**ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО МЕХАНОТЕХНИКА, ЕЛЕКТРОНИКА, ТЕЛЕКОМУНИКАЦИИ И ТРАНСПОРТ „ХРИСТО БОТЕВ“**



**КУРСОВ ПРОЕКТ**

на Сечкин Салим Рахим ученик от 12А клас

(трите имена на ученика)

**Тема: Уеб приложение за**

**приемане и предлагане на**

**услуги**

**(Freelancer)**

дата: xx.xx.2023 г.

гр. Шумен

1. Съдържание
2. Описание на проекта
3. Заключение

**Съдържание**

1. Използвани езици
2. Цел на уеб приложението
3. Снимки
4. Таблици
5. Основни HTML тагове

**Използвани езици**

Уеб приложението е програмиран с езиците HTML, CSS, JavaScript, MVC и C# във Visual Studio 2022.

HTML-ът е използван за скелета (строежа) на самия сайт. С езика CSS е оформен дизайнът, с JavaScript са направени по-напредналите ефекти и някои прости математически пресмятания, като пресмятане на рождена дата. А със C# са направени оперциите, като например Качване на нова обява, редактирне на вече съществуваща обява и изтриване на както съществуваща и редактирана, така и съществуваща но нередактирана обява.

**HTML** – е стандартният език за маркиране на документи, предназначени за показване в уеб браузър. Тя може да бъде подпомогната от технологии като Cascading Style Sheets (CSS) и скриптови езици като JavaScript. Уеб браузърите получават HTML документи от уеб сървър или от локално хранилище и рендират документите в мултимедийни уеб страници. HTML описва структурата на уеб страницата семантично и първоначално включва знаци за външния вид на документа. HTML елементите са градивните елементи на HTML страниците. С HTML конструкции, изображения и други обекти като интерактивни форми могат да бъдат вградени в рендираната страница. HTML предоставя средство за създаване на структурирани документи чрез обозначаване на структурна семантика за текст като заглавия, параграфи, списъци, връзки, цитати и други елементи. HTML елементите са очертани от *тагове*, написани с помощта на ъглови скоби.

**CSS** – **Cascading Style Sheets** е език за стилизиране на HTML, използван за описване на представянето на документ, написан на маркиращ език като HTML или XML. Използва се основно за описание на онлайн представянето на уеббазиран 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.

**JavaScript** – често съкратено като JS**,** е език за програмиране, който е една от основните технологии на World Wide Web, заедно с HTML и CSC.  Всички основни уеб браузъри имат специален JavaScript двигател за изпълнение на кода на устройствата на потребителите.

JavaScript е компилиран език на високо ниво, често точно навреме, който отговаря на стандарта ECMAScript. Той има динамично писане, прототипна обектна ориентация и първокласни функции. Това е мулти-парадигма, поддържаща стилове на програмиране, задвижвани от събития, функционални и наложителни стилове на програмиране. Той има приложно-програмни интерфейси (API) за работа с текст, дати, регулярни изрази, стандартни структури от данни и Document Object Model (DOM).

**C#**  – обектно ориентиран език за програмиране, разработен от Microsoft като част от софтуерната платформа .NET. Стремежът още при създаването на C# езика е бил да се създаде прост, модерен, обектно ориентиран език с общо предназначение. Основа за C# са C++, Java и донякъде езици като Delphi, VB.NET и C. Той е проектиран да балансира мощност (C++) с възможност за бързо разработване (Visual Basic и Java). Те представляват съвкупност от дефиниции на класове, които съдържат в себе си методи, а в методите е разположена програмната логика – инструкциите, които компютърът изпълнява. Програмите на C# представляват един или няколко файла с разширение .cs, в които се съдържат дефиниции на класове и други типове.

MVC – **Модел-Изглед-Контролер** (Model-View-Controller или MVC) е архитектурен шаблонен дизайн в програмирането, основан на разделянето на бизнес логиката от графичния интерфейс и данните в дадено приложение. За пръв път този шаблон за дизайн е използван в програмния език.

* **Модел (Model)** – ядрото на приложението, предопределено от областта, за която се разработва; обикновено това са данните от реалния свят, които се моделират и над които се работи – въвеждане, промяна, показване и т.н. Трябва да се прави разлика между реалния обкръжаващ свят и въображаемият абстрактен моделен свят, който е продукт на разума, който се възприема като твърдения, формули, математическа символика, схеми и други помощни средства.
* **Изглед (View)** – тази част от изходния код на приложението, отговорна за показването на данните от модела. Например изгледът може да се състои от PHP шаблонни класове, JSP страници, Asp страници. Зависи от това какъв графичен инерфейс се прави и каква платформа се използва;
* **Контролер (Controller)** – тази част от сорс кода (клас или библиотека), която взима данните от модела или извиква допълнителни методи върху модела, предварително обработва данните, и чак след това ги дава на изгледа.

**Цел на уеб приложението**

Целта на уеб приложението е да помага на хора, които имат определени знания и искат да печелят пари, но не знаят от къде и как да започнат. Сайтът позволява качване и приемане на обяви, създаване на специална заявка, комуникация между продавача и купувача, комуникация с администратора. Продавачът може да види напредъка на купувача, но само ако купувача го отбележи в самата обява. Отбелязването става в проценти. Купувачът има право да избере допълнителни услуги от сравнителните таблици, според зависи цената. Ревизията, времето за предаване и другите неща зависят от цената.

**Интернет**

* Определение - Устройства, които а свързани помежду си в една обща мрежа.
* Как работи? – работи с мрежови протоколи – HTTP, FTP, IP и др.
* Пакети – Съдържат IP адресите на потребителит на тези, които изпращат и получават.
* Протоколи – IPv4, IPv6

**Протоколи**

* **HTTP**

**Протокол за пренос на хипертекст** : ***H****yper****t****ext****T****ransfer****P****rotocol, HTTP*) е Мрежов протокол, от приложния слой на OSI модел за пренос на информация в компютърната мрежа. Създаден като средство за публикуване на HTML страници, протоколът довежда до формирането на Свтовната мрежа.

* **Методи на заявките**

HTTP определя 8 различни клиентски метода за заявки:

* **HEAD** – иска изпращане на заглавията отговарящи на посочения с URL ресурс. Отговорът на сървъра е идентичен с този на GET, но е с липсващо тяло. По този начин може да се вземе само метаинформацията, свързана с ресурса, спестявайки трансфера на целия файл.
* **GET** – с него клиентът прави заявка за ресурс, зададен чрез URL. Могат да се изпращат и ограничено количество данни, закодирани директно в самия URL (отделени чрез въпросителен знак).
* **POST** – позволява клиентът да изпрати данни на сървъра. Тази заявка обикновено се генерира при изпращането на уеб формуляр, а данните могат да бъдат: текст, написан от потребителя във формуляра; файл на клиентския компютър и др.
* **PUT** – качва файл, който в бъдеще ще отговаря на посочения URL
* **DELETE** – изтрива посочения ресурс.
* **TRACE** – сървърът връща получената заявка със статус OK. Позволява да се провери в какъв вид пристига заявката при сървъра и дали (и как) е била модифицирана по трасето от междинни прокси сървъри.
* **OPTIONS** – сървърът трябва да отговори с поддържаните от него клиентски методи, съответстващи на зададения URL, или с поддържаните от сървъра методи като цяло, ако е зададено \* вместо URL.
* **CONNECT** – използва се при комуникация през прокси.

Тези методи се пишат в тага <form method=” POST/GET…”>…</form>.

**Съдържание на проекта**

1. Папки
   1. wwwroot
      * всички снимки
      * css
   2. Areas
      * Логин и регистър файловете се появяват в тази папка след скафолдването.
   3. Controller
      * Входната точка на уеб приложението, която поема всяка заявка от браузъра. Всеки контролер представлява клас, който наследява класа Controller. Името на контролера задължително трябва да съдържа две неща: име + Controller.
      * Вески такъв клас притежава методи, които се наричат екшъни. Имената на екшъните задължително трябва да съвпадат с имената на HTML файловете, иначе конкретният файл няма да се отвори и ще даде грешка.
      * HomeController – контролер, който се появява автоматично след създаването на проекта.
      * JobController – съдържа CRUD операциите...
      * RequestController
      * CategoryController – съдържа само CRUD операциите на файла за добавяне на категория.
   4. Model
   5. View
   6. Data
   7. Layout