



**НАЦИОНАЛНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ  
ПО КОМПЮТЪРНИ ТЕХНОЛОГИИ И СИСТЕМИ  
гр.ПРАВЕЦ при ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ**

# **ДИПЛОМНА РАБОТА**

**ДЪРЖАВЕН ИЗПИТ ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА  
ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ – ЧАСТ ПО ТЕОРИЯ НА  
ПРОФЕСИЯТА**

на

Ученик:

Росен Пламенов Климентов

ученик от **19225** (XII) клас

Тема:

**УЕБСАЙТ ЗА ФИТНЕС ПРОДУКТИ**

**ПРОФЕСИОНАЛНО НАПРАВЛЕНИЕ:** 481 КОМПЮТЪРНИ НАУКИ

**ПРОФЕСИЯ:** 481020 СИСТЕМЕН ПРОГРАМИСТ

**СПЕЦИАЛНОСТ:** 4810201 СИСТЕМНО ПРОГРАМИРАНЕ

Ученик: .....

Ръководител: .....

*Тенчо Гочев*

## Съдържание

Увод .....	4
Глава 1 Литературен обзор.....	5
I. Езици за разработка .....	6
<b>1.HTML</b> .....	6
<b>За какво се използва HTML?</b> .....	6
<b>2. CSS</b> .....	7
<b>Същност:</b> .....	7
.....	8
<b>Какво е Javascript?</b> .....	8
<b>Описание:</b> .....	8
<b>За какво се използва Javascript?</b> .....	8
<b>4.PHP</b> .....	9
.....	9
<b>Какво е PHP?</b> .....	9
II. Среда за разработка: .....	10
<b>1.Visual Studio Code</b> .....	10
<b>1.MySQL Workbench</b> .....	11
III.Технологии: .....	13
<b>1.Xampp</b> .....	13
<b>Компонентите на XAMPP:</b> .....	14
IV.Работни рамки: .....	16
<b>1. Bootstrap</b> .....	16
<b>Характеристика</b> .....	16
Глава 2 Теоретична част.....	17
I. Структура на директориите в проекта .....	17
II. Преглед на съществуващи подобни сайтове .....	18
III. Инсталиране и конфигуриране на уеб сървър за сайта.....	19
IV. Проектиране на база данни .....	20
V. Проектиране на уеб сайта.....	23
5.1.Връзка с база данни .....	23
5.2. Вход и Регистрация в сайта .....	24
5.2.1. Регистрация в сайта.....	24

5.2.2.Вход в сайта.....	26
5.2.3. eyeFunction() (login.js).....	28
5.3.Валидация за вход и използване на сесии .....	29
5.4.Добавяне на продукт към количка(addToCart.php).....	30
5.5.Завършване на поръчката (checkout.php).....	32
5.7.Добавяне на продукт от админ .....	36
(admin_panel_add_products.php) .....	36
6. Ръководство за потребителя .....	38
6.1.Потребител .....	38
6.2.Админ .....	43
Глава 3 Литературен обзор.....	45
Извод.....	45
Възможни подобрения и перспективи за развитие .....	45
Използвана литература .....	46

## Увод

Настоящата дипломна работа представлява разработката на уеб сайт за фитнес продукти и услуги, съчетаващ функционалност за заявки за тренировки с треньор, създаване на персонализирани фитнес диети и програми за трениране. Уеб сайт е създаден с използването на езиците **HTML, CSS, PHP и JavaScript**. Цялата система е направена с помощта на програмата **VISUAL STUDIO CODE**. Системата използва База Данни чрез **MySql**-поддръжка с помощта на програмата **MySQL Workbench**.

Целта на уеб сайта е да предостави на потребителите информация за различни фитнес продукти, включително хранителни добавки, спортни облекла и аксесоари, както и да предостави възможност за връзка с професионалисти за записване на тренировки с професионален треньор, създаване на персонализирани фитнес диети и програми за трениране.

Идеята на уеб сайта е да предостави удобна и лесна за използване платформа, където фитнес ентусиасти могат да намерят всички необходими ресурси за подобряване на своето здраве и физическа форма. Основната целева аудитория на уеб сайта са хора, които се интересуват от здравословния начин на живот, фитнес ентусиасти и спортисти, които търсят персонализирани решения за тренировки и хранене.

Уеб сайтът предлага следните основни функционалности:

1. Регистрация и вход на потребители.
2. Подаване на заявка за :
  - Тренировка с професионален треньор.
  - Създаване на персонализирани фитнес диети, включително препоръки за хранене и рецепти.
  - Създаване на програми за трениране, включително упражнения, серии и повторения.
3. Възможност за покупка на фитнес продукти през уеб сайта.

Идеята е уеб сайтът да предложи цялостно решение за подобряване на физическата форма и здравословния начин на живот на потребителите, като им предостави лесен и удобен достъп до различни фитнес услуги и ресурси и контакт с професионалисти.

## Глава 1 Литературен обзор

В настоящия проект са използвани следните технологии, езици и среда за разработка /I D E – Integrated Developement Enviroment/ и работни рамки /FrameWorks/:

Използвани технологии за разработка

I. Езици за разработка:

1. HTML
2. CSS
3. JAVASCRIPT
4. PHP

II. Среда за разработка:

1. VISUAL STUDIO CODE
2. MySQL Workbench

III. Технологии

1. XAMPP

IV. Работни рамки:

1. Bootstrap

# I. Езици за разработка

## 1.HTML



HTML (съкращение от термина на английски: HyperText Markup Language, произнасяно най-често като „ейч-ти-ем-ел“, в превод „език за маркиране на хипертекст“) е основният маркиращ език за описание и дизайн на уеб страници. [1] HTML е стандарт в интернет, а неговите стандарти се определят от международния консорциум W3C. Текущата версия на стандарта е HTML 5.0 (от 28 октомври 2014 г.), а предходната стабилна версия е HTML 4.1. Първото публично достъпно описание на HTML е документът „HTML тагове“, споменат за първи път в интернет от

Тим Бърнърс-Лий в края на 1991 г.. Той описва 18 те елемента, които съставляват оригиналния, сравнително прост HTML дизайн. С изключение на маркера за хипервръзка, те бяха силно повлияни от SGMLguid, вътрешен формат на документация, базиран на стандартния обобщен език за маркиране (SGML), в CERN. Единадесет от тези елементи все още съществуват в HTML 4.

### За какво се използва HTML?

HTML се използва за създаване на уеб страница. Описанието на документа става чрез специални елементи, наречени HTML елементи или маркери, които се състоят от етикети или тагове (HTML tags) и ъглови скоби (като например елемента `<h1>`). HTML елементите са основната градивна единица на уеб страниците. Чрез тях се оформят отделните части от текста на една уеб страница, като заглавия, цитати, раздели, хипертекстови препратки и т.н. Най-често HTML елементите са групирани по двойки. В повечето случаи HTML кодът е написан в текстови файлове и се хоства на сървъри, свързани към Интернет.

## 2. CSS



CSS (Cascading Style Sheets) е език за описание на стилове (език за стилови файлове, style sheet language) – използва се основно за описание на онлайн представянето на уеб базиран документ, който написан на език за маркиране. Най-често се използва допълнително към чистия HTML, но се прилага и върху XML уебстраници и документи. Спецификацията на CSS официално се поддържа от W3C. CSS още в началото на развитието на www започва да се добавя към стандартния HTML с цел да бъдат разделени съдържанието и структурата на уеб страниците отделно от тяхното визуално представяне. Преди стандартите за

CSS, установени от W3C през 1995 г., съдържанието на сайтовете и стила на техния дизайн са писани в една и съща HTML страницата. В резултат на това HTML кодът се превръща в сложен и нечетлив, а всяка промяна в проекта на даден сайт изисквала корекцията да бъде нанасяна в целия сайт страница по страница. Използвайки CSS, настройките за форматиране могат да бъдат поставени в един-единствен файл и тогава промяната ще бъде отразена едновременно на всички страници, които използват този CSS файл.

### Същност:

CSS позволява да се определя как да изглеждат елементите на една HTML страница – шрифтове, размери, цветове, фонове, и др. CSS кодът се състои от последователност от стилови правила, всяко от които представлява селектор, последван от свойства и стойности.

Например в следния CSS код: `p {font-size: 9pt;}` има едно правило. То се състои от селектора `p` и свойството `font-size`, на което е зададена стойност `9pt`. Това правило ще направи размера на шрифта във всички параграфи 9 точки.

## 3.JAVASCRIPT



### Какво е Javascript?

JavaScript е интерпретируем език за програмиране, разпространяван с повечето уеб браузъри. Поддържа обектно ориентиран и функционален стил на програмиране. Създаден е в Netscape през 1995 г. Най-често се прилага към HTML кода на интернет страница с цел добавяне на функционалност и зареждане на данни. Може да се ползва също за писане на сървърни скриптове JSON, както и за много други приложения. JavaScript не трябва да се бърка с Java, съпадението на имената е резултат от маркетингово решение на Netscape. Javascript е стандартизиран под името EcmaScript.

JavaScript е разработен първоначално от Брендан Айк под името Mocha, като по късно е преименуван на LiveScript и накрая на JavaScript. LiveScript е официалното име на езика когато за първи път бива пуснат в бета версиите на Netscape Navigator 2.0 през септември 1995 г., но е преименуван на JavaScript на 4 декември 1995 г. Oracle, SQLite и др.

### Описание:

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

### За какво се използва Javascript?

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



## 4.PHP



### Какво е PHP?

PHP е скриптов език върху сървърната (обслужваща) страна.[3] Той е език с отворен код, който е проектиран за уеб програмиране и е широко използван за създаване на сървърни приложения и динамично уеб съдържание. Автор на езика е канадецът от датски произход Размус Лердорф. PHP е рекурсивен акроним от PHP: Hypertext Preprocessor (като в самото начало има значение, дадено от създателите му, на Personal Home Page). Пример за PHP приложение е МедияУики – софтуерът, използван от Уикипедия.

### За какво се използва PHP?

PHP е скриптов език със синтаксис, базиран на C и Perl. Използва се предимно в интернет среда за изпълнение на широк кръг от услуги и е един от най-популярните езици за програмиране в интернет.

При поискване кодът, който е написан на PHP, се интерпретира от уеб сървър, на който е качен, и резултатът се връща на уеб браузър. Потребителят не може да види чистия PHP код, без да има достъп до самия файл, в който той е записан. По този начин се осигурява защитата. PHP файловете могат да съдържат текст, HTML, CSS, JavaScript и PHP код и имат разширение \*.php. Самият език е преносим на много изчислителни архитектури и операционни системи като GNU/Linux, UNIX, macOS, Windows. Съществуват множество модули (разширения) за PHP, които добавят различни функционалности и позволяват много по-бързо и ефективно разработване. Такива допълнителни функционалности към езика са:

- функции за обработка (създаване, редактиране) на изображения
- функции за работа с низове и регулярни изрази
- функции за работа с XML съдържание
- функции за работа със сокет (гнезда)
- функции за дата и час
- математически функции
- функции за управление на сесии и работа с бисквити

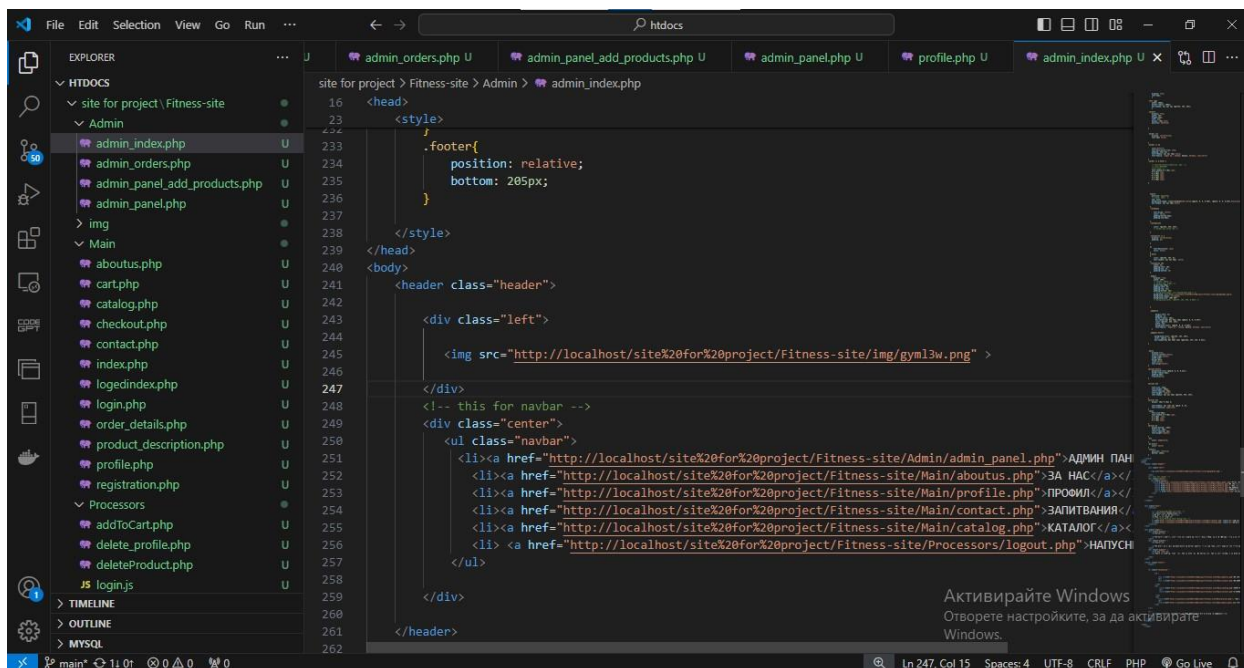
- функции за компресия и шифриране/дешифриране
- функции за COM и .NET за (Windows)
- функции за SOAP
- функции за работа с различни бази данни
- функции за работа с принтер
- функции за създаване на приложения с графичен потребителски интерфейс, базирани на библиотеката GTK
- функции за изпращане на e-mail съобщения
- хранилище за разширения и приложения на PHP: PEAR

PHP може да работи с повечето модерни данни – MySQL, PostgreSQL, Microsoft SQL Server, Oracle, SQLite и др. бази. На официалния сайт на PHP се намира обширна библиотека с информация за езика и модулите му, която може да се използва както за основно запознаване с езика, така и като справочник по време на работата с него.

## II. Среда за разработка:

### 1. Visual Studio Code

Visual Studio Code е редактор на програмен код за Windows, Linux и OS X. Това е първият редактор на Microsoft, който може да се ползва под Linux и macOS. Поддържа богат набор от инструменти за разработване като дебъгване, вграден Git Control, IntelliSense, „Side-by-Side Editing“ (позволява работа едновременно върху 2 файла отворени един до друг) и др. Той също така дава възможност за персонализиране, което означава, че потребителите могат да променят темата на редактора, клавишните комбинации, настройките и др. Все още не е известно дали потребителите ще могат да разширяват функциите на редактора чрез разработване на различни разширения, както е във Visual Studio. Редакторът е продукт на Microsoft и е безплатен, публично достъпен за преглед. Visual Studio Code е базиран на Electron, който е базиран на Chromium, използван да разгръща io.js приложения за десктопа. Visual Studio Code използва Blink layout engine, за да направи интерфейса (Фиг. 1).



Фиг. 1 Интерфейс на VISUAL STUDIO CODE

## 1. MySQL Workbench

MySQL Workbench е графичен инструмент (Фиг.2) за управление на бази данни, предназначен за проектиране, моделиране, разработка и администриране на MySQL бази данни. Един от ключовите аспекти на MySQL Workbench е, че обединява различни функционалности в един интуитивен интерфейс, което прави работата с базите данни по-лесна и ефективна.

Ето някои от основните функции и възможности на MySQL Workbench:

### 1. Проектиране и моделиране на бази данни:

- MySQL Workbench предоставя визуален редактор за проектиране на бази данни, който позволява на потребителите да създават и управляват различни аспекти на структурата на базата данни.
- Потребителите могат да дефинират таблиците в базата данни, атрибутите и техните типове данни, връзките между таблиците (външни ключове) и ограниченията (например уникалност, NULL стойности и други).
- Визуалният редактор позволява лесно променяне и модифициране на структурата на базата данни чрез графичен интерфейс.

## 2. **Разработка на бази данни:**

- MySQL Workbench включва SQL редактор, който позволява на потребителите да създават, редактират и изпълняват SQL заявки и скриптове директно във вградения текстов редактор.
- Потребителите могат да създават нови бази данни, таблиците в тях, изгледи, процедури, тригери и събития чрез изпълнение на съответните SQL команди.
- Редакторът поддържа синтаксисно оцветяване и автоматично допълване на кода, което улеснява писането и редактирането на SQL скриптове.

## 3. **Администриране на бази данни:**

- MySQL Workbench предоставя различни инструменти за администриране на бази данни, включително управление на потребителите и техните права за достъп, мониторинг на състоянието на сървъра и изпълнение на административни задачи като резервно копиране и възстановяване на данни.
- Интерфейсът за администриране на бази данни позволява на потребителите да изпълняват различни административни задачи чрез графичен интерфейс, който предоставя лесен и интуитивен начин за управление на базите данни.

## 4. **Визуализация и анализ на данни:**

- MySQL Workbench предоставя възможности за визуализация на данните в базите данни чрез различни инструменти като диаграми на базите данни, отчети и графики.
- Визуализациите могат да бъдат генерирани автоматично от базата данни или да бъдат ръчно създадени от потребителя чрез интуитивния графичен интерфейс на MySQL Workbench.

Хостът на сървъра в MySQL Workbench е адресът или името на сървъра, към който се свързвате за достъп до базата данни. Това може да бъде IP адрес или домейн име, съчетано с номер на порт (по подразбиране, портът за MySQL е 3306).

Базите данни в MySQL Workbench представляват организирани колекции от данни, които се съхраняват и обработват във формат, удобен за управление и използване.

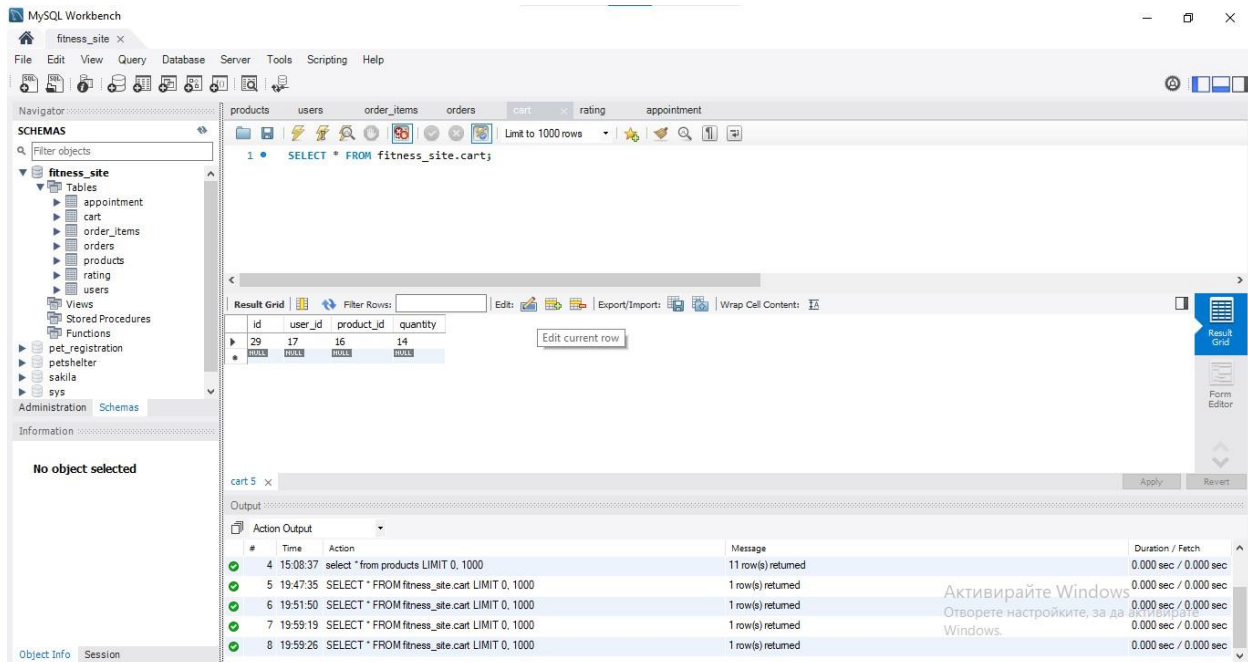
Таблицы в MySQL Workbench са структурирани съвкупности от данни, организирани в редове и колони. Всяка таблица има уникално име и определени колони със специфични типове данни.

Изгледи (views) представляват виртуални таблици, съдържащи резултатите от заявки или изчисления върху реални таблици в базата данни.

Процедурите в MySQL Workbench са съхранени програми, които изпълняват серия от инструкции за манипулиране на данни в базата данни.

Тригерите в MySQL Workbench са инструкции, които се активират автоматично при определени събития, като вмъкване, обновяване или изтриване на ред от таблица.

Събитията в MySQL Workbench са действия, които се изпълняват в определен момент във времето, като например изпълнение на определена SQL заявка.



Фиг.2 Интерфейсът на MySQL Workbench

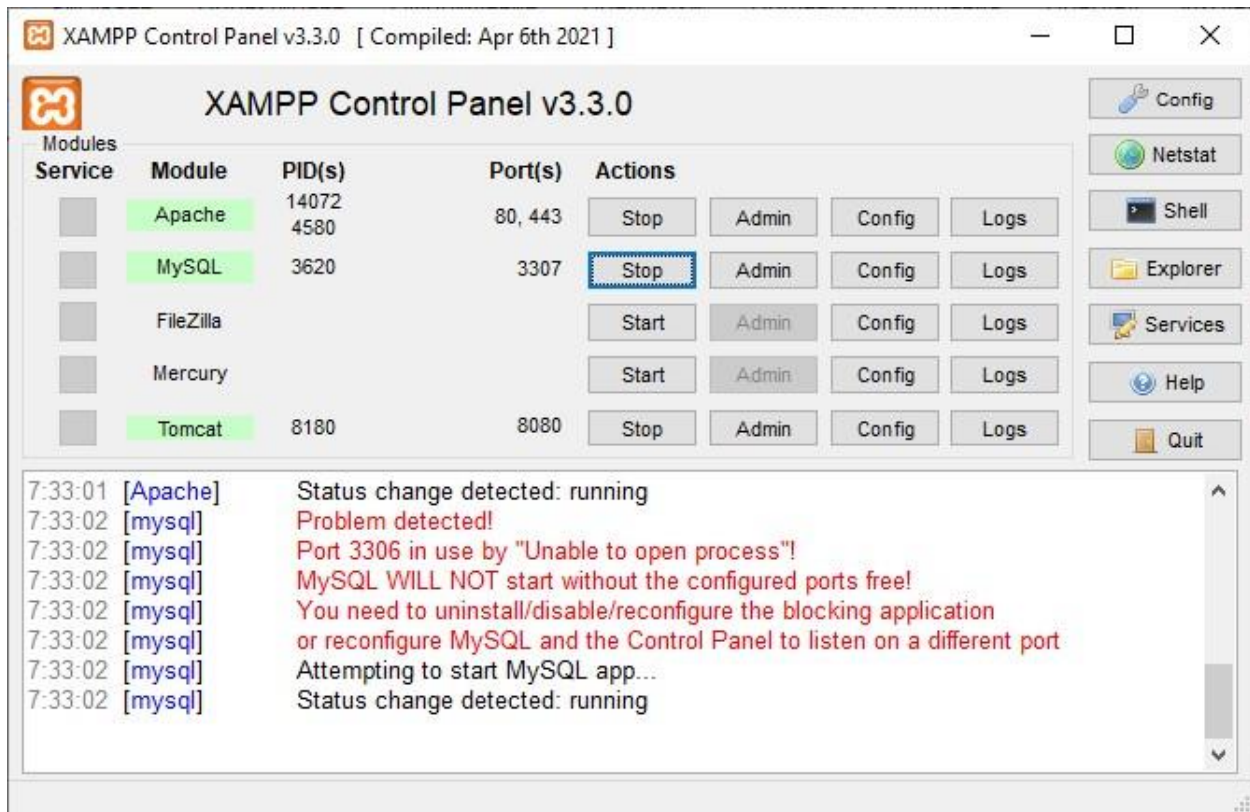
## III. Технологии:

### 1. Xampp



подобен вид.

ХАМРР е съкращение за междуплатформени, Apache, MySQL, PHP и Perl и ви позволява да създавате сайт офлайн, на локален уеб сървър на вашия компютър (фиг.3.1). Това просто и леко решениеработи на Windows, Linux и Mac – отгук и „кросплатформената“ част. Може да бъде срещнато и под наименование WEB-Stack, което не е синоним на конкретното използвания набор, а по-скоро обобщаващо наименование за всички пакети от



Фиг.3 1 XAMPP контрол панел.

Тъй като не е самостоятелно приложение, XAMPP предоставя два основни компонента за неговото инсталиране – Apache, който се използва за създаване на локалния сървър, и MySQL, който можете да използвате като база данни за вашия уебсайт.

Може би се чудите защо и как разработчиците използват локален сървър . Отговорът е прост – позволява им да създадат локално копие на сайта, в което могат да изпробват нови актуализации на плъгини, преди да ги внедрят в неговата версия на живо. По този начин те могат да предотвратят и открият потенциални грешки и проблеми, които могат да възникнат.

## Компонентите на XAMPP:

**1. Apache** е най-широко използваният софтуер за уеб сървър. Разработен и поддържан от Apache Software Foundation, Apache е софтуер с отворен код, достъпен безплатно. Той работи на 67% от всички уеб сървъри в света. Той е бърз, надежден и сигурен. Той може да бъде силно персонализиран, за да отговори на нуждите на много различни среди чрез използване на разширения и модули.

Така че основно уеб сървърът е софтуерът, който получава вашата заявка за достъп до уеб страница. Той изпълнява няколко проверки за сигурност на вашата HTTP заявка и ви отвежда до уеб страницата. В зависимост от страницата, която сте заявили, страницата може да поиска от сървъра да стартира няколко допълнителни модула, докато генерира документа, който да ви служи. След това ви връчва искания от вас документ.

**2. MySQL** е система за управление на база данни. [4]Базата данни е структурирана колекция от данни. Това може да бъде всичко отобикновен списък за пазаруване до галерия със снимки или огромни количества информация в корпоративна мрежа. За добавяне, достъп и обработка на данни, съхранявани в компютърна база данни, ви е необходима система за управление на бази данни като MySQL Server. Тъй като компютрите са много добри в боравене с големиколчества данни, системите за управление на бази данни играят централна роля визчисленията, като самостоятелни помощни програми или като части от други приложения.

### **MySQL бази данни са релационни.**

Релационната база данни съхранява данни в отделни таблици, вместо да поставя всички данни в едно голямо хранилище. Структурите на базата данни саорганизиранни във физически файлове, оптимизирани за скорост. Логическият модел, собекти като бази данни, таблици, изгледи, редове и колони, предлага гъвкава среда запрограмиране. Вие задавате правила, управляващи връзките между различни полета сданни, като едно към едно, едно към много, уникални, задължителни или незадължителни и „ указатели “ между различни таблици. Базата данни налага тези правила, така че с добрепроектирана база данни, вашето приложение никога не вижда непоследователни ,дублирани, остарели, остарели или липсващи данни.

**SQL** частта на “ MySQL ” означава “ Structured Query Language ” . SQL е най разпространеният стандартизиран език, използван за достъп до бази данни. В зависимост от вашата програмна среда, можете да въведете SQL директно (например за генериране наотчети), да вградите SQL изрази в код, написан на друг език, или да използвате специфичен за език API, който скрива синтаксиса на SQL.

### **Софтуерът MySQL е с отворен код.**

Отвореният код означава, че е възможно всеки да използва и променя софтуера. Всеки може да изтегли софтуера MySQL от Интернет и да го използва, без да плаща нищо.

**MySQL Database Server** е много бърз, надежден, мащабируем и лесен за използване.

MySQL Server може да работи удобно на настолен компютър или лаптоп, заедно с други ваши приложения, уеб сървъри и т.н., изисквайки малко или никакво внимание. Ако посветите цяла машина на MySQL, можете да коригирате настройките, за да се възползвате от цялата налична памет, мощност на процесора и I/O капацитет. MySQL може също да се мащабира до клъстери от машини, свързани в мрежа. MySQL Server първоначално е разработен за обработка на големи бази данни много по-бързо от съществуващите решения и успешно се използва в силно взискателни производствени среди от няколко години. Въпреки че е в непрекъснато развитие, MySQL Server днес предлага богат и полезен набор от функции. Неговата свързаност, скорост и сигурност правят MySQL Server изключително подходящ за достъп до бази данни в Интернет.

## IV.Работни рамки:

### 1. Bootstrap



Bootstrap е безплатна CSS рамка с отворен код, насочена към отзивчиво, ориентирано към мобилни устройства уеб програмиране отпред. Той съдържа HTML, CSS и (по избор) базирани на JavaScript шаблони за дизайн за типография, формуляри, бутони, навигация и други компоненти на интерфейса.

### Характеристика

Bootstrap е HTML, CSS и JS библиотека, която се фокусира върху опростяването на разработването на информативни уеб страници (за разлика от уеб приложенията). Основната цел на добавянето му към уеб проект е да се приложат изборите на Bootstrap за цвят, размер, шрифт и оформление към този проект. Като такъв, основният фактор е дали отговорните разработчици намират тези избори по свой вкус.

Веднъж добавен към проект, Bootstrap предоставя основни дефиниции на стил за всички HTML елементи. Резултатът е единен външен вид за проза, таблици и елементи на

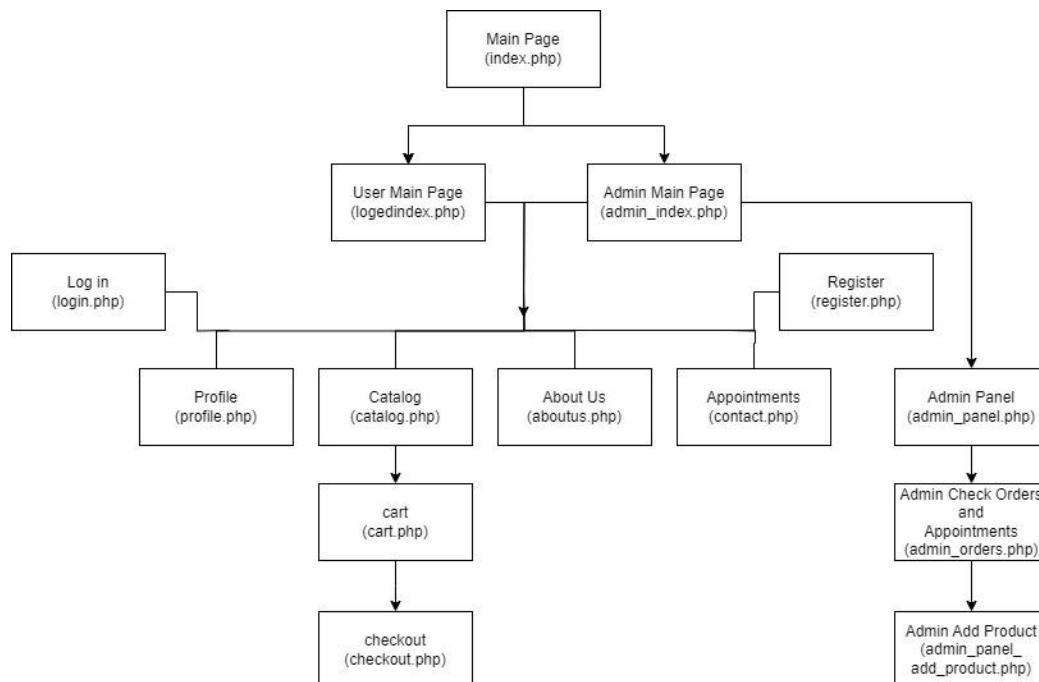


формуляри в уеб браузърите. В допълнение, разработчиците могат да се възползват от CSS класове, дефинирани в Bootstrap, за да персонализират допълнително външния вид на тяхното съдържание. Например, Bootstrap е предвидил таблици със светъл и тъмен цвят, заглавия на страници, по-забележими кавички за изтегляне и текст с подчертаване.

Bootstrap също идва с няколко компонента на JavaScript, които не изискват други библиотеки като jQuery. Те предоставят допълнителни елементи на потребителския интерфейс като диалогови прозорци, подсказки, ленти за напредъка, падащи менюта за навигация и въртележки. Всеки компонент на Bootstrap се състои от HTML структура, CSS декларации и в някои случаи придружаващ JavaScript код. Те също така разширяват функционалността на някои съществуващи интерфейсни елементи, включително например функция за автоматично попълване за полета за въвеждане.

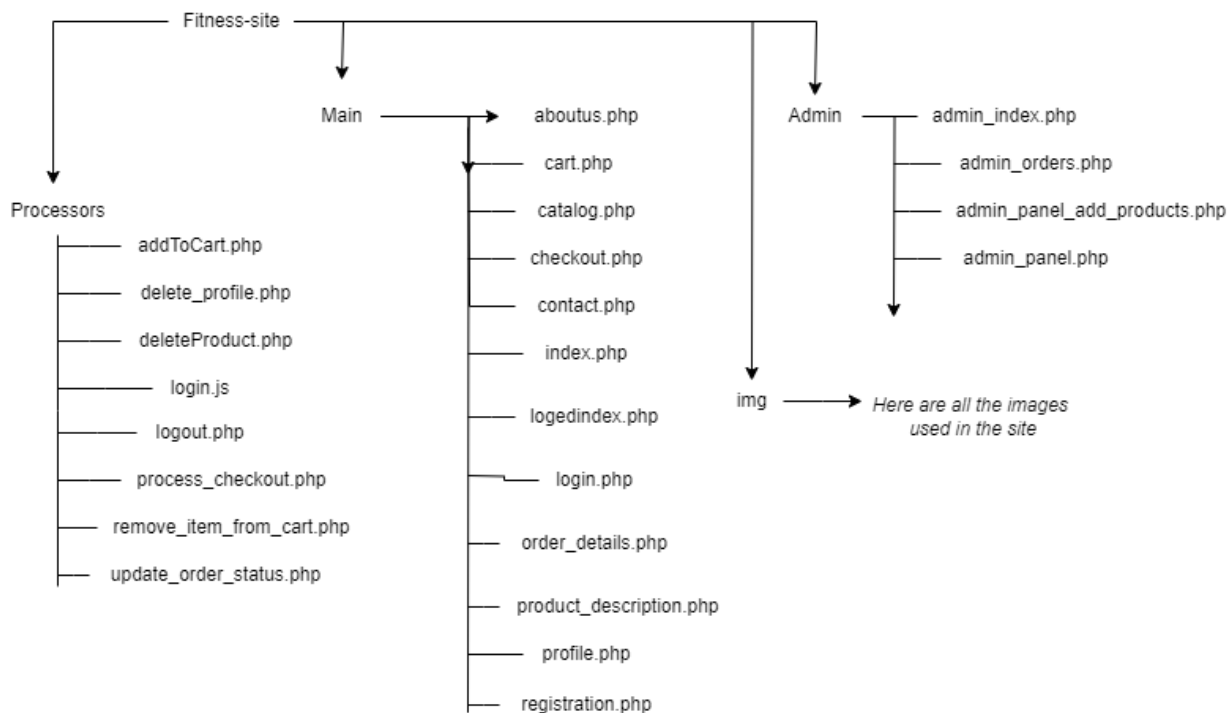
## Глава 2 Теоретична част

### I. Структура на директориите в проекта



Фиг. 4 Сitemap на проекта

Проекта е разположен в директория, като в нея се съдържа отделните файлове отговарящи за неговата работа, както следва :



Фиг.5 Структура на директориите в проекта

## II. Преглед на съществуващи подобни сайтове

Ето примерен анализ на някои български уеб сайтове, които предлагат фитнес продукти, услуги или информация:

1. **Fitnessi.bg** ([www.fitnessi.bg](http://www.fitnessi.bg))
  - Fitnessi.bg е платформа, която предоставя информация за фитнес и здравословен начин на живот на българския потребител. Сайтът включва статии, съвети, тренировъчни програми и продукти за фитнес и здраве.
2. **Fitshop.bg** ([www.fitshop.bg](http://www.fitshop.bg))
  - Fitshop.bg е онлайн магазин за фитнес уреди, аксесоари и хранителни добавки. Те предлагат разнообразие от продукти за домашно и професионално ползване, както и съвети и ръководства за тренировки.
3. **Flexshop.bg** ([www.flexshop.bg](http://www.flexshop.bg))

- Flexshop.bg е специализиран магазин за хранителни добавки и спортни храни. Сайтът предлага широка гама от продукти за спортисти и хора, които се интересуват от здравословния начин на живот.

### Анализ и заключение:

Прегледът на тези български уеб сайтове ни дава представа за тенденциите и предпочитанията на местната аудитория относно фитнес продуктите и услугите. Виждаме, че има разнообразие от магазини за фитнес оборудване и хранителни добавки, както и сайтове с информация и съвети за тренировки и здравословен начин на живот. Тази информация ни помага да определим какви функционалности и съдържание ще са подходящи за нашия уеб сайт за фитнес продукти в българския контекст.

## III. Инсталиране и конфигуриране на уеб сървър за сайта

Инсталирането и конфигурирането на уеб сървър за нашия уеб сайт чрез XAMPP е доста лесно и удобно. XAMPP е безплатен пакет за софтуер, който включва всичко необходимо за създаването на уеб сървър на нашето локално устройство. Ето стъпките, които трябва да следваме:

1. **Изтеглете XAMPP:** Посетете уеб сайта на XAMPP (<https://www.apachefriends.org/index.html>) и изтеглете най-новата версия за вашия операционна система (Windows, Linux, macOS).
2. **Инсталирайте XAMPP:** След като изтеглите инсталационния файл, стартирайте го и следвайте инструкциите на екрана за инсталиране на XAMPP на вашия компютър. По подразбиране, XAMPP ще бъде инсталиран в директория като "C:\xampp" (за Windows).
3. **Стартирайте XAMPP Control Panel:** След успешната инсталация отворете XAMPP Control Panel от стар меню или от инсталираната директория (по подразбиране "C:\xampp"). С този контролен панел можете да стартирате и спирате различните компоненти на вашия уеб сървър (Apache, MySQL, PHP и други).
4. **Стартирайте Apache и MySQL:** За да използвате уеб сървъра и базата данни, трябва да стартирате Apache и MySQL от XAMPP Control Panel. Просто кликнете върху бутона "Start" до съответните компоненти.
5. **Конфигурирайте вашия уеб сайт:** Сега можете да създадете ваш уеб сайт и да го разположите в директорията "htdocs" в инсталираната папка на XAMPP (по подразбиране "C:\xampp\htdocs"). Всяка директория или файл, който поставите тук, ще бъде достъпен през вашия уеб браузър чрез локален хост (например <http://localhost/mywebsite>).
6. **Тествайте вашия уеб сайт:** Отворете вашия уеб браузър и въведете "<http://localhost>" в адресната лента. Това ще ви даде достъп до началната страница

на вашия локален уеб сървър. Ако всичко е наред, можете да достъпите вашия уеб сайт и базата данни през ХАМРР.

## IV. Проектиране на база данни



Фиг.6. Таблиците в дата базата

column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
address	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
city	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
country	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
dob	date		NO			select,insert,update,references	
email	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
firstname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
gender	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
ID	int		NO			select,insert,update,references	auto_increment
is_admin	varchar(45)	0	NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
lastname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
mobile	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
password	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	

Фиг.6.1. Таблица users

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
appointment_date	date	curdate()	YES			select,insert,update,references	DEFAULT_GENERATED
category	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
comment	text		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
email	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
firstname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
idappointment	int		NO			select,insert,update,references	auto_increment
lastname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
mobile	varchar(15)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
user_id	int		YES			select,insert,update,references	

Фиг.6.2. Таблицата appointments

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
id	int		NO			select,insert,update,references	auto_increment
product_id	int		YES			select,insert,update,references	
quantity	int		YES			select,insert,update,references	
user_id	int		YES			select,insert,update,references	

Фиг.6.3 Таблицата cart

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
id	int		NO			select,insert,update,references	auto_increment
order_id	int		NO			select,insert,update,references	
product_id	int		NO			select,insert,update,references	
quantity	int		NO			select,insert,update,references	

Фиг. 6.4 Таблицата order\_items

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
email	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
rating	int		NO			select,insert,update,references	auto_increment
rating	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
user_id	int		YES			select,insert,update,references	

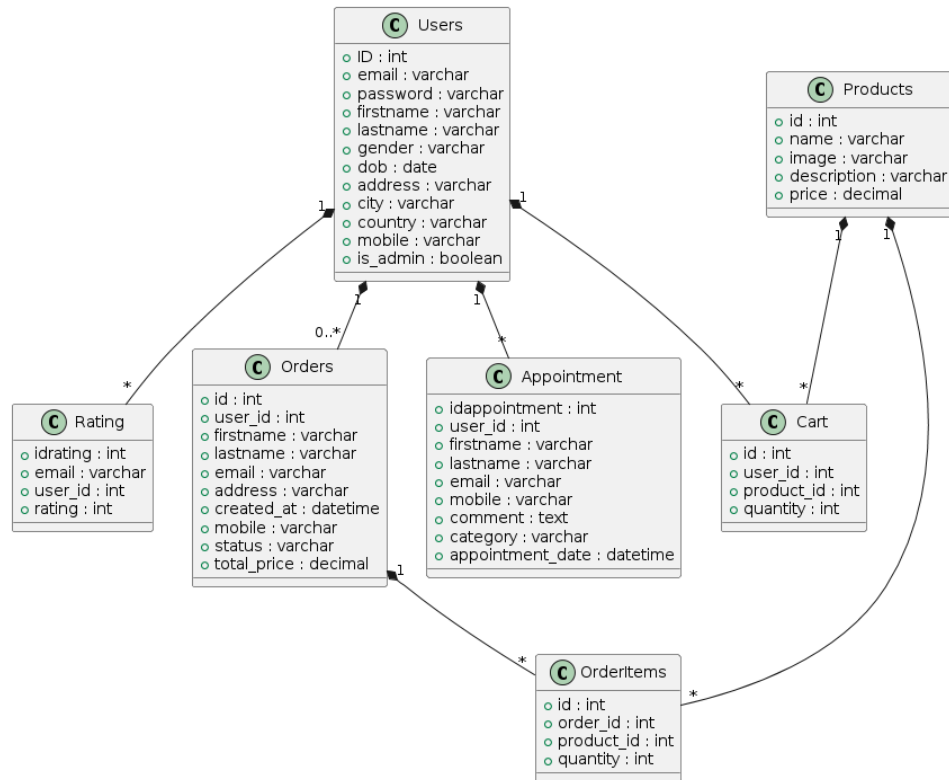
Фиг. 6.5 Таблицата rating

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
address	text		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
created_at	timestamp	CURRENT_TIMESTAMP	NO			select,insert,update,references	DEFAULT_GENERATED
email	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
firstname	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
id	int		NO			select,insert,update,references	auto_increment
lastname	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
mobile	varchar(20)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
status	enum('in progress','s... in progress		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
total_price	decimal(10,2)	0.00	NO			select,insert,update,references	
user_id	int		YES			select,insert,update,references	

Фиг. 6.6 Таблицата orders

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
description	text		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
id	int		NO			select,insert,update,references	auto_increment
image	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
name	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
price	decimal(10,2)		NO			select,insert,update,references	

Фиг. 6.7 Таблицата products



Фиг.6.8 Class диаграма на дата базата

На Фиг.6.8 са показани връзките между таблиците :

- Връзката между таблиците **Users** и **Orders**:
  - Всяка поръчка (**Orders**) принадлежи на един потребител (**Users**), следователно има външен ключ **user\_id** в таблицата **Orders**, който сочи към полето **ID** в таблицата **Users**.
- Връзката между таблиците **Orders** и **OrderItems**:
  - Всяка поръчка (**Orders**) може да съдържа множество артикули (**OrderItems**), затова в таблицата **OrderItems** има външен ключ **order\_id**, който сочи към полето **id** в таблицата **Orders**.
- Връзката между таблиците **Users** и **Rating**:
  - Всяка оценка (**Rating**) е дадена от един потребител (**Users**), следователно има външен ключ **user\_id** в таблицата **Rating**, който сочи към полето **ID** в таблицата **Users**.
- Връзката между таблиците **Users** и **Cart**:
  - Всяка покупка (**Cart**) е направена от един потребител (**Users**), следователно има външен ключ **user\_id** в таблицата **Cart**, който сочи към полето **ID** в таблицата **Users**.
- Връзката между таблиците **Users** и **Appointment**:

- Всяка запазена среща (**Appointment**) е направена от един потребител (**Users**), следователно има външен ключ **user\_id** в таблицата **Appointment**, който сочи към полето **ID** в таблицата **Users**.
6. Връзката между таблиците **Products** и **OrderItems**:
- Всяка поръчка артикул (**OrderItems**) е свързана с определен продукт (**Products**), затова има външен ключ **product\_id** в таблицата **OrderItems**, който сочи към полето **id** в таблицата **Products**.
7. Връзката между таблиците **Products** и **Cart**:
- Всеки артикул в кошницата (**Cart**) е свързан с определен продукт (**Products**), затова има външен ключ **product\_id** в таблицата **Cart**, който сочи към полето **id** в таблицата **Products**.

## V. Проектиране на уеб сайта

### 5.1.Връзка с база данни

```
// Database connection
$servername = 'localhost';
$username = 'root';
$dbpassword = '1234';
$dbname = 'fitness_site';

// Create a new mysqli instance
$conn = new mysqli($servername, $username, $dbpassword, $dbname);

// Check if the connection is successful
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

Този код се използва за създаване на връзка с базата данни чрез PHP и MySQLi (MySQL Improved) API. Да разгледаме всяка част от кода:

#### 1. Database connection parameters:

- Тук са дефинирани параметрите за връзката с базата данни: **\$servername**, **\$username**, **\$dbpassword** и **\$dbname**.
- **\$servername** указва името на сървъра, на който е хостната база данни. В нашият случай е 'localhost', което означава, че базата данни е хостната на същия компютър, на който се изпълнява скрипта.



- **\$username** и **\$dbpassword** са потребителското име и паролата, които се използват за влизане в базата данни.
  - **\$dbname** е името на базата данни, към която ще се осъществи връзката.
2. **Creating a new mysqli instance:**
- След като са дефинирани параметрите за връзката, се създава нова инстанция на класа **mysqli**, като се предоставят параметрите за връзката - сървър, потребителско име, парола и име на базата данни.
  - Тази инстанция представлява връзката с базата данни, чрез която ще се изпращат и получават заявки.
3. **Checking if the connection is successful:**
- След създаването на връзката с базата данни се прави проверка дали връзката е успешна.
  - Ако възникне грешка при връзката (например, грешно потребителско име или парола), скриптът ще спре изпълнението си и ще се изведе съобщение за грешка.
  - Функцията **die()** се използва за прекратяване на изпълнението на скрипта, а **connect\_error** съдържа текста на грешката при възникване на такава.

## 5.2. Вход и Регистрация в сайта

### 5.2.1. Регистрация в сайта

```
$firstname = $lastname = $dob = $gender = $mobile = $email = $password = $address
= $city = $country = "";

if (isset($_POST["submit"])) {

    $firstname = $_POST["firstname"];
    $lastname = $_POST["lastname"];
    $dob = $_POST["dob"];
    $gender = $_POST["gender"];
    $mobile = $_POST["mobile"];
    $email = $_POST["email"];
    $password = $_POST["password"];
    $address = $_POST["address"];
    $city = $_POST["city"];
```



```

$country = $_POST["country"];

$mobileCheckSql = "SELECT * FROM users WHERE mobile = '$mobile'";
$mobileResult = $conn->query($mobileCheckSql);
if ($mobileResult->num_rows > 0) {

    echo "Mobile number already exists!";
} else {

    $emailCheckSql = "SELECT * FROM users WHERE email = '$email'";
    $emailResult = $conn->query($emailCheckSql);
    if ($emailResult->num_rows > 0) {

        echo "Email already exists!";
    } else {

        $hashedPassword = password_hash($password, PASSWORD_DEFAULT);

        $sql = "INSERT INTO users (firstname, lastname, dob, gender, mobile,
email, password, address, city, country)
            VALUES ('$firstname', '$lastname', '$dob', '$gender',
'$mobile', '$email', '$hashedPassword', '$address', '$city', '$country')";

        if ($conn->query($sql) === TRUE) {

            header("Location: http://localhost/site%20for%20project/Fitness-
site/Main/login.php");
            exit();
        } else {
            echo "Error: " . $sql . "<br>" . $conn->error;
        }
    }
}
}

```

Този код реализира логиката за регистрация на потребители във нашия уеб сайт за фитнес продукти. Да разгледаме всяка част от кода:

#### 1. Инициализация на променливи:

- Първоначално се инициализират променливите **\$firstname**, **\$lastname**, **\$dob**, **\$gender**, **\$mobile**, **\$email**, **\$password**, **\$address**, **\$city** и **\$country** с празни стойности. Това се прави, за да се уверим, че всички променливи са дефинирани преди да се извърши проверката за изпращане на формата.

#### 2. Проверка за изпращане на формата:

- След това се проверява дали формата е изпратена. Това става чрез проверка дали в глобалния масив `$_POST` съществува елемент с ключ "submit".
3. **Извличане на данни от формата:**
    - След като формата е изпратена, данните от формата се извличат от глобалния масив `$_POST` и се присвояват на съответните променливи.
  4. **Проверка за уникалност на мобилен номер и имейл адрес:**
    - След извличането на данните се прави проверка дали мобилният номер и имейл адресът, въведени от потребителя, вече съществуват в базата данни. За целта се изпълняват SQL заявки, които търсят записи в таблицата **users**, където мобилният номер или имейл адресът съвпадат с тези, въведени от потребителя.
  5. **Хеширане на паролата:**
    - Преди да бъде вмъкната в базата данни, паролата се хешира чрез функцията **password\_hash()**, за да се осигури повишена сигурност на потребителските данни.
  6. **Изпълнение на заявката за добавяне на потребител в базата данни:**
    - Ако мобилният номер и имейл адресът са уникални, тогава се изпълнява заявката за добавяне на нов потребител в базата данни. Ако заявката бъде успешно изпълнена, потребителят се пренасочва към страницата за вход в системата чрез функцията **header()**. Ако възникне грешка при изпълнението на заявката, се извежда съобщение за грешка.

### 5.2.2.Вход в сайта

```
$query = "SELECT * FROM users WHERE email = '$email'";
$result = $conn->query($query);

// Check if the login is successful
if ($result->num_rows > 0) {
    $user = $result->fetch_assoc();

    // Verify the password
    if (password_verify($password, $user['password'])) {
        // Set session variables to indicate successful login
        $_SESSION['logged_in'] = true;
        $_SESSION['email'] = $user['email'];

        // Check if the user is an admin
        if ($user['is_admin'] == 1) {
            $_SESSION['is_admin'] = true;
            header('Location:http://localhost/site%20for%20project/Fitness-site/Admin/admin_index.php');
        }
    }
}
```

```

        exit();
    } else {
        header('Location:http://localhost/site%20for%20project/Fitness-
site/Main/logedindex.php');
        exit();
    }
} else {
    echo "Invalid email or password.";
}
} else {
    echo "Invalid email or password.";
}
}

```

Този код реализира логиката за вход в системата на потребителите на вашия уеб сайт за фитнес продукти. Да разгледаме всяка част от кода:

1. **Изпълнение на SQL заявка:**
  - Създава се SQL заявка, която се използва за извличане на потребител със съответния имейл адрес от базата данни.
2. **Изпълнение на заявката и проверка за резултати:**
  - Заявката се изпълнява и се проверява дали има потребител със съответния имейл адрес в базата данни. Ако има, се извличат резултатите от заявката.
3. **Проверка на паролата:**
  - След като се извлекат данните за потребителя, се използва функцията **password\_verify()** за сравнение на предоставената от потребителя парола с хешираната парола в базата данни. Ако паролите съвпадат, влизането в системата се счита за успешно.
4. **Задаване на сесийни променливи:**
  - Ако входът е успешен, се задават сесийни променливи, които указват, че потребителят е влезнал в системата и кой е неговият имейл адрес. Също така се проверява дали потребителят е администратор, и ако е така, се задава сесийна променлива, която указва това.
5. **Пренасочване на потребителя:**
  - В зависимост от резултата от проверката за администратор, потребителят се пренасочва към съответната страница на сайта - административната панела или основната страница за потребителите.
6. **Обработка на грешки:**
  - Ако няма намерен потребител със съответния имейл адрес или ако паролата не съвпада с тази в базата данни, се извежда съобщение за невалиден имейл или парола.

### 5.2.3. eyeFunction() (login.js)

```
function eyeFunction()
{
    var x = document.getElementById("psw");
    if (x.type === "password")
    {
        x.type = "text";
        document.getElementById("color").style.color = "red";
    } else
    {
        x.type = "password";
        document.getElementById("color").style.color = "white";
    }
}

function eyeFunction2()
{
    var x = document.getElementById("psw-repeat");

    if (x.type === "password")
    {
        x.type = "text";
        document.getElementById("color2").style.color = "red";
    } else
    {
        x.type = "password";
        document.getElementById("color2").style.color = "white";
    }
}
```

Файлът **login.js** съдържа две JavaScript функции: **eyeFunction()** и **eyeFunction2()**. Тези функции се използват за превключване на видимостта на паролни полета във формите за вход и регистрация на потребители.

#### 1. Функция **eyeFunction()**:

- Тази функция се използва за превключване на видимостта на паролното поле за парола във формата за вход.

- Когато потребителят кликне върху иконата за око (👁), функцията променя типа на паролното поле от **"password"** на **"text"**, което позволява видимо представяне на въведената информация.
- Също така, цветът на текста се променя на червено, за да предупреди потребителя, че паролата се показва.
- Когато потребителят отново кликне върху иконата, типът на паролното поле се връща към **"password"**, а текстът се връща към бял цвят.

## 2. Функция **eyeFunction2()**:

- Тази функция има сходна функционалност с **eyeFunction()**, но се използва за превключване на видимостта на паролното поле за повторно въвеждане на парола във формата за регистрация.
- При клик на иконата за око, функцията променя видимостта на полето за повторно въвеждане на парола, а също и цвета на текста в зависимост от текущото състояние на полето.

Тези функции се използват във формите за вход и регистрация, като предоставят на потребителя възможността да види или скрие въведената парола за по-добра сигурност или удобство при въвеждане.

## 5.3.Валидация за вход и използване на сесии

```
session_start();

//Check if the user is already logged in
if (isset($_SESSION['email'])) {
    // Redirect the user to the appropriate page based on their role
    if (isset($_SESSION['is_admin']) && $_SESSION['is_admin']) {
        header('Location: http://localhost/site%20for%20project/Fitness-
site/Admin/admin_index.php');
        exit();
    }
}
```

Този код служи за защита на определени страници на вашия уеб сайт, като гарантира, че само потребителите, които са влезли в системата и имат права да видят съответната страница, имат достъп до нея. Ето обяснение на всяка част от кода:

### 1. **session\_start()**:

- Функцията **session\_start()** се използва, за да стартира сесията на потребителя, то се поставя в началото на всяка страница. Това е необходимо, за да се използват сесийни променливи за запомняне на влизането на потребителите в системата.

### 2. Проверка за вече влезнали потребители:

- Следващата част от кода проверява дали вече има създадена сесия за потребителя. Ако вече има създадена сесия (т.е. потребителят вече е влязъл в системата), то този код предотвратява повторно влизане и го пренасочва към съответната страница, в зависимост от неговата роля.

### 3. Пренасочване към съответната страница:

- Ако потребителят е влязъл в системата, кодът проверява дали той е администратор. Ако е администратор, той бива пренасочен към административната страница. Ако потребителят не е администратор, той бива пренасочен към основната страница за потребители.

### 4. Обработка на изход:

- След изпълнението на пренасочването, използва се функцията **exit()**, за да се предотврати изпълнението на останалия код на страницата след пренасочването. Това се извършва, за да се гарантира, че потребителят ще бъде пренасочен само към една страница и няма да може да достъпи други части от кода след това.

Кода за административните страници работи със същата логика, но с разликата, че ако потребителят не е администратор, той бива пренасочен към основната страница за потребители (loggedindex.php), вместо към административната панела.

## 5.4. Добавяне на продукт към количка (addToCart.php)

```
<?php
session_start();

// Check if the request contains the necessary parameters
if(isset($_POST["product_id"]) && isset($_POST["user_id"]) &&
isset($_POST["quantity"])) {
    // All parameters are present
    $productId = $_POST["product_id"];
    $userId = $_POST["user_id"];
    $quantity = $_POST["quantity"];

    // Insert or update the product in the cart table in the database (replace
    placeholders with your actual database connection and table name)
    $servername = "localhost";
    $username = "root";
    $password = "1234";
    $dbname = "fitness_site";

    $conn = new mysqli($servername, $username, $password, $dbname);
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }
}
```

```

    }

    // Check if the product already exists in the cart for the user
    $checkQuery = "SELECT * FROM cart WHERE user_id = '$userId' AND product_id = '$productId'";
    $checkResult = $conn->query($checkQuery);

    if ($checkResult->num_rows > 0) {
        // Product already exists in the cart, update the quantity
        $updateQuery = "UPDATE cart SET quantity = quantity + '$quantity' WHERE user_id = '$userId' AND product_id = '$productId'";
        if ($conn->query($updateQuery) === TRUE) {
            echo "Quantity updated successfully";
        } else {
            echo "Error updating quantity: " . $conn->error;
        }
    } else {
        // Product does not exist in the cart, insert a new row
        $insertQuery = "INSERT INTO cart (user_id, product_id, quantity) VALUES ('$userId', '$productId', '$quantity')";
        if ($conn->query($insertQuery) === TRUE) {
            echo "Product added to cart successfully";
        } else {
            echo "Error adding product to cart: " . $conn->error;
        }
    }

    $conn->close();
} else {

    // Output error message
    echo "Error: Missing parameters";
}
?>

```

Този код се използва за добавяне на продукт към количката във вашата уеб страница за пазаруване. Ето обяснение на логиката на кода:

1. **Проверка за наличие на необходимите параметри:**
  - Първоначално кодът проверява дали са предоставени всички необходими параметри за добавяне на продукт към количката. Тези параметри са **product\_id**, **user\_id** и **quantity**.
2. **Извличане на предоставените параметри:**

- Следващата стъпка е да извлечем стойностите на параметрите **product\_id**, **user\_id** и **quantity** от заявката.
3. **Свързване с базата данни:**
    - След това кодът се свързва с базата данни, като използва предоставените данни за сървъра, потребителско име и парола.
  4. **Проверка за наличие на продукта в количката:**
    - След като е установена връзката с базата данни, кодът изпълнява заявка за проверка дали продуктът вече съществува в количката на потребителя.
  5. **Актуализиране на количеството или добавяне на нов продукт:**
    - Ако продуктът вече съществува в количката, кодът актуализира количеството на продукта в количката, като добавя предоставеното количество към съществуващото. В противен случай, ако продуктът не е в количката, кодът добавя нов запис за продукта в базата данни.
  6. **Извеждане на съобщения за успешно добавяне или грешка:**
    - След като операцията приключи успешно или възникне грешка, кодът извежда съответно съобщение за успешно добавяне на продукт в количката или съобщение за грешка.
  7. **Затваряне на връзката с базата данни:**
    - Накрая, след като операцията е приключила, връзката с базата данни се затваря, за да се освободят ресурсите.

## 5.5.Завършване на поръчката (checkout.php)

```
<?php
session_start();

// Check if the request contains the necessary parameters
if(isset($_POST["firstname"]) && isset($_POST["lastname"]) &&
isset($_POST["email"]) && isset($_POST["mobile"]) && isset($_POST["address"])) {
    // All parameters are present
    $firstname = $_POST["firstname"];
    $lastname = $_POST["lastname"];
    $email = $_POST["email"];
    $mobile = $_POST["mobile"];
    $address = $_POST["address"];

    // Retrieve user ID from session or database (assuming it's stored in
$_SESSION['user_id'])
    $userId = $_SESSION['user_id'];

    // Include database connection
    $servername = "localhost";
$username = "root";
$password = "1234";
```



```

$dbname = "fitness_site";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// SQL query to retrieve cart items for the user
$sql = "SELECT * FROM cart WHERE user_id = $userId";
$result = $conn->query($sql);

// Initialize total price
$totalPrice = 0;

if ($result->num_rows > 0) {
    // Loop through each cart item and calculate total price
    while ($row = $result->fetch_assoc()) {
        $quantity = $row['quantity'];
        $productId = $row['product_id'];

        // Fetch product price from the products table
        $productSql = "SELECT price FROM products WHERE id = $productId";
        $productResult = $conn->query($productSql);

        if ($productResult->num_rows > 0) {
            $productRow = $productResult->fetch_assoc();
            $price = $productRow['price'];
            $totalPrice += $price * $quantity;
        }
    }
}

// Insert the order into the database
$conn->begin_transaction();

// Prepare and bind the SQL statement to insert order
$stmtOrder = $conn->prepare("INSERT INTO orders (user_id, firstname,
lastname, email, mobile, address, total_price) VALUES (?, ?, ?, ?, ?, ?, ?)");
$stmtOrder->bind_param("issssd", $userId, $firstname, $lastname, $email,
$mobile, $address, $totalPrice);

```

```

// Execute the statement to insert order
if ($stmtOrder->execute() === TRUE) {
    // Get the order ID of the inserted order
    $orderId = $conn->insert_id;

    // Prepare and bind the SQL statement to insert order items
    $stmtItems = $conn->prepare("INSERT INTO order_items (order_id,
product_id, quantity) VALUES (?, ?, ?)");
    $stmtItems->bind_param("iii", $orderId, $productId, $quantity);

    // Execute the statement to insert order items
    $result->data_seek(0); // Reset result set pointer
    while ($row = $result->fetch_assoc()) {
        $productId = $row['product_id'];
        $quantity = $row['quantity'];
        $stmtItems->execute();
    }

    // Commit the transaction
    $conn->commit();

    // Clear cart after successful order placement
    $stmtRemoveCart = $conn->prepare("DELETE FROM cart WHERE user_id = ?");
    $stmtRemoveCart->bind_param("i", $userId);
    $stmtRemoveCart->execute();
    echo "Order placed successfully";
    header("Location: http://localhost/site%20for%20project/Fitness-
site/Main/cart.php");
    exit();
} else {
    // Rollback the transaction if order insertion fails
    $conn->rollback();
    echo "Error placing order: " . $conn->error;
}

// Close statements and connection
$stmtOrder->close();
$stmtItems->close();
$stmtRemoveCart->close();
$conn->close();
} else {
    // Output error message
    echo "Error: Missing parameters";
}

```



Този код се използва за завършване на поръчката след като потребителят подаде своите данни за доставка. Ето обяснение на логиката на кода:

1. **Проверка за наличие на необходимите параметри:**
  - Кодът първоначално проверява дали са предоставени всички необходими параметри за завършване на поръчката. Тези параметри са **firstname**, **lastname**, **email**, **mobile** и **address**.
2. **Извличане на предоставените параметри:**
  - Следващата стъпка е да извлечем стойностите на параметрите от заявката за име, фамилия, имейл, телефон и адрес.
3. **Свързване с базата данни:**
  - Кодът се свързва с базата данни, използвайки предоставените данни за сървър, потребителско име и парола.
4. **Изчисляване на общата цена на поръчката:**
  - За да се изчисли общата цена на поръчката, кодът извлича продуктите от количката на потребителя и умножава техните цени по количеството.
5. **Вмъкване на поръчката в базата данни:**
  - Кодът стартира транзакция, за да осигури атомарност при вмъкването на поръчката и нейните продукти в базата данни.
  - Подготвя и свързва SQL заявката за вмъкване на поръчката в таблицата **orders**, като включва и общата цена на поръчката.
  - Изпълнява заявката и взема идентификатора на вмъкнатата поръчка.
  - Подготвя и изпълнява заявка за вмъкване на продуктите от количката в таблицата **order\_items**, свързвайки ги с вмъкнатата поръчка.
6. **Потвърждение на поръчката и изтриване на количката:**
  - Ако поръчката е добавена успешно, транзакцията се потвърждава, а данните в количката на потребителя се изтриват.
  - Потребителят се пренасочва към страницата за количка със съобщение за успешното завършване на поръчката.
7. **Обработка на грешки:**
  - Ако възникне грешка по време на вмъкването на поръчката или изтриването на количката, транзакцията се отменя, и се извежда съобщение за грешка.
8. **Затваряне на връзката с базата данни:**
  - Накрая, връзката с базата данни се затваря, за да се освободят ресурсите.

## 5.7. Добавяне на продукт от админ

(admin\_panel\_add\_products.php)

```
<?php
session_start();

// Check if the user is already logged in
if (isset($_SESSION['email'])) {
    // Redirect the user to the appropriate page based on their role
    if (isset($_SESSION['is_admin']) xor $_SESSION['is_admin']) {
        header('Location: http://localhost/site%20for%20project/Fitness-
site/Main/logedindex.php');
        exit();
    }
}

// Define a variable to hold the success message
$successMessage = "";

// Check if the form is submitted
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Process the form data and add the product to the database
    $imageFileName = $_FILES['image']['name']; // Get the name of the uploaded
file
    $imageTempName = $_FILES['image']['tmp_name']; // Get the temporary file name
on the server
    $name = $_POST['name'];
    $price = $_POST['price'];
    $description = $_POST['description'];

    // Move the uploaded file to the img folder
    $uploadDirectory = '..\img\';
    $targetFilePath = $uploadDirectory . basename($imageFileName);
    if (move_uploaded_file($imageTempName, $targetFilePath)) {
        // File uploaded successfully, proceed to insert data into database
        // Database connection parameters
        $servername = "localhost";
        $username = "root";
        $password = "1234";
        $dbname = "fitness_site";

        // Create connection
        $conn = new mysqli($servername, $username, $password, $dbname);
```

```

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Prepare SQL statement to insert product
$sql = "INSERT INTO products (image, name, price, description) VALUES (?, ?, ?, ?)";
$stmt = $conn->prepare($sql);

// Bind parameters and execute statement
$stmt->bind_param("ssds", $imageFileName, $name, $price, $description);
if ($stmt->execute()) {
    // Product added successfully, set success message
    $successMessage = "Успешно добавен продукт! &#128515; ";
} else {
    // Error occurred while adding product
    $successMessage = "Неуспешно добавен продукт! &#128546; ";
}

// Close statement and connection
$stmt->close();
$conn->close();
} else {
    $successMessage = "Неуспешно добавяне на снимката.Моля, опитай отново!";
}
}
?>

```

### 1. Проверка за аутентикация и права на достъп:

- Преди да се даде възможност на потребителя да добави нов продукт, се проверява дали той е влезнал в системата и дали е администратор. Това е важно за сигурността и защитата на данните.
- За да се осъществи тази проверка, се използва механизмът за сесии в PHP. Ако потребителят е влезнал в системата, сесията ще съдържа ключа **email**. Ако потребителят е администратор, сесията ще съдържа ключа **is\_admin**, който ще има стойност **true**.
- Ако сесията е валидна и потребителят е администратор, той ще има достъп до страницата за добавяне на продукт. Ако не е администратор, ще бъде пренасочен към стандартната страница след вход.

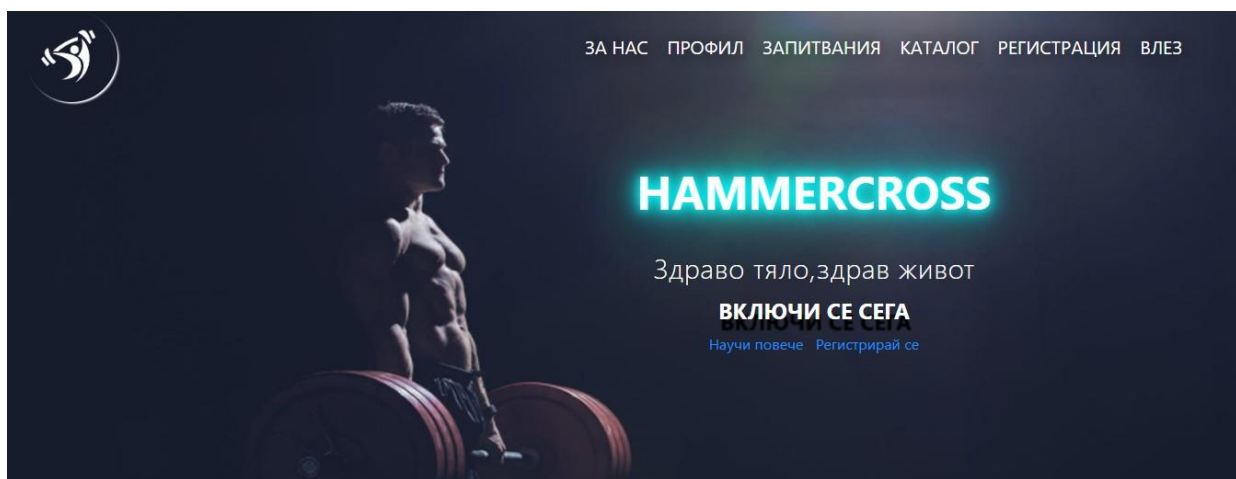
### 2. Обработка на формата за добавяне на продукт:

- Когато формата за добавяне на продукт бъде изпратена, PHP скриптът ще обработи заявката и ще извлече информацията за новия продукт от **\$\_POST**.
- Също така, скриптът ще получи информация за изображението на продукта от **\$\_FILES**. Това изображение се прехвърля към директорията, където се съхраняват изображенията на продуктите.
- След това скриптът ще изпълни SQL заявка за добавяне на новия продукт в базата данни. В заявката ще се включат данните за името на продукта, цената, описанието и пътят към изображението в съответните колони на таблицата за продукти.

Така се осигурява, че само администраторите могат да достъпват страницата за добавяне на продукти и че само те имат право да добавят нови продукти към системата. Ако потребителят е администратор и успешно добави нов продукт, той ще бъде уведомен за успешното добавяне. Ако възникне грешка по време на добавянето, потребителят ще бъде информиран за неуспеха.

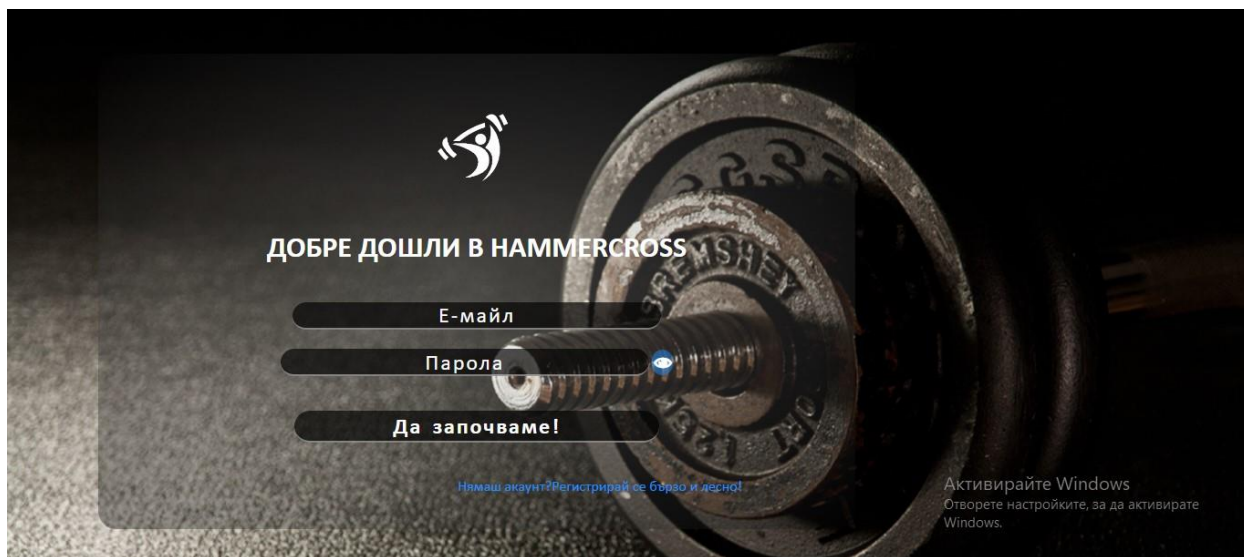
## 6. Ръководство за потребителя

### 6.1.Потребител

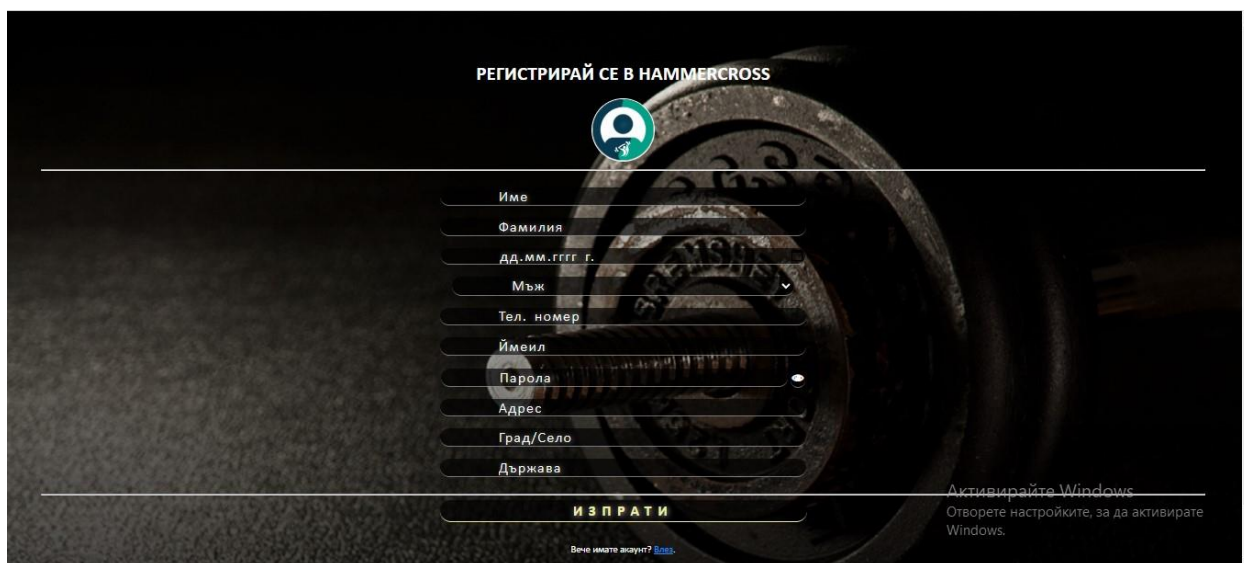


Фиг. 7.1. Начална страница ,за не влезли или регистрирани потребители.

Това е началната страница на сайта, от тук потребителите трябва да изберат дали да се регистрират ,ако нямат акаунт или да влезнат , ако вече имат съществуващ такъв ,за да имат достъп до останалите страници в сайта.



Фиг.7.2. Вход в сайта.



Фиг.7.3. Регистрация в сайта в сайта.



Фиг.7.4.Страница за запитвания.

От тук потребителят може да направи запитване за тренировка с треньор, създаване на персонална фитнес диета, създаване на персонален фитнес режим за тренировки или запитване за нещо друго свързано с фитнес или по сайта. Също така има и нашите контакти.

First Name	Last Name	Mobile	Total Price	Status	Created At
Пламен	Климентов	0895757182	180.00	in progress	2024-04-28 21:15:30

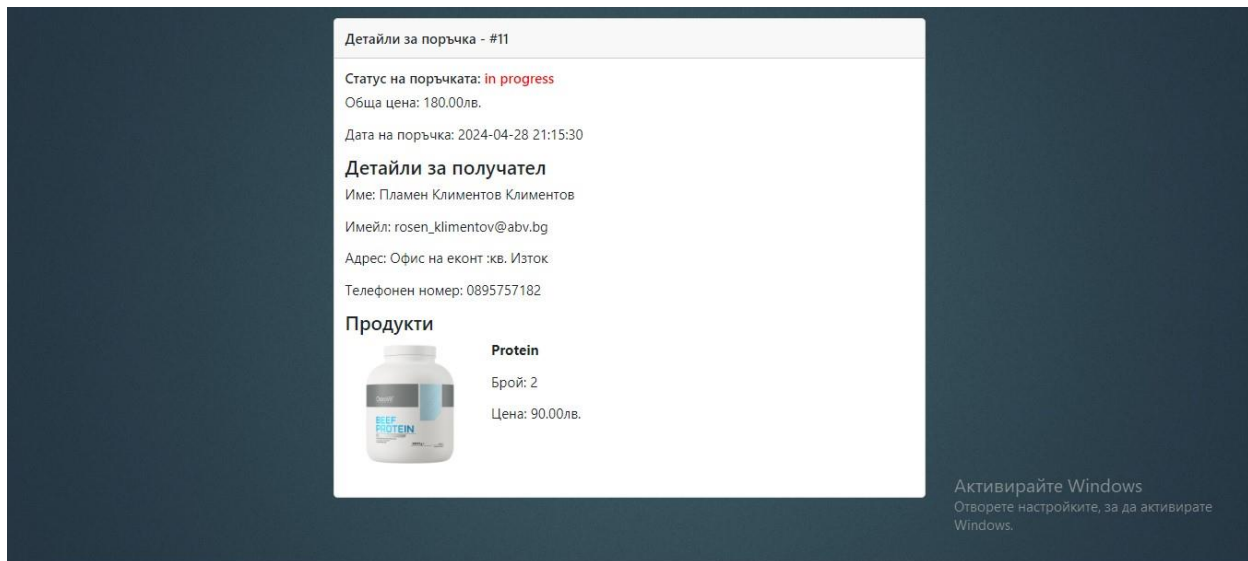
  

First Name	Last Name	Mobile	Email	Category	Comment	Appointment Date
Росен	Климентов	0898333290	rosen.klimentov01@abv.bg	Тренировка с треньор	Искам да си запиша час за тренировка с професионален треньор.	2024-04-28

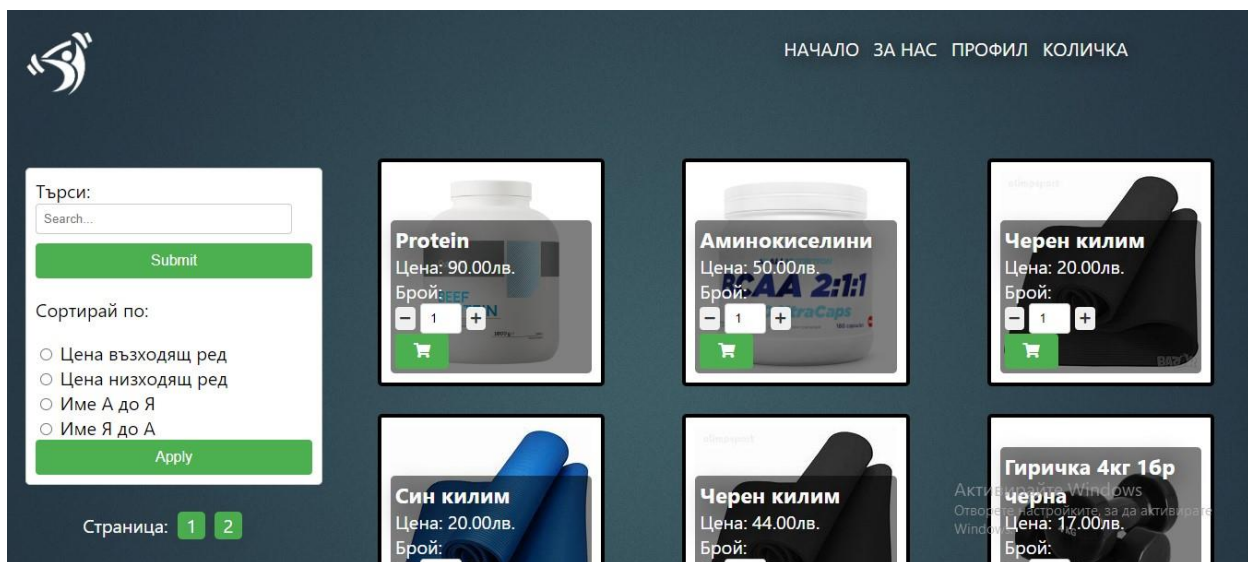
Фиг.7.5.Страницата с профилът на потребителя

Тук потребителят проверява своя профил, може да го изтрие или редактира. Може да проверява своите поръчки и заявки. Потребителят може да натисне върху датата на своята поръчка, която е линк, който я изпраща към страница с детайлите и продуктите в поръчката (Фиг.7.6.).



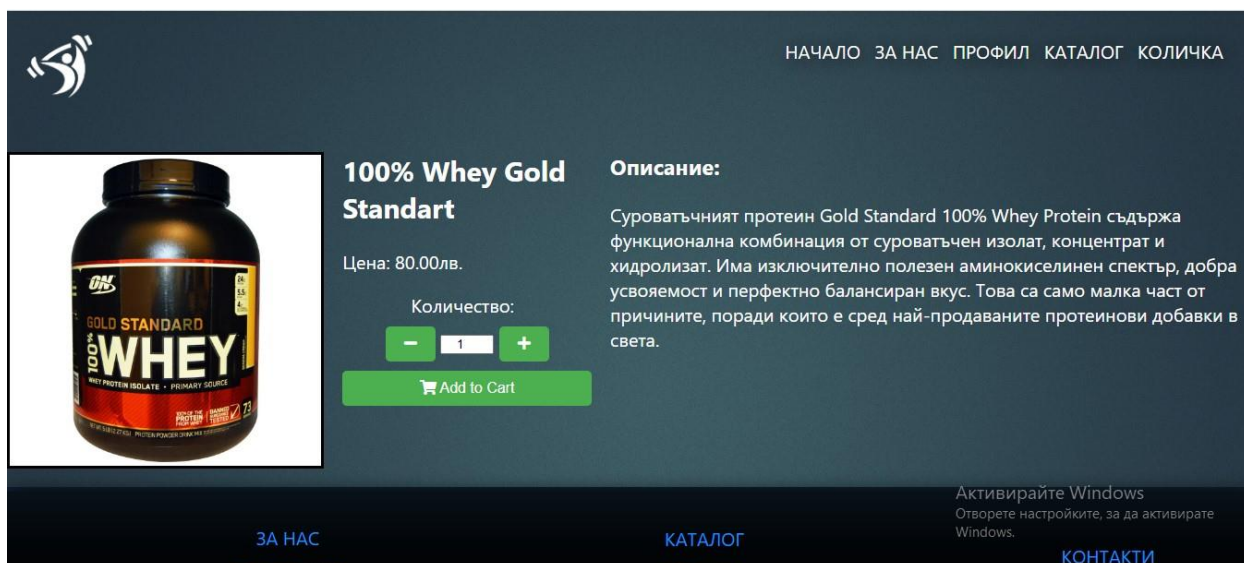


Фиг.7.6.Страницата с детайли за поръчката.

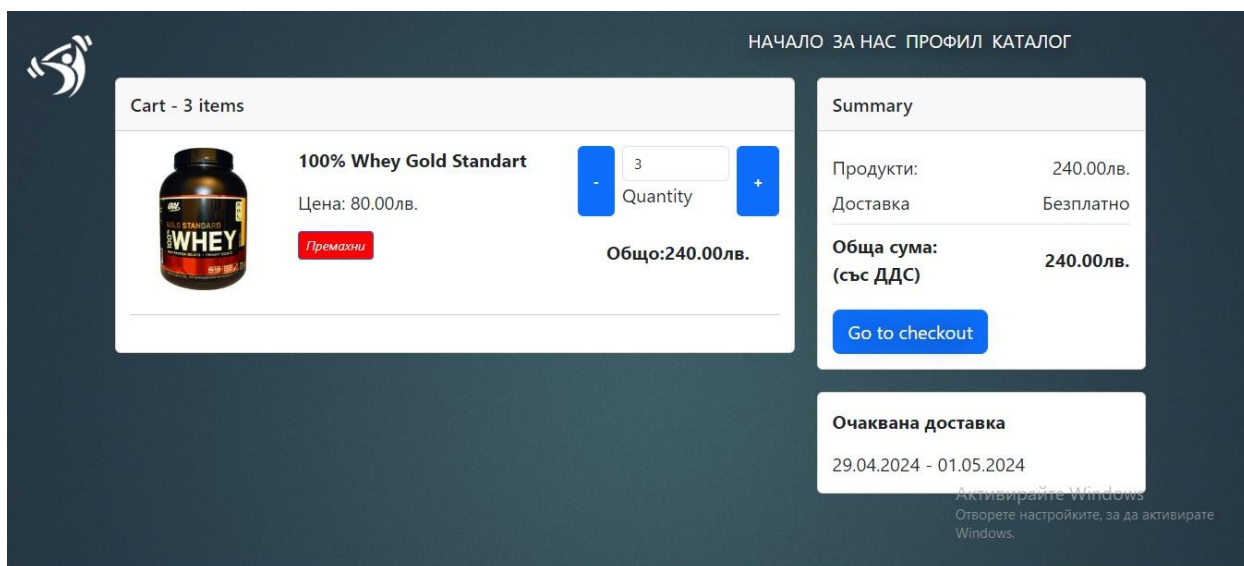


Фиг.7.7. Каталог с продукти.

В страницата с каталога се намират всички продукти, потребителят може да сортира продуктите или да търси чрез търсачката даден продукт. Потребителят добавя продукт и избира броят ,който да добави в количката си. Потребителят може да цъкне върху продукт и ще бъде препратен към страница с описание на даденият продукт(Фиг.7.8).



Фиг.7.8.Страница с описание за даден продукт.



Фиг.7.9.Страница с количката.

В страницата с количката потребителят може да види крайната цена на продуктите ,може да променя броя им или да го премахне от количката. Когато е сигурен, че е избрал продуктите ,които иска потребителят натиска бутона „Go to checkout“ ,който го отвежда до страницата за попълване на данните за доставка и финализиране на поръчката(Фиг.7.10.).

**Checkout**

Обща сума за плащане: 240.00лв. с ДДС

Име:

Фамилия:

Имейл:

Тел. номер:

Моля въведете адрес за доставка(работим само с наш куриер до личен адрес):

☐ I agree to the Terms of Service

**Завърши поръчката**

Активирайте Windows  
Отворете настройките, за да активирате Windows.

Фиг.7.10.Страница за финализиране на поръчката.

## 6.2.Админ

Име

Фамилия

Адрес

Град

Държава

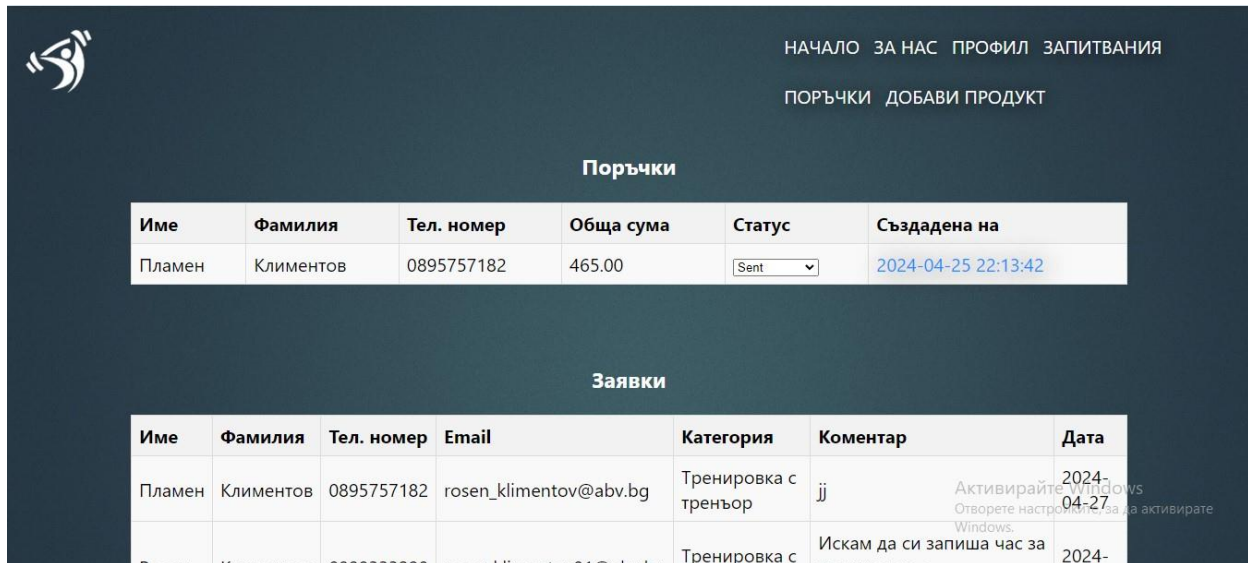
**Submit**

Име	Фамилия	Имейл	Тел. номер	Адрес	Град	Държава	Админ	Изберете действие:
admin	admin	admin@admin.com	1234567801	adim	admin	България	Admin	<input type="button" value="Delete"/> <input type="button" value="Edit"/> <input type="button" value="Remove Admin"/>
Росен	Климентов	rosen.klimentov01@abv.bg	0898333290	Офис на екопт :кв. Изток	Ботевград	България	Not Admin	<input type="button" value="Delete"/> <input type="button" value="Edit"/> <input type="button" value="Make Admin"/>

Активирайте Windows  
Отворете настройките, за да активирате Windows.

Фиг.8.1.Страница на админа за проверка на профилите в сайта.

В тази страница админа може да проверява профилите регистрирани в сайта. Може да ги редактира, изтрие или да направи потребител админ.



НАЧАЛО ЗА НАС ПРОФИЛ ЗАПИТВАНИЯ

ПОРЪЧКИ ДОБАВИ ПРОДУКТ

### Поръчки

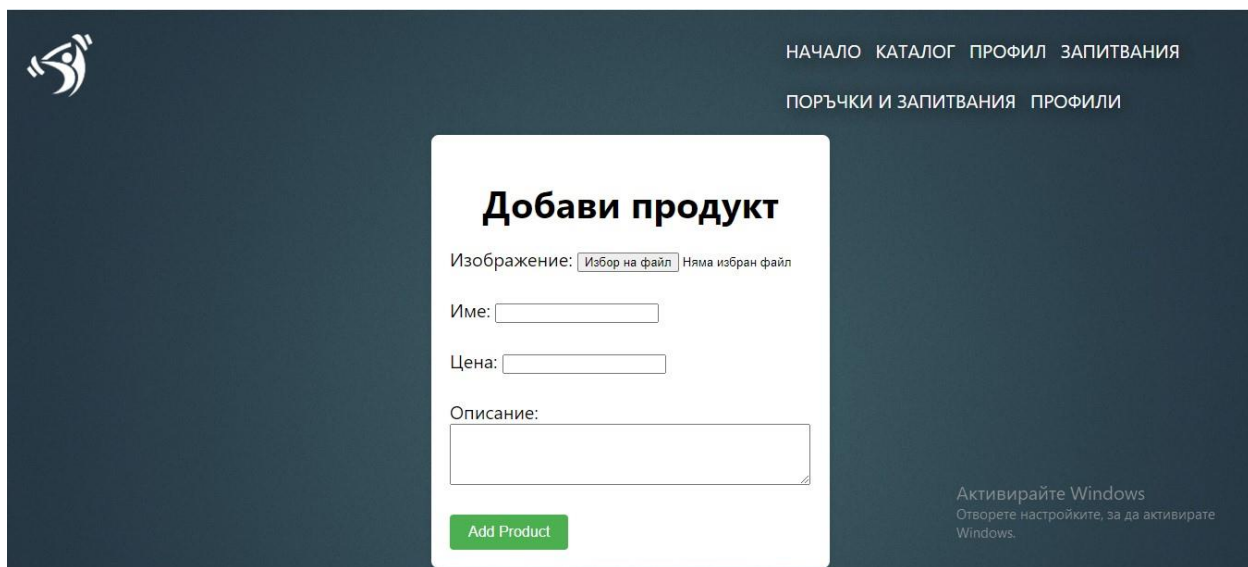
Име	Фамилия	Тел. номер	Обща сума	Статус	Създадена на
Пламен	Климентов	0895757182	465.00	Sent	2024-04-25 22:13:42

### Заявки

Име	Фамилия	Тел. номер	Email	Категория	Коментар	Дата
Пламен	Климентов	0895757182	rosen_klimentov@abv.bg	Тренировка с треньор	jj	2024-04-27
Росен	Климентов	0895757182	rosen.klimentov@abv.bg	Тренировка с треньор	Искам да си запиша час за	2024-04-27

Фиг.8.2.Страница на админа за проверка на направените поръчки и заявки в сайта.

В тази страница админът може да проверява направените поръчки и заявки ,може да променя статуса на поръчките на: в разработка ,изпратена или доставена.



НАЧАЛО КАТАЛОГ ПРОФИЛ ЗАПИТВАНИЯ

ПОРЪЧКИ И ЗАПИТВАНИЯ ПРОФИЛИ

## Добави продукт

Изображение:  Няма избран файл

Име:

Цена:

Описание:

Фиг.8.3.Страница на админа за добавяне на продукт в сайта.

В тази страница админът добавя продуктите в каталога.

## Глава 3 Литературен обзор

### Извод

Създаденият уеб сайт за фитнес продукти представлява мощен инструмент за предоставяне на широк спектър от продукти и услуги, свързани със здравословния начин на живот. Чрез внимателно проектиране и изграждане на уеб платформата, ние успяхме да създадем удобно и лесно за използване пространство, което предоставя на нашите потребители удовлетворяващо преживяване при търсенето, поръчката и използването на продуктите и услугите, които предлагаме.

Презентирането на разнообразие от фитнес продукти, както и предоставянето на подробна информация за тях, помага на нашите потребители да направят информиран избор при покупката. Също така, функционалността за добавяне на продукти в количката и завършване на поръчка е лесна и удобна, което допринася за приятното изживяване на онлайн пазаруването.

В заключение, уеб сайтът за фитнес продукти представлява надежден източник на информация и пазарно място за фитнес ентусиасти и всички, които се стремят към здравословен начин на живот. Нашата цел е да продължим да подобряваме и развиваме уеб платформата, за да отговаряме на нуждите и очакванията на нашите потребители в бъдеще.

### Възможни подобрения и перспективи за развитие

1. Разглеждаме възможността за разработка на мобилно приложение, което да допълни функциите на нашия уеб сайт. Планираме да включим лесен достъп до информацията и услугите на сайта, както и да добавим допълнителни функции като проследяване на прогреса, напомняния и интерактивни инструменти.
2. Интеграцията със здравословни устройства е още една опция, която разглеждаме. Желаем да предоставим възможност за свързване с фитнес тракери, смарт часовници и други подобни устройства. Това ще помогне на потребителите да проследят и анализират своя фитнес прогрес по-лесно и ефективно.
3. Постоянно разширяваме нашата продуктова гама, като добавяме нови и иновативни фитнес продукти. Извършваме изследвания и анализираме

предпочитанията на потребителите, за да предложим продукти, които отговарят на техните нужди и предпочитания.

4. Обогащаването на съдържанието в блога на нашия уеб сайт е също важен аспект от развитието ни. Планираме да създадем полезни и ценни съдържание, което да предоставя съвети, насоки и информация по въпроси, свързани със здравословния начин на живот, фитнес и диета.
5. Накрая, разглеждаме възможности за сключване на партньорства с фитнес марки, треньори, здравни организации и други свързани с фитнес индустрията партньори. Тези сътрудничества могат да донесат допълнителни предимства като специални оферти, събития и ресурси за нашите потребители.

## Използвана литература

1. HTML- "HTML5 – Hypertext Markup Language – 5.0" Internet Engineering Task Force. 28 October 2014. Archived from the original on October 28, 2014. Retrieved November 25, 2014. This document recommends HTML 5.0 after completion.  
<https://www.w3.org/2014/10/html5-rec.html.en>
2. CSS- "W3C CSS validation service". Archived from the original on 2011-02-14. Retrieved 2012-06-30. <https://jigsaw.w3.org/css-validator/>
3. PHP- "PHP: Hypertext Preprocessor". www.php.net. Retrieved 2020-02- 12.  
<https://www.php.net/>
4. MYSQL- "Supported Platforms: MySQL Database". Oracle. Retrieved 24 March 2014.  
<https://www.mysql.com/support/supportedplatforms/database.html>