

Ministerul Educației, Culturii și Cercetării
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică
Departamentul Ingineria Software și Automatică

Raport de Practică

Tema: Elaborarea unui Web site de sport utilizând ASP.NET MVC

Efectuat: st.gr. TI-207 Bunescu Gabriel.
Verificat: asis.univ.Lisnic Inga.

Chișinău 2022

Cuprins

Introducere	3
1 Analiza domeniului problemei. Considerații teoretice.....	4
1.1 Tehnologii software utilizate în cadrul realizării serviciului web	5
1.1.1 HTML.....	6
1.1.2 CSS.....	7
1.1.3 Bootstrap.....	10
1.1.4 C#	13
2 Descrierea principiului de lucru al tematicii alese,un site web de sport.....	14
2.1 Familiarizarea cu structura unui document HTML/CS	15
2.2 Utilizarea ASP.NET pentru proiectarea unui Web Site.....	17
2.3 Familiarizarea cu structura modelului de proiectării BusinessLogic	19
2.4 Framework și conectarea la proiectul ASP.NET	23
3 Prezentarea generală a web site-ului de sport.....	28
Concluzie	31
Bibliografie	32

Introducere

O pagină web este o resursă aflată în spațiul web din Internet, de obicei în format HTML și având link-uri pentru navigarea simplă de la o pagină sau secțiune de pagină la alta.

Crearea paginilor HTML este un lucru relativ simplu, învățarea etichetelor HTML și crearea unor imagini ducând la realizarea de pagini web de o complexitate medie. Odată cu dezvoltarea internetului, site-urile au devenit din ce în ce mai complexe, cu un număr mai mare de pagini, cerințele privind grafica și elemente din pagina au devenit mai pretențioase și astfel proiectarea paginilor web a devenit o sarcină ceva mai dificilă.[1]

CSS folosește stiluri, acestea înglobează, sub un anumit nume, attribute de formatare care se aplică asupra unui element individual din pagină, asupra unui grup de elemente sau la nivelul întregului document. CSS funcționează cu HTML, însă nu este HTML. El extinde funcționalitatea HTML, permițând redefinirea etichetelor HTML existente.[7]

ASP.NET este o platformă pentru dezvoltatori alcătuită din instrumente, limbaje de programare și biblioteci pentru construirea multor tipuri diferite de aplicații. El extinde platforma pentru dezvoltatori .NET cu instrumente și biblioteci special pentru construirea de aplicații web. Cu ASP.NET, puteți utiliza mai multe limbi, editori și biblioteci pentru a crea pentru web, mobil, desktop, jocuri și IoT. Puteți scrie aplicații ASP.NET în C#, F# sau Visual Basic. ASP.NET Framework acceptă site-uri web, servicii, aplicații desktop și multe altele pe Windows. ASP.NET funcționează mai rapid decât orice cadru web popular. Pentru a extinde funcționalitatea, Microsoft și alții mențin un ecosistem de pachete sănătos, construit pe .NET Standard. NuGet este un manager de pachete creat special pentru .NET, care conține peste 90.000 de pachete.[10]

Web site-ul de sport are ca scop prezentarea, motivarea și atragerea a persoanelor de a se ocupa în săli de sport. Vizualizarea imaginilor printre care poate chiar să ajuga și imaginea ta! Comentariilor pe care le lasă antrenori profesioniști aduc ca regulă la motivare de a încerca ceva nou pentru fiecare, ei se staruie să se împartă cu cât mai multă informație despre cum trebuie să te antrenezi corect, care diete ar fi mai potrivite ce grafic de antrenament să respecti și care sălii de sport sunt dotate cu tot necesar pentru a ajuta oamenii să se aducă la o forma a corpului cât mai potrivită pentru fiecare în parte.

1 Analiza domeniului problemei. Considerații teoretice

ASP.NET este o tehnologie Microsoft pentru crearea de aplicații web și servicii web. ASP.NET este succesorul lui ASP (Active Service Pages) și beneficiază de puterea platformei de dezvoltare .NET, și de setul de instrumente oferite de mediul de dezvoltare al aplicației “Visual Studio .NET”.

Cîteva dintre avantajele ASP.NET sunt:

- ASP.NET are un set larg de componente, bazate pe XML, oferind astfel un model de programare orientat pe obiect (OOP).
- ASP.NET rulează cod compilat, ceea ce crește performanțele aplicației web. Codul sursă poate fi separat în două fișiere, unul pentru codul executabil, iar un altul pentru conținutul paginii (codul HTML și textul din pagină).
- ASP.NET este compatibil cu peste 20 de limbaje diferite, cele mai utilizate fiind C# și Visual Basic.

Pattern-ul Model-View-Controller(MVC) separă o aplicație în trei componente principale:

- M: model (Model).
- V: vizualizare(View User Interface).
- C: controller (Controller).

Observație:

Ceea ce am definit mai sus constituie pattern-ul MVC pentru UI – User Interface.

Pattern-ul MVC nu spune nimic despre modul cum accesăm datele, cum interacționează serviciile, etc.[1]

Interacțiunea într-o aplicație ASP.NET poate fi reprezentată astfel:

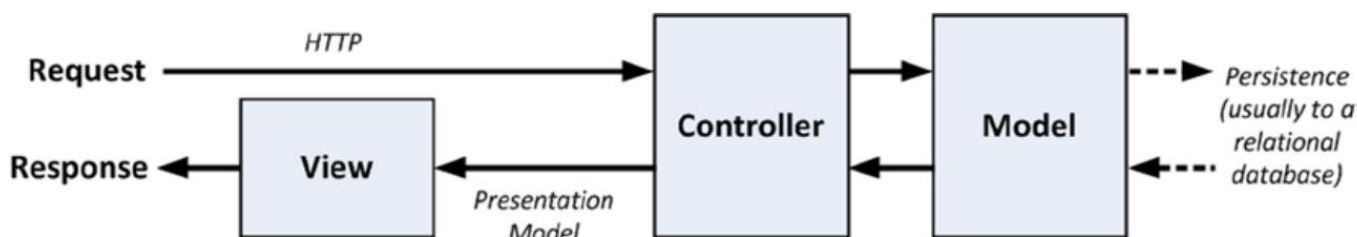


Figura 1- Interacțiunea într-o aplicație ASP.NET MVC.[2]

În ASP.NET MVC, MVC este:

Model conține clase ce reprezintă domeniul aplicației. Aceste obiecte încapsulează adesea date memorate într-o bază de date precum și cod folosit pentru a procesa datele și a executa acțiuni specifice logicii aplicației.

View definește cum va arăta interfașa aplicației. View este un template pentru a genera în cod dinamic HTML. Controller este o clasă specifică ce gestionează relațiile dintre View și Model.

Controller-ul răspunde la acțiunile utilizatorului, comunică cu modelul și decide ce vizualizări va afișa (dacă există o structură unei aplicații ASP.NET MVC).

Director – Scop.

Controllers – Conține clasele pentru Controller ce gestionează cererile URL.

Models – Conține clasele ce reprezintă datele modelului, gestionarea acestor obiecte.

View – Conține fișiere template UI responsabile pentru afișarea rezultatului.

Scripts – Conține biblioteci JavaScript și scripturi (.js).

Images – Conține imaginile folosite în cadrul aplicației.

Content – Putem pune CSS sau alt conținut dar nu scripturi și/sau imagini.

Filters – Conține codul pentru filtre.

App_Date - Conține fișierele de date Read/Write.

App_Start – Conține cod de configurare, grupări de fișiere și Web API.

Observație:

Această structură este generată (nu în totalitate) de către Visual Studio. Se poate folosi și o structură personalizată în sensul adăugării de noi directoare.[3]

1.1 Tehnologii software utilizate în cadrul realizării serviciului web

Un site web constă în mare parte din fișiere text, așa că pentru un proces de dezvoltare distractiv și plăcut, ar trebui să vă alegeți cu înțelepciune editorul de text.

Numărul mare de opțiuni este puțin copleșitor, deoarece un editor de text este atât de important pentru informatică (da, dezvoltarea web este informatică).

În general, orice editor de text poate deschide orice fișier text. Acest lucru funcționează excelent pentru a scrie note pentru dvs., dar atunci când faceți dezvoltare web și scrieți în HTML, CSS și JavaScript, puteți produce câteva fișiere destul de mari și complexe. Ușurați-vă pentru dvs. alegând un editor de text care înțelege tehnologiile cu care lucrați. Multe editoare de text vă ajută cu funcții precum

- **Colorarea codului.** Faceți fișierul mai lizibil prin codificarea culorilor cuvintelor cheie în funcție de tehnologia pe care o utilizați.
- **Completarea codului.** Economisiți timp prin completarea automată a structurilor recurente (de exemplu, închiderea automată a etichetelor HTML sau sugerând valori valide pentru o anumită proprietate CSS).
- **Fragmente de cod.** După cum ați văzut când începeți un nou document HTML, multe tehnologii folosesc aceeași structură de document din nou și din nou. Economisiți-vă bătălia de a reseta toate acestea folosind un fragment de cod pentru a vă completa documentul în prealabil.

Majoritatea editorilor de text acceptă acum colorarea codului, dar nu neapărat celelalte două caracteristici. Asigurați-vă, în special, că editorul dvs. de text codifică HTML, CSS și JavaScript .[11]

1.1.1 HTML

HTML este cel mai răspândit limbaj pentru a scrie pagini web .

- **Hipertextul** se referă la modul în care paginile Web (documentele HTML) sunt legate între ele. Astfel, linkul disponibil pe o pagină web se numește Hypertext.
- După cum sugerează și numele, HTML este un **limbaj de marcare**, ceea ce înseamnă că utilizați HTML pentru a „marca” pur și simplu un document text cu etichete care spun unui browser web cum să-l structureze pentru a-l afișa.

Inițial, HTML a fost dezvoltat cu intenția de a defini structura documentelor, cum ar fi titluri, paragrafe, liste și așa mai departe, pentru a facilita schimbul de informații științifice între cercetători.

Acum, HTML este utilizat pe scară largă pentru a formata pagini web cu ajutorul diferitelor etichete disponibile în limbajul HTML.

Etichete HTML

După cum sa spus mai devreme, HTML este un limbaj de marcare și folosește diverse etichete pentru a formata conținutul. Aceste etichete sunt incluse în acolade unghiulare **<Numele etichetei>** . Cu excepția câtorva etichete, majoritatea etichetelor au etichetele de închidere corespunzătoare. De exemplu, **<html>** are eticheta de închidere **</html>** și eticheta **<body>** are eticheta de închidere **</body>** etc.[4]

Următorul este un exemplu clasic :

```
<!DOCTYPE html>
< html >
  < head >
    < title > Acesta este un titlu </ title >
  </ head >
  < body >
    < div >
      < p > Bună lume! </ p >
    </ div >
  </ body >
</ html >
```

HTML are o mulțime de cazuri de utilizare, și anume:

- Dezvoltare web . Dezvoltatorii folosesc codul HTML pentru a proiecta modul în care un browser afișează elementele paginii web, cum ar fi text, hyperlinkuri și fișiere media.

- Navigare pe internet . Utilizatorii pot naviga cu ușurință și insera legături între paginile conexe și site-urile web, deoarece HTML este foarte folosit pentru a încorpora hyperlinkuri.
- Documentație web . HTML face posibilă organizarea și formatarea documentelor, în mod similar cu Microsoft Word.

Toate paginile HTML au o serie de elemente HTML, constând dintr-un set de etichete și atribute. Elementele HTML sunt elementele de bază ale unei pagini web. O etichetă spune browserului web unde începe și se termină un element, în timp ce un atribut descrie caracteristicile unui element.

Cele trei părți principale ale unui element sunt:

- Etichetă de deschidere – folosită pentru a indica locul în care un element începe să aibă efect. Eticheta este înfășurată cu paranteze unghiulare de deschidere și de închidere. De exemplu, utilizați eticheta de început `<p>` pentru a crea un paragraf.
- Conținut – acesta este rezultatul pe care îl văd alți utilizatori.
- Eticheta de închidere – la fel ca eticheta de deschidere, dar cu o bară oblică înainte de numele elementului. De exemplu, `</p>` pentru a încheia un paragraf.

Avantaje și dezavantaje ale HTML

La fel ca orice alt limbaj de calculator, HTML are punctele sale forte și limitările sale. Iată avantajele și dezavantajele HTML-ului:

Pro:

- Prietenos pentru începători. HTML are un marcaj curat și consistent, precum și o curbă de învățare superficială.
- A sustine. Limba este folosită pe scară largă, având în spate o mulțime de resurse și o comunitate mare.
- Accesibil. Este open-source și complet gratuit. HTML rulează nativ în toate browserele web.
- Flexibil. HTML este ușor de integrat cu limbaje backend precum PHP și Node.js.

Contra:

- Static. Limbajul este folosit în principal pentru pagini web statice. Pentru funcționalitate dinamică, poate fi necesar să utilizați JavaScript sau un limbaj back-end, cum ar fi PHP.
- Pagina HTML separată. Utilizatorii trebuie să creeze pagini web individuale pentru HTML, chiar dacă elementele sunt aceleași.
- Compatibilitate browser. Unele browsere adoptă noi funcții încet. Uneori, browserele mai vechi nu redau întotdeauna etichete mai noi.[5]

1.1.2 CSS

CSS, este un limbaj de design simplu menit să simplifice procesul de a face paginile web prezentabile.

CSS gestionează aspectul și senzația parte a unei pagini web. Folosind CSS, puteți controla culoarea textului, stilul fonturilor, distanța dintre paragrafe, modul în care sunt dimensionate și aranjate coloanele, ce imagini sau culori de fundal sunt utilizate, modele de aspect, variații ale afișajului pentru diferite dispozitive și dimensiuni de ecran. precum și o varietate de alte efecte.

CSS este ușor de învățat și de înțeles, dar oferă un control puternic asupra prezentării unui document HTML. Cel mai frecvent, CSS este combinat cu limbajele de marcare HTML sau XHTML.

Avantajele CSS

- CSS economisește timp - Puteți scrie CSS o dată și apoi reutiliza aceeași foaie în mai multe pagini HTML. Puteți defini un stil pentru fiecare element HTML și îl puteți aplica la câte pagini Web doriți.
- Paginile se încarcă mai repede - Dacă utilizați CSS, nu trebuie să scrieți atribute de etichetă HTML de fiecare dată. Doar scrieți o regulă CSS a unei etichete și aplicați-o la toate aparițiile etichetei respective. Deci, mai puțin cod înseamnă timpuri mai rapide de descărcare.
- Întreținere ușoară - Pentru a face o schimbare globală, schimbați pur și simplu stilul și toate elementele din toate paginile web vor fi actualizate automat.
- Stiluri superioare HTML – CSS are o gamă mult mai largă de atribute decât HTML, astfel încât puteți oferi un aspect mult mai bun paginii dvs. HTML în comparație cu atributele HTML.
- Compatibilitate cu mai multe dispozitive - Foile de stil permit optimizarea conținutului pentru mai mult de un tip de dispozitiv. Folosind același document HTML, pot fi prezentate versiuni diferite ale unui site web pentru dispozitive portabile, cum ar fi PDA-uri și telefoane mobile sau pentru imprimare.
- Standarde web globale - Acum atributele HTML sunt depreciate și se recomandă utilizarea CSS. Deci, este o idee bună să începeți să utilizați CSS în toate paginile HTML pentru a le face compatibile cu viitoarele browsere.[6]

Modalități de a include CSS într-un document

Regulile CSS pot fi localizate atât în documentul web în sine , al cărui aspect îl descriu, cât și în fișierele externe care au extensia `.css` . Formatul CSS este un fișier text care conține o listă de reguli CSS și comentariile acestora.

Stilurile CSS pot fi incluse sau încorporate în documentul web pe care îl descriu în patru moduri:

- când definiția stilului este într-un fișier separat, aceasta poate fi atașată documentului prin intermediul unui element `<link>` inclus în elementul `<head>` :

```
<!DOCTYPE html>
```

```
< html >
```

```
< head >
```



```

.....
< link rel = "stylesheet" type = "text/css" href = "style.css" >
</ head >
< body >
.....
</ body >
</ html >

```

- când fișierul de stil este plasat separat de documentul părinte, acesta poate fi inclus în document cu o instrucțiune @import pe elementul `<style>`:

```

<!DOCTYPE html>
< html >
  < head >
    .....
    < style media = "toate" >
      @ import url ( stil . css ) ;
    </ style >
  </ head >
</ html >

```

Anatomia unui set de reguli CSS

Întreaga structură este numită un set de reguli. Notați și numele părților individuale:

Selector:

- Numele elementului HTML la începutul setului de reguli. Selectează elementul(ele) la care să se aplice stilul (în acest caz, elemente p). Pentru a stila un alt element, trebuie doar să schimbați selectorul.

Declarație:

- singură regulă, de exemplu, color: red; specifică care dintre proprietățile elementului doriți să stilați.

Proprietăți:

- Modalitățile în care puteți stila un anumit element HTML (în acest caz, coloro proprietate asupra elementelor p). În CSS, alegeți ce proprietăți doriți să afectați în regula dvs.

Valoarea proprietății:

- În dreapta proprietății, după două puncte, avem o valoare a proprietății care selectează una dintre numeroasele caracteristici posibile pentru acea proprietate (sunt multe valori colorpe lângă red).

Observați părțile importante ale sintaxei:

- Fiecare set de reguli (cu excepția selectorului) trebuie să fie înfășurat în acolade ({}).

- Fiecare declarație trebuie să folosească două puncte (:) pentru a separa proprietatea de valorile sale.
- În fiecare set de reguli, trebuie să utilizați punct și virgulă (;) pentru a separa fiecare declarație de următoarea.

Astfel, pentru a schimba mai multe valori de proprietate simultan, trebuie doar să le scrieți separate prin punct și virgulă, astfel[7]:

```
p {
  color: red;
  width: 500px;
  border: 1px solid black;
}
```

1.1.3 Bootstrap

Bootstrap este un set de instrumente puternic - o colecție de instrumente HTML, CSS și JavaScript pentru crearea și construirea de pagini web și aplicații web.

Cu Bootstrap, dezvoltatorii web se pot concentra pe munca de dezvoltare, fără a-și face griji în privința designului, și pot obține rapid un site web arătos. În schimb, oferă designerilor web o bază solidă pentru crearea unor teme Bootstrap interesante.

Noțiuni de bază cu Bootstrap

Bootstrap este disponibil în două forme, ca versiune precompilată și ca versiune de cod sursă. Versiunea codului sursă folosește preprocesorul Less CSS. Pentru a facilita utilizarea prefixelor de furnizor CSS, Bootstrap folosește Autoprefixer .

Structura fișierului

Ne vom concentra pe versiunea precompilată. Când descărcați arhiva zip și o decompriți, structura de bază a fișierului arată astfel:

```
bootstrap/
```

```
├── css/
```

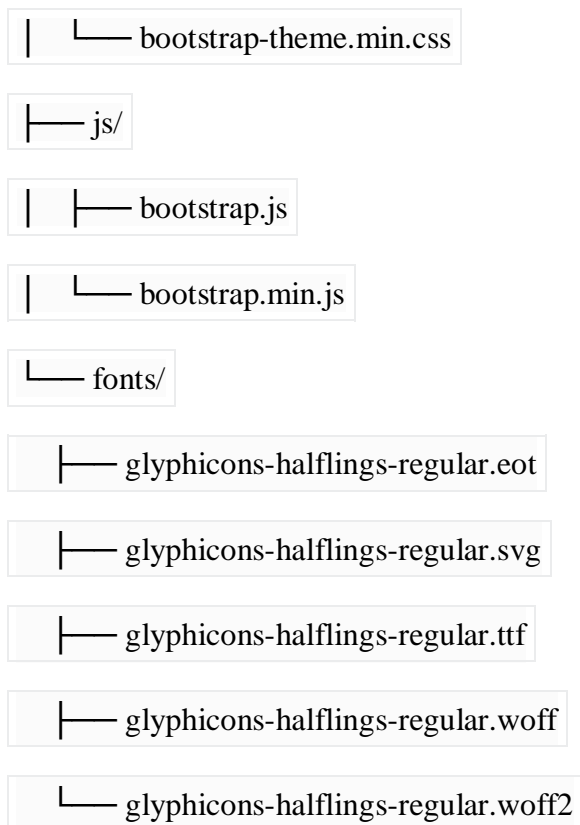
```
|   ├── bootstrap.css
```

```
|   ├── bootstrap.css.map
```

```
|   ├── bootstrap.min.css
```

```
|   ├── bootstrap-theme.css
```

```
|   ├── bootstrap-theme.css.map
```



Structura Bootstrap este destul de simplă și explicită de la sine. Include fișiere precompilate care permit utilizarea rapidă în orice proiect web. Pe lângă fișierele CSS și JS compilate și minimizate, include și fonturi din Glyphicons și tema bootstrap de pornire opțională.

Această structură poate fi încorporată cu ușurință în structura de fișiere a propriului proiect prin includerea fișierelor Bootstrap exact așa cum ies din arhiva zip sau, dacă se potrivește mai bine proiectului dvs., puteți rearanja aceste fișiere și le puteți plasa oriunde doriți. Doar asigurați-vă că folderul cu fonturi Glyphicons este la același nivel cu folderul CSS.

Un șablon HTML Bootstrap de bază ar trebui să arate cam așa:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap Template</title>
    <link href="css/bootstrap.min.css" rel="stylesheet">
  </head>
  <body>
    <h1>Hello, world!</h1>
```

```

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script src="js/bootstrap.min.js"></script>
</body>
</html>

```

Sistemul de grilă Bootstrap:

Dacă combinăm diferite niveluri de clasă, vom obține aspecturi diferite pe vizualizările mobile și pe desktop. În exemplul de mai jos, pe desktop vor fi 4 coloane într-o linie, iar pe mobil vor avea lățimea completă și se vor stivui unele pe altele.

```

<div class="row">
  <div class="col-xs-12 col-md-3">.col-xs-12 .col-md-3</div>
  <div class="col-xs-12 col-md-3">.col-xs-12 .col-md-3</div>
  <div class="col-xs-12 col-md-3">.col-xs-12 .col-md-3</div>
  <div class="col-xs-12 col-md-3">.col-xs-12 .col-md-3</div>
</div>

```



Figura 2 – Aspecte diferite de vizualizării mobile și pe desktop.[8]

Imagini și icoane

Imaginile din Bootstrap pot fi adaptate, pur și simplu oferindu-le `.img-responsive` clasa. Acest lucru se va aplica `max-width:100%; height:auto;` și `display:block;` pentru imaginea în cauză, astfel încât să se scaleze la elementul părinte.[8]

1.1.4 C#

Există o mulțime de limbaje de programare de uz general, dar dezvoltatorii pot fi cu siguranță de acord că C# este unul dintre cele mai bune. Probabil că asta are de-a face cu faptul că C# este foarte versatil, oferă o curbă de învățare blândă și este orientat pe obiecte.

Proiectat inițial pentru a rivaliza cu Java, C# este un limbaj de programare sage de tip modern, care permite inginerilor să construiască aplicații care rulează în ecosistemul .NET. După cum vă puteți imagina, are legături puternice cu familia de limbi C, așa că orice inginer cu o bună înțelegere a C și C++ nu va avea cu siguranță nicio problemă să înceapă cu C#.

De la prima sa lansare, acum mai bine de 20 de ani, C# a fost constant printre cele mai populare limbaje din lumea programării. De fapt, de ani de zile a ocupat locul 4 în PYPL Popularity of Programming Language Index , chiar sub alte categorii grele din industrie precum Python, Java și JavaScript.

C# este un limbaj de programare de uz general, cu o abordare multiparadigma, care cuprinde mai multe discipline de programare, cum ar fi scrierea statică, programarea imperativă, declarativă, funcțională, orientată pe obiecte și orientată pe componente. Această abordare este ceea ce permite C# să fie atât de versatil până la punctul în care îl puteți folosi pentru o mulțime de proiecte diferite.

Dezvoltat de Microsoft în 2000, C# a fost creat pentru a răspunde cererii emergente pentru aplicații web. În timp ce compania Redmond avea Visual Basic și C++ pentru a lucra la acest tip de aplicație, realitatea este că ambele limbi au avut probleme în a lansa software de înaltă performanță. De aceea, C# a găsit atât de repede o nișă de inginerie, deoarece arhitectura sa urmează cele mai bune practici Java pentru a oferi o abordare mai bună a dezvoltării aplicațiilor.

Alte caracteristici remarcabile ale C# includ capacitatea sa de a reutiliza componente pentru o dezvoltare mai rapidă și tipurile de date flexibile, fără erori. De parcă nu ar fi de ajuns, C# are o gamă largă de componente care pot impulsiona cu ușurință orice proiect, fie că este orientat spre sistem sau orientat spre afaceri.

Ca limbaj de programare de uz general, puteți folosi C# pentru a dezvolta aproape orice vă puteți gândi, de la aplicații mobile și desktop până la software de întreprindere și platforme bazate pe cloud. Cu toate acestea, C# strălucește cel mai mult atunci când îl utilizați pentru 3 tipuri specifice de proiecte.

- dezvoltare web - ca parte a platformei .NET, C# este o potrivire naturală pentru construirea de site-uri web și aplicații dinamice. Natura sa orientată pe obiecte îl face perfect pentru dezvoltarea de site-uri web care se laudă cu eficiență ridicată și sunt ușor scalabile.
- aplicații Windows - deoarece C# a fost dezvoltat de Microsoft, este firesc ca acesta să fie utilizat pe scară largă pentru a construi aplicații desktop Windows. De fapt, acesta ar putea fi cel mai puternic caz de utilizare pentru acest limbaj - creând aplicații adaptate special arhitecturii sistemului de operare Microsoft.

- Dezvoltarea jocului - C# a fost larg recunoscut ca unul dintre cele mai bune limbaje de programare pentru jocuri, în special jocurile Unity. C# se integrează cu motorul Unity pentru a oferi cel mai bun mediu pentru dezvoltarea jocurilor mobile – și îl puteți folosi chiar și pentru a dezvolta jocuri de consolă cu tehnologii multi-platformă precum Xamarin.

Dezavantajele utilizării C#

Oricât de minunat este C#, are o serie de dezavantaje pe care trebuie să le iei în considerare înainte de a-l adopta pentru proiectele tale digitale. Cele mai notabile includ:

- Bazat pe Windows - deoarece C# face parte din ecosistemul .NET, aplicațiile sale sunt aproape exclusiv pentru sisteme bazate pe Windows. Dacă alegeți să lucrați cu un sistem de operare diferit, este posibil să descoperiți că unele dintre funcțiile C# nu funcționează sau nu sunt disponibile.
- Dependentă de .NET - în timp ce C# este versatil și poate servi unui număr destul de mare de proiecte, această abilitate vine cu o avertizare: aveți nevoie de framework-ul .NET pentru ca totul să funcționeze fără probleme.
- Imposibilitatea de a codifica soluții de nivel scăzut - C# este un limbaj de nivel înalt, ceea ce nu înseamnă doar că abordările de sintaxă și codare sunt mai abstracte, ci și că interfațarea produselor C# cu hardware-ul este imposibilă.[9]

2 Descrierea principiului de lucru al tematicii alese,un site web de sport

Implementarea cunoștințelor acumulate în decursul practicii pentru realizarea unui site web de sport utilizând mai multe limbaje pentru crearea lui: HTML, CSS, C#. Cu ajutorul HTML am creat paginile web, cu ajutorul CSS i-am redat stilurile necesare.

ASP.NET (Active Server Pages) este un set de tehnologii care permite crearea de aplicații web, beneficiind de suportul platformei de dezvoltare Microsoft.NET. Una dintre cele mai importante calități ale ASP.NET este timpul redus necesar dezvoltării aplicațiilor web. ASP.NET cuprinde toate tehnologiile necesare pentru a dezvolta o aplicație web scriind cantitatea minimă de cod.

Limbajele de programare care pot fi utilizate pentru a crea aplicații ASP.NET sunt cele suportate de platforma .NET – cum sunt Visual Basic .NET și C#, iar o altă caracteristică importantă a acestor limbaje este că au fost create având în vedere paradigma programării orientată pe obiecte.

ASP.NET MVC este un framework pentru web development dezvoltat de Microsoft care combină arhitectura model-view-controller cu cele mai bune componente ale platformei ASP.NET și cu cele mai recente idei și tehnici de dezvoltare de tip Agile. ASP.NET MVC este o tehnologie Open Source, o reală alternativă la tradiționalul ASP.NET Web Forms, având avantaje considerabile în special pentru proiectele web de mare anvergură.

2.1 Familiarizarea cu structura unui document HTML/CS

HTML (HyperText Markup Language) – este un limbaj care descrie formatul primar în care documentele sunt distribuite și văzute pe Web. Multe din trasaturile lui, cum ar fi independența față de platforma, structurarea formătărilor și legaturile hypertext, fac din el un foarte bun format pentru documentele Internet și Web.

main>

```
<!--? slider Area Start-->
<div class="slider-area position-relative">
  <div class="slider-active">
    <!-- Single Slider -->
    <div class="single-slider slider-height d-flex align-items-center">
      <div class="container">
        <div class="row">
          <div class="col-xl-9 col-lg-9 col-md-10">
            <div class="hero__caption">
              <span data-animation="fadeInLeft" data-delay="0.1s">Hi This is Zacson</span>
              <h1 data-animation="fadeInLeft" data-delay="0.4s">Gym Trainer</h1>
              <a href="@Url.Action("Courses","")" class="border-btn hero-btn"
                data-animation="fadeInLeft" data-delay="0.8s">My Courses</a>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

CSS – este un limbaj de stilizare al elementelor html, al tagurilor html. Denumirea CSS provine din expresia Cascading Style Sheets. În Web Design-ul modern, pentru stilizarea paginilor web se folosește numai CSS. Acest lucru înseamnă că de la culoarea literelor și a backgroundului până și la poziționarea elementelor de pe o pagină web, totul este stilizat prin CSS.

body{

```
margin: 0;
padding: 0;
display: flex;
```

```

        justify-content: center;
        align-items: center;
        min-height: 100vh;
        font-family: 'Jost', sans-serif;
        background: linear-gradient(to bottom, #0f0c29, #302b63, #24243e);
    }
    #chk{
        display: none;
    }
    .signup{
        position: relative;
        width:100%;
        height: 100%;
    }
    label{
        color: #fff;
        font-size: 2.3em;
        justify-content: center;
        display: flex;
        margin: 60px;
        font-weight: bold;
        cursor: pointer;
        transition: .5s ease-in-out;
    }

```

Bootstrap – este un framework CSS gratuit și opened-source, direcționat către dezvoltarea web front-end receptivă și mobilă. Conține șabloane de design bazate pe CSS și JavaScript pentru tipografie, formulare, butoane, navigare și alte componente de interfață.

<!-- Traning categories Start -->

```

<section class="traning-categories black-bg">
    <div class="container-fluid">
        <div class="row">
            <div class="col-xl-6 col-lg-6">
                <div class="single-topic text-center mb-30">
                    <div class="topic-img">
                        

```



```

<div class="topic-content-box">
  <div class="topic-content">
    <h3>Personal training</h3>
    <p>You'll look at graphs and charts in Task One, how to approach the task and <br>
      the language needed for a successful answer.</p>
    <a href="@Url.Action("Courses","")" class="border-btn">View Courses</a>
  </div>
</div>
</div>
</div>
</div>

```

2.2 Utilizarea ASP.NET pentru proiectarea unui Web Site

MVC este un concept foarte răspândit în programarea Web. Scopul MVC este de a ține separate logică business-ului și interfață utilizator, astfel încât cei care întrețin aplicația să schimbe mult mai ușor o parte, fără a afecta alte părți. În MVC, modelul conține informațiile (datele) și regulile business; view conține elemente din interfață utilizator (texte, input-uri ale formularelor etc); controller-ul gestionează comunicația dintre model și view. În afară de MVC, Yii introduce un front-controller, cu numele application, care reprezintă contextul în care se execută procesarea cererii client. Application rezolvă cererea utilizator și o trimite mai departe controller-ului corespunzător care va trata efectiv cererea.

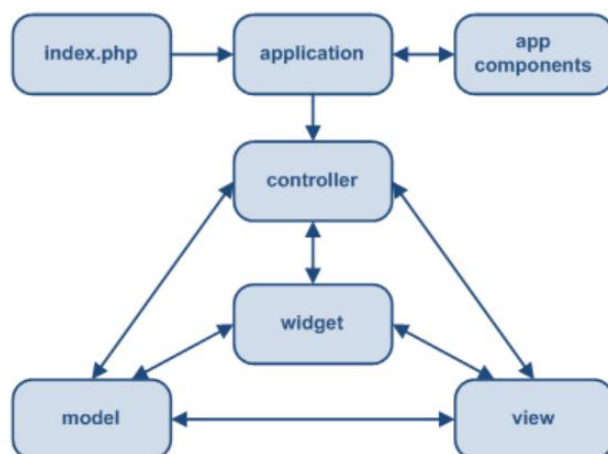


Figura 3 – Exemplu de HTML.

Bundle este o caracteristică nouă în ASP.NET 4.5 care facilitează combinarea sau gruparea mai multor fișiere într-un singur fișier. Puteți crea CSS, JavaScript și alte pachete. Mai puține fișiere înseamnă mai puține solicitări HTTP și asta poate îmbunătăți performanța de încărcare a primei pagini.

```

BundleConfig.cs  X _Layout.cshtml
WebApplication1  WebApplication1.Web.BundleConfig  RegisterBundles(BundleCollection bundles)
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.Optimization;
6
7  namespace WebApplication1.Web
8  {
9      public class BundleConfig
10     {
11
12         internal static void RegisterBundles(BundleCollection bundles)
13         {
14             bundles.Add(new StyleBundle("~/bundles/bootstrap/css").Include("~/Content/bootstrap.min.css", new CssRewriteUrlTransform()));
15             bundles.Add(new StyleBundle("~/bundles/owl.carousel/css").Include("~/Vendors/Css/owl.carousel.min.css", new CssRewriteUrlTransform()));
16             bundles.Add(new StyleBundle("~/bundles/slicknav/css").Include("~/Vendors/Css/slicknav.css", new CssRewriteUrlTransform()));
17             bundles.Add(new StyleBundle("~/bundles/flaticon/css").Include("~/Vendors/Css/flaticon.css", new CssRewriteUrlTransform()));
18             bundles.Add(new StyleBundle("~/bundles/gijgo/css").Include("~/Vendors/Css/gijgo.css", new CssRewriteUrlTransform()));
19             bundles.Add(new StyleBundle("~/bundles/animate/css").Include("~/Vendors/Css/animate.min.css", new CssRewriteUrlTransform()));
20             bundles.Add(new StyleBundle("~/bundles/animated-headline/css").Include("~/Vendors/Css/animated-headline.css", new CssRewriteUrlTransform()));
21             bundles.Add(new StyleBundle("~/bundles/magnific-popup/css").Include("~/Vendors/Css/magnific-popup.css", new CssRewriteUrlTransform()));
22             bundles.Add(new StyleBundle("~/bundles/fontawesome-all/css").Include("~/Vendors/Css/fontawesome-all.min.css", new CssRewriteUrlTransform()));
23             bundles.Add(new StyleBundle("~/bundles/themify-icons/css").Include("~/Vendors/Css/themify-icons.css", new CssRewriteUrlTransform()));
24             bundles.Add(new StyleBundle("~/bundles/slick/css").Include("~/Vendors/Css/slick.css", new CssRewriteUrlTransform()));
25             bundles.Add(new StyleBundle("~/bundles/nice-select/css").Include("~/Vendors/Css/nice-select.min.css", new CssRewriteUrlTransform()));
26             bundles.Add(new StyleBundle("~/bundles/style/css").Include("~/Vendors/Css/style.css", new CssRewriteUrlTransform()));
27
28             bundles.Add(new ScriptBundle("~/bundles/modernizr-3.5.0/js").Include("~/Vendors/vendor/modernizr-3.5.0.min.js"));
29             bundles.Add(new ScriptBundle("~/bundles/jquery-1.12.4/js").Include("~/Vendors/vendor/jquery-1.12.4.min.js"));
30             bundles.Add(new ScriptBundle("~/bundles/popper/js").Include("~/Vendors/popper.min.js"));
31             bundles.Add(new ScriptBundle("~/bundles/jquery.slicknav/js").Include("~/Vendors/jquery.slicknav.min.js"));
32             bundles.Add(new ScriptBundle("~/bundles/owl.carousel/js").Include("~/Vendors/owl.carousel.min.js"));
33             bundles.Add(new ScriptBundle("~/bundles/slick/js").Include("~/Vendors/slick.min.js"));
34             bundles.Add(new ScriptBundle("~/bundles/animated-headline/js").Include("~/Vendors/animated-headline.js"));
35             bundles.Add(new ScriptBundle("~/bundles/jquery.magnific-popup/js").Include("~/Vendors/jquery.magnific-popup.js"));
36             bundles.Add(new ScriptBundle("~/bundles/gijgo/js").Include("~/Vendors/gijgo.min.js"));
37             bundles.Add(new ScriptBundle("~/bundles/jquery.nice-select/js").Include("~/Vendors/jquery.nice-select.min.js"));
38             bundles.Add(new ScriptBundle("~/bundles/jquery.sticky/js").Include("~/Vendors/jquery.sticky.js"));
39             bundles.Add(new ScriptBundle("~/bundles/jquery.counterup/js").Include("~/Vendors/jquery.counterup.min.js"));
40             bundles.Add(new ScriptBundle("~/bundles/waypoints/js").Include("~/Vendors/waypoints.min.js"));
41             bundles.Add(new ScriptBundle("~/bundles/jquery.validate/js").Include("~/Vendors/jquery.validate.min.js"));
42             bundles.Add(new ScriptBundle("~/bundles/jquery.validate.unobtrusive/js").Include("~/Vendors/jquery.validate.unobtrusive.min.js"));
43             bundles.Add(new ScriptBundle("~/bundles/contact/js").Include("~/Vendors/contact.js"));
44             bundles.Add(new ScriptBundle("~/bundles/jquery.form/js").Include("~/Vendors/jquery.form.js"));
45             bundles.Add(new ScriptBundle("~/bundles/jquery.validate.js").Include("~/Vendors/jquery.validate.min.js"));
46             bundles.Add(new ScriptBundle("~/bundles/mail-script/js").Include("~/Vendors/mail-script.js"));
47             bundles.Add(new ScriptBundle("~/bundles/jquery.ajaxchimp/js").Include("~/Vendors/jquery.ajaxchimp.min.js"));
48             bundles.Add(new ScriptBundle("~/bundles/plugins/js").Include("~/Vendors/plugins.js"));
49             bundles.Add(new ScriptBundle("~/bundles/main/js").Include("~/Vendors/main.js"));
50
51         }
52     }
53 }

```

Figura 4 – Configurarea Bundle.

Layout - în scopul eficientizării și economisirii de spațiu și timp, a fost creat fișierul Layout. În cadrul acestuia se introduc rândurile de cod ce se referă la structuri statice din cadrul proiectului, care pot fi vizualizate pe orice pagină din cadrul acestuia, precum navbar sau footer.

```

BundleConfig.cs  _Layout.cshtml  X
1  using System.Web.Optimization;
2
3  <doctype html>
4  <html class="no-js" lang="xxx">
5  <head>
6      <meta charset="utf-8">
7      <meta http-equiv="x-ua-compatible" content="ie=edge">
8      <title>Gym trainer | Template </title>
9      <meta name="description" content="">
10     <meta name="viewport" content="width=device-width, initial-scale=1">
11     <link rel="manifest" href="site.webmanifest">
12     <link rel="shortcut icon" type="image/x-icon" href="~/Images/Favicon.ico">
13
14     <!-- CSS here -->
15     @Styles.Render("~/bundles/bootstrap/css")
16     @Styles.Render("~/bundles/owl.carousel/css")
17     @Styles.Render("~/bundles/slicknav/css")
18     @Styles.Render("~/bundles/flaticon/css")
19     @Styles.Render("~/bundles/gijgo/css")
20     @Styles.Render("~/bundles/animate/css")
21     @Styles.Render("~/bundles/animated-headline/css")
22     @Styles.Render("~/bundles/magnific-popup/css")
23     @Styles.Render("~/bundles/fontawesome-all/css")
24     @Styles.Render("~/bundles/themify-icons/css")
25     @Styles.Render("~/bundles/slick/css")
26     @Styles.Render("~/bundles/nice-select/css")
27     @Styles.Render("~/bundles/style/css")
28
29 </head>
30
31 <!-- Scroll Up -->
32 <body class="black-bg">
33     <div id="back-top">
34         <a title="Go to Top" href="#"> <i class="fas fa-level-up-alt"></i></a>
35     </div>
36

```

Figura 5.1 – Layout-ul site-ului.

```

36
37 @Html.Partial("_Header")
38
39 @RenderBody()
40
41 @Html.Partial("_Footer")
42
43 <!-- JS here -->
44 @Scripts.Render("~/bundles/modernizr-3.5.0/js")
45 <!-- JQuery, Popper, Bootstrap -->
46 @Scripts.Render("~/bundles/jquery-1.12.4/js")
47 @Scripts.Render("~/bundles/popper/js")
48 @Scripts.Render("~/bundles/bootstrap/js")
49 <!-- JQuery Mobile Menu -->
50 @Scripts.Render("~/bundles/jquery.slicknav/js")
51 <!-- JQuery Slick , Owl-Carousel Plugins -->
52 @Scripts.Render("~/bundles/owl.carousel/js")
53 @Scripts.Render("~/bundles/slick/js")
54 <!-- One Page, Animated-Headlin -->
55 @Scripts.Render("~/bundles/ow/js")
56 @Scripts.Render("~/bundles/animated-headline/js")
57 @Scripts.Render("~/bundles/jquery.magnific-popup/js")
58 <!-- Date Picker -->
59 @Scripts.Render("~/bundles/gijgo/js")
60 <!-- Nice-select, sticky -->
61 @Scripts.Render("~/bundles/jquery.nice-select/js")
62 @Scripts.Render("~/bundles/jquery.sticky/js")
63 <!-- counter , waypoint,Hover Direction -->
64 @Scripts.Render("~/bundles/jquery.counterup/js")
65 @Scripts.Render("~/bundles/waypoints/js")
66 @Scripts.Render("~/bundles/jquery.countdown/js")
67 @Scripts.Render("~/bundles/hover-direction-snake/js")
68 <!-- contact js -->
69 @Scripts.Render("~/bundles/contact/js")
70 @Scripts.Render("~/bundles/jquery.form/js")
71 @Scripts.Render("~/bundles/jquery.validate/js")
72 @Scripts.Render("~/bundles/mail-script/js")
73 @Scripts.Render("~/bundles/jquery.ajaxchimp/js")
74 <!-- JQuery Plugins, main JQuery -->
75 @Scripts.Render("~/bundles/plugins/js")
76 @Scripts.Render("~/bundles/main/js")
77 </body>

```

Figura 5.2 – Layout-ul site-ului.

2.3 Familiarizarea cu structura modelului de proiectării BusinessLogic

Proiectul MVC ASP.NET poate fi împărțit în mod condiționat în 3 niveluri:

- Nivelul de prezentare, nivelul BusinessLogic și nivelul de acces la date. Această separare îmbunătățește procesul de dezvoltare și îmbunătățește performanța sistemului.
- Nivelul BusinessLogic încorporează întreaga logică de afaceri a proiectului, toate calculele necesare. Acest strat obține obiecte din stratul de acces la date și le transmite la nivelul reprezentărilor sau invers. Obiectele de afaceri stochează date și comportamentul, nu doar date.
- Pentru a simplifica compararea claselor de modele în proiectele MVC ASP.NET, se folosește extensia AutoMapper. Cu ajutorul acestei biblioteci devine posibilă conversia unui obiect în altul. Cartografia poate fi utilă în cazul acestora când obiectul depășește limitele aplicației sau nivelului.

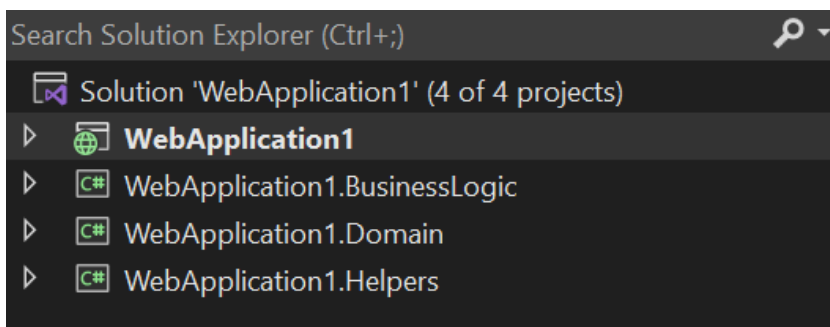


Figura 6 – Adăugarea bibliotecii claselor BusinessLogic, Domain, Helpers.

Crearea interfeței Isession în BusinessLogic în care vom crea UloginResp, HttpCookie și UserMinimal.

```

1  using System.Web;
2  using WebApplication1.Domain.Entities.User;
3
4  namespace WebApplication1.BusinessLogic.Interfaces
5  {
6      public interface ISession
7      {
8          ULoginResp UserLogin(ULoginData data);
9          HttpCookie GenCookie(string loginCredential);
10         UserMinimal GetUserByCookie(string apiCookieValue);
11     }
12 }

```

Figura 7 – Crearea clasei ISession în BusinessLogic.

Creare clasei SessionBL pentru scocarea datelor care sunt returnate din clasa Isession.

```

1  using System.Web;
2  using WebApplication1.BusinessLogic.Core;
3  using WebApplication1.BusinessLogic.Interfaces;
4  using WebApplication1.Domain.Entities.User;
5
6  namespace WebApplication1.BusinessLogic
7  {
8      public class SessionBL : UserApi, ISession
9      {
10         public ULoginResp UserLogin(ULoginData data)
11         {
12             return UserLoginAction(data);
13         }
14
15         public HttpCookie GenCookie(string loginCredential)
16         {
17             return Cookie(loginCredential);
18         }
19
20         public UserMinimal GetUserByCookie(string apiCookieValue)
21         {
22             return UserCookie(apiCookieValue);
23         }
24     }
25 }

```

Figura 8 – Crearea clasei SessionBL în BusinessLogic.

Aceasta clasă conține metoda care returnează un obiect care incapsuleaza interfața Isession.

```

1  using WebApplication1.BusinessLogic.Interfaces;
2
3  namespace WebApplication1.BusinessLogic
4  {
5      public class BussinesLogic
6      {
7          public ISession GetSessionBL()
8          {
9              return new SessionBL();
10         }
11     }
12 }

```

Figura 9 – Adăugarea clasei BusinessLogic.

Structura bibliotecii BusinessLogic este alcătuită din mai multe dosare: Core, DBModel, Interfaces și fișiere necesare a face legătura cu baza de date pentru a putea înregistra datele de pe web site.

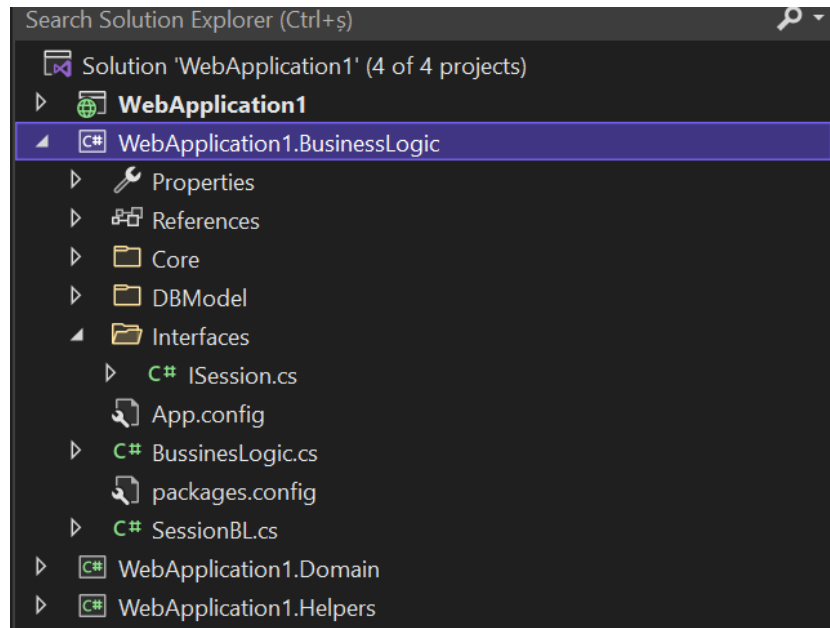


Figura 10 – Structura bibliotecii BusinessLogic.

În proiectul Domain se creaza dosarele Entities și Enums. Entities conține intităţi care vor fi utilizate mai târziu în lucrul cu baza de date. În dosarul Entities se crează un subfolder User cu două clase în interior: ULoginResp, UloginData.

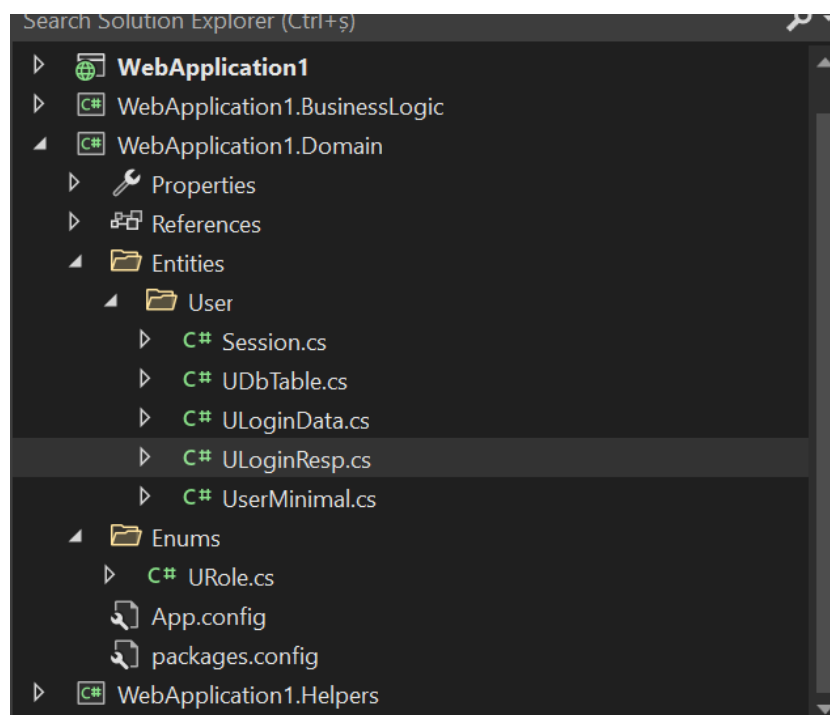
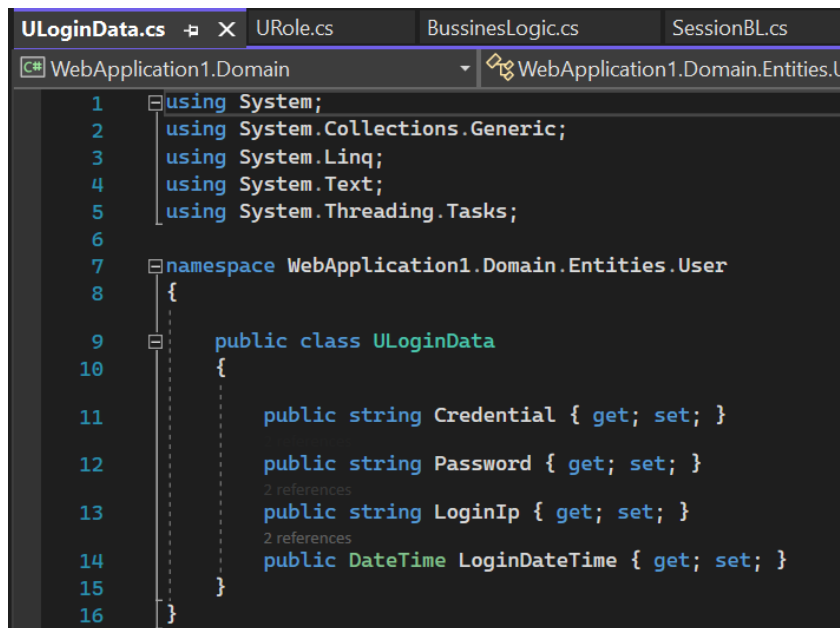


Figura 11 – Structura bibliotecii Domain.

În UloginData este utilizat pentru stocarea de date pentru logarea unui utilizator.



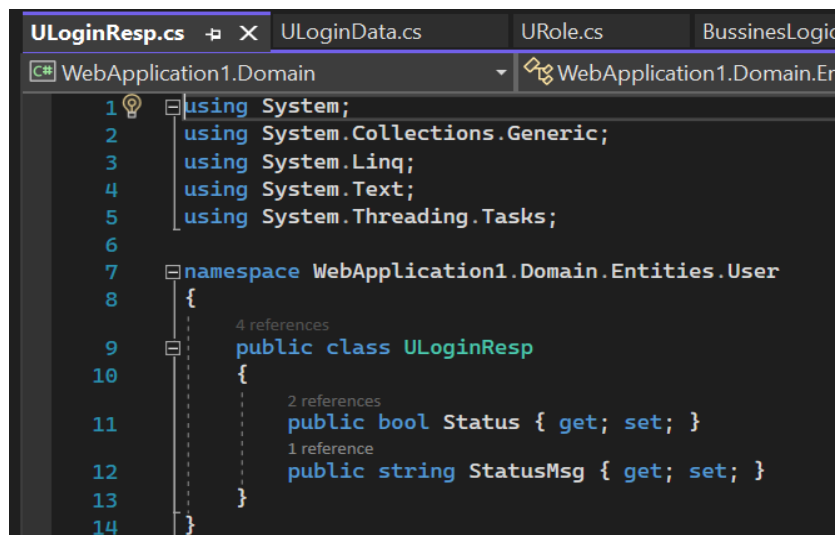
```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace WebApplication1.Domain.Entities.User
8  {
9      public class ULoginData
10     {
11         public string Credential { get; set; }
12         public string Password { get; set; }
13         public string LoginIp { get; set; }
14         public DateTime LoginDateTime { get; set; }
15     }
16 }

```

Figura 12 – Crearea clasei SessionBL în BusinessLogic.

În UloginResp se stochează datele referitor la statusul utilizatorului.



```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace WebApplication1.Domain.Entities.User
8  {
9      public class ULoginResp
10     {
11         public bool Status { get; set; }
12         public string StatusMsg { get; set; }
13     }
14 }

```

Figura 13 – Crearea clasei SessionBL în BusinessLogic.

Structura bibliotecii Helper este formată din clase, una care generează CookieGenerator și LoginHelper pentru logare.

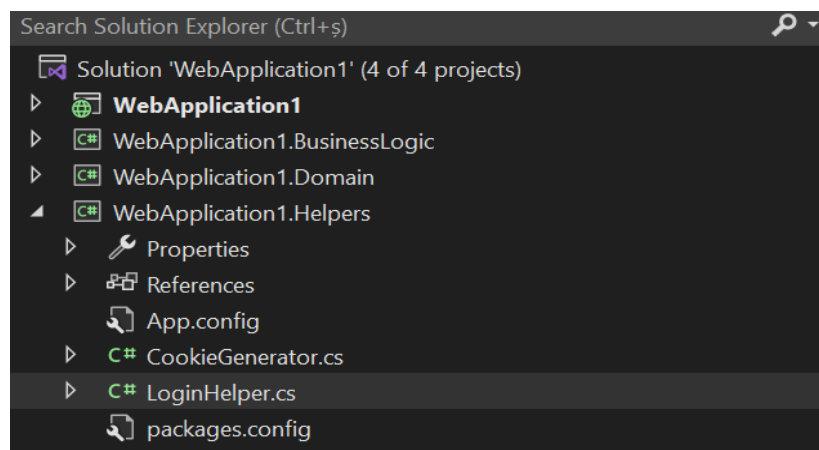


Figura 14 – Structura bibliotecii Helper.

2.4 Framework și conectarea la proiectul ASP.NET

O bază de date este o colecție de date care este stocată în conformitate cu schema de date proiectată. Definiția unei baze de date nu trebuie confundată cu definirea unui sistem de gestionare a bazelor de date (DBMS). DBMS - un set de instrumente software folosite pentru a crea și administra o bază de date. Acest laborator utilizează unul dintre sistemele de gestionare a bazelor de date client-server, Microsoft SQL Server, dezvoltat de Microsoft.

Entity Framework este o tehnologie orientată pe obiecte bazată în .NET framework pentru lucrul cu datele. Acest framework va permite lucrarea cu datele, indiferent de tipul acestora. Astfel, dacă la nivel fizic, dezvoltatorul operează cu tabele, chei, apoi la nivelul furnizat de cadrul entității, el lucrează cu obiecte, ceea ce simplifică foarte mult lucrul cu datele ca întreg. Conceptul central al Entity Framework este conceptul unei entități, care este un set de date asociate unui anumit obiect. Anume din această cauză tehnologia Entity Framework presupune lucrul cu obiectele, nu cu tabelele.

Instalarea NuGet package-ului Entity Framework pentru proiectul Web și BusinessLogic.

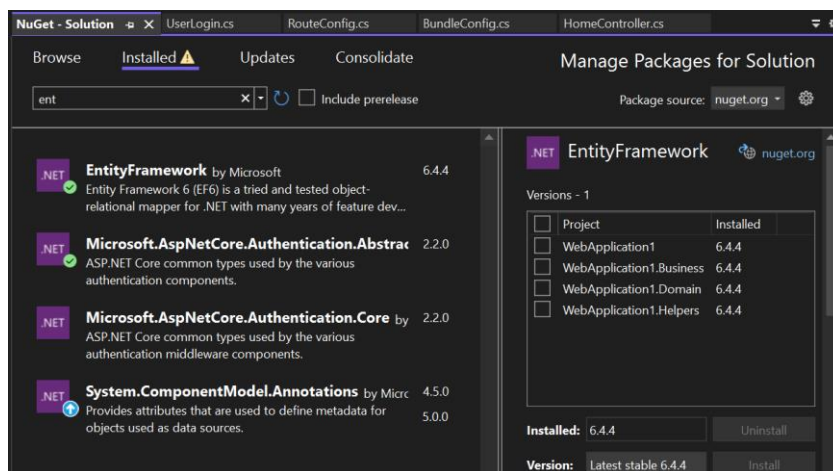


Figura 15 – Instalarea Entity Framework.

Adăugarea referinței System.ComponentModel.DataAnnotations pentru Domain layer.

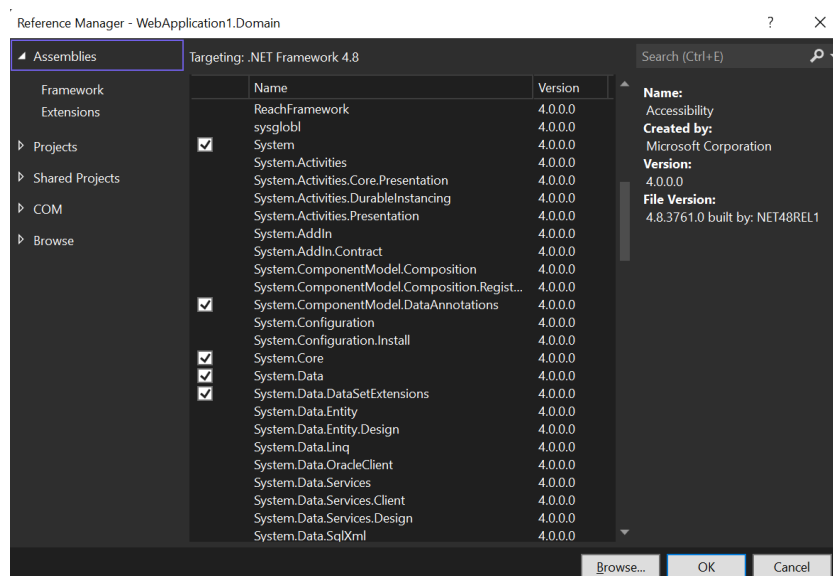


Figura 16 – Manager-ul referințelor pentru Domain.

Crearea acestei clase este una esențială în lucrul cu baza de date, astfel anume în această clasă se vor salva toată informația despre utilizator. După cum se observă, fiecare variabilă este anotată cu unele reguli, cum ar fi, Id primește [Key] ceea ce înseamnă că anume acest câmp va fi cheia principală din baza de date. [Required] reprezintă modul obligatoriu de a fi completat câmpul, [StringLength] indică lungimea maximă și minimă a câmpului.

```
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using WebApplication1.Domain.Enums;
namespace WebApplication1.Domain.Entities.User
{
    public class UDbTable
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        [Display(Name = "Username")]
        [StringLength(30, MinimumLength = 5, ErrorMessage = "Username cannot be longer than 30
characters.")]
        public string Username { get; set; }

        [Required]
        [Display(Name = "Password")]
        [StringLength(50, MinimumLength = 8, ErrorMessage = "Password cannot be shorter than 8
characters.")]
        public string Password { get; set; }

        [Required]
        [Display(Name = "Email Address")]
        [StringLength(30)]
        public string Email { get; set; }

        [DataType(DataType.Date)]
        public DateTime LastLogin { get; set; }

        [StringLength(30)]
        public string LasIp { get; set; }
```



```

    public URole Level { get; set; }
}
}

```

Crearea fișierului UserContext.cs în proiectul Business Logic

Datorită anumei acestei clase se va executa conexiunea la baza de date, iar datele fiind salvate în clasa UsersDbTable.

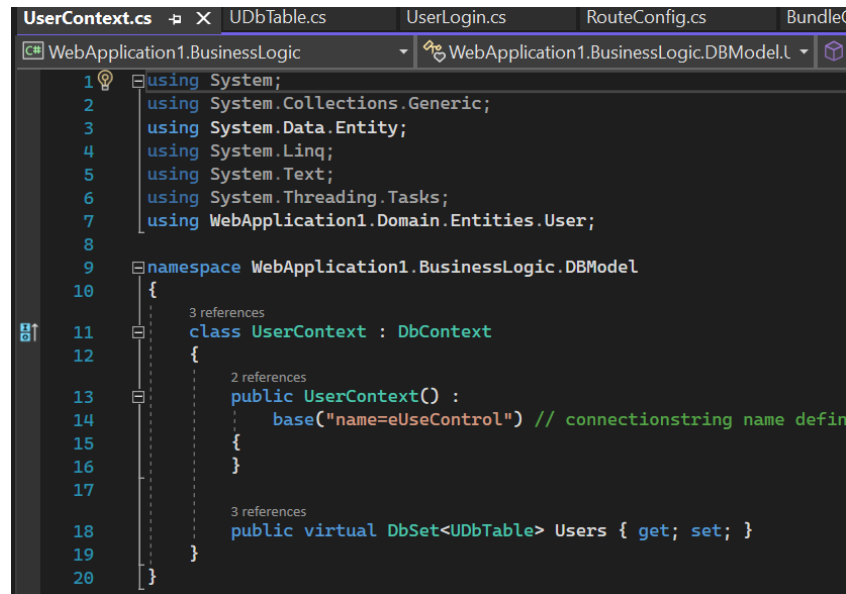


Figura 17 – UsersDbConext.cs.

Modificarea funcției de logare si inregistrare din Business Logic/UserAPI, pentru a ne conecta la baza de date.

```

using System;
using System.Collections.Generic;
using System.Data.Entity.Validation;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using WebApplication1.BusinessLogic.DBModel;
using WebApplication1.Domain.Entities.User;
using WebApplication1.Domain.Enums;
namespace WebApplication1.BusinessLogic.Core
{
    public class UserApi
    {
        internal ULoginResp UserLoginAction(ULoginData data)
        {

```

```

UsersDbTable result;
using (var db = new UserContext())
{
    result = db.Users.FirstOrDefault(u => u.Username == data.Credential && u.Password
    == data.Password);
}
if (result == null)
{
    return new ULoginResp { Status = false, StatusMsg = "The username or password is
incorrect" };
}
return new ULoginResp { Status = true };
}
internal ULoginResp UserRegisterAction(URegisterData data)
{
    UsersDbTable result;
    var user = new UsersDbTable();
    user.Email = data.Email;
    user.Username = data.Username;
    user.Password = data.Password;
    user.RegisterDate = data.RegisterDateTime;
    user.Level = URole.Guest;
    try
    {
        using (var db = new UserContext())
        {
            result = db.Users.FirstOrDefault(u => u.Username == data.Username);
            if (result == null)
            {
                db.Users.Add(user);
                db.SaveChanges();
                return new ULoginResp { Status = true };
            }
        }
    }
    else
    {

```

```

return new ULoginResp
{
    Status = false,
    StatusMsg = "Invalid Username."
};
} } }

catch (DbEntityValidationException e)
{
    throw;
} } } }

```

Adăugarea conexiunii în Web.config cu baza de date.

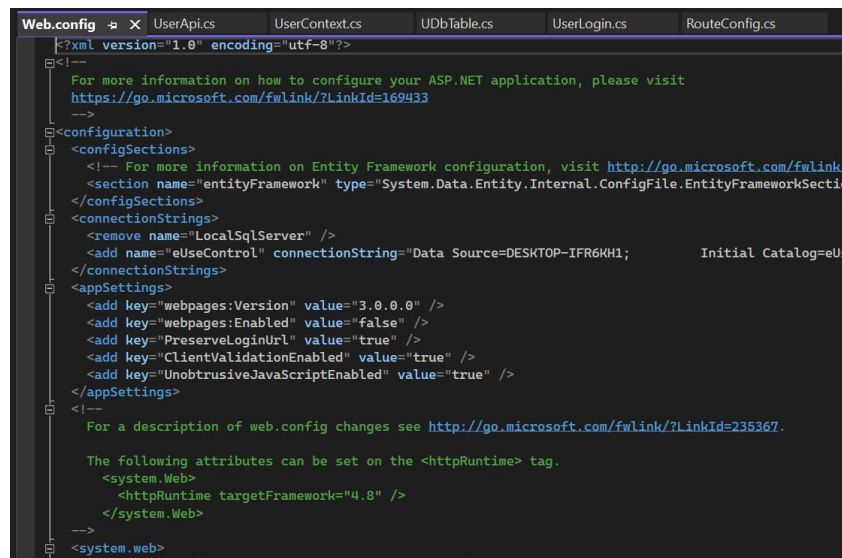


Figura 18 – Web.config.

Mai întâi trebuie de create conexiunea cu baza noastră de date.

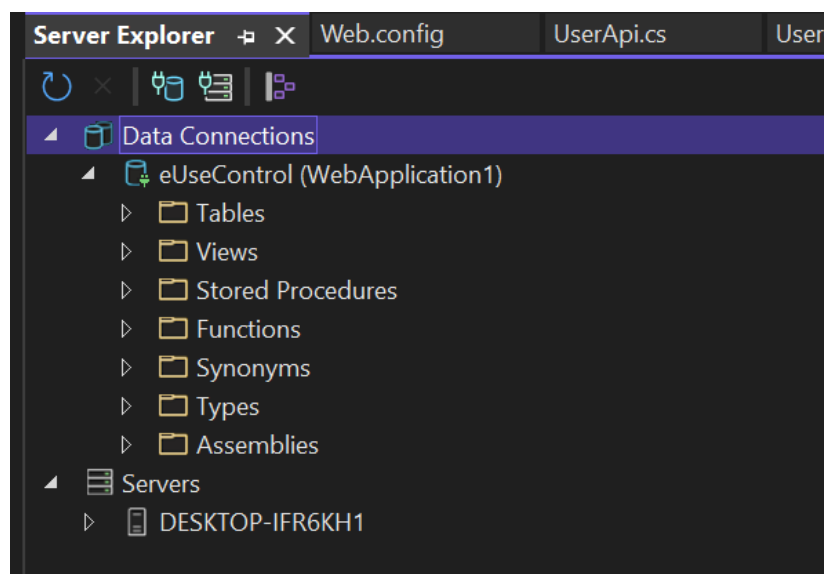


Figura 19 – Cu conexiune la baza de date.

Executăm conectarea la SQL-Server introducând numele calculatorul.

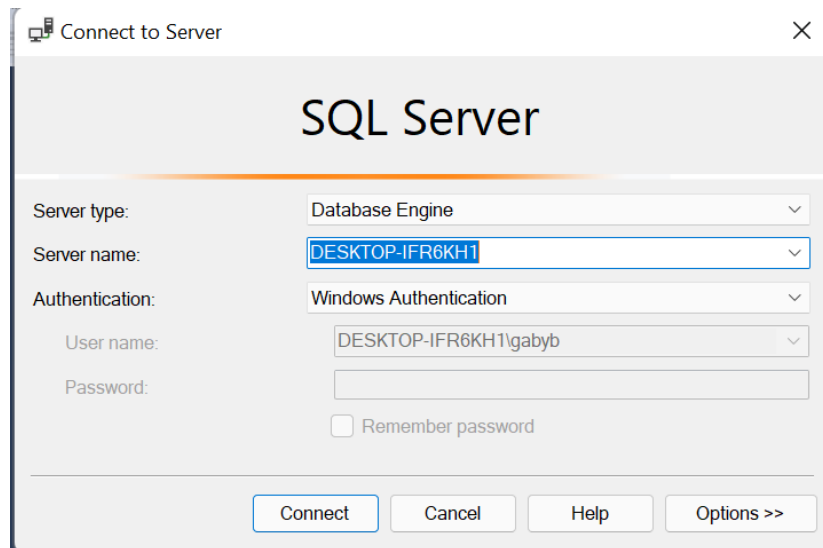


Figura 20 – Conectarea la server.

Verificarea dacă baza de date Microsoft SQL Server Management Studio lucrează și se stochează datele în ea de la înregistrare și logare. Pe care le putem vizualiza în tabelul sub numele UdbTable.

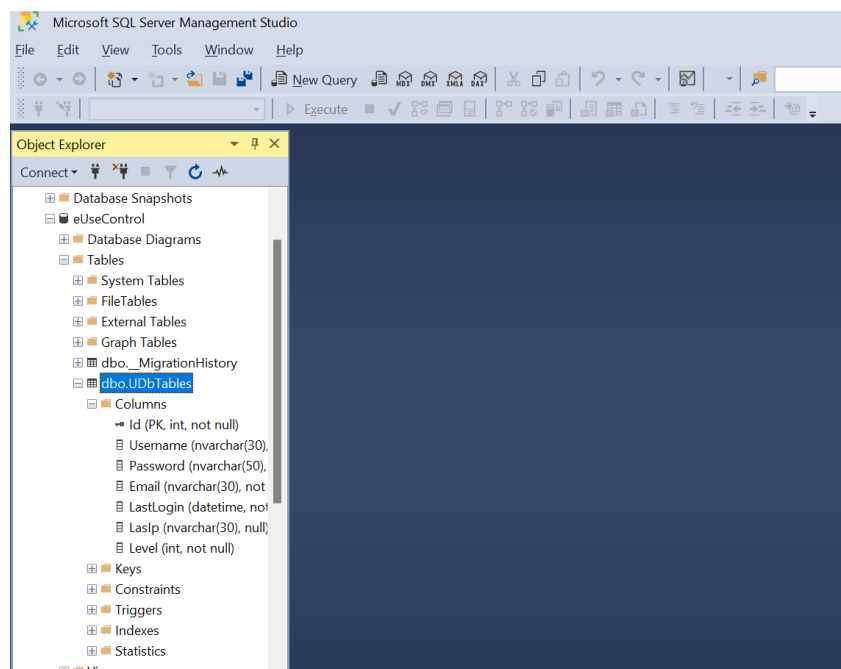


Figura 21 – Vizualizarea bazei de date.

3 Prezentarea generală a web site-ului de sport

În versiunea finală a site-ului, am realizat pagina de Home, About, Courses, Gallery, Blog, înregistrare/logare, Contact me. Navigarea între aceste pagini se realizează bine, prin butoanele care sunt apasate deasupra sau la sfârșit de pagină.

Vizualizarea rezultatului:

Pagina principală ne face cunoștință cu site-ul crea, ne spune care este denumirea site-ului, ne arată tematica lui și bara de meniu, care ne spune ce pagini există pe site-ul dat.



Figura 22 – Pagina principală.

Pagina About Me este creată pentru informația despre o persoană care se antrenează și dorește să se împartă cu detalii despre antrenamentele sale, sau cu sfaturi.

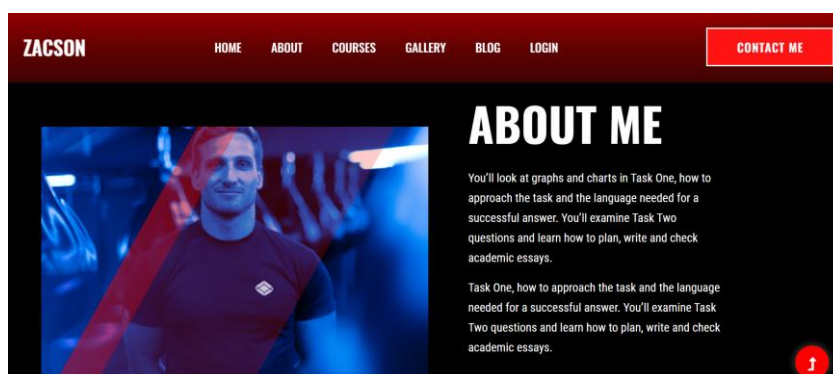


Figura 23 – Pagina Despre mine.

Pagina Courses este alcătuită din o mulțime de cursuri pe care se poate de înregistrat pentru frecventare.

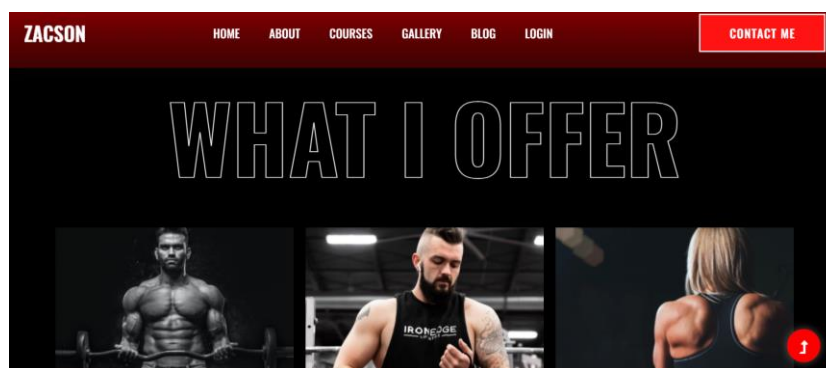


Figura 24 – Pagina Cursurilor.

Pe pagina Gallery este reprezentat multe imagini a persoanelor în cadrul antrenamentelor sale în salele de sport.

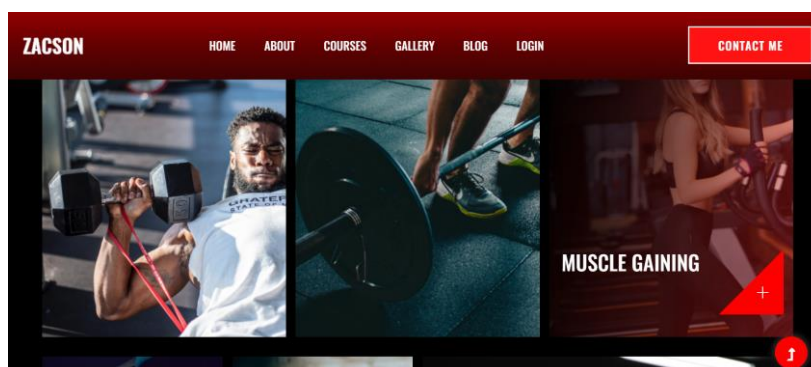


Figura 25 – Gallery.

Pagina Blog se ocupă cu informarea și discuția între persoane pe diferite teme sportive. Ca de exemplu ce rațion trebuie sa menții, ce tipuri de grafic este mai potrivit și multe altele.

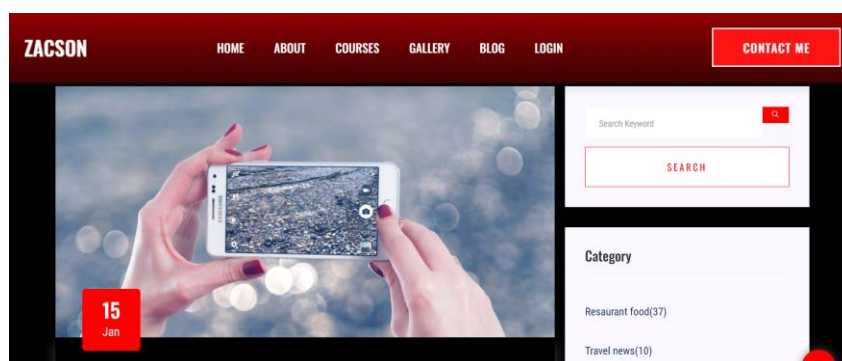


Figura 26 – Pagina Blog.

Pentru a contacta administratorul intrați pe pagina Contact Me pentru a putea scrie un mesaj.

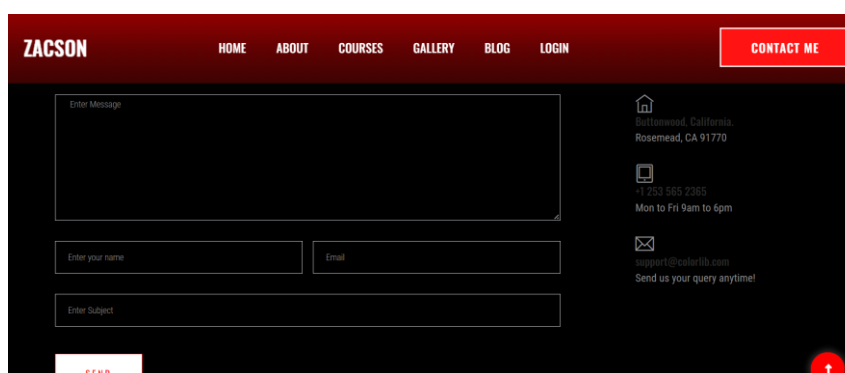


Figura 27 – Pagina de Contact.

Pentru a ne putea înregistra pe site intrăm pe pagina login și alegem Sign up și Login pentru a ne putea loga pe site.

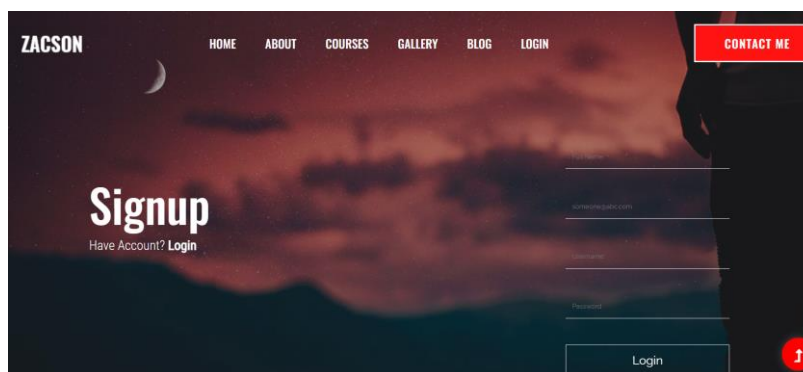


Figura 28 – Pagina de Înregistrare.

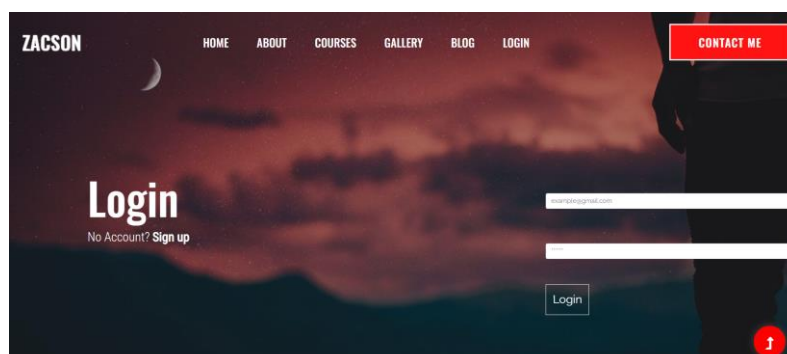


Figura 29 – Pagina de Login.

Concluzie

Pe parcursul realizării practicii, s-a analizat realizarea unui web site de sport utilizând tehnologia ASP.NET MVC. De aceea, a trebuie să se cerceteze și familiarizeze mai aprofundat cu limbajul C#, crearea claselor cât și crearea unui controler, view, model în cadrul aplicației.

Astfel, s-a studiat tehnologia ASP.NET, structura unei aplicații MVC, selectarea ulterioară a mediului de programare a aplicației și anume Visual Studio. În acest mod, crearea unui proiect ASP.NET în Visual Studio vine cu toate fișierele șablon de care avem nevoie pentru crearea serviciilor web ales, înainte de a adăuga ceva. Pattern-ul Model-View-Controller (MVC) separă o aplicație în trei componente principale: M: model (Model), V: vizualizare (View User Interface), C: controller (Controller).

Realizarea unui serviciu web necesită cunoștințe, implicare cât și dorință de a realiza un produs cât mai calitativ. Astfel în cadrul practicii am avut posibilitatea de a studia mai profund acest domeniu și anume: despre limbajul HTML, cum se poate simplu și ușor de creat pentru un web site. Despre CSS, cu ajutorul căruia am creat stilurile pentru ca totul să fie cât mai plăcut și de asemenea cu o tehnologie pentru crearea a unui sistem de design ca Bootstrap. Și deja limbajul C# cu ajutorul căruia am scris codul pentru controller, pentru Login și Sign.

Crearea mai multor biblioteci ca, BusinessLogic, Domain și Helper. Cu ajutorul Entity Framework și SQL au fost prelucrate date dintr-o bază de date în cadrul a ASP.NET. Cu ajutorul entităților de date s-a creat o bază de date cu tabele în ea, respective și cu câmpuri care urmează a fi completate de utilizator.

În concluzie, toate cunoștințele acumulate în mod practic, formează o bază bine pusă în tehnologia ASP.NET, cât și obținerea unor deprinderi practice de a realiza pagini HTML, CSS pentru anumite pagini din produsul web. Realizarea produsului web sunt și astăzi în plină dezvoltare, de aceea, se poate afirma cu certitudine că deprinderile acumulate în urma elaborării practicii vor fi utile și pe viitor.

Bibliografie

1. ASP.NET Formulare web – Tutorial. [Resursă electronică], accesat la data 09.04.2022. – Regim de acces: https://www.w3bai.com/ro/aspnet/aspnet_intro.html
2. ASP.NET Core 3.0. [Resursă electronică], accesat la data 09.04.2022. – Regim de acces: https://profs.info.uaic.ro/~iasimin/Special/ASP_NET_Core_2020.pdf
3. Noțiuni introductive cu ASP.NET MVC 5. [Resursă electronică], accesat la data 09.04.2022. – Regim de acces: <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/getting-started>
4. HTML - Prezentare generală. [Resursă electronică], accesat la data 10.04.2022. – Regim de acces: https://www.tutorialspoint.com/html/html_overview.htm
5. Ce este HTML? Limbajul de marcare. [Resursă electronică], accesat la data 10.04.2022. – Regim de acces: <https://www.hostinger.com/tutorials/what-is-html>
6. Ce este CSS? [Resursă electronică], accesat la data 10.04.2022. – Regim de acces: https://www.tutorialspoint.com/css/what_is_css.htm
7. Cum este structurat CSS. [Resursă electronică], accesat la data 09.04.2022. – Regim de acces: https://developer.mozilla.org/ru/docs/Learn/CSS/First_steps/How_CSS_is_structured
8. Ce este Bootstrap? [Resursă electronică], accesat la data 10.04.2022. – Regim de acces: <https://www.toptal.com/front-end/what-is-bootstrap-a-short-tutorial-on-the-what-why-and-how>
9. Limbajul de programare C#. [Resursă electronică], accesat la data 10.04.2022. – Regim de acces: <https://www.bairesdev.com/technologies/csharp/>
10. ASP.NET. [Resursă electronică], accesat la data 11.04.2022. – Regim de acces: <https://dotnet.microsoft.com/en-us/apps/aspnet>
11. Ce este un editor de text? [Resursă electronică], accesat la data 11.04.2022. – Regim de acces: https://developer.mozilla.org/ru/docs/Learn/Common_questions/Available_text_editors