

Ministerul Educației, Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Ingineria Software și Automatică

Raport de practică

Tema: Elaborarea unui magazin aplicol utilizând ASP.NET MVC

A efectuat: st. gr. TI-181 Vrabie Aliona

A controlat: lector universitar, Duca Ludmila

Chișinău 2020

Cuprins

Introducere.....	3
1. Analiza domeniului problemei.....	4
1.1. Considerații teoretice ASP.NET.....	4
1.2. Tehnologii software utilizate în cadrul realizării serviciului web.....	5
1.2.1. HTML.....	5
1.2.2.CSS.....	6
1.2.3. Bootstrap.....	7
1.2.4 C#.....	7
2. Descrierea principiului de lucru al tematicii alese.....	8
2.1. Implementarea si folosirea ASP.NET MVC în realizarea magazinului apicol.....	8
2.1.1 ASP.NET Controler C#.....	8
2.1.2 ASP.NET View C#.....	9
2.3 Proiectarea magazinului apicol utilizînd ASP.NET.....	10
2.2 Accesarea, proiectarea bazelor de date (EntityFramework).....	18
2.3 Prezentarea magazinului apicol.....	23
Concluzie.....	26
Bibliografie.....	27

Introducere

Astăzi suntem prezenți la o eră digitală fiind cea mai bună inovație a tuturor timpurilor.

Comerțul electronic oferă o capacitate imensă de conectivitate prin activitățile de cumpărare și vânzare din întreaga lume. În ultimele două decenii noile concepte de afaceri au evoluat datorită popularității internetului, oferind noi oportunități de afaceri pentru organizațiile comerciale și sunt în continuare influențate de activitățile utilizatorilor noilor aplicații ale Internetului. Tranzacțiile de afaceri sunt posibile printr-o combinație de procesare securizată a datelor, tehnologii de rețea și funcții de interactivitate. Modelele de afaceri sunt, de asemenea, supuse unor forțe externe continue de evoluție tehnologică, soluții inovatoare derivate prin concurență, crearea unor limite legale prin legislație și schimbări sociale.

Magazinul virtual are ca scop prezentarea și vinderea unor produse. Există posibilitatea realizării unor magazine virtuale pe platforme e-commerce software liber ca de exemplu: Open Cart, Magento, Woocommerce, Presta Shop.

Majoritatea comercianților mari, corporațiile sau brandurile, își dezvoltă propriile departamente de programare web și își creează magazine virtuale folosind soluții e-commerce avansate. Din această categorie amintim: Adidas, Puma, Boss, Nike etc. Astfel de magazine virtuale prezintă o multitudine de instrumente care ajută userii/clientii să ia decizii importante cu privire la comenzile lor.

Realizarea unui magazine apicol a devenit astăzi o necesitate și pentru sfera apicolă, această nouă opțiune în domeniu permite apicultorilor să își realizeze produsele mult mai rapid și comod pentru clienți.

1. Analiza domeniului problemei

1.1. Considerații teoretice ASP.NET

ASP.NET este o tehnologie Microsoft pentru crearea de aplicații web și servicii web. ASP.NET este succesorul lui ASP (Active Server 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 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: controler (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 MVC poate fi reprezentată astfel:

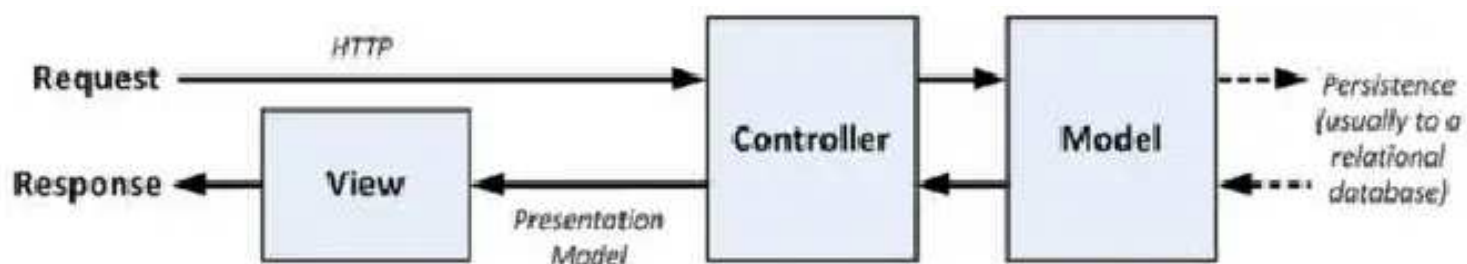


Figura 1 Interacțiunea într-o aplicație ASP.NET MVC

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 interfața aplicației. View este un template pentru a genera în mod dinamic HTML. Controller este o clasă specială ce gestionează relațiile dintre View și Model. Controller-ul răspunde la acțiunile utilizatorului, comunica cu modelul și decide ce vizualizare va afișa (daca exista unaStructura 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.

Views Conține fișiere template UI responsabile pentru afișarea rezultatului.

Scripts Conține biblioteci JavaScript si scripturi (.js).

Images Conține imaginile folosite în cadrul aplicației.

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

Filters Conține codul pentru filtre.

App_Data Conține fișierele de date Read/Write.

App_Start Conține cod de configurare, grupări de fișiere si 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[2]

1.2. Tehnologii software utilizate în cadrul realizării serviciului web

1.2.1. HTML

HTML este un limbaj de marcare folosit pentru crearea paginilor web, ce definește conținutul paginilor și modul de organizare și afișare al componentelor acestora prin utilizarea etichetelor și a atributelor[3]. Etichetele sunt reprezentate de cuvinte cheie plasate între caracterele „<” și „>” și sunt folosite pentru a defini elemente HTML. Marea majoritate a etichetelor formează perechi, ceea ce înseamnă că folosirea uneia dintre etichete presupune, implicit, și folosirea celeilalte. O astfel de pereche este alcătuită dintr-o etichetă de început și una de încheiere. Ambele coțin același cuvânt cheie plasat între caracterele „<” și „>”, diferența dintre ele fiind dată de faptul că, în cazul etichetei de încheiere, cuvântul cheie este precedat de caracterul „/”. Un exemplu de astfel de pereche ar fi <p> și </p>, folosită pentru definirea unui paragraf. Există însă și etichete ce nu prezintă o pereche. De exemplu, pentru definirea unei imagini este suficientă folosirea etichetei [4].

Pentru ca o pagină web să poată fi interpretată corect de către browser, și ,prin urmare, să poată fi afișată, trebuie să prezinte etichetele esențiale:<html> și perechea </html>, <head> și corespondentul </head>, <body> și eticheta de încheiere </body>. Prima pereche este folosită pentru a i se specifica browser-ului faptul că are de interpretat o pagină HTML. A treia pereche marchează secțiunea numită head ce conține informații asupra paginii precum titlul acesteia, referințele către fișerele JavaScript, etc iar ultima pereche marchează secțiunea numită body care conține toate componentele paginii vizibile în browser[5].

Având în vedere cele menționate anterior, codul utilizat pentru realizarea unei pagini web cu titlul „Acesta este un test”, ce afișează textul „Acesta este un paragraf” este:

```
<html>

<head>

<title>Acesta este un test</title>

</head>

<body>

<p>Acesta este un paragraf</p>

</body>

</html>
```

În ceea ce privește atributele, ce sunt specificate în eticheta de început, acestea sunt folosite pentru a oferi informații suplimentare asupra uneia dintre componentele paginii web (un paragraf, o imagine, etc). În general, atributele au asociat un nume și o valoare. Astfel, pentru alinierea centrală a unui paragraf se va folosi atributul cu numele align și valoarea center după cum urmează:

```
<p align="center">Acesta este un paragraf</p>
```

1.2.2.CSS

CSS este un limbaj de stilizare folosit pentru a descrie modul de prezentare a unui document scris în limbajul HTML. Cu alte cuvinte, CSS descrie cum vor fi afișate elementele unei pagini

web. La ora actuală, controlul aspectului unei pagini se face numai cu ajutorul CSS, ceea ce înseamnă că de la poziționarea unei componente pe pagină și până la culoarea textului și a fundalului, totul este implementat folosind acest limbaj. CSS elimină dezavantajul menționat anterior și simplifică procesul de dezvoltare al unui site web cu mai multe pagini prin faptul că toate formele de stilizare ce vor fi aplicate elementelor HTML vor fi menționate într-un fișier extern cu extensia .css. Astfel, pentru schimbarea aspectului mai multor pagini web vor fi necesare modificări doar în fișierul .css. Pentru ca diferitele forme de stilizare precizate în fișierele .css să poată fi încorporate într-o pagină web este necesară includerea acestora în pagină, mai exact, în fișierul .html, astfel: A doua linie de cod este folosită pentru a crea o legătură (un link) către fișierul .css. Atributul rel specifică relația dintre documentul curent (în cazul nostru pagina web) și documentul către care s-a creat legătura (fișier .css) și ia valoarea stylesheet, așa cum era de așteptat iar atributul type specifică tipul media al fișierului extern.

1.2.3. Bootstrap

Bootstrap este cel mai popular framework HTML, CSS și JavaScript folosit pentru dezvoltarea paginilor web a căror interfață se adaptează la rezoluția ecranului dispozitivului (desktop, laptop, tabletă, smartphone) de pe care este accesată pagina[19]. Bootstrap oferă numeroase facilități, punând la dispoziția dezvoltatorilor șabloane HTML și CSS pentru formulare, butoane, tabele, diferite animații, etc. Astfel, dezvoltatorul este scutit de o parte din muncă, rămânându-i sarcina de a particulariza șablonul prescriptiv, de exemplu, prin utilizarea propriului fișier .css. Pentru ca facilitățile oferite de Bootstrap să poată fi folosite este necesară includerea fișierelor .css și .js specifice Bootstrap în fișierul .html. Cea mai avantajoasă variantă este cea a includerii directe ținând cont de faptul că fișierele se află pe server-ele din rețeaua Bootstrap CDN.

1.2.4 C#

Limbajul C# este unul dintre cele mai utilizate limbaje de programare multiparadigmă din lume, fiind clasat în acest moment pe locul 5 mondial, ca nivel de popularitate. Este un limbaj simplu, modern, cu o flexibilitate foarte mare în ceea ce privește dezvoltarea de aplicații și portabilitatea acestora. Istoria acestui limbaj își are originile la începutul mileniului în care ne aflăm. În iulie 2000 a fost lansată prima distribuție a limbajului C#, deși zvonurile apariției unui limbaj puternic au apărut încă de prin anul precedent. Microsoft a fost corporația care a decis să intre mult mai în forță în lumea dezvoltatorilor, întrucât limbajul Basic își pierduse popularitatea ceea ce a dus la o ușoară scădere a nivelului economic al companiei.

Unul dintre principalele motive care au inițiat crearea acestui limbaj de programare a fost legat de faptul că în anul 1995, cei de la Sun Microsystems au creat limbajul Java, primul limbaj multiparadigmă cu o portabilitate foarte mare. A înregistrat un nivel de popularitate exponențial în

primii ani de la lansare, și chiar în zilele noastre, este pe primul loc mondial în ceea ce privește popularitatea. Astfel, Microsoft a decis să creeze un limbaj care să ofere mult mai multe elemente dezvoltatorilor.

Printre principalele calități ale limbajului, putem distinge:

- modernitate, simplitate, utilitate generală, productivitate mare
- stabilitate în cadrul aplicațiilor complexe, durabilitate
- este un limbaj total orientat pe obiect (orice entitate din acest limbaj este de fapt un obiect)
- oferă suport complet pentru dezvoltarea de componente necesare în medii distribuite, deci este și un limbaj
- orientat către componente

Limbajul C# are aplicabilitate foarte mare în cadrul sistemelor embeded, care includ dispozitive precum mp3 playere, semafoare, automate (vending machines), controlere, telefoane mobile, ceasuri digitale, s.a.m.d.

Deci acum știți că atunci când sunteți opriți la semafor, un algoritm transpus în C# coordonează controller-ul care dirijază fluxul de trafic.

2. Descrierea principiului de lucru al tematicii alese

2.1. Implementarea și folosirea ASP.NET MVC în realizarea magazinului apicol

2.1.1 ASP.NET Controller C#

Fiecare cerere este mapată la un controller particular. Controller-ul este responsabil pentru generarea răspunsului.

Un controller este o clasă derivată din `System.Web.Mvc.Controller`. Un controller expune acțiuni. Acțiunea este în fapt o metodă din cadrul clasei ce definește controller-ul.

- acțiune din controller trebuie să fie metoda publică în clasă.
- metoda folosită ca acțiune nu poate fi supraîncărcată și nu poate fi statică.

Rezultatul acțiunii – moștenite din clasa `ActionResult`.

O acțiune returnează ceea ce se numește rezultat acțiune. Tipurile suportate ca rezultat al unei acțiuni sunt :

1. `ViewResult` - Reprezintă HTML și markup.
2. `EmptyResult` – Reprezintă nici un rezultat.
3. `RedirectResult` – Reprezintă o redirectare la un nou URL.
4. `JsonResult` – Reprezintă un rezultat “JavaScript Object Notation” ce poate fi folosit într-o

aplicație AJAX.

5. `JavaScriptResult` - Reprezintă un script JavaScript.

6. `ContentResult` – Reprezintă un rezultat text.
7. `FileContentResult` – Reprezintă un fișier downladabil cu continut binar.
8. `FilePathResult` - Reprezintă un fișier downladabil (cu cale).
9. `FileStreamResult` - Reprezintă un fișier downladabil (cu stream).

Creare “Action” in cadrul unui Controller (C#)

Listing 1 - Controllers\HomeController.cs

```
using System.Web.Mvc;
using eUseControl.Web.Extension;
using eUseControl.Web.Models;

namespace eUseControl.Web.Controllers
{
    public class HomeController : BaseController
    {
        [HandleError]
        public ActionResult Index()
        {
            SessionStatus();
            var user = System.Web.HttpContext.Current.GetMySessionObject();
            if (user == null)
            {
                return View();
            }

            UserData u = new UserData
            {
                Username = user.Username
            };
            return View(u);
        }
    }
}
```

Pentru ca o metoda sa fie acțiune trebuie îndeplinite următoarele condiții;

- Trebuie sa fie publică.
- Nu poate fi statică.
- Nu poate fi o metoda extinsă.

2.1.2. ASP.NET View C#

View definește cum va arăta interfața aplicației. View este un document standard (X)HTML ce poate conține scripturi. Scripturile pot adăuga conținut dinamic la view.

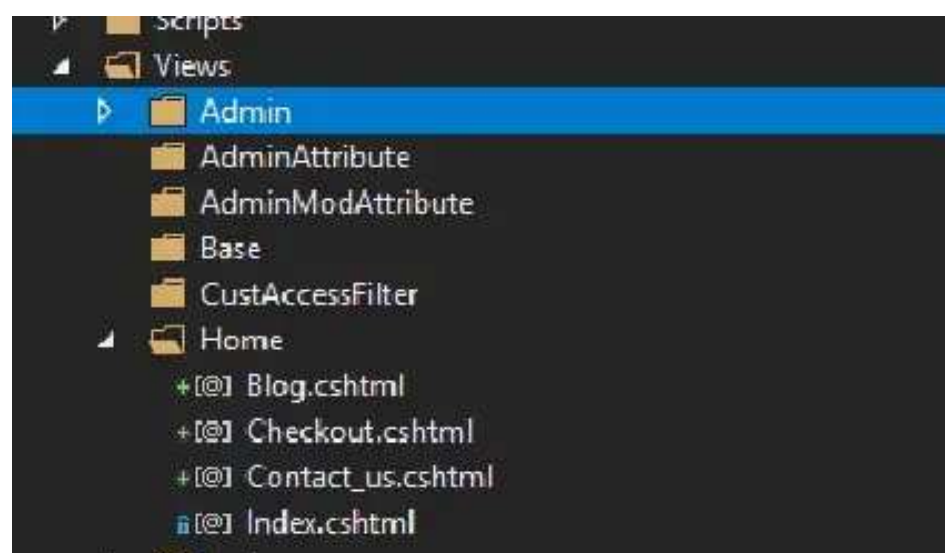


Figura 2. View în aplicație

2.3 Proiectarea magazinului apicol utilizând ASP.NET

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

1. Crearea unei soluții noi

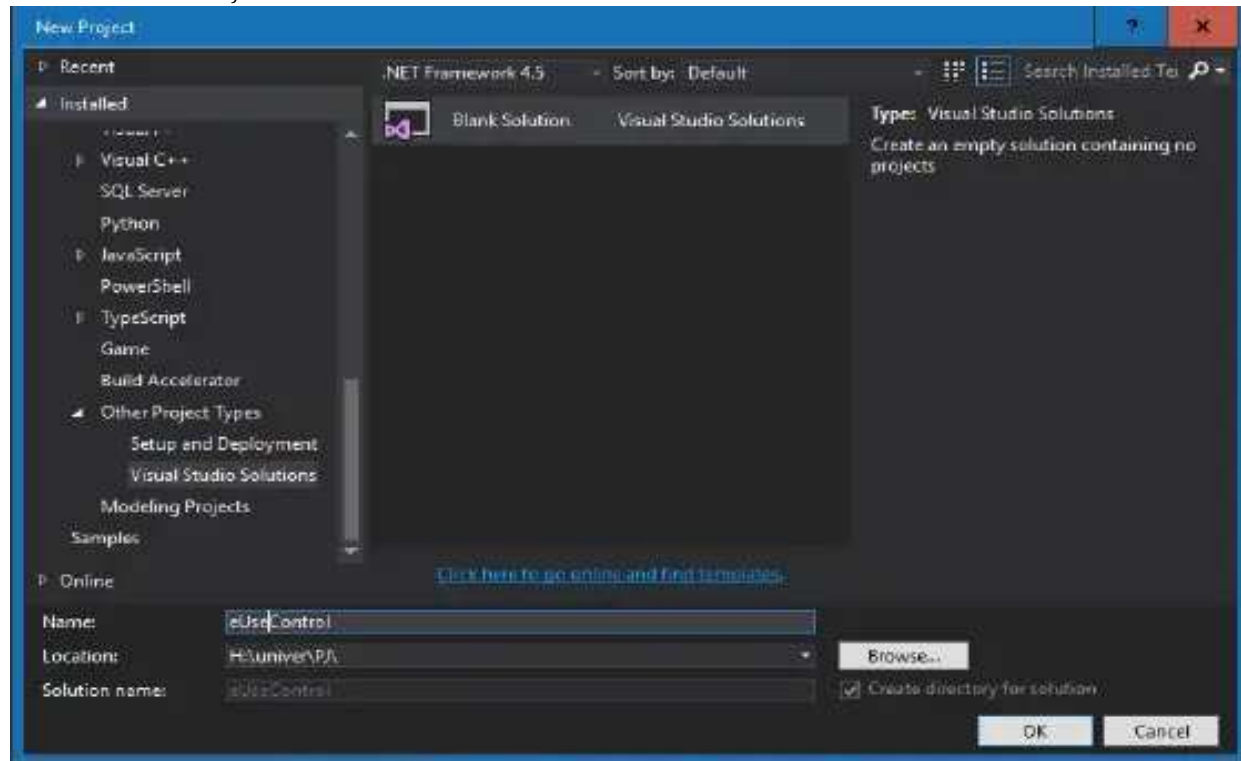


Figura3 Crearea unei soluții noi

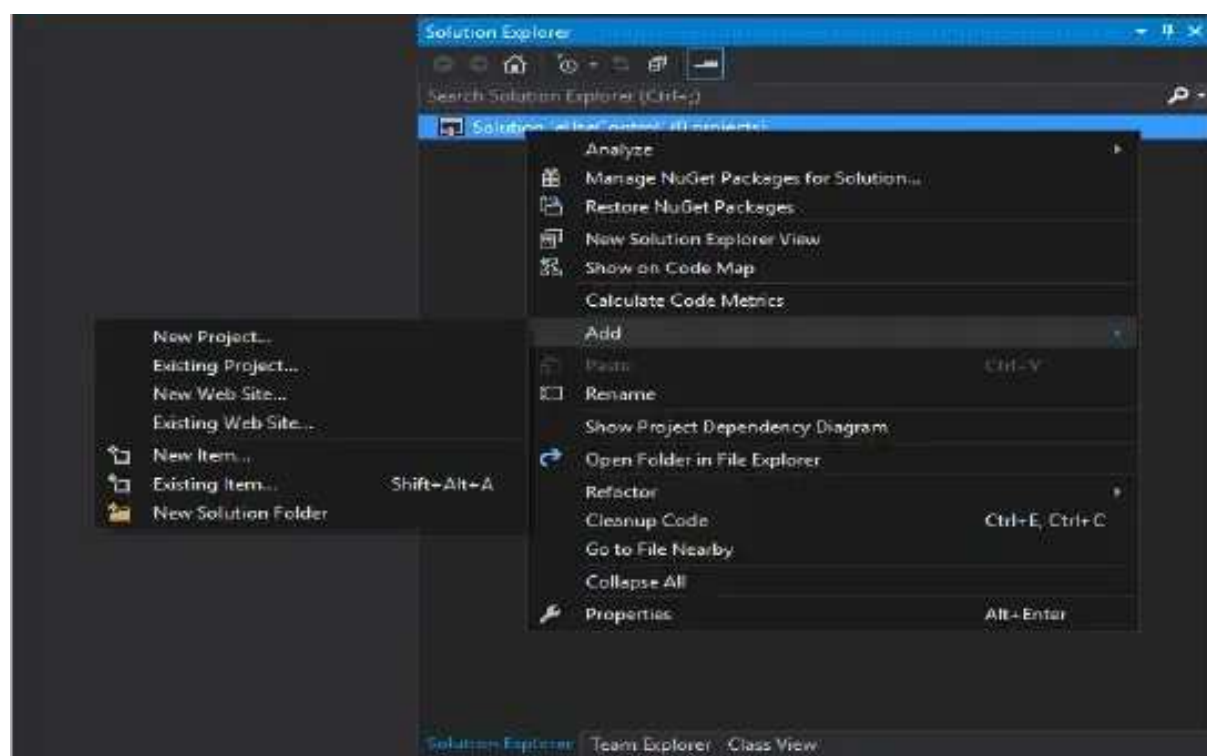


Figura 4 Adaugare proiect la soluție

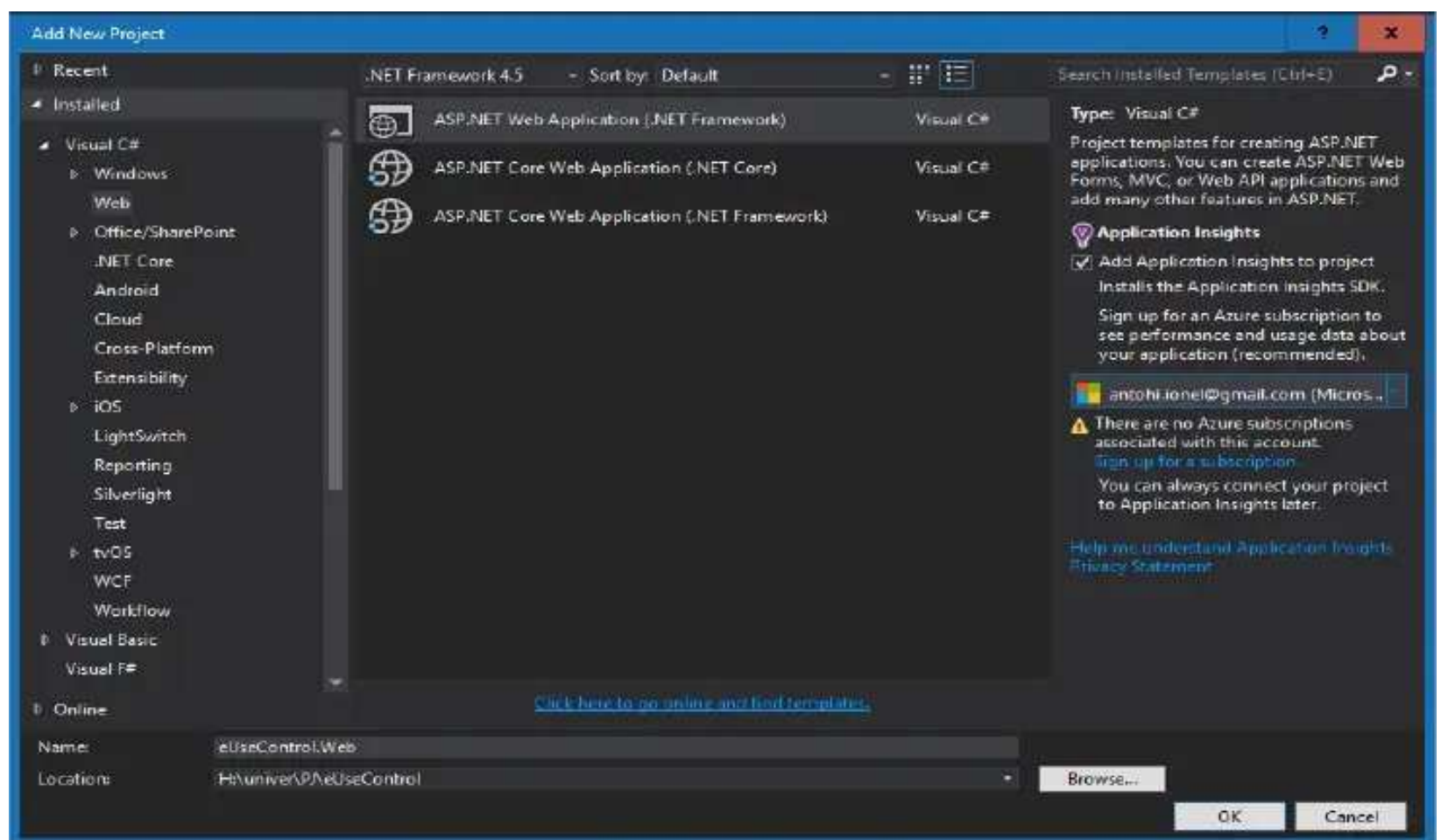


Figura 5 Crearea proiectului ASP.NET

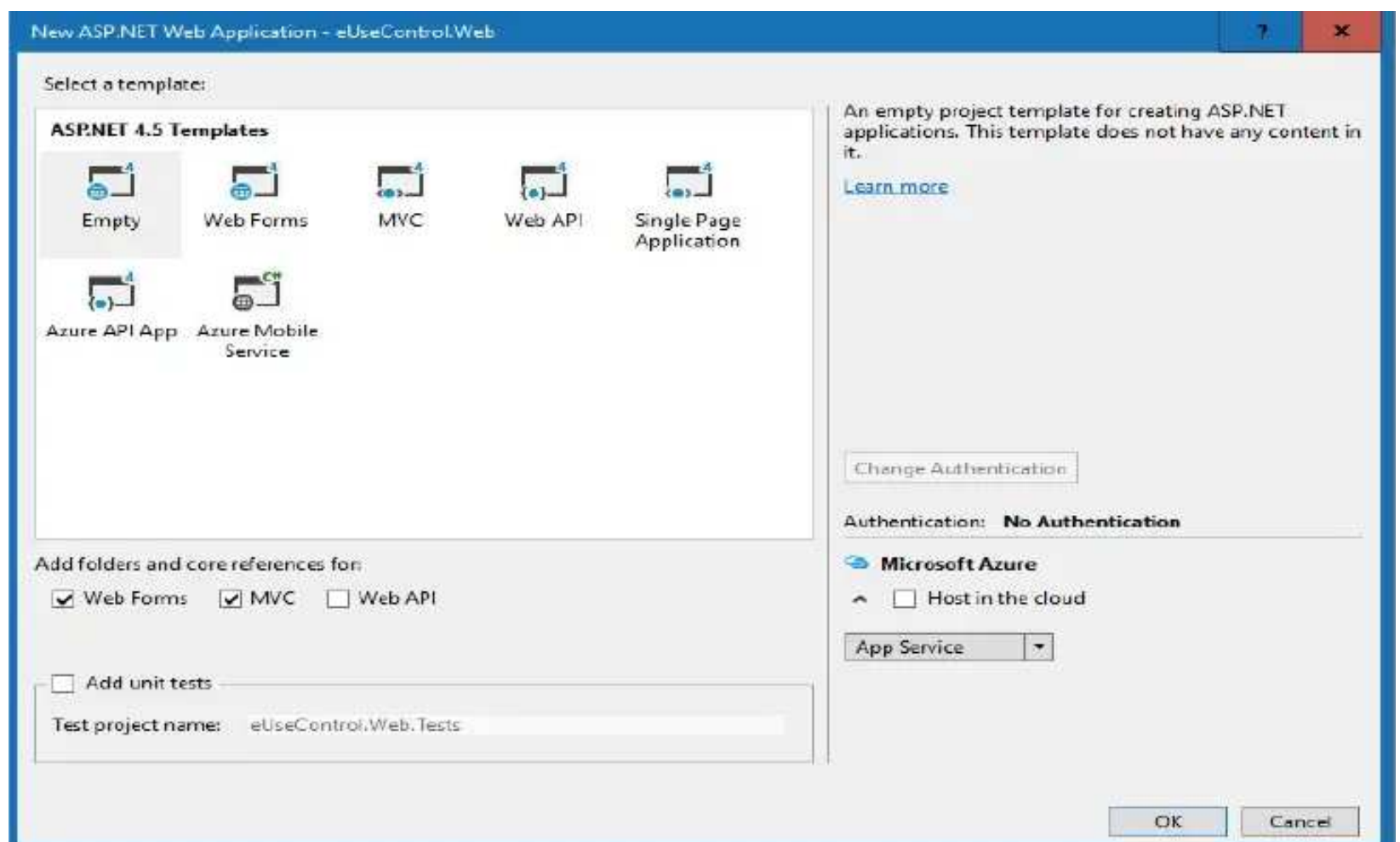


Figura 6 Parametrii proiectului

2. Instalarea tuturor packages prin NuGet

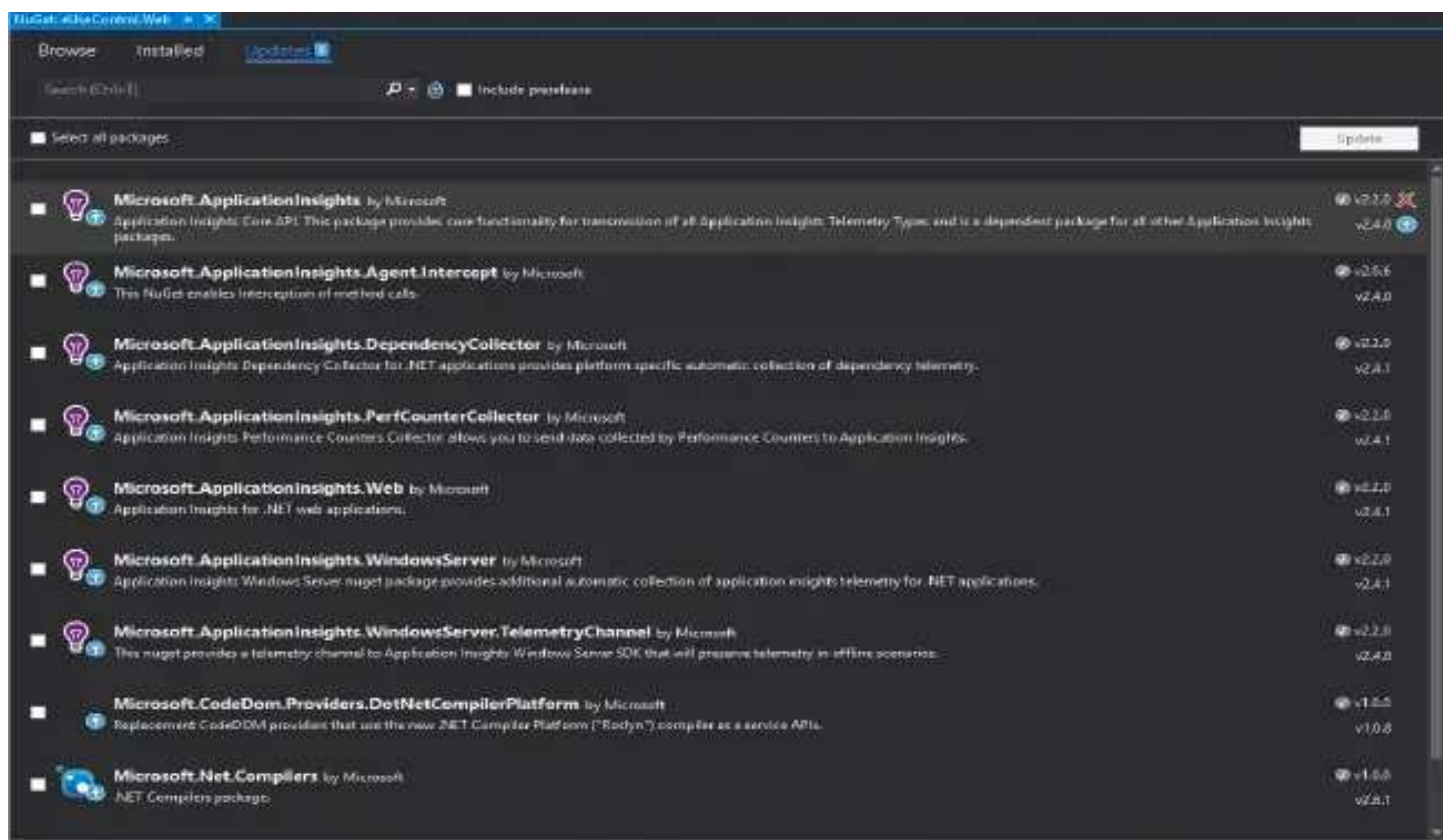


Figura 7 Instalarea packages

3. Crearea clasei BundleConfig în mapa App_Start

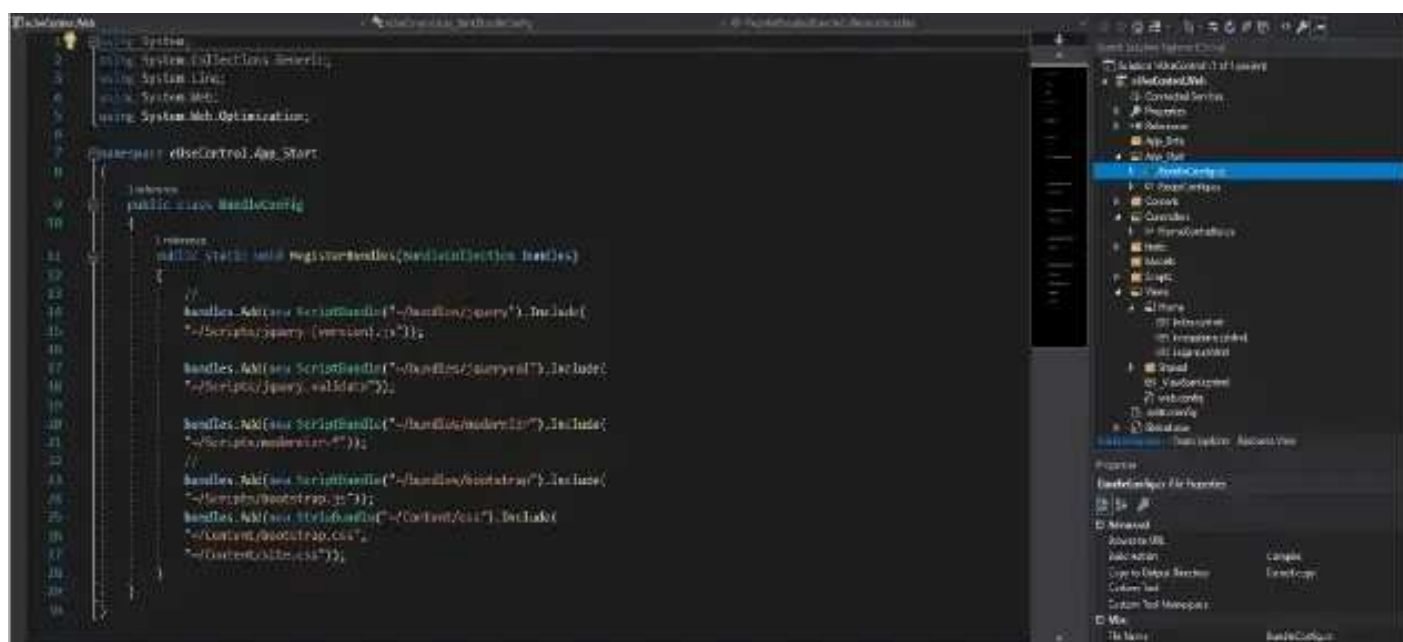


Figura 6 Fișierele care le conține BundleConfig.cs

Codul:

```
bundles.Add(new ScriptBundle("~/bundles/jquery").Include(
    "~/Scripts/jquery-{version}.js"));

bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include(
    "~/Scripts/jquery.validate"));

bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(
    "~/Scripts/modernizr-*"));
//
```



```

bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(
    "~/Scripts/bootstrap.js"));
bundles.Add(new StyleBundle("~/Content/css").Include(
    "~/Content/bootstrap.css",
    "~/Content/site.css"));

```

Înregistrarea clasei în Global.asax

```
BundleConfig.RegisterBundles(BundleTable.Bundles);
```

4. Crearea HomeController în mapa Controllers

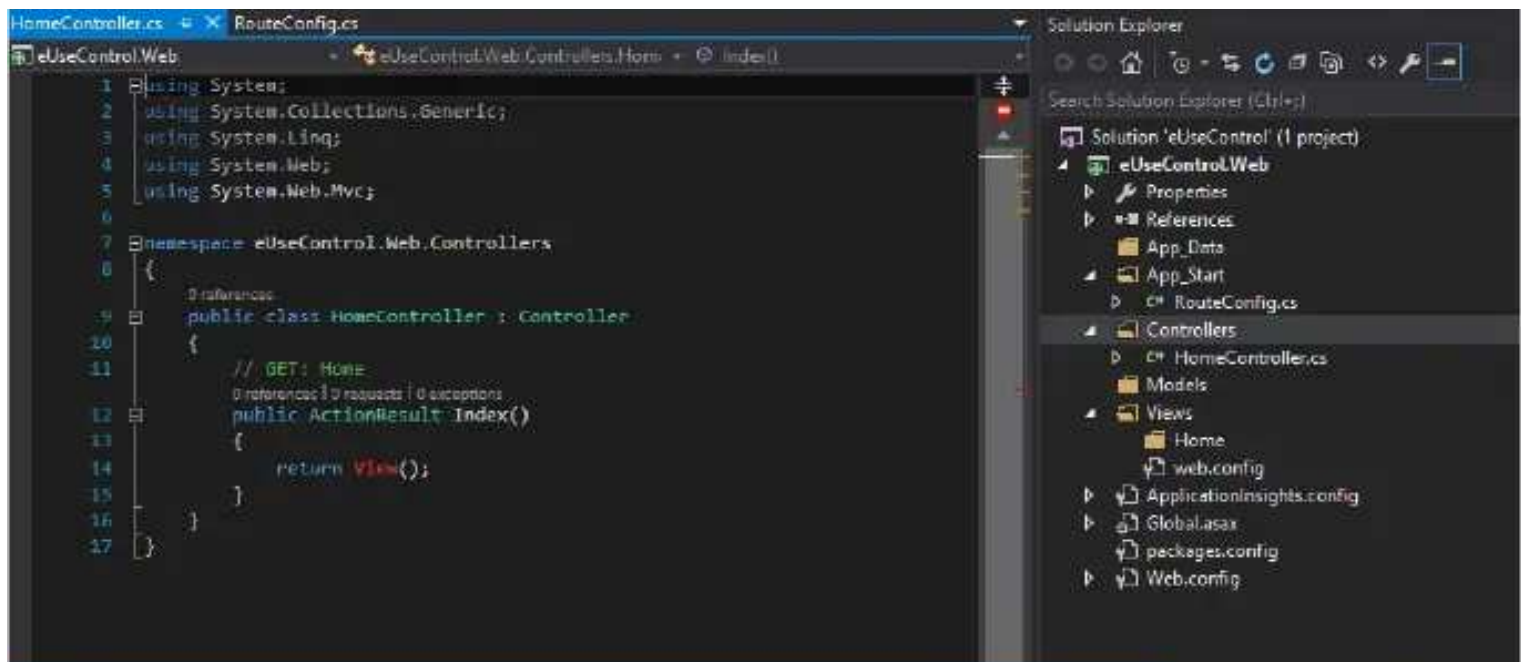


Figura 8 Crearea HomeController

5. Crearea Fisierului Index in Views/Home



Figura 9 Conținutul Indexului

Index.cshtml conține codul paginii principale a paginii web.

În mapa Views se creaza mapa Shared cu fişierele cshtml _Layout, _Footer, _Header.

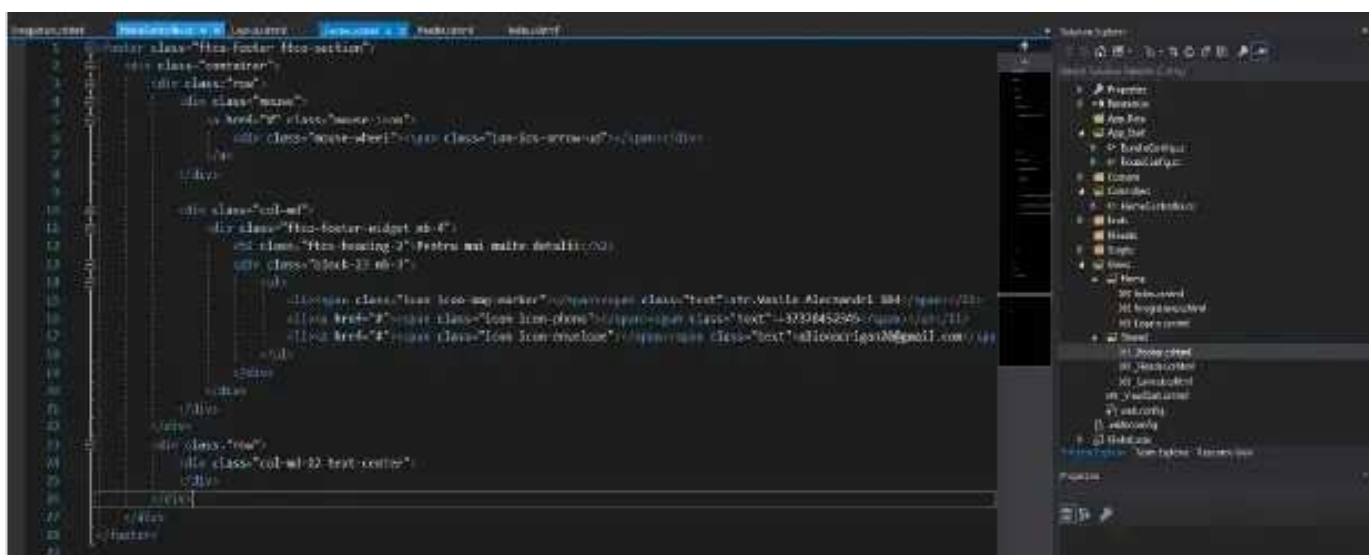


Figura 10 _Footer

Conținutul _Header

```
<div class="py-1 bg-primary">
  <div class="container">
    <div class="row no-gutters d-flex align-items-start align-items-center px-md-0">
      <div class="col-lg-12 d-block">
        <div class="row d-flex">
          <div class="col-md pr-4 d-flex toppler align-items-center">
            <div class="icon mr-2 d-flex justify-content-center align-items-center"><span class="icon-phone2"></span></div>
            <span class="text">068345851</span>
          </div>
          <div class="col-md pr-4 d-flex toppler align-items-center">
            <div class="icon mr-2 d-flex justify-content-center align-items-center"><span class="icon-paper-plane"></span></div>
            <span class="text">alionacrigan20@gmail.com</span>
          </div>
          <div class="col-md-5 pr-4 d-flex toppler align-items-center text-lg-right">
            <span class="text">Pentru o viata sanatoasa & Fericitate</span>
          </div>
        </div>
      </div>
    </div>
  </div>
```

```

        </div>
</div>
<nav class="navbar navbar-expand-lg navbar-dark ftco_navbar bg-dark ftco-navbar-light"
id="ftco-navbar">
    <div class="container">
        <a class="navbar-brand" href="@Url.Action("Index","Home") ">ApiAlbina </a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#ftco-nav" aria-controls="ftco-nav" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="oi oi-menu"></span> Menu
        </button>

        <div class="collapse navbar-collapse" id="ftco-nav">
            <ul class="navbar-nav ml-auto">
                <li class="nav-item active"><a href="index.html" class="nav-
link">Despre noi</a></li>
                <li class="nav-item dropdown">
                    <a class="nav-link dropdown-toggle" href="#"
id="dropdown04" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">Produsele
noastre</a>
                    <div class="dropdown-menu" aria-labelledby="dropdown04">
                        <a class="dropdown-item" href="shop.html">Miere</a>
                        <a class="dropdown-item"
href="wishlist.html">Produse din propolis</a>
                        <a class="dropdown-item" href="product-
single.html">Lapte de regina</a>
                        <a class="dropdown-item" href="cart.html">Polen
crud</a>
                        <a class="dropdown-item"
href="checkout.html">Produse din ceara</a>
                    </div>
                </li>
                <li class="nav-item"><a class="nav-link">Noutati</a></li>
                <li class="nav-item"><a class="nav-link">Blog</a></li>
                <li class="nav-item"><a class="nav-link">Blog</a></li>
                <li class="nav-item"><a class="nav-link">Contacte</a></li>
                <li class="nav-item cta cta-colored"><a class="nav-link"><span
class="icon-shopping_cart"></span>[0]</a></li>
                <li class="nav-item"><a class="nav-link"
href="@Url.Action("Logare","Home")">Logare</a></li>
                <li class="nav-item"><a class="nav-link"
href="@Url.Action("Inregistrare","Home")">Inregistrare</a></li>
            </ul>

```

```

        </div>
    </div>
</nav>

```

Coținutul _Layout

```

@using System.Web.Optimization
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <title>@ViewBag.Title|ApiAlbina</title>
    <link href="~/Content/Site.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/bootstrap.min.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/animate.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/aos.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/owl.carousel.min.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/open-iconic-bootstrap.min.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/owl.theme.default.min.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/magnific-popup.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/ionicons.min.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/bootstrap-datepicker.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/jquery.timepicker.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/flaticon.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/icomoon.css" rel="stylesheet" type="text/css" />
    <script src="~/Scripts/modernizr-2.8.3.js"></script>
    <link href="https://fonts.googleapis.com/css?family=Poppins:200,300,400,500,600,700,800&display=swap" rel="stylesheet">
    <link href="https://fonts.googleapis.com/css?family=Lora:400,400i,700,700i&display=swap" rel="stylesheet">
    <link href="https://fonts.googleapis.com/css?family=Amatic+SC:400,700&display=swap" rel="stylesheet">

    @Styles.Render("~/Content/css")
    @Styles.Render("~/bundles/animate/css")
    @Scripts.Render("~/bundles/modernizr")

    <!--Add local styles, mostly for plugins css file-->
    @if (IsSectionDefined("Styles"))
    {
        @RenderSection("Styles", required: false)
    }

    <!--Primary Home style-->
    @Styles.Render("~/bundles/main/css")

```



```

</head>
<body>
  <div class="wrapper">
    <!--Header-->
    <nav>
      @Html.Partial("_Header");
    </nav>

    <!--main view-->
    <section id="home-section" class="hero">
      @RenderBody()

      <!--footer-->

      @Html.Partial("_Footer")
    </section>

    <!--Section for main scripts render-->
    @Scripts.Render("~/Scripts/jquery-3.4.1.min.js")
    @Scripts.Render("~/bundles/bootstrap/js")

    <!--Handler for local scripts-->
    @RenderSection("scripts", required: false)

  </div>

  <script src="Scripts/jquery.min.js"></script>
  <script src="Scripts/jquery-migrate-3.0.1.min.js"></script>
  <script src="Scripts/popper.min.js"></script>
  <script src="Scripts/bootstrap.min.js"></script>
  <script src="Scripts/jquery.easing.1.3.js"></script>
  <script src="Scripts/jquery.waypoints.min.js"></script>
  <script src="Scripts/jquery.stellar.min.js"></script>
  <script src="Scripts/owl.carousel.min.js"></script>
  <script src="Scripts/jquery.magnific-popup.min.js"></script>
  <script src="Scripts/aos.js"></script>

```

```

<script src="Scripts/jquery.animateNumber.min.js"></script>
<script src="Scripts/bootstrap-datepicker.js"></script>
<script src="Scripts/scrollax.min.js"></script>
<script src="https://maps.googleapis.com/maps/api/js?
key=AIzaSyBWaKrjvy3MaE7SQ74_uJiULgl1JY0H2s&sensor=false"></script>
<script src="Scripts/google-map.js"></script>
<script src="Scripts/main.js"></script>

</body>
</html>

```

2.2 Accesarea, proiectarea bazelor de date (EntityFramework)

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. În figura 3 se ilustrează modul în care Cadrul de entități se încadrează în aplicația dvs. Din figură se vede că Entity Framework se încadrează între entitățile de afaceri (clase de domeniu) și baza de date. Salvează datele stocate în proprietățile entităților comerciale și, de asemenea, preia datele din baza de date și le transformă automat în obiecte de entități de afaceri.

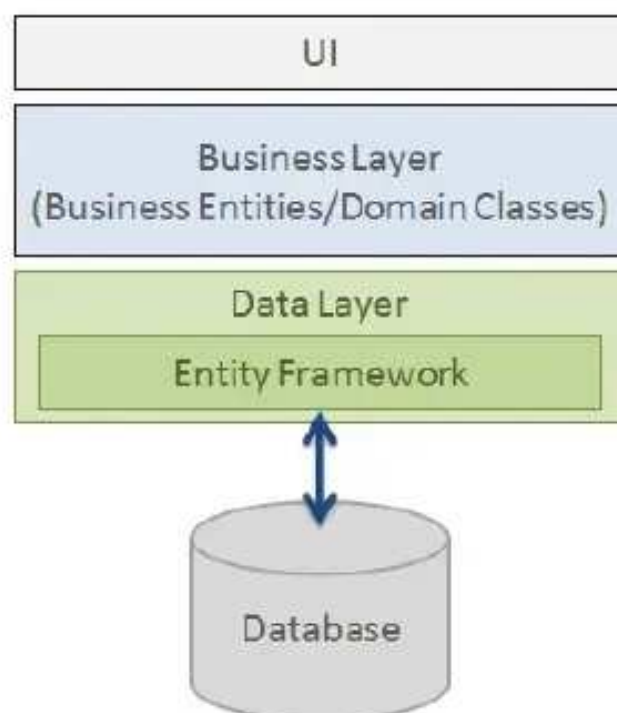


Figura 11. Entity Framework.

1. La primul pas se instalează NuGet package-ului Entity Framework pentru proiectul Web și BusinessLogic.

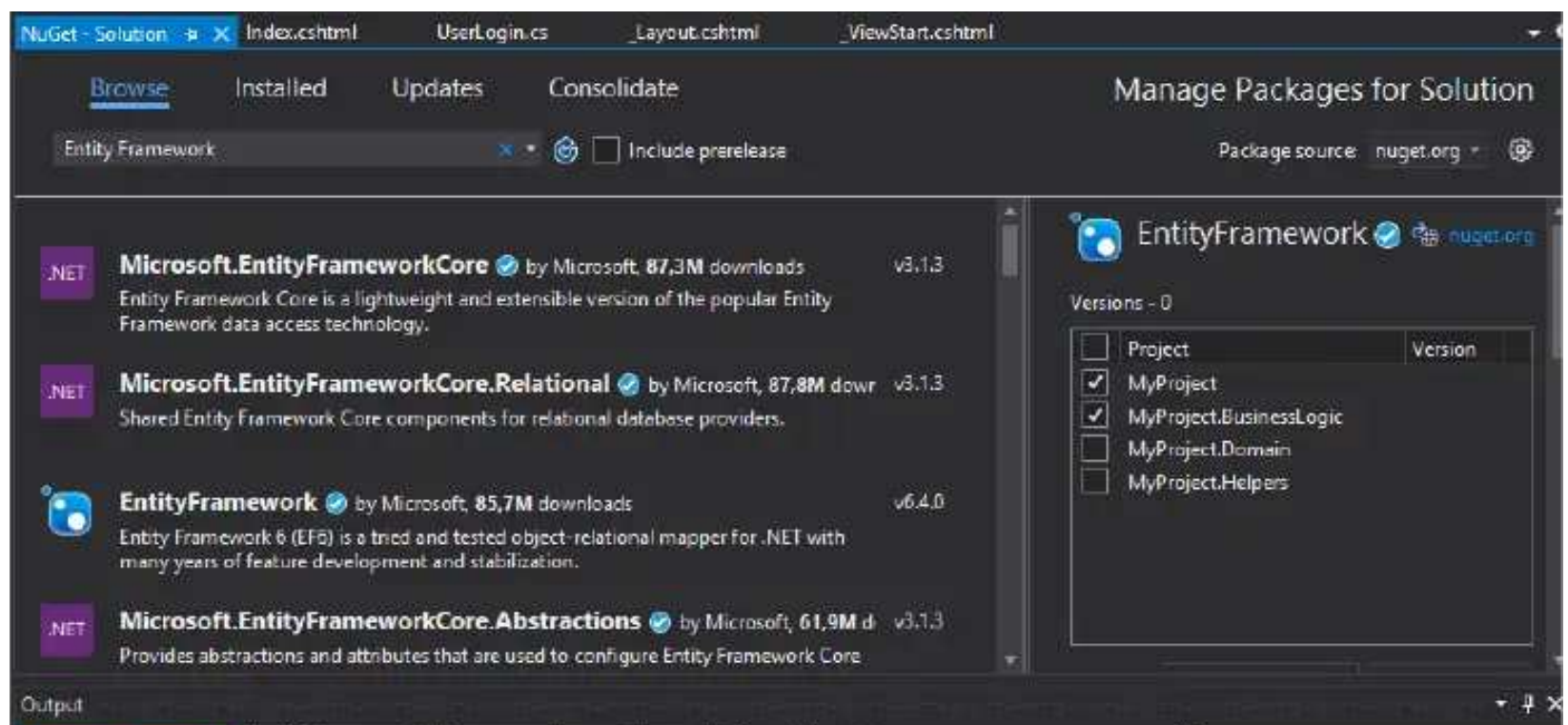


Figura 12. NuGet package-ului Entity Framework.

2. Adăugarea referinței `System.ComponentModel.DataAnnotations` pentru Domain layer. Spațiul de nume [System.ComponentModel.DataAnnotations](#) oferă clase de atribut care sunt utilizate pentru a defini metadatele pentru controalele de date ASP.NET MVC și ASP.NET. (Întrucât Domain nu conținea nici o referință, s-a instalat spațiul de nume din pachetul NuGet, aceasta este arătat în figura 5).

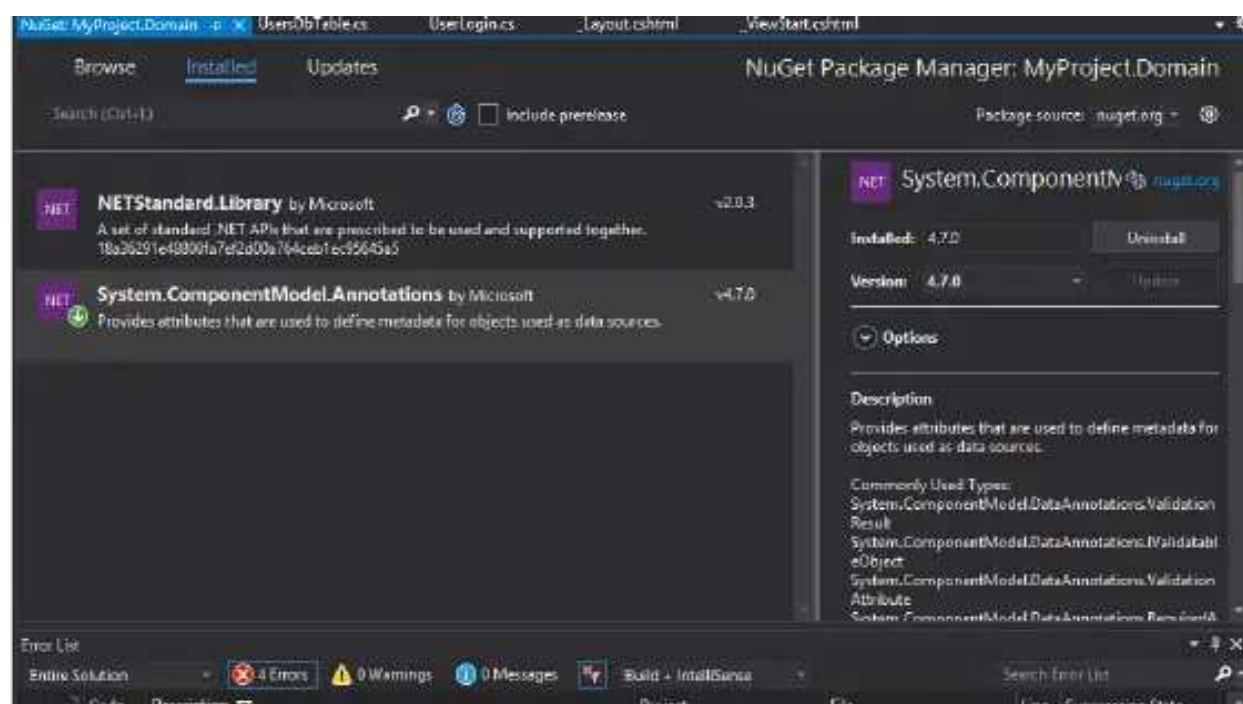


Figura 13. Spațiul de nume [System.ComponentModel.DataAnnotations](#)

3. Crearea fișierului `UsersDbTable.cs` în proiectul 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.

```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.Threading.Tasks;
5  using System.ComponentModel.DataAnnotations;
6  using System.ComponentModel.DataAnnotations.Schema;
7  using System.Data;
8
9  namespace MyProject.Domain.Entities.User
10 {
11     [References]
12     public class UsersDbTable
13     {
14         [Key]
15         [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
16         public int Id { get; set; }
17         [Required]
18         [Display(Name = "Username")]
19         [StringLength(30, MinimumLength = 4, ErrorMessage = "Username must be less than 30 characters")]
20         public string Username { get; set; }
21         [Required]
22         [Display(Name = "Password")]
23         [StringLength(40, MinimumLength = 8, ErrorMessage = "Password must be more than 8 characters")]
24         public string Password { get; set; }
25     }
26 }

```

Figura 14. UsersDbTable.

4. 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.

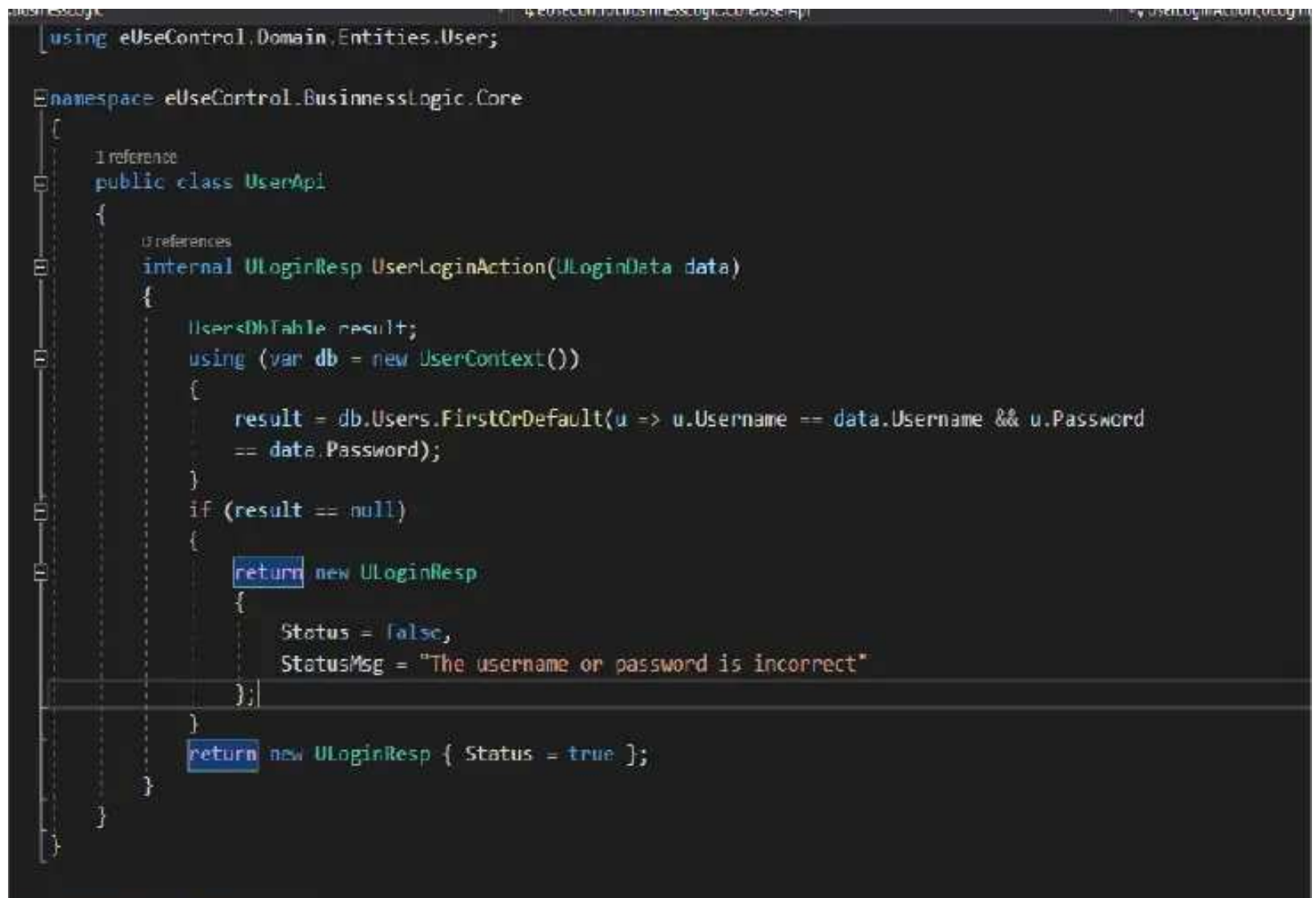
```

1  using System;
2  using System.Collections.Generic;
3  using System.Data.Entity;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using eUseControl.Domain.Entities.User;
8
9  namespace eUseControl.BusinessLogic.DBModel.Seed
10 {
11     [2 references]
12     public class UserContext : DbContext
13     {
14         [1 reference]
15         public UserContext() :
16             base("name=eUseControl")
17         {
18         }
19         [1 reference]
20         public virtual DbSet<UsersDbTable> Users { get; set; }
21     }
22 }

```

Figura 15. UserContext.

5. Modificarea funcției de logare din Business Logic/*UserAPI*, astfel să ne putem conecta la bază.

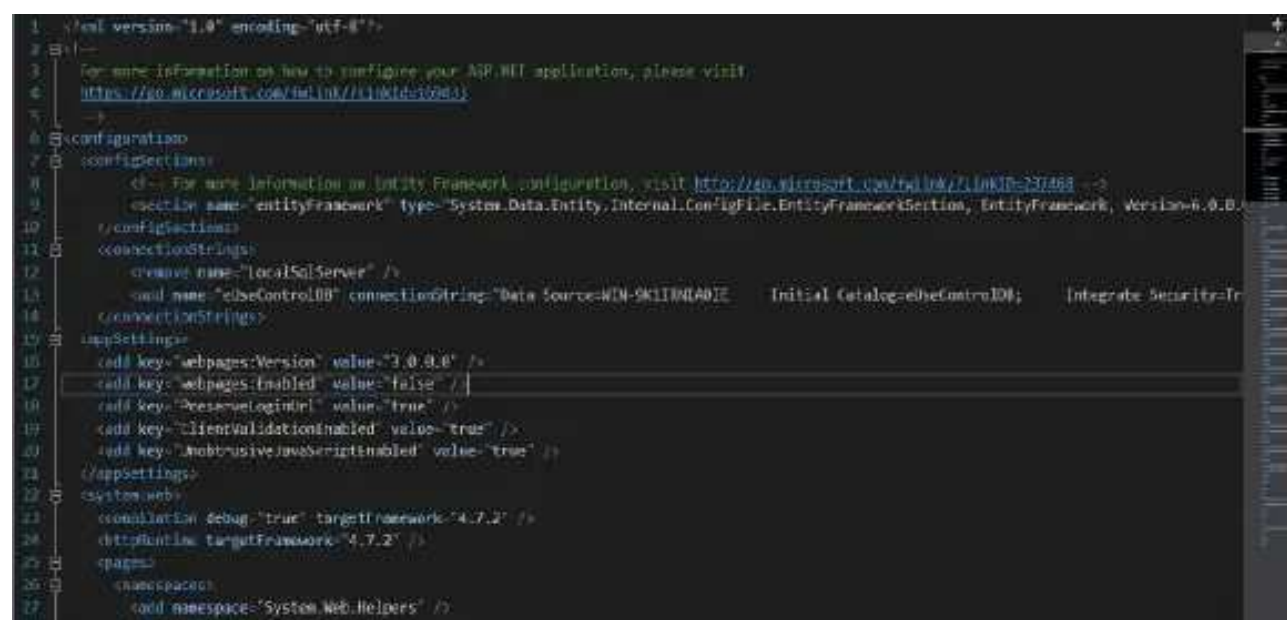


```
using eUseControl.Domain.Entities.User;

namespace eUseControl.BusinessLogic.Core
{
    [reference]
    public class UserApi
    {
        [references]
        internal ULoginResp UserLoginAction(ULoginData data)
        {
            UsersDbTable result;
            using (var db = new UserContext())
            {
                result = db.Users.FirstOrDefault(u => u.Username == data.Username && u.Password == data.Password);
            }
            if (result == null)
            {
                return new ULoginResp
                {
                    Status = false,
                    StatusMsg = "The username or password is incorrect"
                };
            }
            return new ULoginResp { Status = true };
        }
    }
}
```

Figura 16. UserApi.

6. Adăugarea conexiunii în *Web.config*



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <!--
3 For more information on how to configure your ASP.NET application, please visit
4 https://go.microsoft.com/fwlink/?linkid=301885
5 -->
6 <configuration>
7   <configSections>
8     <!-- For more information on Entity Framework configuration, visit https://go.microsoft.com/fwlink/?linkid=301885 -->
9     <section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
10   </configSections>
11   <connectionStrings>
12     <remove name="localSqlServer" />
13     <add name="eUseControlDB" connectionString="Data Source=WIN-9K11RM407E; Initial Catalog=eUseControlDB; Integrate Security=True" />
14   </connectionStrings>
15   <appSettings>
16     <add key="webpages:Version" value="3.0.0.0" />
17     <add key="webpages:Enabled" value="false" />
18     <add key="ResourceLocalization" value="true" />
19     <add key="ClientValidationEnabled" value="true" />
20     <add key="UnobtrusiveJavaScriptEnabled" value="true" />
21   </appSettings>
22   <system.web>
23     <compilation debug="true" targetFramework="4.7.2" />
24     <httpRuntime targetFramework="4.7.2" />
25     <pages>
26       <namespaces>
27         <add namespace="System.Web.Helpers" />
28       </namespaces>
29     </pages>
30   </system.web>
31 </configuration>
```

Figura 17. Web.config.

7. Configurarea a MSSQL



Figura 18. Conectarea la server.

8. În baza de date s-a creat o tabelă cu numele UsersDbTable și cu câmpurile indicate în fișierul UsersDbTable.cs

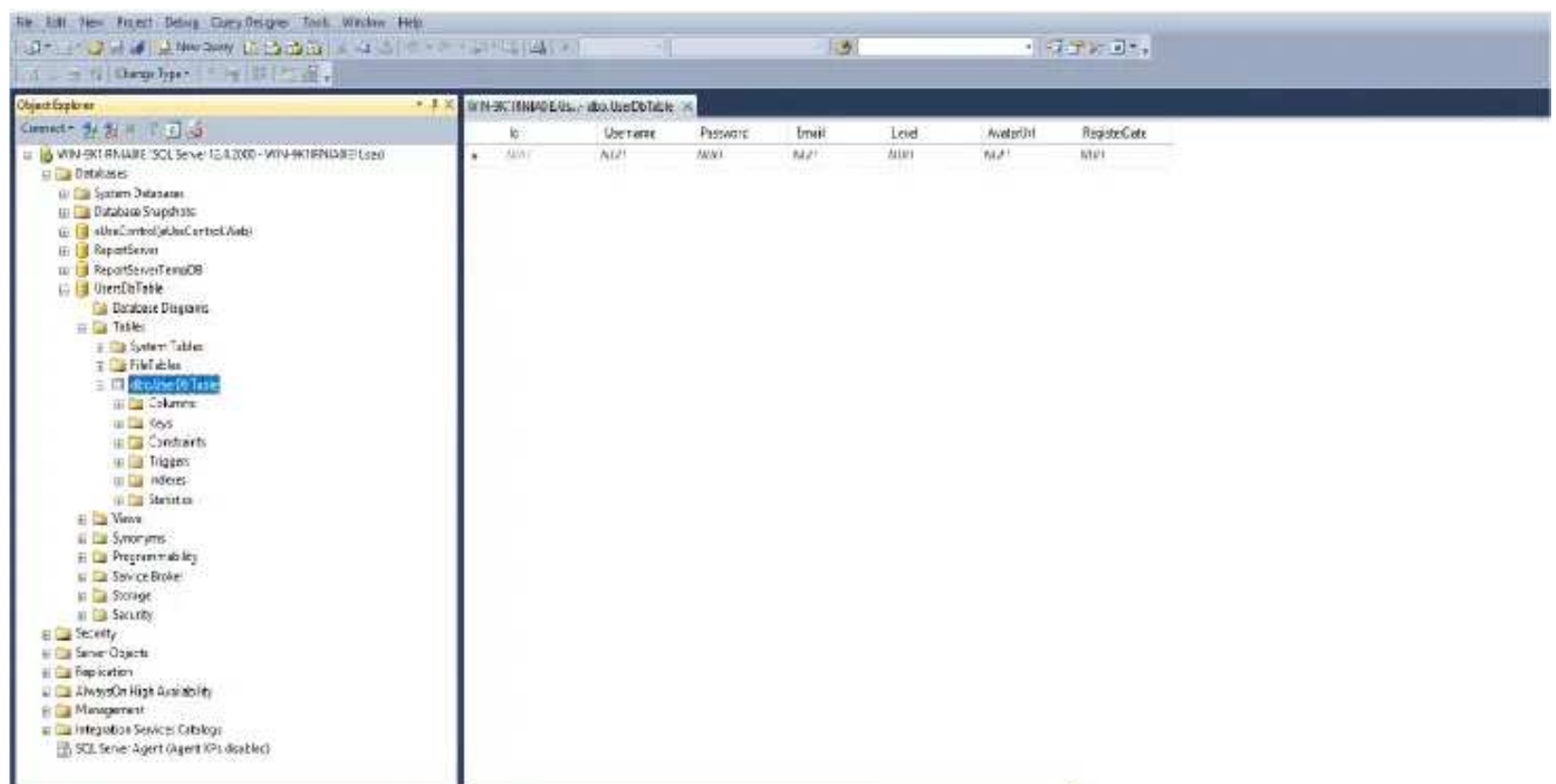


Figura 19. Tabela UsersDbTable.

2.3 Prezentarea magazinului apicol

În versiunea finală a site-ului, am realizat pagina de înregistrare/logare, checkout, blog. Pe viitor se poate de îmbunătățit situl prin adăugarea posibilității de a adăuga imagini în propria galerie, de a scrie bloguri și a le distribui altor utilizatori.

Previzualizarea rezultatului:



Figura 20. Pagina principală



Figura 21. Pagina de produse



Figura 22. Pagina despre produse recomandate



Figura 23. Pagina de produse la reducere

APIALBINA

DESPRE NOI PRODUSELE NOASTRE NOU TATI BLOG CONTACTE 🔍 LOGARE INREGISTRARE

Credential

Username or email address

Password

Remember me

Log In

Forgot your password?

Don't have an account? Sign Up

Go to Home

Figura 24. Pagina de logare

APIALBINA

DESPRE NOI PRODUSELE NOASTRE NOU TATI BLOG CONTACTE 🔍 LOGARE INREGISTRARE

Username

Enter your username

Email Address

Enter your email

Password

Enter your password

Repeat Password

Confirm your password

Sign Up

Already have account? Sign In

Go to Home

Figura 25. Pagina de înregistrare

Concluzie

Pe parcursul realizării practicii, s-a analizat realizarea unui magazin apicol utilizând tehnologia ASP.NET MVC. De aceea, a trebuit 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 serviciului 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: controler (Controller).

Realizarea unui serviciu web necesită cunoștințe, implicare cât și dorința de a realiza un produs cât mai calitativ. Astfel în cadrul practicii am avut posibilitatea de a studia mai profund acest domeniu.

În concluzie, toate aceste cunoștințe 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 produselor 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]. – Regim de acces: http://www.w3bai.com/ro/aspnet/aspnet_intro.html
2. Curs ASP.NET MVC. [Resursă electronică]. – Regim de acces: https://profs.info.uaic.ro/~iasimin/Special/Curs_ASPMVC.pdf
3. Scrierea cu diacritice: indicator al gradului de cultură și de profesionalism, <http://legestart.ro/scrierea-cu-diacritice-indicator-al-gradului-de-cultura-si-de-profesionalism/>, accesat la data: 01.06.2016 și Bootstrap. [Resursă electronică]. – Regim de acces: <https://getbootstrap.com/>
4. What is HTML?, <http://www.simplehtmlguide.com/whatishtml.php>, accesat la data de: 07.06.2016 INTRODUCERE IN SQL. [Resursă electronică]. – Regim de acces: <http://cursuri.cs.pub.ro/~radulescu/bd/sql7/oracle2.html>
5. HTML, <http://www.computerhope.com/jargon/h/html.htm>, accesat la data: 07.06.2016 [18] CSS-Introduction, http://www.w3schools.com/css/css_intro.asp, accesat la data: 07.06.2016