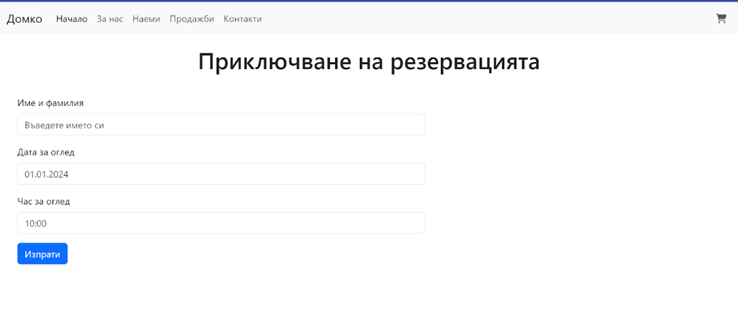
* Промених “Количка” в навигационния бар - текстът се заменя с икона и се поставя в дясната страна на навигационния бар.
* Добавих съдържание в “Количка | Домко” - създадох компонент, който съдържа информация за имота, който е добавен за оглед. Страницата съдържа бутон, който води към друга страница за въвеждане на данните за желаната резервация. (фиг. 8)

фиг. 8



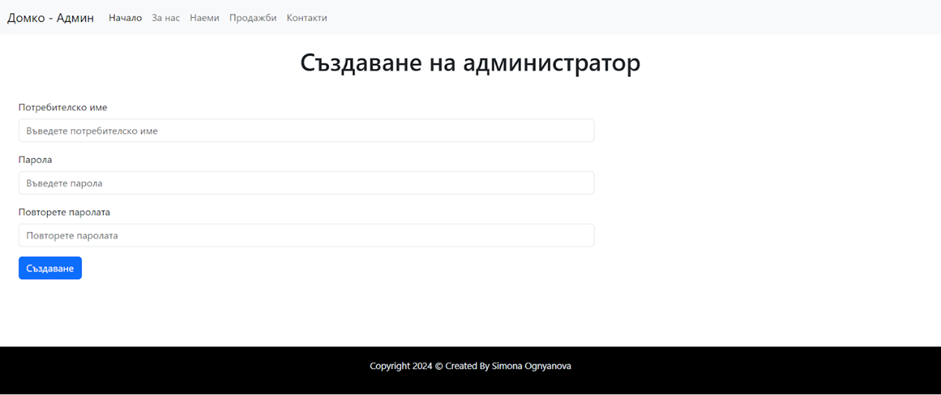
* Създадох страница “Завършване на резервация | Домко” - страницата съдържа форма, в която да се въвеждат данните на потребителя, датата и часа, на които да се извърши огледа. Бутонът изпраща данните към администратора и изчиства количката. (фиг. 9)

фиг. 9



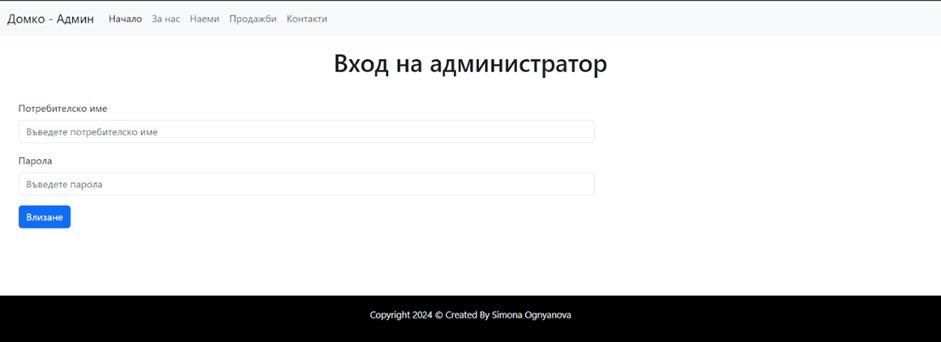
* Добавих долна част за всички страници - секция, която отделя края на страницата от останалите компоненти.
* Добавих страница “Създаване на администратор | Администратор” - съдържа html форма, в която да бъде въведено потребителско име на администратор, парола и повторение на паролата, а после тези данни да бъдат предоставени на новия админ, за да се използват. (фиг. 10)

фиг. 10



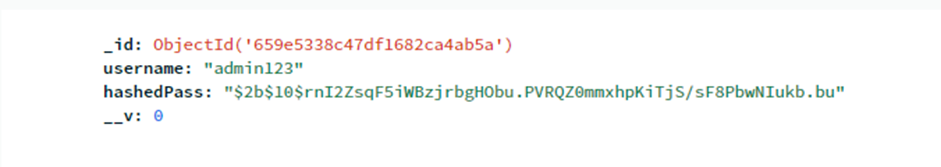
* Добавих страница “Вход за администратор | Администратор” - съдържа html форма, в която да бъде въведено потребителско име и паролата. Тези данни ще бъдат предварително създадени от стария администратор. (фиг. 11)

фиг. 11



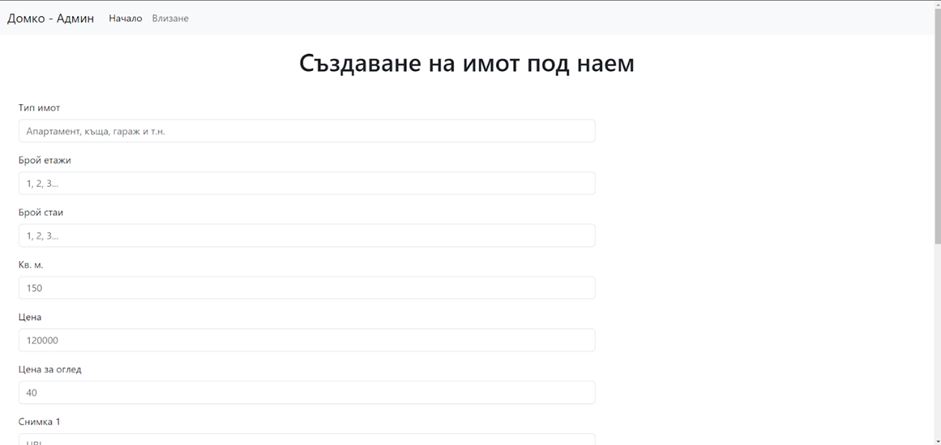
* Създадох динамичен навигационен бар към страницата на администратора - ако има сесия на администратор (администраторът е влязъл в профила си), навигационния бар дава достъп до другите страници, ако не е влязъл, достъпът е само до страницата за влизане.
* Създадох заявка към базите данни за регистрация на администратор - взимат се данните от формата и чрез пост заявка се изпращат към базите. С помощта на bcrypt паролата се хешира и се изпраща криптирана към сървъра, за да може ако хакер е проникнал в профила, паролите да бъдат защитени. (фиг. 12)

фиг. 12



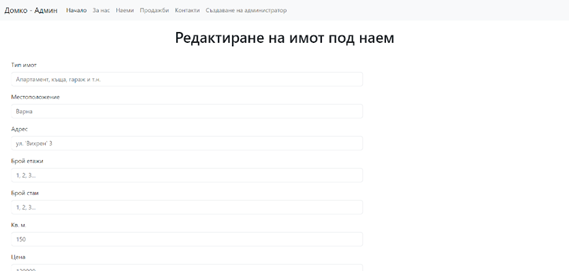
* Създадох заявка към базите данни за вход на администратора - взимат се данните от формата и чрез пост заявка се изпращат към базите. Чрез потребителското име се намира администраторът от базите данни. Ако потребителското име е намерено в базата, с помощта на bcrypt, паролата се сравнява с въведената от потребителя и ако съвпада с паролата от базите данни, то администраторът влиза успешно. Данни: username: admin123, парола: admin123
* Създадох заявка към базите данни за създаване на съдържание на страницата “За нас | Домко” - администраторът въвежда съдържание във форма, което съдържание ще се визуализира на клиента.
* Промених страницата “Наеми | Администратор” - поставя се условие, ако има имот, за него се създава отделна визуализация, ако няма имот, излиза текст: “Няма създадени имоти под наем”.
* Във файла, който рендерира началната страница създадох проверка. Ако администраторът е влязъл в профила си, на “/admin” се появява началната страница, ако не е влязъл, същият адрес води към страницата за влизане.
* Добавих страница “Създаване на имот под наем | Администратор” - създадох страница, която съдържа форма за попълване на данни за конкретния имот. В нея ще бъдат въведени различни характеристики на имота, които след това ще бъдат визуализирани на потребителя. (фиг. 13)

фиг. 13



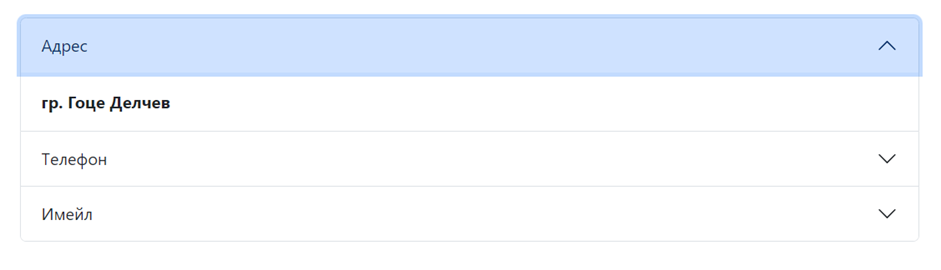
* Добавих post заявка към страницата “Създаване на имот под наем | Администратор” - заявката изпраща въведените данни към базите данни, където те се запазват.
* Добавих get заявка към страницата “Създаване на имот под наем | Администратор” - заявката взима всички създадени данни от базата данни за наеми и чрез темплейт създава карта за всеки съществуващ имот.
* Добавих страница “Създаване на имот за продажба | Администратор” - създава се страница, която съдържа форма за попълване на данни за конкретния имот. В нея ще бъдат въведени различни характеристики на имота, които след това ще бъдат визуализирани на потребителя.
* Добавих post заявка към страницата “Създаване на имот за продажба | Администратор” - заявката изпраща въведените данни към базите данни, където те се запазват.
* Добавих get заявка към страницата “Създаване на имот за продажба | Администратор” - заявката взима всички създадени данни от базата данни за продажби и чрез темплейт създава карта за всеки съществуващ имот.
* Добавих delete заявка към страницата “Наеми | Администратор” - когато бутонът “Изтрий” бъде натиснат се появява прозорец за потвърждение и ако събитието бъде потвърдено, данните от базите данни се изтриват и имотът се премахва от визуализацията.
* Добавих страница “Редактиране на имоти под наем | Администратор” - форма, в която да се въвеждат нови данни за имота и да се редактират в базата данни.
* Добавих post заявка към страницата “Редактиране на имот под наем | Администратор” - намира се конкретният имот по id в базата данни и се редактират неговите характеристики. (фиг. 14)

фиг. 14



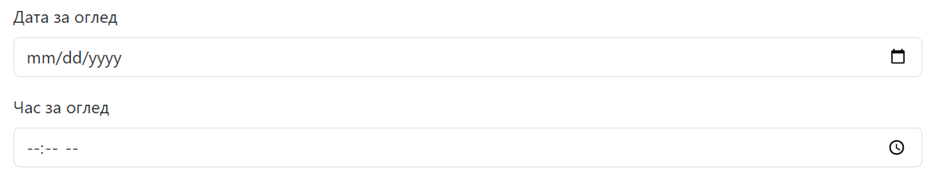
* Добавих страница “Редактиране на имоти за продажба | Администратор” - форма, в която да се въвеждат нови данни за имота и да се редактират в базата данни.
* Добавих post заявка към страницата “Редактиране на имот за продажба | Администратор” - намира се конкретният имот по id в базата данни и се редактират неговите характеристики.
* Добавих get заявка към “За нас | Домко” - Заявката взима данните, създадени от администратора и ги поставя на местата, където потребителят трябва да ги вижда. (фиг. 15)

фиг. 15



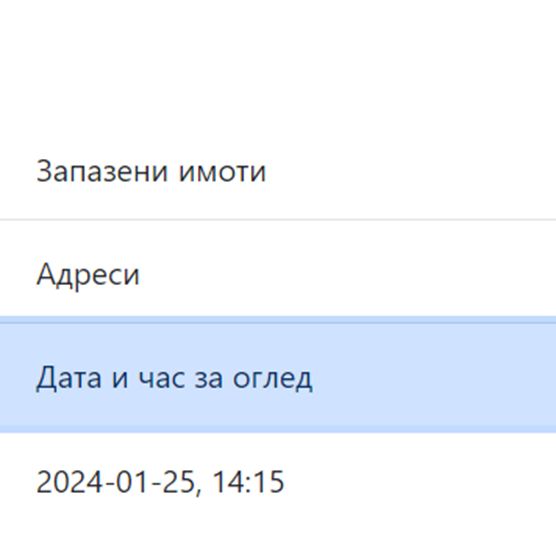
* Добавих get заявка към “Наеми | Домко” - заявката взима всички създадени данни от базата данни за наеми и чрез темплейт създава карта за всеки съществуващ имот.
* Добавих get заявка към “Продажби | Домко” - заявката взима всички създадени данни от базата данни за наеми и чрез темплейт създава карта за всеки съществуващ имот.
* Добавих get заявка към “Контакти | Домко” - заявката взима всички създадени данни от базата данни за контакти и чрез темплейт създава компонент, в който да се визуализират данните.
* Добавих съдържание в страницата “Галерия | Домко” - след като е натиснат бутонът “Галерия” от “Наеми | Домко”, задейства се събитие, което взима Id на конкретния апартамент и води към собствената му галерия. Чрез get заявка се вземат данните за конкретния апартамент и се изпращат към темплейта, в който ще бъдат визуализирани данните.
* Добавих съдържание в страницата “Галерия | Домко” - след като е натиснат бутонът “Галерия” от “Продажби | Домко”, задейства се събитие, което взима Id на конкретния имот и води към собствената му галерия. Чрез get заявка се вземат данните за него и се изпращат към темплейта, в който ще бъдат визуализирани.
* Създадох post заявка за добавяне в количката - след натискане на бутона “Добави в количката”, всички заредени данни в галерията се изпращат към базата данни за добавяне в количката.
* Създадох get заявка за визуализация на добавените данни в количката - Заявката взима всички данни от базите данни и ги визуализира за потребителя в страницата на количката.
* Създадох delete заявка към страницата “Количка | Домко” - когато е натиснат бутонът “Премахни” се изпраща заявка до базите данни, която изтрива имота от количката.
* Добавих страница “Регистрация | Домко” - съдържа html форма, в която да бъде въведено потребителско име, имейл, парола и повторение на паролата, а после тези данни да бъдат изпратени към базата данни.
* Добавих post заявка към страницата “Регистрация | Домко” - заявката взима данните от формата, валидира ги и ги изпраща към базите данни, където да се съхраняват.
* Добавих страница “Вход | Домко” - съдържа html форма, в която да бъде въведено потребителско име, имейл и парола, която след това ще бъдат обработени от заявка.
* Добавих post заявка към страницата “Вход | Домко” - заявката взима данните от формата, проверява дали съществуват тези данни, сравнява ги и ако потребителят има създаден профил, създава сесия и го води към началната страница.
* Добавих ownerId към модела на базата данни за количката - необходимо е, за да се визуализират данни само на потребителя, който ги е създал.
* Добавих визуализация за потребителското име в навигационния бар, ако потребителят е влязъл.
* Добавих изход за потребителя - заявката изтрива сесията на потребителя и е необходимо отново да влезе в профила си.
* Добавих изход за администратора - заявката изтрива сесията на админа и е необходимо отново да влезе в профила си.
* Добавих “totalPrice” в темплейта за количката - извежда се общата цена за оглед на всички добавени продукти в количката
* Редактирах формуляра за завършване на резервация - текстовите полета се заменят с полета за дата и час, за да бъде по-удобно за потребителите при резервация.
* Създадох заявка за приключване на резервация - потребителят въвежда имената си, желания час и дата, които след това се запазват в базата данни, а данните от количката се изтриват. (фиг. 16)

фиг. 16



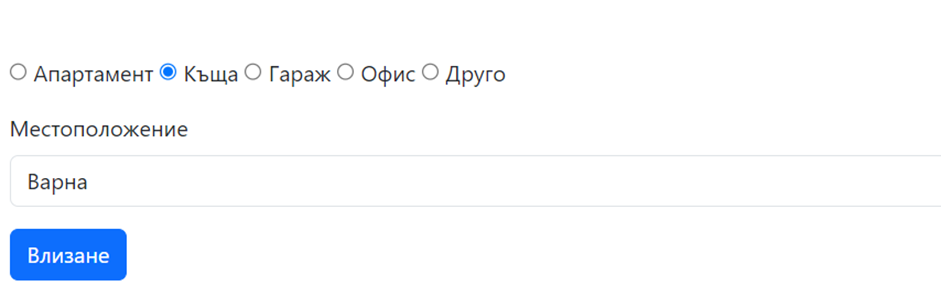
* Добавих страница “Резервации | Администратор” - страница, която ще визуализира направените резервации от потребителите. (фиг. 17)

фиг. 17



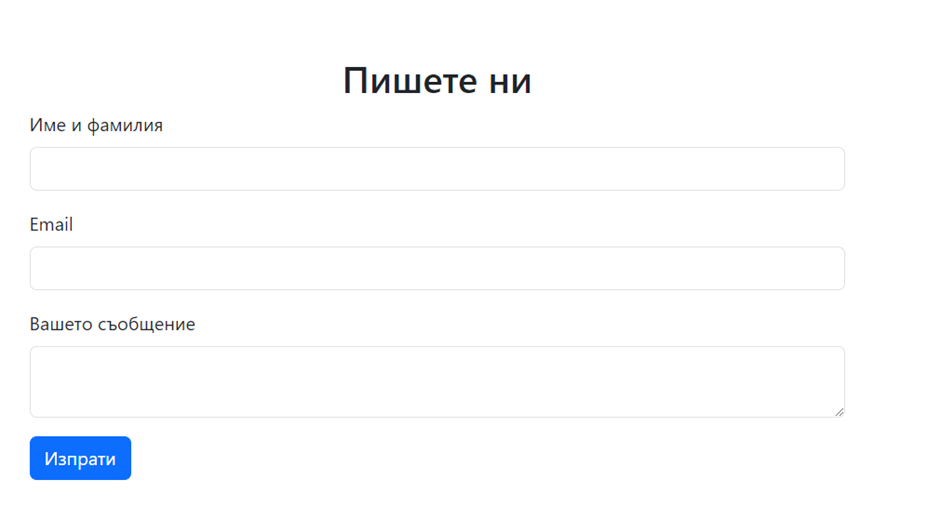
* Добавих заявка за персонално извеждане на данните. Взимат се всички апартаменти, които потребителят е запазил и се визуализират в страницата на администратора.
* Добавих reservationChaking заявка - когато бъде натиснат знакът за отметка, се изпълнява заявка, която отбелязва резервацията като изпълнена и я изтрива от базите данни.
* Създадох страница, в която ще се визуализират филтрираните данни. След като бъдат взети от базата данни, ще се извеждат в нея. Ако данни с тези характеристики не съществуват, ще се извежда текст “Няма създадени имоти”.
* Добавих филтриране към “Наеми | Домко” - въвеждат се два критерия (Тип на имота и местоположение), които изпращат заявка и визуализират имотите, които имат тези характеристики.
* Добавих филтриране към “Продажби | Домко” - въвеждат се два критерия (Тип на имота и местоположение), които изпращат заявка и визуализират имотите, които имат тези характеристики. (фиг. 18)

фиг. 18



* Добавих страница “Потребители със съобщения | Администратор” - страницата ще визуализира имената на клиентите оставили съобщения от клиентската страница.
* Добавих форма за съобщение в страницата “Контакти | Домко” - в нея потребителят ще има възможност да остави данните си и съобщението, които ще бъдат изпратени на администратора. (фиг. 19)

фиг. 19



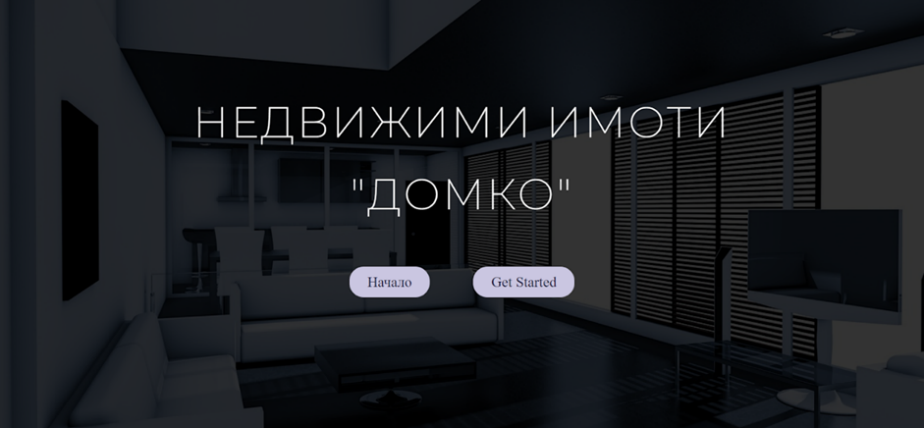
* Добавих post заявка към “Контакти | Домко” за изпращане на съобщение от потребителя.
* Създадох get заявка към “Потребители със съобщения | Администратор” - заявката взима всички съобщения от базата данни и визуализира имената на потребителите, оставили съобщения. (фиг. 20)

фиг. 20

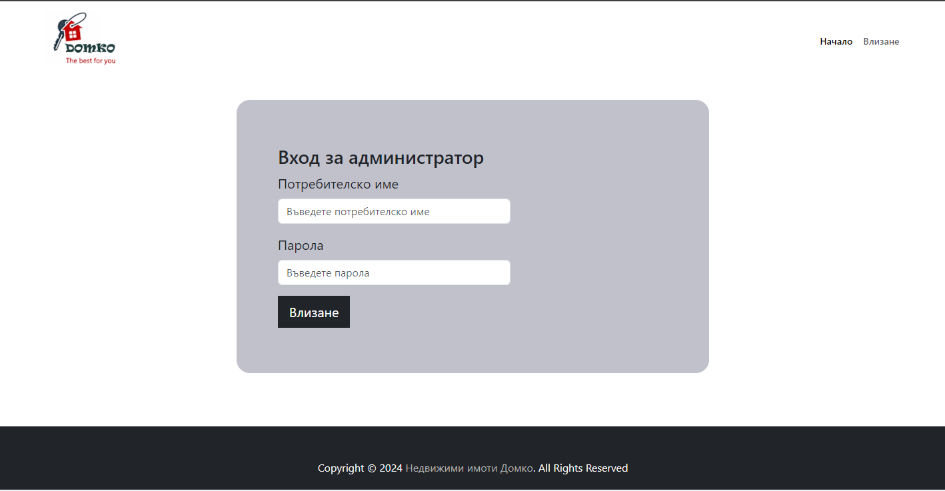
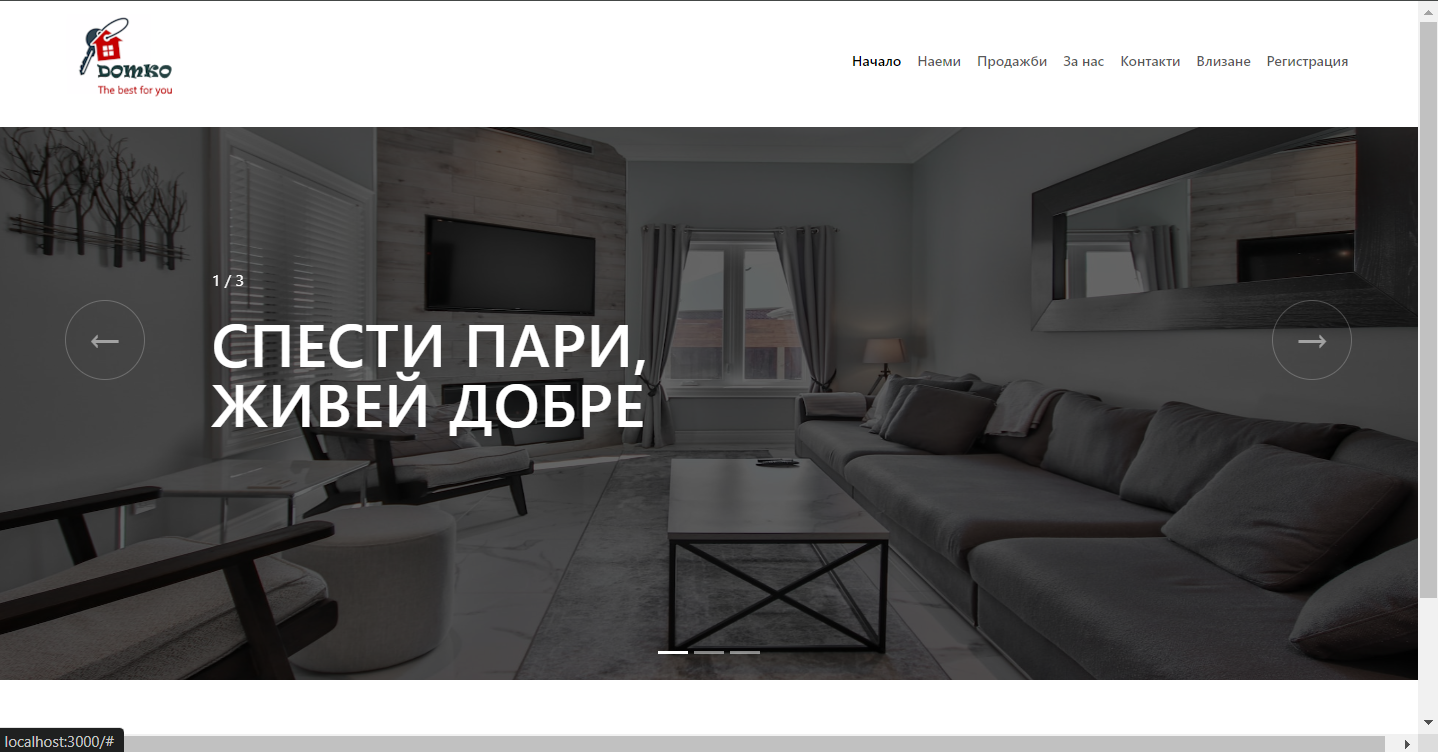


* Създадох get заявка към “Съобщения от потребители | Администратор” - заявката взима съобщението на конкретния потребител и го визуализира в страницата на администратора.
* Създадох дизайн на отдела за потребителя. Първо беше необходимо да определя целите на уеб сайта и изискванията към него, защото всеки тип сайт има различни изисквания заради целевата група и начина, по който ще се използва. След това създадох базова концепция, която включва схеми, които подготвих, преди да започна със създаването на реалния дизайн. Непосредствено след това създадох папка "my-design", в която започнах да създавам дизайна на моя сайт. Подбрах цветове, шрифтове, графика и други елементи, които да осигурят желаното от мен визуално изживяване за потребителите. Важно беше да обърна внимание на потребителския интерфейс (UI) и на потребителското изживяване (UX), за да осигуря лесна навигация и приятен дизайн. Накрая създадох прототип, за да съм сигурна, че визуално изглежда достатъчно добре и ще работи максимално ефективно в моя сайт. (фиг. 21, фиг. 22)

фиг. 21



фиг. 22



* Създадох дизайн на отдела за администратора. Първо беше необходимо да проуча как точно трябва да изглежда този раздел, защото това е панел за управление и трябва да има максимално опростен дизайн, като в същото време работи достатъчно ефективно. След това създадох базова концепция, която включва схеми, които подготвих, преди да започна със създаването на реалния дизайн. Непосредствено след това добавих в папката "my-design" файлове, с имената на страниците, които съдържа раздела на моя проект. Подбрах цветове, шрифтове, графика и други елементи, които да осигурят желаното от мен визуално изживяване за потребителя. Важно беше да създадом прототип, за да съм сигурна, че дизайнът е подходящ за моето приложение.
* В първоначалната разработка на проекта, когато потребителят въвежда невалидни данни, сървърът връщаше нова страница с текст "Невалидни данни". При тестването на проекта прецених, че това не е необходимо и го замених със съобщение, което се връща на клиента със същия текст.
* Следващото нещо, което претърпя промени в процеса на разработка е завършването на поръчката. Първоначално, когато клиентът заяви своята поръчка, сървърът го връщаше обратно към началната страница. Но потребител, който влиза за първи път не би разбрал дали неговата поръчка е приета и дали резервацията е запазена. Затова създадох нова страница, която да дава тази информация на клиентите и те да бъдат сигурни, че резервацията им е запазена.
* Добавих нова страница “Потребител | Профил”. В нея се извеждат всички данни, които потребителят е въвел при регистрацията. От тази страница може също да вижда имотите, които е поставил в “любими”. Данните се взимат от базата данни и се визуализират. Страницата съдържа и линк, който води към любими продукти, създадени от потребителя.
* Добавих страница “Любими | Домко”. В нея се извеждат всички имоти, които потребителят е добавил в любими. По този начин той по-лесно ще може да достъпва имотите, които са направили най-силно впечатление и няма да се налага повторно търсене.
* Добавих бутон “Добави в любими” към галерията на конкретния имот, който след като бъде натиснат ще изпраща имотите в схема “Любими” и след това ще бъдат визуализирани за потребителите.
* Добавих още полета за въвеждане на данни при регистрация на потребител. Бяха необходими, за да се визуализират повече данни в страницата на потребителя. Това наложи и промяна в заявката за изпращане към базите данни, защото и в нея трябваше да се добавят повече полета.
* В момента на тестване на сайта установих, че един продукт може да бъде добавен в количката и в любими имоти повече от веднъж. Логически това не е правилно, защото по този начин ще бъде заявена една и съща резервация два пъти и цената ще се удвои. Затова добавих допълнителна проверка, която да обходи базите данни и ако имотът вече е добавен да извежда “Имотът е добавен в количката/любими имоти” и да не бъде повторно добавян.

### 

### *3.4 Приключване:*

* Направих тестване на целия проект. Първо изпробвах всички функционалности, за да се уверя, че работят достатъчно добре и достатъчно бързо. След това предоставих проекта на мои приятели, които не бяха предварително запознати със същността му, за да придобия реална представа за това доколко сайтът ми е достъпен за потребителите, лесно ли се използва и как работи на различни устройства. Направих няколко вида тестване:

**Ръчно тестване:** Прегледът на уеб сайта от потребителска гледна точка, за да се уверя, че всичко работи както трябва. Проверих всяка страница за визуални и функционални проблеми, включително работата на връзките, формулярите, изображенията и други елементи на сайта.

**Крос Браузърно тестване:** Уверих се, че уеб сайтът работи коректно в различни браузъри като Google Chrome, Mozilla Firefox, Safari, Microsoft Edge и други. Нужно е, защото в някои случаи могат да възникнат разлики в изгледа или функционалността в различни браузъри.

**Responsive тестване:** Трябваше да бъда сигурна, че уеб сайтът е респонсивен и се адаптира към различни устройства и размери на екрана, като например настолни компютри, таблети и мобилни телефони.

**Тестване на функционалността:** Проверих всички функции на уебсайта, включително формулярите, филтрирането, добавянето на продукти в количката и редактирането на страниците работят добре.

**Сигурностно тестване:** Проверих за възможни сигурностни проблеми като уязвимости за XSS (Cross-Site Scripting) или SQL инжекции.

* Проверих дали възложените цели са изпълнени - сайтът максимално достъпен ли е за избраната аудитория, дава ли се достатъчно ясна информация, дали функционално работи, дизайнът подходящ ли е за този тип сайт, изграден ли е от подходящи компоненти, които да осигурят ефективната му работа.
* Изготвих цялостната документация - създадох теоретична част с всички поставени изисквания, описах проекта и се направих цялостен извод за функционалността му, аудиторията, която ще го използва и функциите, които ще изпълнява.
* Предадох проекта - предадох готовият проект на възложителите за оценка. Теоретичната и практическата част защитих пред публика.

Чрез дипломния проект успях да изградя уеб сайт на фирма за недвижими имоти. Изпълних всички поставени задачи, като добавих и неща, които не бяха заложени в заданието. Литературни източници не използвах, помогнаха ми предимно лекциите от платените курсове на softuni.bg.

# 

# **Заключение**

Крайният резултат е успешно изграден уеб сайт на фирма за недвижими имоти "Домко". Успешното изпълнение на заданието и решаването на проблемите, поставени в заданието направи възможно реализирането на функционален продукт, който да може да бъде използван в реална среда.

Заключението на дипломния проект за уеб сайт на фирма за недвижими имоти представлява съвкупността от анализа, проектирането, разработването и тестването на уебсайта. Проектът има за цел да предостави ефективно и удобно решение за клиентите на фирмата за недвижими имоти, като им дава възможност за лесно намиране на подходящи имоти за покупка или наем.

Процесът на разработка на уеб сайта включва детайлно изследване на потребителските нужди и предпочитания, създаване на дизайн, който е функционален, интуитивен и атрактивен, и изграждане на платформа, която е стабилна, сигурна и лесна за управление от администраторите на фирмата.

В резултат на проекта се получава уеб сайт, който предоставя богат каталог от недвижими имоти с детайлна информация за всяко обявление, възможност за търсене и филтриране на имотите според различни критерии, както и удобен начин за свързване с агентите на фирмата.

В заключение, дипломният проект успешно демонстрира професионализма и компетентността на ученичката за разработка, като представя функционален и качествен уеб сайт, който отговаря на нуждите на клиентите и подобрява ефективността на бизнеса на фирмата за недвижими имоти.

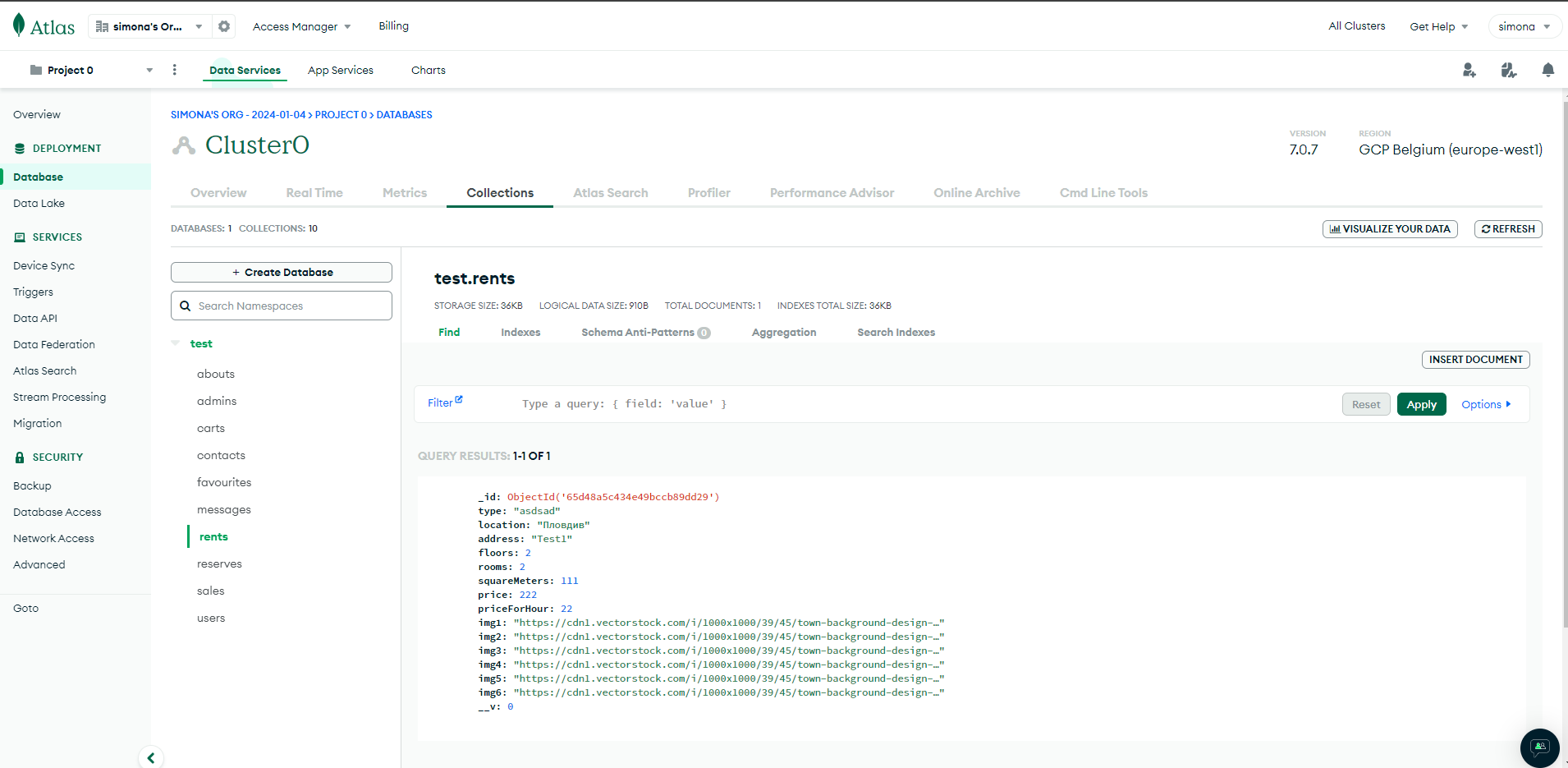
# **Използвана литература**

1. [**https://softuni.bg/**](https://softuni.bg/) **- Наков, Светлин & Колектив**
2. [**https://js-book.softuni.bg/**](https://js-book.softuni.bg/) **- Наков, Светлин & Колектив**
3. [**https://www.mongodb.com/docs/atlas/getting-started/**](https://www.mongodb.com/docs/atlas/getting-started/) **- Колектив, MongoDB**
4. [**https://www.w3schools.com/html/default.asp**](https://www.w3schools.com/html/default.asp) **- Колектив, w3schools**
5. [**https://www.youtube.com/watch?v=UrjZ3qn44uE**](https://www.youtube.com/watch?v=UrjZ3qn44uE) **– INFY TECH**
6. [**https://www.youtube.com/watch?v=kUMe1FH4CHE**](https://www.youtube.com/watch?v=kUMe1FH4CHE) **– freeCodeCamp.org**
7. [**https://www.youtube.com/watch?v=Oe421EPjeBE**](https://www.youtube.com/watch?v=Oe421EPjeBE) **- freeCodeCamp.org**
8. [**https://www.youtube.com/watch?v=bxsemcrY4gQ**](https://www.youtube.com/watch?v=bxsemcrY4gQ) **– Net Ninja**
9. [**https://www.youtube.com/watch?v=4CXtw1CIauQ&t=303s**](https://www.youtube.com/watch?v=4CXtw1CIauQ&t=303s) **– Cobr Kai**
10. [**https://www.youtube.com/watch?v=lY6icfhap2o**](https://www.youtube.com/watch?v=lY6icfhap2o) **– Web Dev Simplified**

# **Приложения**

## Приложение 1

***Схеми на базите данни***



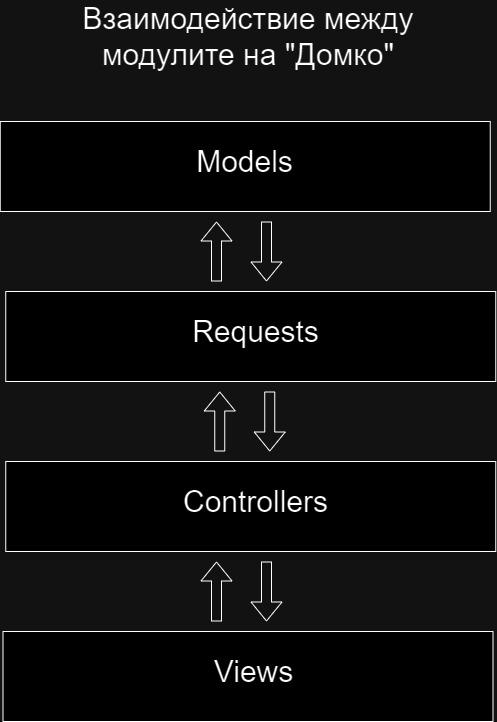
## Приложение 2

***Схема на работа на сайта***

******

## Приложение 3

***Взаимодействие между файловете***

******