ЧАСТНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ ЗА ДИГИТАЛНИ НАУКИ „СОФТУНИ БУДИТЕЛ“, гр. София

**ДИПЛОМЕН ПРОЕКТ**

На Мирослав Мирославов Мирчев

ученик от XII Г клас

професия- код: **481030**, **“Приложен програмист”**

специалност- код: **4810301**, **“Приложно програмиране”**

**Тема: Разработване на уеб приложение за препродаване на дрехи**

**Ръководител-консултант: Лечо Лечев**

Сесия: май-юни 2024г.

Дата:.........................

# Съдържание

[1. Увод 3](#_Toc162784136)

[***1.1.*** ***Описание*** 3](#_Toc162784137)

[***1.2.*** ***Целта*** 3](#_Toc162784138)

[***1.3. Очаквания*** 4](#_Toc162784139)

[***1.4.*** ***Предизвикателства*** 4](#_Toc162784140)

[2. Изложение 5](#_Toc162784141)

[***2.1.*** ***Описание на Софтуерните технологии*** 5](#_Toc162784142)

[2.1.1 Уеб приложение 5](#_Toc162784143)

[2.1.2. HTML 6](#_Toc162784144)

[2.1.3. CSS 7](#_Toc162784145)

[2.1.4 JavaScript 9](#_Toc162784146)

[2.1.5 Package manager 10](#_Toc162784148)

[2.1.6 Node.JS 11](#_Toc162784149)

[2.1.7 Firebase 14](#_Toc162784150)

[2.1.8 Използвани пакети 15](#_Toc162784151)

[2.1.9 Nodemon 17](#_Toc162784152)

[2.1.10 AWS SDK 18](#_Toc162784153)

[2.1.11 Express 20](#_Toc162784154)

[2.1.12 Bcrypt 22](#_Toc162784155)

[2.1.13 Nodemailer 23](#_Toc162784156)

[2.1.14 dotenv 24](#_Toc162784157)

[***2.2. Реализация на приложението*** 26](#_Toc162784158)

[***2.3. Поставени цели за проекта*** 26](#_Toc162784159)

[***2.4.*** ***Ръководство на потребителя*** 26](#_Toc162784160)

[2.4.1 Ръководство за потребител, който ще използва сайта за собствен девелопмент 26](#_Toc162784161)

[2.4.2 Ръководство за потребител, който ще бъде клиент на сайта 27](#_Toc162784162)

[3. Заключение 29](#_Toc162784163)

[4. Информационни източници 30](#_Toc162784164)

# Увод

## ***Описание***

Във века на цифровата технология, електронната търговия се утвърждава като ключов играч във световната икономика. С разрастването на онлайн пазарите и нарастващата осведоменост за устойчивия начин на живот, интересът към покупката на дрехи втора употреба се увеличава значително. Много от хората имат проблем, а именно струпването на излишни стари дрехи, които не са носени и не се намира вече полза от тях. Това съчетание от фактори отваря вратата за развитието на уеб сайтове, специализирани в продажбата на дрехи втора ръка.

Проектът „ StyleSphere” е любителски проект, който пресъздава съществуващи решения на проблема, като OLX и се използва за демонстрация на различни технологии в уеб разработката. Този проект е предизвикателство, но предоставя изследване на различни аспекти от създаването на сайт, като създаване на потребителско преживяване (UX), архитектура на данни и сигурност. В основата на „StyleSphere” са използвани най-новите технологии, за да може да бъде използвано от възможно най-много потребители. Този проект има за цел да отърве хората от излишните им дрехи и да освободи място в гардероба им за нови покупки.

## ***Целта***

Чрез обзор на съществуващи платформи за онлайн търговия, анализ на потребителските предпочитания и изследване на тенденциите в модната индустрия, дипломната работа ще създаде основата за дизайн и функционалност на уеб сайта. Целта е да се предостави удобно и интуитивно потребителско изживяване, докато се гарантира ефективно управление на продуктовия каталог и сигурност на плащанията. Проектът ще се концентрира също така върху внедряването на мерки за привличане на потребители и развитието на стратегии за маркетинг, които да повишат видимостта на уеб сайта и да подкрепят неговото утвърждаване на пазара. Основната цел е да се предостави любителско копие на OLX в уеб вариант, което да позволява на потребителите да се регистрират, да разглеждат вече качени продукти, да качват своите със снимки и детайлно описание, както и да закупуват и поръчват своите продукти. Това значи, че трябва да съществуват акаунти и регистрации за удостоверяване на всеки един от потребителите.

## ***1.3. Очаквания***

Очаква се StyleSpherе да постигне няколко академични и професионални цели. От една страна, проекта ще помогне да добием практически опит в разработването и инициализирането на комплексни уеб приложения, използвайки модерни технологии като Firebase, HTML, CSS и JavaScript. Освен това ще ни помогне да придобием опит в UX дизайна, архитектурата на данните и базите данни, както и сигурността на приложенията. Очакваме също, че този проект ще ни помогне в бъдеще с демонстрация на уменията ни пред работодатели и клиенти.

## ***Предизвикателства***

Създаването на такова приложение идва с няколко предизвикателства. Едно от най-значимите предизвикателства е проектирането на цялостното преживяване(UI/UX), както и интерфейса на приложението, тъй като те трябва да бъдат интуитивни и удобни за потребителя, като същевременно са функционални по начин, който помага на това всеки да може да продаде дрехите си.

Друго предизвикателство е включването на система за удостоверяване на потребители. И не по-малко от другите, интегрирането на различните услуги на приложението Firebase. Това може да бъде много сложен процес, изискващ много време и внимание, за да се осигури безпроблемна работа.

# Изложение

## ***Описание на Софтуерните технологии***

## 2.1.1 Уеб приложение

Уеб приложение е софтуерно приложение, което е достъпно и управлявано изцяло чрез уеб браузър или по-точно казано през интернет. За разлика от традиционните софтуерни приложения, които изискват инсталиране на устройството на потребителя, уеб приложенията се хостват на отдалечени сървъри и се осъществяват достъп чрез уеб браузър, което ги прави независими от платформата и достъпни от всяко устройство с интернет връзка. Уеб приложенията следват модел на архитектура клиент-сървър, където логиката и данните на приложението се намират на отдалечен сървър (backend), а потребителят взаимодейства с приложението чрез уеб браузър (клиент). Тази архитектура позволява безпроблемна комуникация между клиента и сървъра, като клиентът изпраща заявки до сървъра и получава отговори, съдържащи исканите данни или функционалност. Потребителският интерфейс на уеб приложение се представя на потребителя чрез уеб браузър. Състои се от HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) и JavaScript, които заедно определят структурата, стила и интерактивността на интерфейса на приложението.

Съвременните уеб приложения често включват усъвършенствани UI framework-ове и библиотеки като в нашия случай React за създаване на динамични и отзивчиви потребителски интерфейси. Бекендът на уеб приложение се състои от код от страна на сървъра, отговорен за обработката на клиентски заявки, изпълнението на бизнес логиката и взаимодействието с бази данни или външни услуги. Обичайните технологии, използвани в backend разработката, включват езици за програмиране като JavaScript (Node.js) и framework-ове като Express.js, Django, Flask, Rails и Firebase. Бекендът отговаря за задачи като удостоверяване на потребителя, валидиране на данни и операции с бази данни. Уеб приложенията често изискват постоянно съхранение на данни, за да управляват потребителска информация, данни за приложения и други ресурси. Базите данни се използват за ефективно съхраняване и извличане на данни. Популярните избори за бази данни за уеб приложения включват релационни бази данни и бази данни от типа NoSQL като Firebase Firestore. Уеб приложенията разчитат на стандартни комуникационни протоколи като HTTP (Hypertext Transfer Protocol) и HTTPS (HTTP Secure), за да улеснят комуникацията между клиента и сървъра. HTTP се използва за предаване на данни между браузъра и уеб сървъра, докато HTTPS добавя слой на криптиране, за да защити предаването на данни, като гарантира поверителността и целостта на потребителските данни.

Уеб приложенията често трябва да управляват потребителски сесии, състояние на приложението и данни в множество взаимодействия. Техниките за управление на сесии като бисквитки, сесии и токени се използват за поддържане на удостоверяването на потребителя и проследяване на потребителските взаимодействия(Firebase Auth). Освен това модерните уеб приложения могат да използват библиотеки за управление на състоянието от страна на клиента като Redux, MobX или контекстен API в React за управление на сложни състояния на приложения. Веднъж разработени, уеб приложенията трябва да бъдат разположени на уеб сървър, за да станат достъпни през интернет. Внедряването включва конфигуриране на сървърната среда, качване на файлове на приложението и осигуряване на правилна свързаност. Уеб приложенията могат да бъдат хоствани на различни платформи, включително облачни услуги като Firebase Hosting.

## HTML

HTML (HyperText Markup Language) служи като гръбнакът на световната мрежа, осигурявайки структурата и съдържанието за почти всяка уеб страница, която срещнете онлайн. Това е език за маркиране, който дефинира структурата на уеб документи чрез използване на система от тагове и атрибути за описание на елементите в една страница. В основата си HTML се състои от набор от таг-ове за маркиране, които определят структурата и съдържанието на уеб страница.

Чрез тях се форматира, графично оформя текста и неговите отделните части в рамките на една уеб страница, като например заглавия, цитати, текстови раздели, хипертекстови препратки и т.н. Тези тагове са затворени в ъглови скоби (< >) и обикновено идват по двойки - отварящ таг и затварящ таг. Отварящият таг показва началото на елемент, докато затварящият таг обозначава края на този елемент. Някои елементи обаче са самозатварящи се и не изискват затварящ таг. HTML елементите могат да бъдат категоризирани в няколко типа: семантични елементи, елементи на текстово съдържание, мултимедийни елементи, елементи на формуляри и елементи на структурата на документа. HTML документите обикновено са структурирани йерархично, с вложени елементи, образуващи дървовидна структура. Тази йерархична структура е от съществено значение за организиране и стилизиране на съдържанието на уеб страница с помощта на CSS (Cascading Style Sheets) и за добавяне на интерактивност и функционалност с помощта на JavaScript. В проекта “StyleSphere” HTML се използва за направата на страниците на сайта и реално е основата на сайта. HTML е ключова технология за разработката на сайта, тъй като тя е гръбнака на всяка една от страниците в проекта.



**Фигура 1 – Лого на HTML**

## 2.1.3. CSS

CSS или Cascading Style Sheets е основна технология в уеб разработката, която допълва HTML чрез контролиране на представянето и оформлението на уеб документите. Той позволява на разработчиците да стилизират и проектират уеб страници, осигурявайки последователност, естетика и приятно потребителско изживяване на различни устройства и размери на екрана. В основата си CSS се състои от набори от правила, които определят как HTML елементите трябва да се показват в браузъра. Тези набори от правила се състоят от селектори, свойства и стойности. Селекторите са насочени към конкретни HTML елементи, докато свойствата определят визуалните характеристики на тези елементи, като цвят, размер, шрифт, разстояние и позициониране.

Стойностите определят специфичните настройки за всяко свойство. CSS предоставя широк набор от възможности за стилизиране на уеб съдържание, включително селектори, модел на кутия, типография, цветове и фонове, оформление, преходи и анимации и медийни заявки. В обобщение, CSS е критична технология в уеб разработката, която позволява на разработчиците да стилизират и проектират уеб страници, да контролират оформлението и типографията, да прилагат цветове и фонове, да създават адаптивни оформления, да добавят преходи и анимации и да оптимизират потребителското изживяване на различни устройства и размери на екрана. Използвайки силата на CSS, разработчиците могат да създават визуално привлекателни и ангажиращи уеб приложения, които завладяват потребителите и ефективно придават съдържание и идентичност на сайта. В проекта CSS се използва за стилизиране на страниците, бутоните, попъпите и другите елементи на сайта, като им придава уникален изглед и помага сайта да изглежда приемливо и красиво. CSS е ключова технология в разработката на уеб приложението, защото без него всяка страница щеше да бъде просто празен бял фон с черен текст отгоре.



**Фигура 2 – Лого на CSS**

## 2.1.4 JavaScript

JavaScript е многофункционален език за програмиране, който обикновено се използва за създаване на интерактивно и динамично уеб съдържание. Като основен компонент на уеб разработката, JavaScript позволява на разработчиците да добавят функционалност, да манипулират данни и да реагират на потребителските взаимодействия в уеб приложенията. JavaScript се използва предимно за скриптове от страна на клиента, което означава, че работи в уеб браузъра на потребителя, а не на отдалечен сървър. Това позволява на JavaScript да взаимодейства с Document Object Model (DOM), който представлява структурата на HTML и XML документи. Чрез достъп до и модифициране на DOM, JavaScript може динамично да актуализира съдържанието и поведението на уеб страниците в отговор на потребителски действия, като кликвания, събития за въвеждане и събития за зареждане на страници.

Ключовите характеристики и възможности на JavaScript включват променливи и типове данни, оператори и изрази, оператори на потока на управление, функции, обекти и обектно-ориентирано програмиране (ООП), обработка на събития, асинхронно програмиране, обработка на грешки, модули и модуларизация и API на браузъра.

В обобщение, JavaScript е мощен език за програмиране, който позволява на разработчиците да създават интерактивни и динамични уеб приложения чрез манипулиране на DOM, обработка на потребителски взаимодействия, извършване на асинхронни задачи и достъп до API на браузъра. Със своята гъвкавост и широк набор от възможности, JavaScript играе централна роля в модерното уеб развитие, като дава възможност на разработчиците да изграждат ангажиращи и отзивчиви уеб изживявания. В приложението JavaScript се използва за направата на всички функционалности, които съществуват в сайта, както и за направата на сървъра на проекта. Без JavaScript проекта нямаше да може да функционира и нямаше да може да се използва който и да е от елементите в проекта.



**Фигура 3 – Лого на JavaScript**

## 2.1.5 Package manager

Мениджърът на пакети е софтуерен инструмент, който помага при управлението, инсталирането и премахването на софтуерни пакети или библиотеки в рамките на компютърна система или специфична среда за програмиране. Той опростява процеса на обработка на зависимости, като гарантира, че всички необходими компоненти или модули са налични и правилно интегрирани за гладкото функциониране на приложенията или проектите.

Мениджърите на пакети са особено важни при програмирането и разработката на софтуер, където проектите често разчитат на различни външни библиотеки или модули. Те рационализират процеса на придобиване и управление на тези зависимости, като улесняват разработчиците да се съсредоточат върху основните си задачи, вместо да боравят ръчно с отделни пакети. BookBnB използва NPM като мениджър на пакетите.

NPM (Node Package Manager) е мениджър на пакети за JavaScript и мениджър на пакети по подразбиране за Node.js, среда за изпълнение, която позволява изпълнение на JavaScript код извън уеб браузър. Той служи като централно хранилище за повторно използваем JavaScript код, предоставяйки лесен начин за разработчиците да споделят, откриват и управляват зависимости в своите проекти.

С NPM разработчиците могат лесно да инсталират, актуализират и премахват пакети (известни също като модули или библиотеки), необходими за техните приложения. Тези пакети могат да варират от малки помощни функции до цялостни рамки, предоставящи решения за различни аспекти на разработката на JavaScript. В проекта NPM се използва за менажиране на пакети и библиотеки. Използва се също за издаване на dependencies, без които проекта не може да функционира.



**Фигура 4 – Лого на NPM**

## 2.1.6 Node.JS

Node.js е платформа, която променя играта в сферата на уеб разработката, като внася JavaScript в контекста на сървъра и позволява на разработчиците да използват един и същи програмен език за клиентската и сървърната част на приложенията си. Тази иновация премахва необходимостта от превключване между различни езици за различни аспекти на приложението, което води до по-бърза и по-ефективна разработка. Създадена върху мощния V8 JavaScript двигател на Google, Node.js предлага висока производителност и ефективност чрез своя събитийно-ориентиран подход и неблокиращ входно-изходен модел. Това я прави идеална за създаването на мащабируеми мрежови приложения, като уеб сървъри, API-та и други системи, които изискват висока производителност и способност за обработка на голям брой едновременни заявки без забавяне.

Node.js значително разширява възможностите на JavaScript, включвайки достъп до файловата система, мрежови операции и други функционалности, които обикновено са извън обсега на браузъра. Това отваря вратата за създаването на комплексни и напълно функционални сървърни приложения, които могат лесно да се интегрират с различни бази данни, външни услуги и системи. Наличието на npm, мениджъра на пакети за Node.js, допълнително засилва тези възможности, предоставяйки достъп до богата колекция от пакети и библиотеки, които могат да бъдат лесно интегрирани във всякакви проекти, спестявайки време и усилия на разработчиците.

Благодарение на своята гъвкавост и мощ, Node.js намира приложение в широк кръг от проекти - от прости уеб сайтове и приложения до сложни системи за микросервизи, облачни решения и приложения за Интернет на нещата (IoT). Тази универсалност, съчетана с високата производителност и мащабност на платформата, я прави предпочитан избор за много разработчици и компании по целия свят. Node.js се ползва от богата и активна общност, която не само допринася за нейното постоянно развитие и подобрение, но също така предлага обширна поддръжка и ресурси за нови разработчици. Форуми, уебинари, конференции и локални групи за срещи насърчават обмена на знания и опит, като по този начин обогатяват екосистемата на Node.js.

Въпреки че Node.js е сравнително млада технология, нейното въздействие върху уеб разработката е неоспоримо. Тя предлага нов подход към разработването на софтуер, където производителността, ефективността и мащабируемостта са в основата на всеки проект. Като предоставя средства за бърза разработка на високопроизводителни приложения, Node.js не само улеснява работата на разработчиците, но също така позволява на бизнесите да отговорят по-бързо на пазарните изисквания и да предоставят висококачествени услуги на своите клиенти. Използването на Node.js в рамките на проектите позволява на екипите да се адаптират към променящите се технологични изисквания и потребителски предпочитания с голяма гъвкавост, като по този начин поддържат своите приложения актуални и конкурентоспособни.

Един от ключовите фактори за успеха на Node.js е нейната модулна структура, която стимулира споделянето и повторната употреба на код. Разработчиците могат лесно да включват различни модули и пакети в своите проекти, което значително ускорява процеса на разработка и намалява вероятността от грешки. Същевременно, тази модулна система насърчава сътрудничеството и споделянето на знания между разработчиците, като допринася за обогатяването на екосистемата на Node.js и улеснява непрекъснатото ѝ развитие.

Node.js се използва в множество индустрии за разработването на разнообразни приложения, включително видео стрийминг платформи, едностранични приложения (SPA), социални мрежи, електронни търговски платформи и много други. Благодарение на своите характеристики като висока производителност при обработка на асинхронни заявки и вградената поддръжка за реално времева комуникация, Node.js е особено подходяща за създаването на приложения, които изискват бърза обработка на данни и незабавен отговор, като например чат приложения или системи за управление на потоци от данни в реално време.

За разработчиците, Node.js предлага не само технически предимства, но и предоставя възможност за кариерно развитие, като отваря врати към работа в разнообразни проекти и индустрии. Знанието и умението за работа с Node.js се оценяват високо на пазара на труда в областта на уеб разработката, като търсенето на квалифицирани специалисти непрекъснато расте.

В заключение, Node.js е платформа, която предлага значителни възможности за разработване на мащабни и ефективни уеб приложения. С нейната помощ, разработчиците могат да създадат бързи, надеждни и гъвкави приложения, които отговарят на съвременните технологични и бизнес изисквания. Силната общност, богатата екосистема и модулният дизайн на Node.js продължават да стимулират иновациите и да подкрепят разработчиците в тяхната кариера, като по този начин укрепват позицията на Node.js като една от най-популярните и важни технологии в света на съвременната уеб разработка. В проекта Node.JS се използва за разработката на сървъра, като разработва цялото приложение и внася Javascript в сървъра на проекта.

Картина, която съдържа лого, символ, Графика, дизайн

Описанието е генерирано автоматично

**Фигура 5 – Лого на Node.JS**

## 2.1.7 Firebase

Основата на backend-а (автентикация, база данни и др.) на проекта е базирана на Firebase. Firebase е цялостна платформа, предоставена от Google, която предлага разнообразие от услуги за бързо и ефективно изграждане на уеб и мобилни приложения. Той предоставя широк набор от инструменти и функции, включително услуги за база данни, удостоверяване, облачно съхранение, хостинг и други, всички интегрирани в една платформа.

Firebase Cloud Firestore е гъвкава и мащабируема NoSQL услуга за база данни, предоставена от Firebase. Той позволява на разработчиците да съхраняват и синхронизират данни в реално време между клиенти и сървъри, което го прави идеален за изграждане на приложения в реално време като приложения за чат, инструменти за сътрудничество и табла за управление на данни на живо. Firestore поддържа мощни възможности за заявки и индексиране, което позволява на разработчиците да извличат и манипулират данни ефективно.

Firebase Cloud Storage е базирана на облак услуга за съхранение, предоставена от Firebase, която позволява на разработчиците да съхраняват и обслужват генерирано от потребителите съдържание, като изображения, видеоклипове и документи, сигурно и надеждно. Той предоставя мащабируеми решения за съхранение с вградени функции за сигурност, включително контрол на достъпа и криптиране, което го прави подходящ за съхранение и обслужване на мултимедийно съдържание в уеб и мобилни приложения.

Firebase Authentication е услуга, предоставена от Firebase, която позволява на разработчиците да удостоверяват потребители и да управляват самоличността на потребителите в техните приложения. Той поддържа различни методи за удостоверяване, включително имейл/парола, социално влизане (напр. Google, Facebook, Twitter), удостоверяване по телефона и персонализирано удостоверяване, което позволява на разработчиците да изберат метода за удостоверяване, който най-добре отговаря на изискванията на тяхното приложение. Firebase Authentication също така предоставя функции като управление на потребители, потребителски роли и многофакторно удостоверяване (MFA) за подобрена сигурност. В проекта Firebase се използва за вдигането на датабаза за проекта и за аутентикация на юзъри, както и следене на статистики за проекта. Firebase е безплатен, както датабазите, така и хостинга и се оперира много лесно с него, затова и го избрах за проекта.

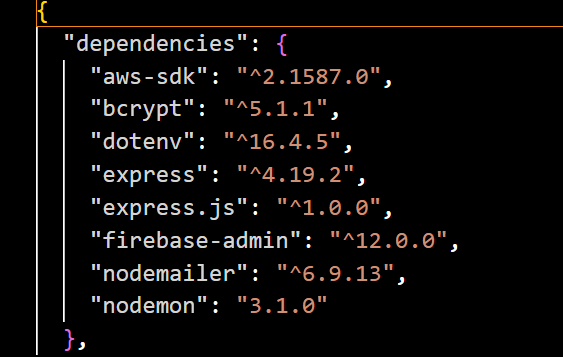
Картина, която съдържа текст, екранна снимка, диаграма

Описанието е генерирано автоматично

**Фигура 6 – Потребителски интерфейс на Firebase**

## 2.1.8 Използвани пакети

Проектът включва сбор от няколко пакета с помощта на които имаме готова функционалност за библиотеките. Те са показани фигуративно и обобщено текстово н следващата страница.



**Фигура 7 – Използваните пакети в сайта**

По-важните от тях са пакетите:

* Firebase - NPM пакетът Firebase е официалната клиентска библиотека на JavaScript за Firebase, която е цялостна платформа за разработка, предоставена от Google. Той позволява на разработчиците лесно да интегрират услугите на Firebase в своите JavaScript проекти, включително уеб и Node.js приложения. Пакетът предоставя унифициран API и набор от функции, които позволяват на разработчиците да взаимодействат с различни услуги на Firebase
* Nodemon - инструмент, разработен за улесняване на процеса на разработка на приложения, като автоматично рестартира нод процеса, когато открие промени във файловете в директорията. Той е особено полезен при разработването на Node.js приложения, където често се налага рестартиране на приложението след всяка промяна в кода, за да се отразят промените.
* AWS-SDK - AWS SDK е набор от инструменти и библиотеки, предоставени от Amazon Web Services (AWS), които позволяват на разработчиците да взаимодействат с различните услуги на AWS чрез програмен код. Този пакет предлага програмен интерфейс (API), който позволява на разработчиците да управляват ресурсите и услугите на AWS директно от техните приложения.
* Express.JS - Express.js е минималистичен уеб приложенчески фреймуърк за Node.js, който предоставя основни функционалности за създаване на уеб приложения и API-та. Той е лек, гъвкав и много популярен сред разработчиците на Node.js, тъй като предлага прост синтаксис и мощни възможности за създаване на сървърни приложения.
* Bcrypt - bcrypt е библиотека за хеширане на пароли, която е особено популярна в средите на софтуерните разработчици за защита на пароли в уеб приложенията. Тя предоставя сигурен и устойчив начин за съхранение на пароли в база данни, като използва сол (salt) и криптиране с хеширане.

## 2.1.9 Nodemon

Nodemon(съкратено от Node Monitor) е неизменна част от ежедневната работа на много разработчици на Node.js приложения поради неговата ефективност и удобство. При разработката на софтуер се изисква непрекъснато тестване и промени в кода, което често води до рестартиране на приложенията, за да се отразят промените. Този процес, въпреки че неизбежен, може да бъде доста досаден и да отнема значително време, особено когато става въпрос за често повтарящи се промени.

Тук влиза в употреба Nodemon със своята изключително полезна функционалност. Той автоматизира процеса на наблюдение и рестартиране на Node.js приложенията при забележени промени в изходния код. Следенето на всички файлове в дадена директория става постоянна задача за Nodemon, като той наблюдава за промени като добавяне, изтриване или модификация на файлове.

Един от ключовите аспекти на Nodemon е неговата способност да предостави интуитивен начин за рестартиране на приложението, когато забележи промени в кода. Това освобождава разработчиците от нуждата ръчно да рестартират приложението след всяка промяна. Също така, Nodemon предлага разнообразни опции за конфигурация, което позволява на разработчиците да го настроят съгласно своите нужди и предпочитания.

Благодарение на своята гъвкавост и удобство за употреба, Nodemon е станал неотменим инструмент в работата на мнозина разработчици на Node.js. Той подобрява процеса на разработка, позволявайки на разработчиците да се съсредоточат върху създаването на качествен софтуер, вместо да губят време с ръчното рестартиране на приложенията си. В проекта Nodemon се използва за улесняване на процеса на рестартиране на сървъра при промени, като се рестартира сам и автоматично актуализира уеб сайта. След всяка направена промяна сървъра се рестартира сам и подпомага бързата работа по проекта, като веднага актуализира страницата с вече променения код.

Картина, която съдържа дизайн

Описанието е генерирано автоматично

**Фигура 8 – Лого на Nodemon**

## 2.1.10 AWS SDK

AWS SDK (Amazon Web Services Software Development Kit) представлява комплект от инструменти и библиотеки, предоставящи разработчиците с обширни възможности за взаимодействие с облачните услуги на Amazon Web Services (AWS). Този SDK е от критично значение за разработката на приложения, които използват инфраструктурата на AWS, като предоставя лесен и удобен начин за интеграция на различни AWS услуги в приложенията.

Основната цел на AWS SDK е да улесни разработката на приложения, като предостави средства и API, които позволяват на разработчиците да изграждат, конфигурират и управляват приложения, които използват широкия набор от услуги, предлагани от AWS. Този комплект от инструменти поддържа няколко програмни езика, включително JavaScript, Java, Python, Ruby, PHP и други, което прави SDK достъпен за широк кръг от разработчици с различни умения и предпочитания.

Чрез AWS SDK, разработчиците могат да взаимодействат с различни услуги на AWS, като например:

- Amazon S3 (Simple Storage Service) за съхранение на обекти в облака.

- Amazon EC2 (Elastic Compute Cloud) за създаване и управление на виртуални сървъри.

- Amazon RDS (Relational Database Service) за управление на релационни бази данни.

- Amazon DynamoDB за управление на NoSQL бази данни.

- Amazon Lambda за изпълнение на код без да се грижите за инфраструктурата.

- Amazon API Gateway за създаване и управление на API.

- Amazon SNS (Simple Notification Service) за изпращане на съобщения до различни крайни точки.

- Amazon SQS (Simple Queue Service) за управление на опашки на съобщения и много други.

Освен това, AWS SDK предоставя различни инструменти за управление на сигурността, мониторинга и анализа на приложенията, които помагат на разработчиците да създадат скалабилни, сигурни и надеждни приложения. Той предлага също и разширени възможности за управление на ресурсите и конфигурациите на приложенията, което позволява на разработчиците да автоматизират различни аспекти от процеса на разработка и управление на приложенията в облака.

В крайна сметка, AWS SDK е критичен инструмент за разработчиците, които работят с облачните услуги на Amazon Web Services. Той предоставя широк набор от инструменти и ресурси, които помагат на разработчиците да създадат стабилни, сигурни и надеждни приложения, които отговарят на изискванията и очакванията на съвременните потребители. В проекта AWS се използва специфично за своята Bucket функция, като тя се използва да генерира линкове за снимките на продуктите, които биват качени за продуктите, след което изпраща този линк като връзка в датабазата, за да бъде достъпно, като така изчиства и улеснява процеса, за да не трябва снимките да бъдат съхранявани в датабазата и потенциално да се объркат в изгледа на сайта.

Картина, която съдържа текст, екранна снимка, лого, диаграма

Описанието е генерирано автоматично

**Фигура 9 – Пояснение на работата на AWS SDK**

## 2.1.11 Express

Express е уеб фреймуърк за Node.js, който предоставя широка гама от функционалности и възможности за създаване на различни видове уеб приложения и API. Създаден с цел да бъде минималистичен, но с мощна функционалност, Express е станал предпочитан избор за разработчиците, които ценят простотата и гъвкавостта на своя код.

Един от основните аспекти на Express е неговото прости и лесно за разбиране API, което позволява на разработчиците бързо и лесно да създават маршрути, обработват HTTP заявки и генерират отговори. Със своя интуитивен синтаксис и гъвкави възможности за конфигуриране, Express прави създаването на уеб приложения приятна и ефективна задача.

Освен основните функционалности за маршрутиране и обработка на заявки, Express предлага и богат набор от middleware, които могат да бъдат използвани за добавяне на различни функционалности към приложенията. Тези middleware включват автентикация, авторизация, обработка на формуляри, компресия на отговорите, обработка на грешки и други, което прави Express много гъвкав и подходящ за различни видове приложения.

Също така, Express е изключително мащабируем и може да бъде използван за създаване на приложения от всякакъв размер и сложност. Той е подходящ както за създаване на прости уеб сайтове и API, така и за по-сложни микросервизни архитектури и приложения. Благодарение на интеграцията си с различни шаблонни езици и бази данни, Express прави работата с различни технологии лесна и удобна.

В крайна сметка, Express е незаменим инструмент за разработчиците, които търсят мощен, но лесен за употреба уеб фреймуърк за създаване на уеб приложения и API с Node.js. Неговата прости синтаксис, гъвкавост и богат набор от функционалности го правят предпочитан избор за много разработчици по целия свят. В проекта Express се използва за фреймуърк, автентикация, авторизация, обработка на формуляри, обработка на грешки, както и обработка на HTTP заявки. Express е лесен за употреба и помага с всичките тези функции по много разбираем и лесен начин. Повече от богатите функционалности на Express са употребени в направата на проекта, като основно са използвани middleware функционалностите.

Картина, която съдържа текст, Шрифт, Графика, лого

Описанието е генерирано автоматично

**Фигура 10 – Лого на Express.JS**

## 2.1.12 Bcrypt

Bcrypt е алгоритъм за хеширане, който играе важна роля в сферата на киберсигурността, особено когато става въпрос за съхранението и управлението на пароли в информационни системи. Създаден през 1999 година от Niels Provos и David Mazieres, този алгоритъм е разработен с цел да осигури високо ниво на сигурност, предимно срещу сериозни заплахи като brute force атаки и използване на rainbow tables.

Основната идея зад bcrypt е да затрудни процеса на откриване или вземане на пароли от страна на злоумишленици чрез използване на различни атаки. За да постигне тази цел, алгоритъмът използва няколко важни концепции, като сол (salt) и work factor. Солта е случайно генерирана стойност, която се прибавя към паролата преди да бъде хеширана. Това предотвратява ефективното използване на предварително изчислени таблиците с хешове (rainbow tables), като прави атаката практически невъзможна. Освен това, work factor или броят итерации, които се използват по време на процеса на хеширане, може да бъде настроен от потребителя. Повечето алгоритми за хеширане обикновено изискват фиксиран брой итерации, но в случая с bcrypt потребителят има контрол върху този процес. Повишаването на броя итерации значително затруднява атаките с brute force, като увеличава времето, необходимо за обработка на паролите.

Освен това, bcrypt използва ключ за разширяване, който увеличава сложността на хеша и прави трудно предсказването на изхода от функцията за хеширане. Този ключ за разширяване е още един механизъм защита срещу различни видове атаки, които биха могли да се опитат да компрометират сигурността на системата.

Важно е да се подчертае, че въпреки че bcrypt е мощен алгоритъм за хеширане, използването му не гарантира абсолютна сигурност на паролите. Правилната конфигурация и използване на алгоритъма са от съществено значение за гарантиране на максимално ниво на защита. Той е широко поддържан в различни програмни езици и е предпочитан инструмент от разработчиците за сигурно съхранение на пароли в различни приложения и системи. В проекта Bcrypt се използва за хеширане на пароли в датабазата и премахването на сензитивна информация от там, където не трябва да бъде виждана и по този начин да се даде поверителност на потребителя в частта от сайта, която само администратор може да види.

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

Описанието е генерирано автоматично

**Фигура 11 – Описание на процеса на хеширане на парола**

## 2.1.13 Nodemailer

Nodemailer е изключително мощна и широко използвана библиотека за Node.js, която играе критична роля във всякакъв вид сървърно базирани приложения и уебсайтове. Тази библиотека предоставя надеждно и гъвкаво средство за изпращане на електронни писма, което позволява на разработчиците да създават и управляват електронна комуникация от всякакъв вид приложения.

Основната функционалност на Nodemailer включва възможността за изпращане на електронни писма чрез различни протоколи и транспортни механизми, включително SMTP, Sendmail, Amazon SES, Mailgun, SendGrid и други. Това осигурява голяма гъвкавост и възможност за изпращане на писма чрез различни доставчици на услуги за изпращане на имейли в зависимост от нуждите на проекта.

Библиотеката предоставя и широк набор от функции за конфигуриране на електронните писма, включително възможности за прикачване на файлове към съобщенията, поддръжка на текстово и HTML форматиране на съдържанието, създаване на персонализирани шаблони за писмата и други. Това прави Nodemailer идеален инструмент за създаване на разнообразни видове електронна комуникация, включително потвърждения за регистрация, забравени пароли, известия за поръчки и други.

Освен това, Nodemailer е изключително лесен за използване и интеграция във всякакъв вид проекти. Благодарение на добре документираното API и обширните възможности за конфигуриране, разработчиците могат бързо и лесно да внедрят електронната комуникация в своите приложения без значителни усилия.

В същото време, Nodemailer е надежден и сигурен инструмент за изпращане на електронни писма, който предлага вградени механизми за контрол на сигурността и защита от злоупотреби. Това го прави предпочитан избор за разработчиците, които ценят надеждността и безопасността на своите приложения. В проекта Nodemailer се използва за изпращане на имейли след завършена поръчка и поддържане на комуникация с клиента през електронна поща. Конфигуриран е да изпраща имейл на човека, направил поръчката и да го уведомява, че поръчката е приета и се обработва.

Картина, която съдържа Шрифт, лого, Графика, графичен дизайн

Описанието е генерирано автоматично

**Фигура 12 – Лого на Nodemailer**

## 2.1.14 dotenv

dotenv е инструмент, който играе критична роля в разработката на софтуер, особено в средата на Node.js. В контекста на уеб и сървърно базираните приложения, той предоставя мощен и гъвкав механизъм за управление на променливи от средата, които са от съществено значение за функционирането и конфигурирането на приложението.

В света на съвременната разработка на софтуер, особено в средите, които се развиват и разширяват бързо, е от съществено значение да се поддържа разграничение между кода на приложението и чувствителните данни като пароли, API ключове, токени и други конфиденциални настройки. dotenv решава този проблем, като позволява на разработчиците да съхраняват такива данни във външни конфигурационни файлове, наречени .env файлове.

С помощта на dotenv, променливите от средата, дефинирани в .env файлове, могат лесно да се заредят и достъпят в приложението по време на неговото стартиране. Това осигурява гъвкавост при управлението на конфигурацията, тъй като позволява лесно прехвърляне между различни среди на разработка, като се осигурява съществуващо средно състояние между разработчиците и средите на експлоатация.

Безопасността е също важен аспект от ролята на dotenv. Чрез съхраняването на чувствителните данни във външни файлове, които не се включват във вашия код, dotenv помага за предотвратяване на нежелан достъп до тези данни. Това не само подобрява сигурността на приложението, но също така предотвратява случайното разкриване на конфиденциални данни в публични репозитории или при споделянето на кода с други разработчици.

В крайна сметка, dotenv не само предоставя удобен и безопасен начин за управление на конфигурацията на вашето приложение, но също така подобрява процеса на разработка и поддръжка, като улеснява управлението на чувствителните данни и гарантира, че те се използват правилно и безопасно. В проекта .env се използва за поддържане на сигурността на данните, използвани при регистрация в уебсайта, като е отговорен за пазенето на имейла и паролата на всеки един създаден акаунт. Също така, .env се използва за съхраняване на чувствителни данни, като тайния ключ за AWS бъкета, в който се съхраняват снимките за целия проект.

Картина, която съдържа жълто, Шрифт, Графика, лого

Описанието е генерирано автоматично

**Фигура 13 – Лого на .env**

## ***2.2. Реализация на приложението***

Приложението и неговите функционалности, макар и лесни, дават много добра идея за силата на уеб технологиите и представените от тях възможности. В едно такова приложение се дава пример за front-end технологии и разработка на тях, както и работа с backend(сесии, бази данни, автентикация на потребители, хостинг и др.). Проектът служи за една добра основа и начална точка за разработчиците на уеб среда.

## ***2.3. Поставени цели за проекта***

- Намиране на актуална тема и нуждата от софтуер, който да е необходим

- Избор на лесни и достъпни за поддръжка технологии

- Избор на backend за приложението

- Избор на сигурен и защитен начин за регистрация и съхраняване на данни на потребителите

- Запознаване с динамичен уеб дизайн

- Проектиране на динамичен уеб дизайн

- Публикуване на приложението

## ***Ръководство на потребителя***

## 2.4.1 Ръководство за потребител, който ще използва сайта за собствен девелопмент

Когато приложението бъде изтеглено от гитхъб, първо трябва да се уверим, че имаме инсталиран npm. За да направим това, влизаме в терминала на IDE-то, което използваме и пишем следната команда:

rm -rf node\_modules; npm install

Ако това не сработи, значи нямате изтеглен Node.JS. За да направим това, влизаме в сайта на Node.JS и теглим последната версия. След като е готово, рестартираме IDE-то си и пишем наново командата по-горе. Това ще изтегли npm пакетите в директорията, в която сте запазили проекта си. След като това е готово е ред да пуснем сървъра. Отиваме в терминала и пишем следната команда:

* node server.js

След като тази команда бъде инициализирана приложението ще изкара следния отговор:

* listening on port:3000

Когато получите това съобщение, значи сървърът е инициализиран. Ако все още не го получавате, трябва да промените Firebase админ файла (фигура 15) на Ваш собствен такъв. След което трябва да промените AWS bucket и AWS region променливите в сървъра (вижте фигура 14).



**Фигура 14 – Данни за AWS S3 Bucket**

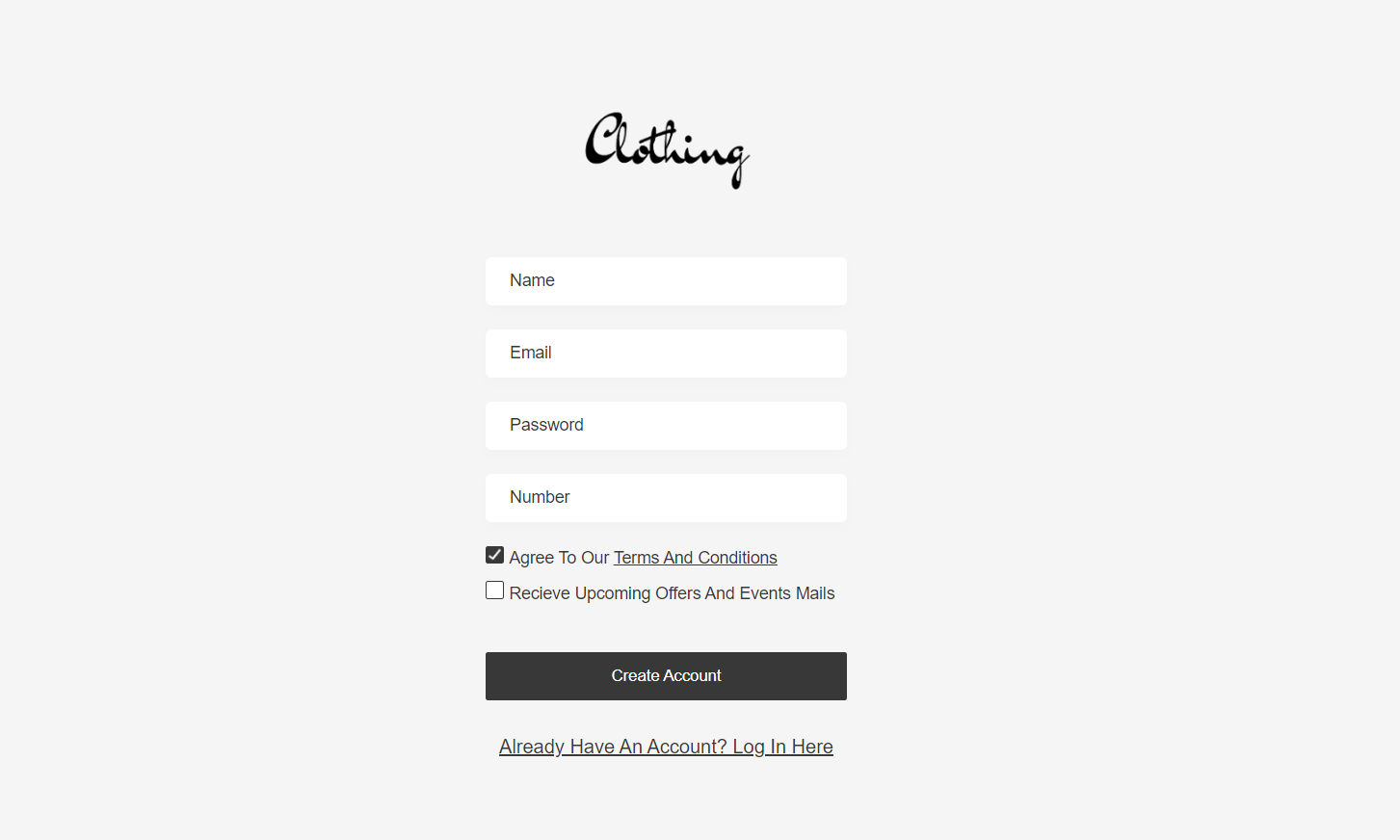


**Фигура 15 – Администраторски файл за Firebase**

След като всички тези промени бъдат направени сайта ще бъде пуснат на локалхост и ще можете да правите желаните и необходими за вас промени.

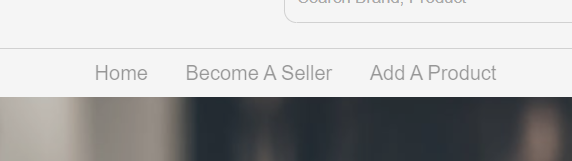
## 2.4.2 Ръководство за потребител, който ще бъде клиент на сайта

Когато влезете в сайта, първото нещо, което ще се изисква от вас е да направите акаунт, ако случайно вече имате такъв, да влезете в него.

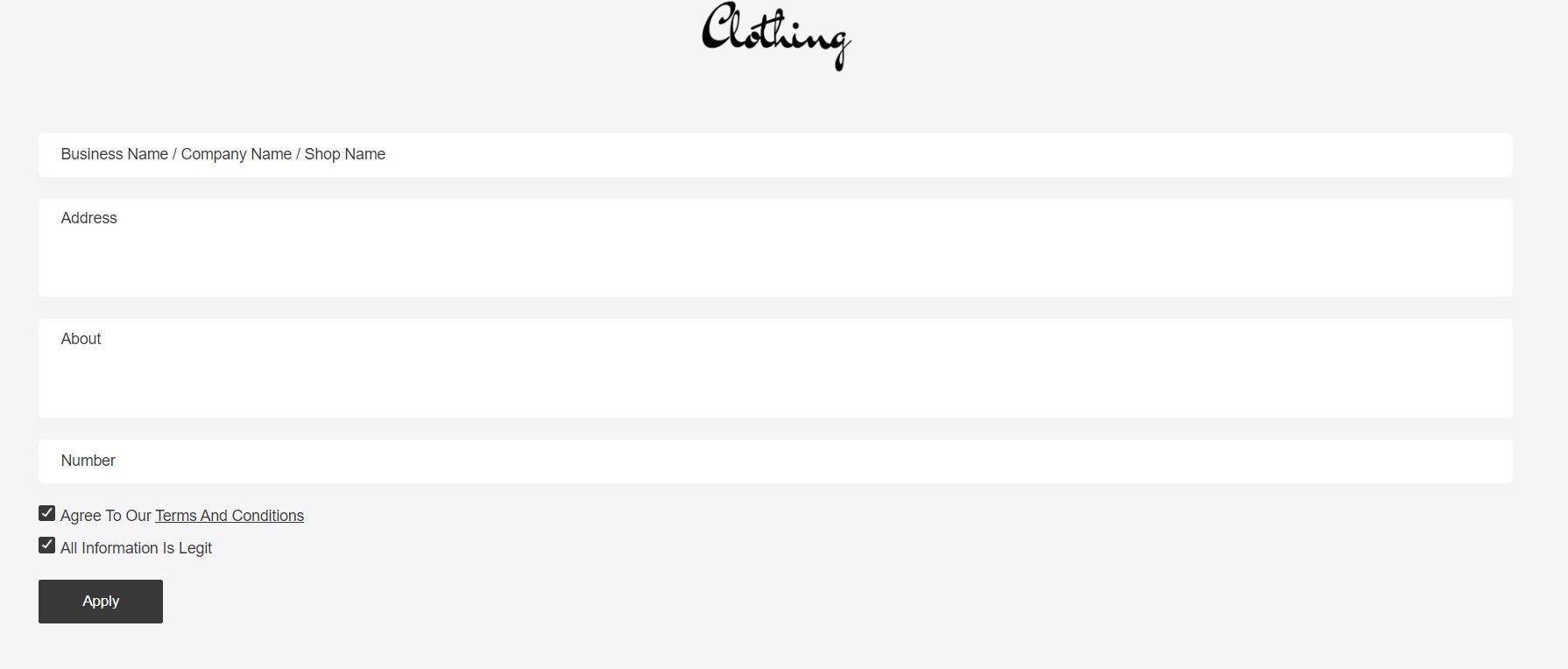


**Фигура 16 - Форма за регистрация**

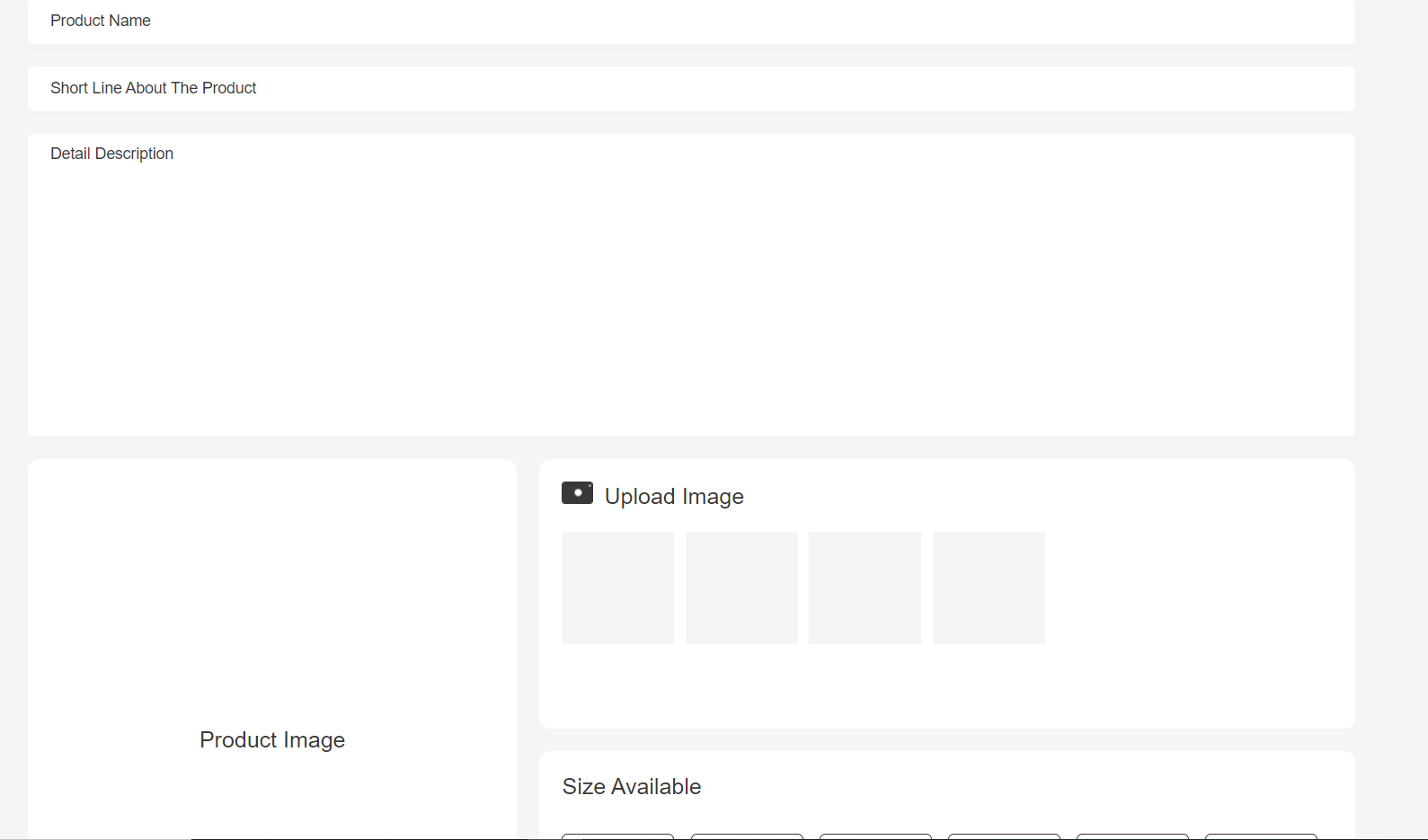
След което в навигационното меню горе в средата има бутон “become a seller” (виж фигура 17). След като натиснете този бутон от вас ще се иска да вкарате име, адрес, кратка информация за вас и телефонен номер(виж фигура 18). След като сте станали продавач, можете да добавяте продукти със снимки, описание, цена, налични размери и бройки(виж фигура 19).



**Фигура 17 – Навигационното меню в сайта**



**Фигура 18 - Форма за ставане на продавач**



**Фигура 19 – Форма за добавяне на продукт**

# Заключение

Заключението на дипломната документация, фокусирано върху разработването на уебсайт за препродажба на дрехи, представя изчерпателен обзор на проекта, включително неговите ключови аспекти, използвани технологии, достигнати цели, значимостта му в контекста на съвременната електронна търговия и перспективите за бъдещо развитие.

Процесът на създаване на уебсайт за препродажба на дрехи е подложен на строг анализ и изследване на съвременните тенденции в модната индустрия и онлайн търговията. Чрез използване на съвременни технологии и инструменти за уеб дизайн и програмиране, успяхме да създадем платформа, която не само е функционална и лесна за използване, но и отговаря на високите очаквания на потребителите в тази област.

Основните характеристики на уебсайта включват интуитивен интерфейс, който прави пазаруването лесно и приятно за потребителите, богат каталог с разнообразие от дрехи, възможности за персонализация на търсенето и филтриране на продуктите, както и сигурни методи за плащане и доставка.

В заключение, разработването на този уебсайт за препродажба на дрехи не само отговаря на съвременните изисквания и очаквания на пазара, но и отваря нови възможности за развитие и растеж в областта на електронната търговия. Вярваме, че този проект има потенциал да се превърне в успешен бизнес и да създаде стойност както за компанията, така и за нейните клиенти.

# Информационни източници

JavaScript - <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

Html - <https://developer.mozilla.org/en-US/docs/Web/HTML>

CSS - <https://developer.mozilla.org/en-US/docs/Web/CSS>

Node.JS - <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction>

Dotenv - <https://www.npmjs.com/package/dotenv>

Nodemailer - <https://nodemailer.com/usage/>

Bcrypt - <https://blog.logrocket.com/password-hashing-node-js-bcrypt/>

Express - <https://devdocs.io/express/>

AWS-SDK - <https://docs.aws.amazon.com/sdk-for-javascript/>

Nodemon - <https://www.npmjs.com/package/nodemon>

Firebase - <https://firebase.google.com/docs>