

Ministerul Educației, Culturii și Cercetării al Republicii Moldova  
Universitatea Tehnică a Moldovei  
Departamentul Ingineria Software și Automatică

# RAPORT

Lucrare de laborator Nr.3  
Disciplina: Tehnologii Web  
Tema: Modele de proiectare. BusinessLogic

A efectuat:

st. gr. TI-216 Arsene Ion  
Vlașițchi Ștefan  
Cozma Daniel  
Untilă Victor

A verificat:

asist. univ Gaidarji Alina

**Sarcina lucrării de laborator:** Familiarizarea cu structura șablonului de proiectare BusinessLogic și modelarea unui proiect ASP.NET, în baza lucrării de laborator Nr2, în conformitate cu modelul BusinessLogic.

### Considerații teoretice:

Proiectul MVC Asp.NET poate fi împărțit pe 3 nivele: nivelul prezentării, nivelul BusinessLogic și nivelul de acces la date. Această împărțire îmbunătățește procesul de dezvoltare și îmbunătățește performanța sistemului.

Nivelul *BusinessLogic* incapsulează toată logica de afaceri a proiectului, toate calculele necesare. Acest nivel primește obiecte din nivelul de acces la date și le transferă la nivelul de prezentare(Web) și invers. Obiectele Business stochează date și comportament, nu numai date.

### Implementarea practică a sarcinilor de laborator

Deoarece principalele niveluri ale aplicației sunt Domain, Model, Data, Web, prin urmare, este necesar de împărțit sistemul proiectat în niveluri corespunzătoare.

Pentru a face acest lucru, este necesar de adăugat încă 3 proiecte suplimentare la soluția MS Visual Studio. Arborele decizional rezultat este prezentat în figura 1.

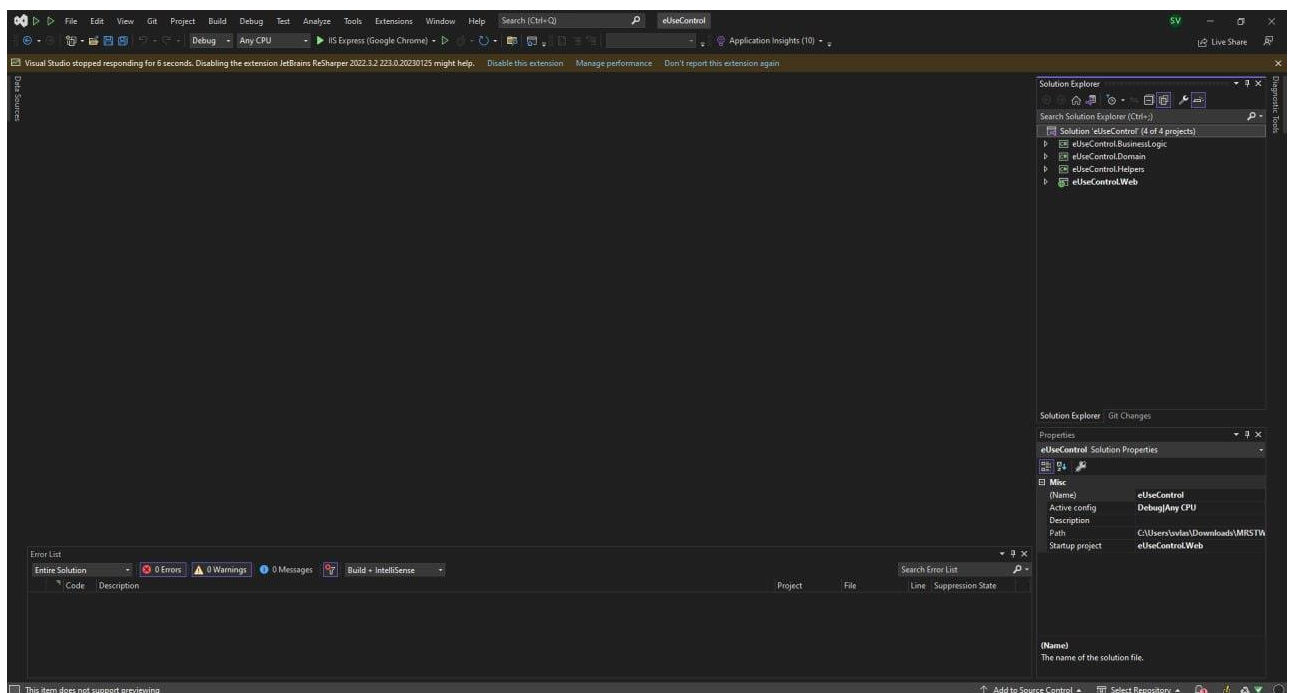


Figura 1 - Proiectele aplicației web

În figura de mai sus este reprezentat structura soluției care constă din 3 proiecte eUseControl, BusinessLogic, Domain și Helpers.

De asemenea este necesar să se stabilească legături (dependențe) între aceste proiecte. Dependențele pentru stratul Businesslogic sunt reprezentate în figura 2.

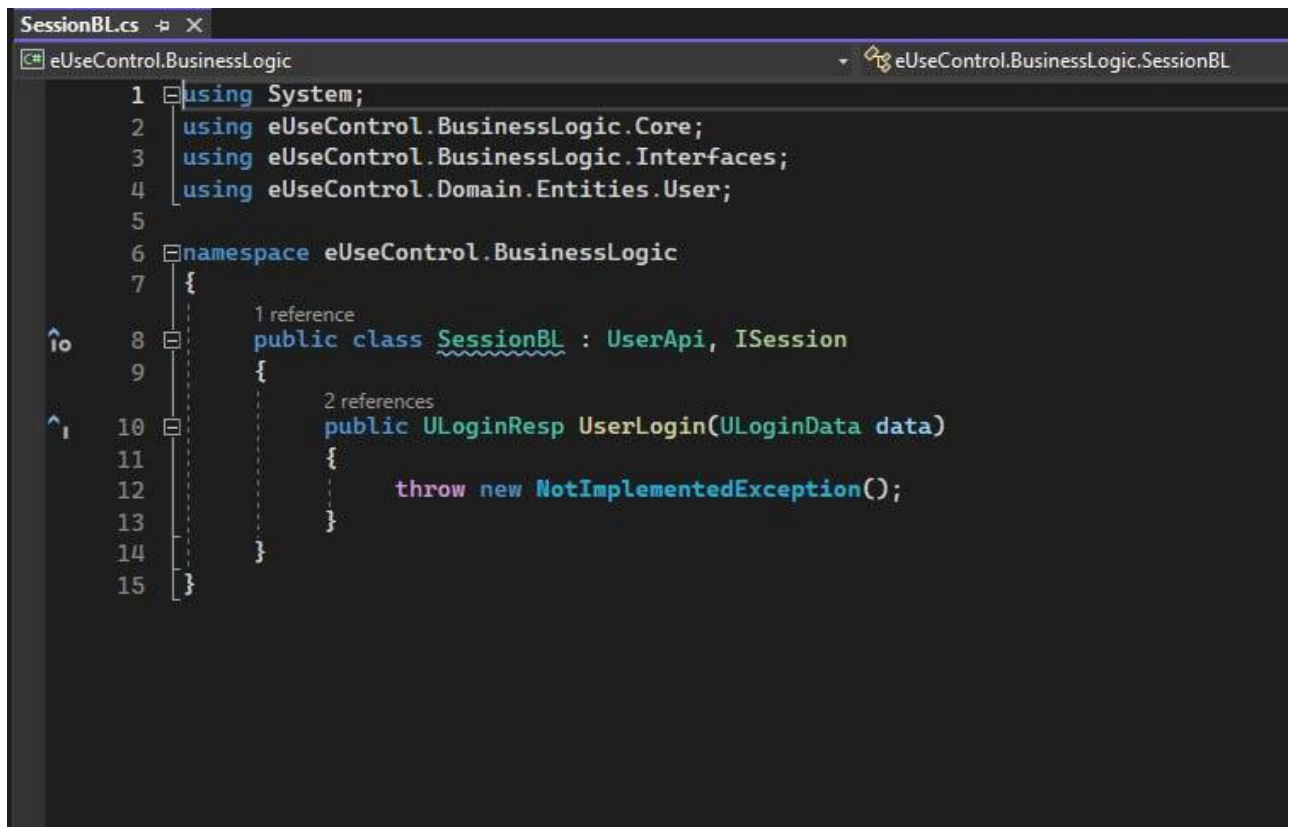


Figura 2 - Dependentele proiectului BusinessLogic

În același mod se stabilesc dependențele și pentru celelalte proiecte:

- Proiectul Domain are referință la proiectul Helpers;
- Proiectul de prezentare eUseControl are referință la proiectele BusinessLogic și Domain;
- Proiectul Helpers nu are nici o referință;

La formarea proiectului BusinessLogic este nevoie de a crea două mape în interiorul său: Core și Interfaces. În mapa Core, se creează 2 clase AdminApi și UserApi.

În mapa Interfaces este creată clasa ISession. Conform regulilor C#, toate denumirile interfețelor încep cu litera majuscula I.

Următorul element creat în cadrul proiectului BusinessLogic este clasa SessionBL, care se află în rădăcina a acestui proiect.

```

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

```

Figura 3 - Conținutul clasei *SessionBL*

Din fragmentul de cod prezentat se poate vedea că clasa *SessionBL* este moștenește clasa *UserApi* și implementează interfața *ISession* creată anterior.

S-a adăugat și clasa *MyBussinesLogic*. Această clasă conține o metodă ce returnează un obiect de tip *SessionBL*.

Structura finală a proiectului *BusinessLogic* este reprezentată în figura 4.

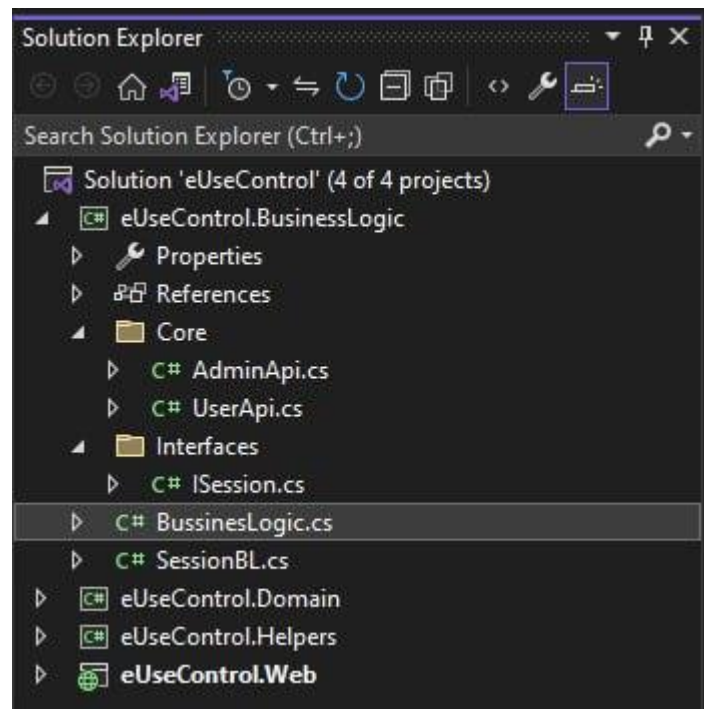
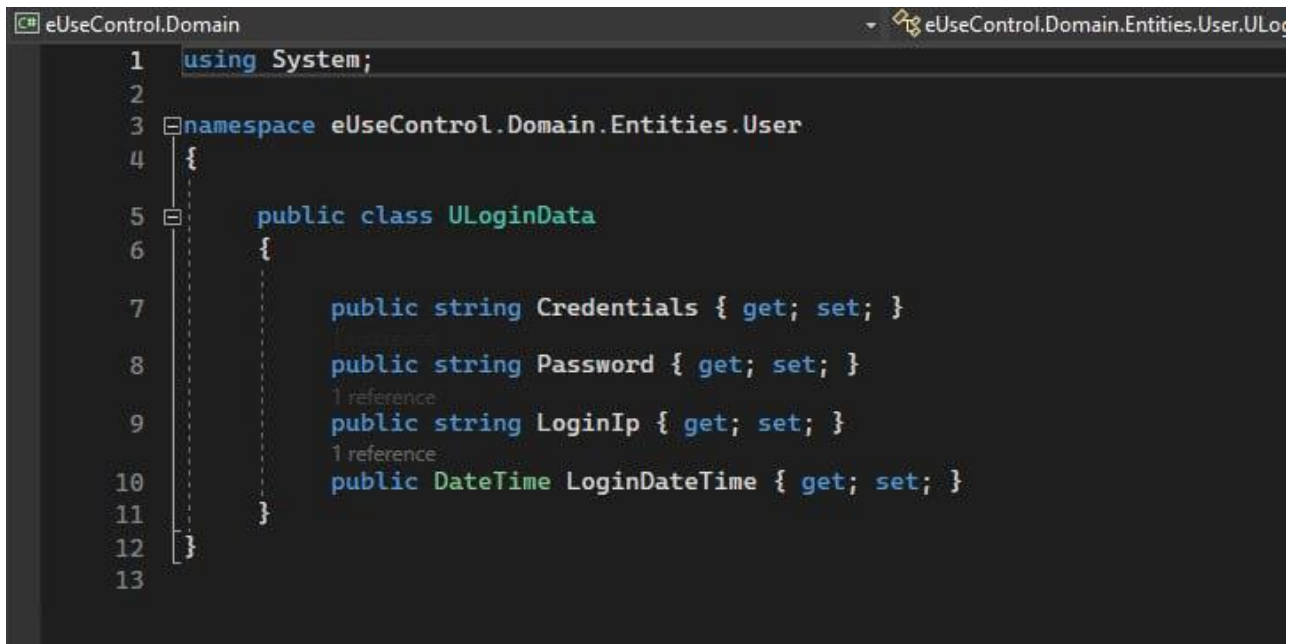


Figura 4 - Structura proiectului *BusinessLogic*

Următorul proiect la care trebuie de implementat funcționalitatea este proiectul *Domain*. Pentru aceasta se creează două mape: *Entities* și *Enums*. *Entities* conține clase care vor fi utilizate în viitoarele lucrări cu baza de date. La această etapă, în mapa *Entities* se creează o mapă *User* ce conține două clase în interiorul său: *UloginData* și *UloginResp*.

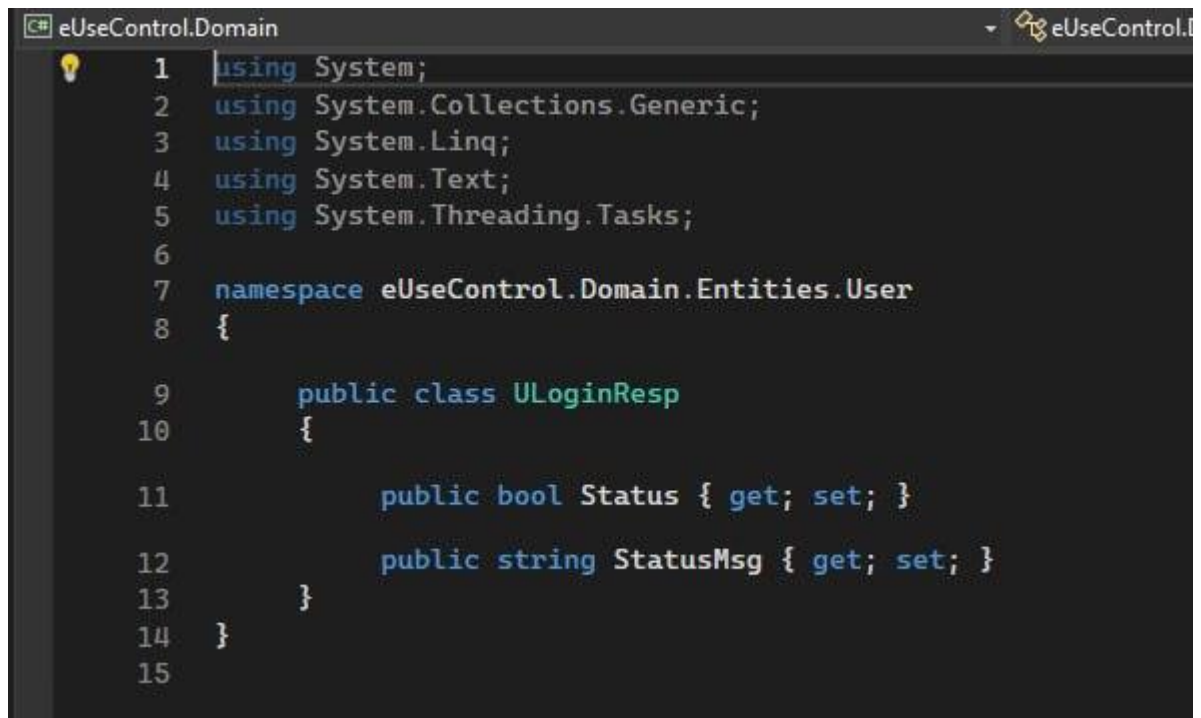
Clasa *ULoginData* conține câmpurile necesare pentru obținerea informațiilor de autentificare a utilizatorului. Pentru comoditate, am folosit proprietăți implementate automat, care sunt niște câmpuri de rezervă privat, anonim, care poate fi accesat numai prin accesorii get și set.



```
1 using System;
2
3 namespace eUseControl.Domain.Entities.User
4 {
5     public class ULoginData
6     {
7         public string Credentials { get; set; }
8         public string Password { get; set; }
9         public string LoginIp { get; set; }
10        public DateTime LoginDateTime { get; set; }
11    }
12 }
13
```

Figura 6 - Clasa *ULoginData*

Clasa *ULoginResp*, la rândul ei, conține câmpuri care descriu răspunsul primit după ce utilizatorul se conectează: starea și mesajul corespunzător.



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace eUseControl.Domain.Entities.User
8 {
9     public class ULoginResp
10    {
11        public bool Status { get; set; }
12        public string StatusMsg { get; set; }
13    }
14 }
15
```

Figura 7 - Clasa *ULoginResp*

Structura finală a proiectului *Domain* este reprezentată în figura 8.

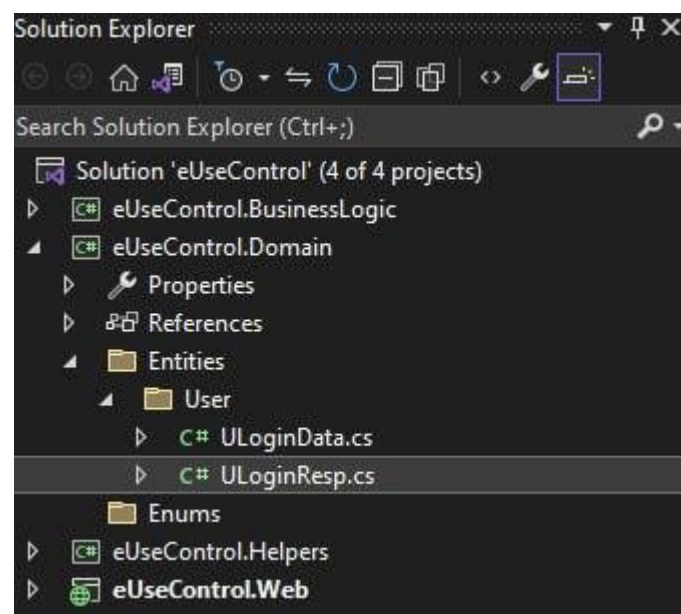
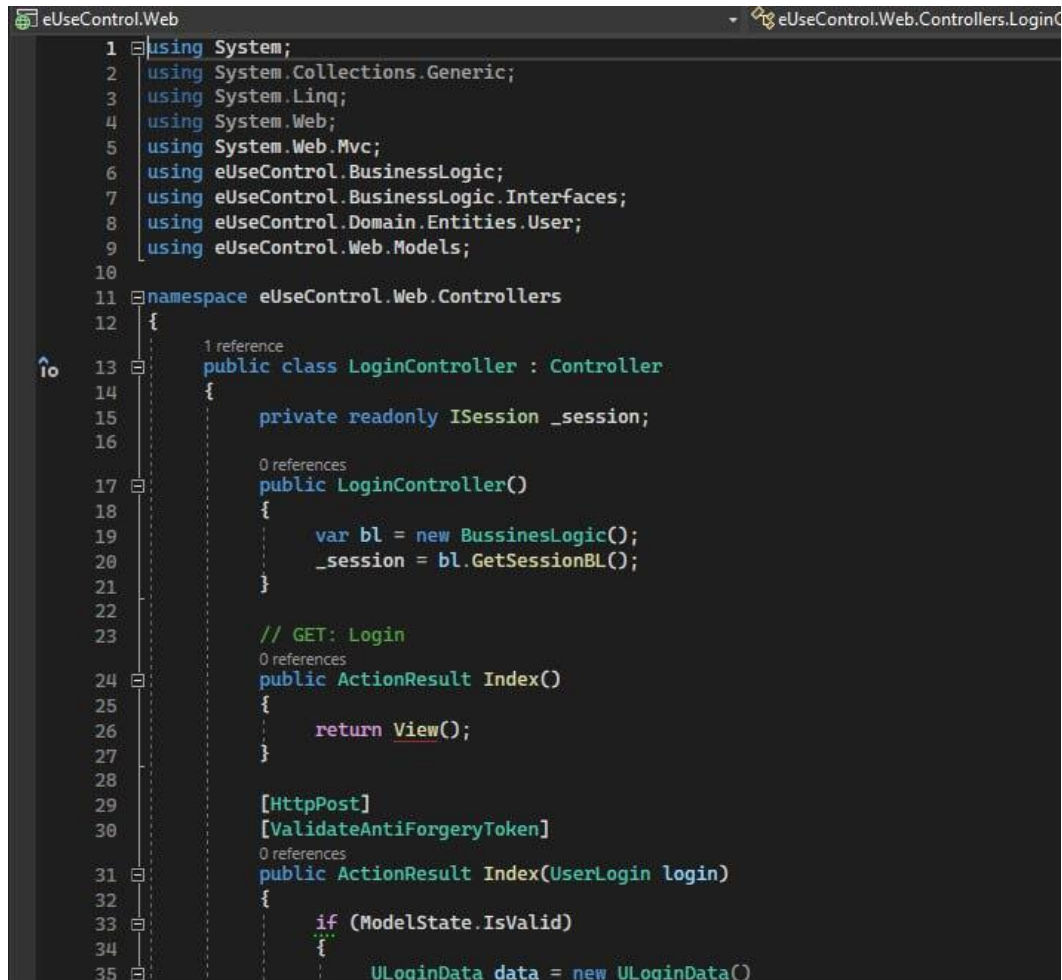


Figura 8 - Structura proiectului *Domain*

În nivelul de prezentare în mapa Controllers s-a creat un nou controller *LoginController*.

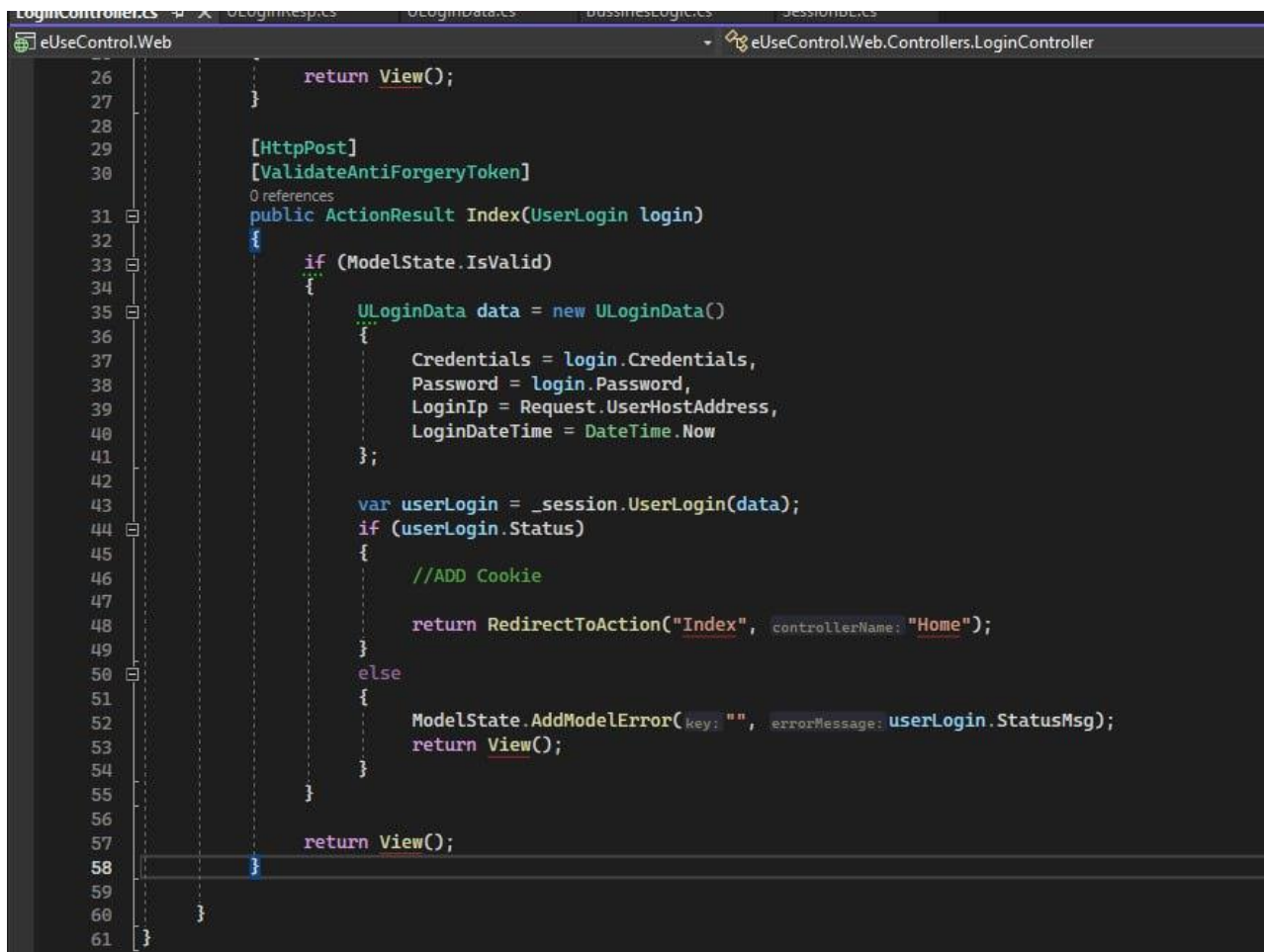


```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.Mvc;
6 using eUseControl.BusinessLogic;
7 using eUseControl.BusinessLogic.Interfaces;
8 using eUseControl.Domain.Entities.User;
9 using eUseControl.Web.Models;
10
11 namespace eUseControl.Web.Controllers
12 {
13     public class LoginController : Controller
14     {
15         private readonly ISession _session;
16
17         public LoginController()
18         {
19             var bl = new BussinesLogic();
20             _session = bl.GetSessionBL();
21         }
22
23         // GET: Login
24         public ActionResult Index()
25         {
26             return View();
27         }
28
29         [HttpPost]
30         [ValidateAntiForgeryToken]
31         public ActionResult Index(UserLogin login)
32         {
33             if (ModelState.IsValid)
34             {
35                 ULoginData data = new ULoginData()
```

Figura 9 – Inițializarea sesiunii



În figura de mai sus se reprezintă constructorul clasei *LoginController*, în interiorul căruia se inițializează sesiunea utilizatorului în aplicația noastră.

The image is a screenshot of a code editor, likely Visual Studio, showing the implementation of the `Index` action method in the `LoginController` class. The file name at the top is `eUseControl.Web.Controllers.LoginController`. The code is written in C# and includes the following elements:

- Line 26: `return View();`
- Line 27: Closing brace for the previous block.
- Line 28: Empty line.
- Line 29: Attribute `[HttpPost]`.
- Line 30: Attribute `[ValidateAntiForgeryToken]`.
- Line 31: Method signature `public ActionResult Index(UserLogin login)`.
- Line 32: Opening brace for the method body.
- Line 33: `if (ModelState.IsValid)` condition.
- Line 34: Opening brace for the `if` block.
- Line 35: `ULoginData data = new ULoginData()`.
- Line 36: Opening brace for the `ULoginData` constructor.
- Line 37: `Credentials = login.Credentials,`
- Line 38: `Password = login.Password,`
- Line 39: `LoginIp = Request.UserHostAddress,`
- Line 40: `LoginDateTime = DateTime.Now`
- Line 41: Closing brace for the `ULoginData` constructor.
- Line 42: `};`
- Line 43: `var userLogin = _session.UserLogin(data);`
- Line 44: `if (userLogin.Status)` condition.
- Line 45: Opening brace for the `if` block.
- Line 46: `//ADD Cookie` comment.
- Line 47: `return RedirectToAction("Index", controllerName: "Home");`
- Line 48: Closing brace for the `if` block.
- Line 49: `else` condition.
- Line 50: Opening brace for the `else` block.
- Line 51: `ModelState.AddModelError(key: "", errorMessage: userLogin.StatusMsg);`
- Line 52: `return View();`
- Line 53: Closing brace for the `else` block.
- Line 54: Closing brace for the `if` block.
- Line 55: Empty line.
- Line 56: `return View();`
- Line 57: Closing brace for the method body.
- Line 58: Closing brace for the class.
- Line 59: Empty line.
- Line 60: Empty line.
- Line 61: Empty line.

Figura 10 – Metoda de acțiune *Index*

În figura de mai sus observăm metoda de acțiune *Index* care răspunde la solicitarea POST a motorului de rutare atunci când este trimis formularul de autentificare. Atributul `[HttpPost]` este un tip de supraîncărcare a metodei.

Modelul `UserLogin` care se folosește în cadrul autentificării, conține câmpurile necesare pentru autentificarea utilizatorului. Această clasă este creată în mapa `Models`. În clasa `UserLogin`, există două câmpuri care stochează adresa de email și parola prin care s-a conectat la sistem.

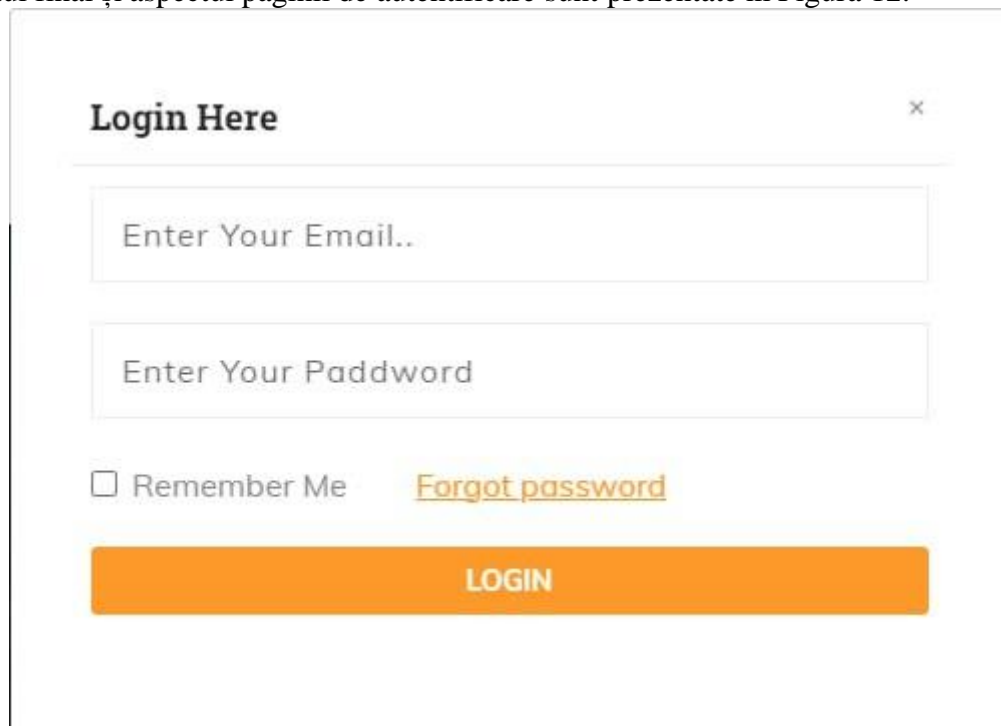


```
eUseControl.Web
1 namespace eUseControl.Web.Models
2 {
3     1 reference
4     public class UserLogin
5     {
6         1 reference
7         public string Credentials { get; set; }
8         1 reference
9         public string Password { get; set; }
10    }
11 }
```

Figura 11 - Clasa *UserLogin* (mapa *Models*)

Pentru a afișa această implementare pe ecranul utilizatorului, trebuie de creat o nouă vizualizare în mapa *Views* → *Login*, cu denumirea *Login.cshtml*. Această vizualizare conține codul de marcare pentru pagina de autentificare.

Rezultatul final și aspectul paginii de autentificare sunt prezentate în Figura 12.



The screenshot displays a web browser window with a login form. The form has a title 'Login Here' and a close button (X). It contains two text input fields: the first is labeled 'Enter Your Email..' and the second is labeled 'Enter Your Paddword'. Below these fields, there is a checkbox labeled 'Remember Me' and a link labeled 'Forgot password'. At the bottom of the form is a large orange button with the text 'LOGIN'.

Figura 12 - Formularul de autentificare

**Concluzii:**

Pe parcursul elaborării acestei lucrări de laborator, a fost analizată structura proiectului ASP.NET MVC conform modelului *Business Logic*. În conformitate cu cunoștințele acumulate șablonul de modelare *BusinessLogic*, proiectul elaborat anterior a fost actualizat și completat cu modificările necesare. În procesul dat, de asemenea, a fost modelată și adăugată la sistem o pagină web pentru autentificarea utilizatorului.

Link-ul proiectului plasat pe GitHub: <https://github.com/IonArsene/MRSTW>