



*Practică De Domeniu*  
*Proiect*

---

*Management Bibliotecă*

---

Prof. îndrumător: As.Univ.Dr.Ing. Tcaciuc Simona-Anda

Student: Beilic Maria

Specializarea: Calculatoare

Grupa: 3122B

An: II

2019/2020

## Cuprins

I. Tema proiectului.....	3
II. Noțiuni teoretice .....	3
III. Descrierea proiectului .....	4
1. Bibliotecar .....	4
-Adăugare Cărți.....	5
-Afișare Cărți.....	7
-Afișare Studenți.....	9
-Împrumut Cărți.....	9
-Adăugare Sancțiuni.....	12
-Returnare Cărți.....	12
-Contact/Trimite Mesaj.....	14
2. Student.....	18
-Înregistrare Cont Student.....	18
-Împrumuturile mele.....	20
-Afișare Cărți.....	21
-Contact/Trimite Mesaj.....	21
IV. Concluzii.....	22
V. Bibliografie .....	22

## I. Tema proiectului

Tema proiectului o reprezintă un sistem de management al unei biblioteci. Proiectul se numește „Management Bibliotecă” și este realizat în Visual Studio 2019 utilizând ASP.NET cu Bootstrap, C# și SQL Server. Acesta suprinde două perspective, student și administrator. Scopul proiectului este de a realiza o aplicație web interactivă ce ușurează munca bibliotecarilor și facilitează interacțiunea dintre mediul academic și student.

## II. Noțiuni teoretice

O aplicație web (sau web app) este un software care îndeplinește o anumită funcție folosind un web browser pe post de „client”. Aplicația poate fi ceva simplu, precum un formular de contact de pe un website, sau ceva mai complex, precum un word processor sau un joc cu mai mulți jucători pe care îl poți downloada pe telefon.

ASP.NET este un set de tehnologii care ne permit crearea de aplicații web. Este evoluția de la Microsoft Active Server Pages (ASP), dar beneficiază de suportul platformei de dezvoltare Microsoft .NET . ASP.NET reprezintă un framework web gratuit utilizat pentru crearea site-urilor, aplicațiilor și serviciilor cu ajutorul HTML, CSS și JavaScript. ASP.NET a fost construit pe Common Language Runtime (CLR), ceea ce permite programatorilor să scrie cod folosind orice limbaj suportat de .NET.

Bootstrap este un cadru front-end inclus în ASP.NET ce ajută în construirea interfeței utilizator cu HTML, CSS și JavaScript. Puteți utiliza Bootstrap pentru a crea o interfață care să arate bine pe orice, de la afișaje mari pentru desktop până la ecrane mici pentru mobil.

C# este un limbaj simplu, cu circa 80 de cuvinte cheie și 12 tipuri de date predefinite. El permite programarea structurată, modulară și orientată obiectual, conform percepțelor moderne ale programării profesionale.

SQL (Structured Query Language) este folosit pentru a comunica cu o bază de date. SQL Server este un sistem de gestionare a bazelor de date relaționale (RDBMS) dezvoltat și comercializat de Microsoft. Ca server de baze de date, funcția principală a SQL Server este stocarea și preluarea datelor utilizate de alte aplicații. Componenta de bază a SQL Server este Database Engine. Acesta constă dintr-un engine relațional care procesează interogări și un engine de stocare care gestionează fișiere de baze de date, pagini, index etc.

### III. Descrierea proiectului

La execuția programului utilizator este îndrumat pe o pagină intermediară de conectare pentru alegerea tipului de utilizator (student sau bibliotecar).

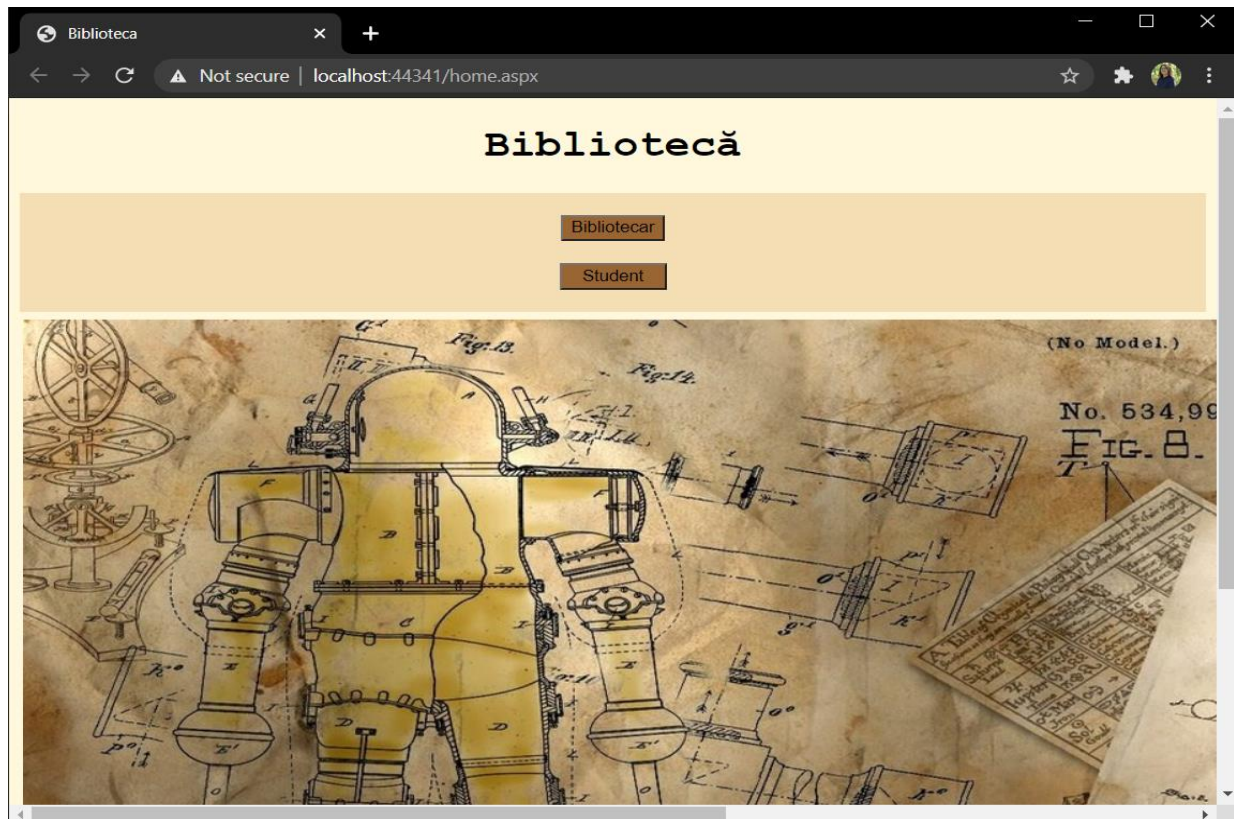


Figura 1. Pagina de start

#### 1. Bibliotecar

Pentru a accesa aplicația din punctul de vedere al administratorului avem nevoie de un utilizator „admin” și de o parolă „3122B”. Dacă utilizatorul introduce alte date, inexistente în baza de date, va primi un mesaj de avertizare.

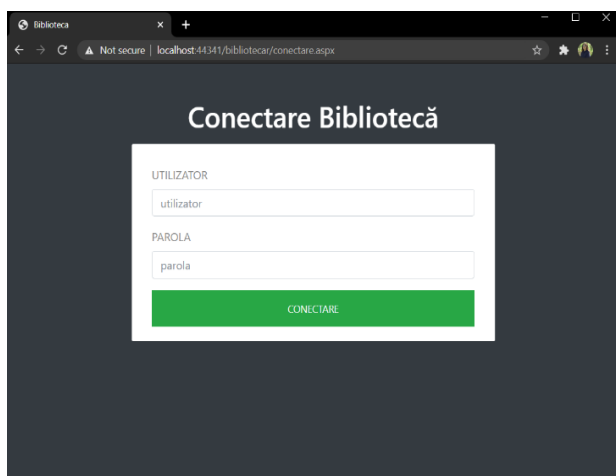


Figura 2. Formă de autentificare

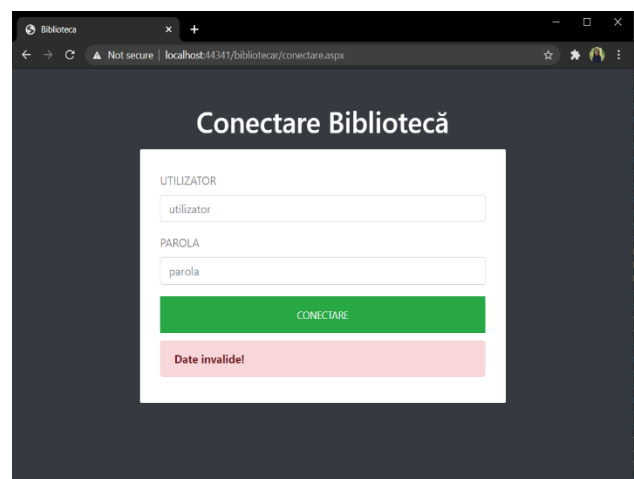
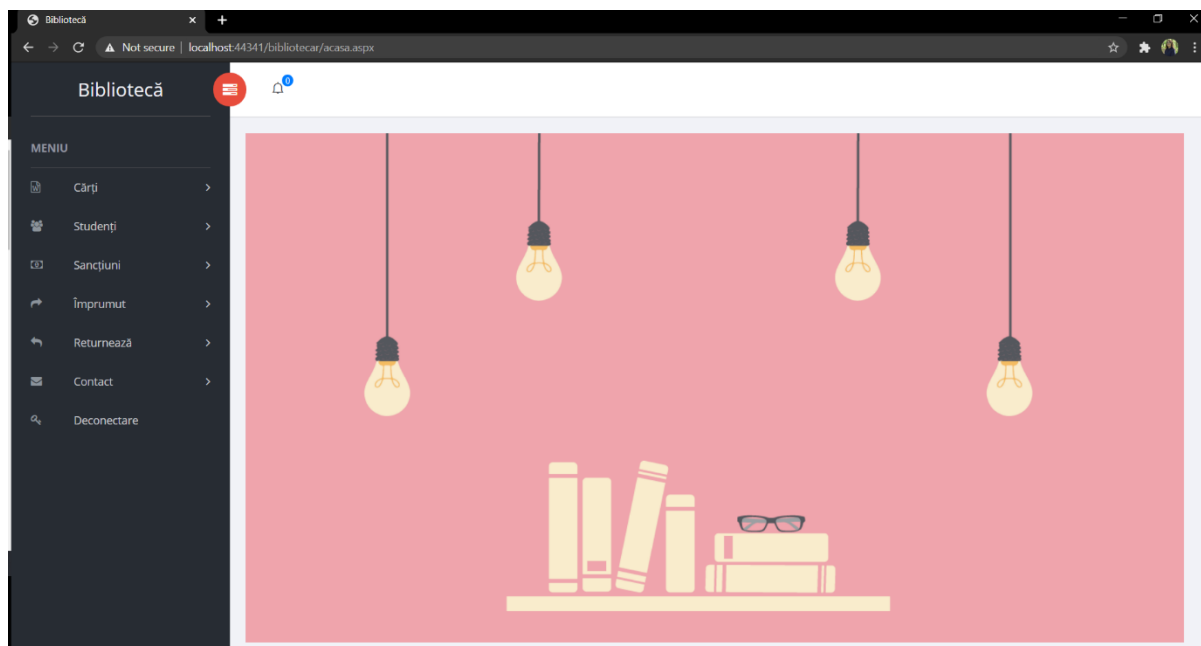


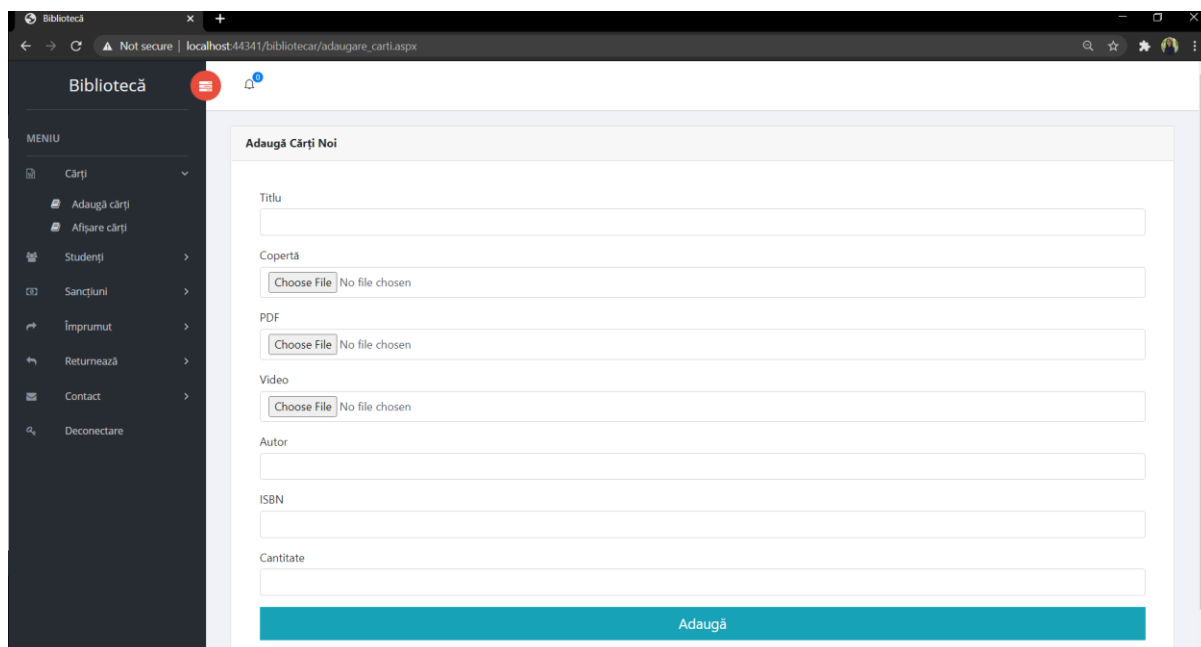
Figura 3. Date invalide

După conectare, administratorul este îndrumat pe pagina principală unde se află meniul cu opțiunile dedicate acestuia: adăgarea/afișarea cărților, afișarea studenților, adăugarea sancțiunilor, împrumut cărți, returnare cărți, trimiterea de mesaje scrise, deconectarea.



*Figura 4. Meniul principal*

Pentru adăugarea unei cărți s-au solicitat următoarele date: titlu, copertă, pdf, video, autor, isbn și cantitatea.



*Figura 5. Adăugare Carte*

În primul rând, am început procesul de adăugarea prin crearea interfeței. Am împărțit proiectul în două fișiere. Am adăugat un Master page în fișierul dedicat administratorului în care am creat un șablon utilizând Bootstrap ce-mi permite afișarea meniului din partea stângă. Pentru pagina de adăugare am introdus în cod etichetele ce specifică tipul datelor introduse și TextBox-uri sau căsuțe de tip file upload fiecare având un id, pentru introducerea datelor.

```

adaugare_carti.aspx*
1  <Page Title="" Language="C#" MasterPageFile="~/bibliotecar/bibliotecar.Master" AutoEventWireup="true" CodeBehind="adaugare_carti.aspx.cs" Inherits="B
2  <asp:Content ID="Content1" ContentPlaceHolderID="c1" runat="server">
3
4  <div class="col-lg-12">
5      <div class="card">
6          <div class="card-header">
7              <strong class="card-title">Adaugă Cărți Noi</strong>
8          </div>
9          <div class="card-body">
10             <div id="pi">
11                 <div class="card-body">
12                     <form action="" id="fo1" runat="server" method="post" novalidate="novalidate">
13                         <div class="form-group">
14                             <label for="" class="control-label mb-1">Titlu</label>
15                             <asp:TextBox ID="titlu" runat="server" class="form-control"></asp:TextBox>
16                         </div>
17
18                         <div class="form-group">
19                             <label for="" class="control-label mb-1">Copertă</label>
20                             <asp:FileUpload ID="f1" runat="server" class="form-control" />
21                         </div>
22                         <div class="form-group">
23                             <label for="" class="control-label mb-1">PDF</label>
24                             <asp:FileUpload ID="f2" runat="server" class="form-control" />
25                         </div>
26                         <div class="form-group">
27                             <label for="" class="control-label mb-1">Video</label>
28                             <asp:FileUpload ID="f3" runat="server" class="form-control" />
29                         </div>
30                         <div class="form-group">
31                             <label for="" class="control-label mb-1">Autor</label>
32                             <asp:TextBox ID="autor" runat="server" class="form-control"></asp:TextBox>
33                         </div>
34                         <div class="form-group">
35                             <label for="" class="control-label mb-1">ISBN</label>
36                             <asp:TextBox ID="isbn" runat="server" class="form-control"></asp:TextBox>
37                         </div>
38                         <div class="form-group">
39                             <label for="" class="control-label mb-1">Cantitate</label>
40                             <asp:TextBox ID="cantitate" runat="server" class="form-control"></asp:TextBox>
41                         </div>
42                         <div>
43                             <asp:Button ID="b1" runat="server" class="btn btn-lg btn-info btn-block" Text="Adaugă" OnClick="b1_Click"/>
44                         </div>
45                         <div class="alert alert-success" id="msg" runat="server" style="margin-top:10px; display:none">
46                             <strong>Cartea a fost adăugată cu succes!</strong>
47                         </div>
48                     </form>
49                 </div>
50             </div>
51         </div>
52     </div>
53 </asp:Content>
54
55

```

Figura 6. Cod interfață-adăugare carte

Pentru salvarea datelor am adăugat în proiect un SQL Server DataBase în care am creat un tabel ce va conține informațiile despre cărți.

Tables	
▶	bibliotecar_conectare
▶	carti
	id
	titlu_carte
	coperta_carte
	pdf_carte
	video_carte
	autor_carte
	isbn_carte
	cantitate_carte

Figura 7. Tabel cu date despre cărți

După realizarea interfeței, am introdus prin cod adăugarea elementelor preluate de la utilizator în baza de date. Pentru a evita suprapunerea datelor salvate în fișier (copertă, pdf, video) am adăugat o funcție ce va salva elementele sub o denumire întâmplătoare. Pentru introducerea informației în baza de date am realizat o comandă ce folosește id-urile corespunzătoare datelor din partea de interfață. Codul de mai jos surprinde acțiunea la apăsarea butonului „Adaugă” (Figura 5).

```
protected void b1_Click(object sender, EventArgs e)
{
    string nume_coperta=Class1.GetRandomPassword(10)+".jpg";
    string pdf_carte = "";
    string video_carte = "";

    string path = "";
    string path2 = "";
    string path3 = "";
    f1.SaveAs(Request.PhysicalApplicationPath + "/bibliotecar/coperta/" + nume_coperta.ToString());
    path = "coperta/" + nume_coperta.ToString();
    if (f2.FileName.ToString() != "")
    {
        pdf_carte = Class1.GetRandomPassword(10)+".pdf";
        f2.SaveAs(Request.PhysicalApplicationPath + "/bibliotecar/pdf_carte/" + pdf_carte.ToString());
        path2 = "pdf_carte/" + pdf_carte.ToString();
    }
    if (f3.FileName.ToString() != "")
    {
        video_carte = Class1.GetRandomPassword(10) + ".mp4";
        f3.SaveAs(Request.PhysicalApplicationPath + "/bibliotecar/video_carte/" + video_carte.ToString());
        path3 = "video_carte/" + video_carte.ToString();
    }
    SqlCommand cmd = con.CreateCommand();
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "insert into carti values('"+ titlu.Text + "','"+ path.ToString() + "','"+ path2.ToString() + "','"+ path3.ToString() + "','"+ autor.Text + "','"+ isbn.Text + "','"+ cantitate.Text + "')";
    cmd.ExecuteNonQuery();
    msg.Style.Add("display", "block");
}
```

Figura 8. Adăugare Carte-cod C#

La acționarea butonului „Afișare cărți” (Figura 5), utilizatorului îi este prezentat un tabel (Figura 9) ce surprinde informațiile din baza de date. Funcționalitatea de căutare, respectiv cea de sortare a fost realizată în CSS și JavaScript (Figura 10).

Copertă	Titlu	PDF	Video	Autor	ISBN	Cantitate	Editare	Ștergere
	Harry Potter and the Philosopher's Stone			J. K. Rowling	0-7475-3269-9	20	Modifică	Șterge
	Harry Potter and the Chamber of Secrets			J. K. Rowling	0-7475-3849-2	20	Modifică	Șterge
	Harry Potter and the Prisoner of Azkaban	<a href="#">pdf_carte/O65bZ4rBzL.pdf</a> <a href="#">șterge pdf</a>	<a href="#">video_carte/Nnb5CAxvW2.mp4</a> <a href="#">șterge video</a>	J. K. Rowling	0-7475-4215-5	12	Modifică	Șterge
	Harry Potter and the Goblet of Fire			J. K. Rowling	0-7475-4624-X	20	Modifică	Șterge
	Harry Potter and the Order of the Phoenix			J. K. Rowling	0-7475-5100-6	20	Modifică	Șterge

Figura 9. Afișare Cărți

Informația a fost preluată din baza de date cu ajutorul denumirilor din Figura. 7. Utilizatorul are opțiunea de a șterge documentul pdf sau video corespunzător unei cărți fără să afecteze celelalte date, această funcționalitate a fost posibilă cu ajutorul unor comenzi ce actualizează baza de date cu informația deja existentă , iar în locul documentului pdf/video lasă loc liber. În Figura 11 este prezentat codul C# pentru afișarea cărților.

```

1  Page Title="" Language="C#" MasterPageFile="~/Biblioteca/Biblioteca.Master" AutoEventWireup="true" CodeBehind="afisare_carti.aspx.cs" Inherits="Biblioteca.Biblioteca.afisare_carti"
2  <asp:Content ID="Content1" ContentPlaceHolderID="c1" runat="server">
3  <link href="https://cdn.datatables.net/1.10.19/css/jquery.dataTables.min.css" type="text/css" rel="stylesheet" />
4  <script src="https://code.jquery.com/jquery-3.3.1.js"></script>
5  <script src="https://cdn.datatables.net/1.10.19/js/jquery.dataTables.min.js"></script>
6  <div class="col-lg-12">
7  <div class="card">
8  <div class="card-header">
9  <strong class="card-title">Cărți Disponibile</strong>
10 </div>
11 <div class="card-body">
12
13 <asp:Repeater ID="r1" runat="server">
14 <headerTemplate>
15 <table class="table" id="example">
16 <thead class="thead-dark">
17 <tr>
18 <th scope="col">Coperta</th>
19 <th scope="col">Titlu</th>
20 <th scope="col">PDF</th>
21 <th scope="col">Video</th>
22 <th scope="col">Autor</th>
23 <th scope="col">ISBN</th>
24 <th scope="col">Cantitate</th>
25 <th scope="col">Editare</th>
26 <th scope="col">Ștergere</th>
27 </tr>
28 </thead>
29 <tbody>
30 <HeaderTemplate>
31 <ItemTemplate>
32 <tr>
33 <td></td>
34 <td>Eval("titlu_carte")</td>
35 <td>Eval("pdf_carte")</td>
36 <td>Eval("video_carte")</td>
37 <td>Eval("autor_carte")</td>
38 <td>Eval("isbn_carte")</td>
39 <td>Eval("cantitate_carte")</td>
40 <td><a href="modifica_carte.aspx?id=Eval("id")">Modifică</a></td>
41 <td><a href="sterge.aspx?id2=Eval("id")">Șterge</a></td>
42 </tr>
43 </ItemTemplate>
44 <FooterTemplate>
45 </tbody>
46 </table>
47 </HeaderTemplate>
48 </FooterTemplate>
49 </asp:Repeater>
50
51 <tbody>
52 </div>
53 </div>
54 </div>
55
56 <script type="text/javascript">
57 $(document).ready(function () {
58     $('#example').DataTable({
59         "pagingType": "full_numbers"
60     });
61 });
62 </script>
63
64 </asp:Content>
65

```

Figura 10. Afișare - cod interfață

```

1  Biblioteca
2  namespace Biblioteca.Biblioteca
3  {
4  public partial class afisare_carti : System.Web.UI.Page
5  {
6  //Initalizeaza obiect nou de tip sqlconnection ce permite accesul la informatia din baza de date si manipulara informatiei
7  SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\hell\OneDrive\Desktop\Practica\BibliotecaApp_Data\lms.mdf;Integrated Security=True");
8
9  //Verifica daca conexiunea este deschisa, daca aceasta nu este o deschide
10 if (con.State == ConnectionState.Open)
11 {
12     con.Close();
13 }
14 con.Open();
15 //Dupa o perioada de inactivitate sau daca incercam sa ne conectam direct la pagina de adaugare suntam trimisi in pagina de conectare
16 if (Session["biblioteca"] == null)
17 {
18     Response.Redirect("conectare.aspx");
19 }
20
21 SqlCommand cmd = con.CreateCommand();
22 cmd.CommandType = CommandType.Text;
23 cmd.CommandText = "select * from carti";
24 cmd.ExecuteNonQuery();
25 DataTable dt = new DataTable(); //tabel in care se afla datele din baza de date
26 SqlDataAdapter da = new SqlDataAdapter(cmd); //pod intre tabelul din C# si SQL Server pentru recuperarea si salvarea datelor
27 da.Fill(dt);
28 r1.DataSource = dt; //repete afisarea in tabel completand cu datele din baza de date
29 r1.DataBind();
30
31 //Informatii
32 public string verificavideo(object myvalue, object id)
33 {
34     if (myvalue.ToString() == "")
35     {
36         return myvalue.ToString();
37     }
38     else
39     {
40         return "<a href='sterge.aspx?id2=" + id + "' style='color:red'>sterge video</a>";
41     }
42 }
43
44 //Informatii
45 public string verificapdf(object myvalue1, object id1)
46 {
47     if (myvalue1.ToString() == "")
48     {
49         return myvalue1.ToString();
50     }
51     else
52     {
53         return "<a href='sterge.aspx?id1=" + id1 + "' style='color:red'>sterge pdf</a>";
54     }
55 }
56 }
57 }
58

```

Figura 11. Afișare cărți – cod C#

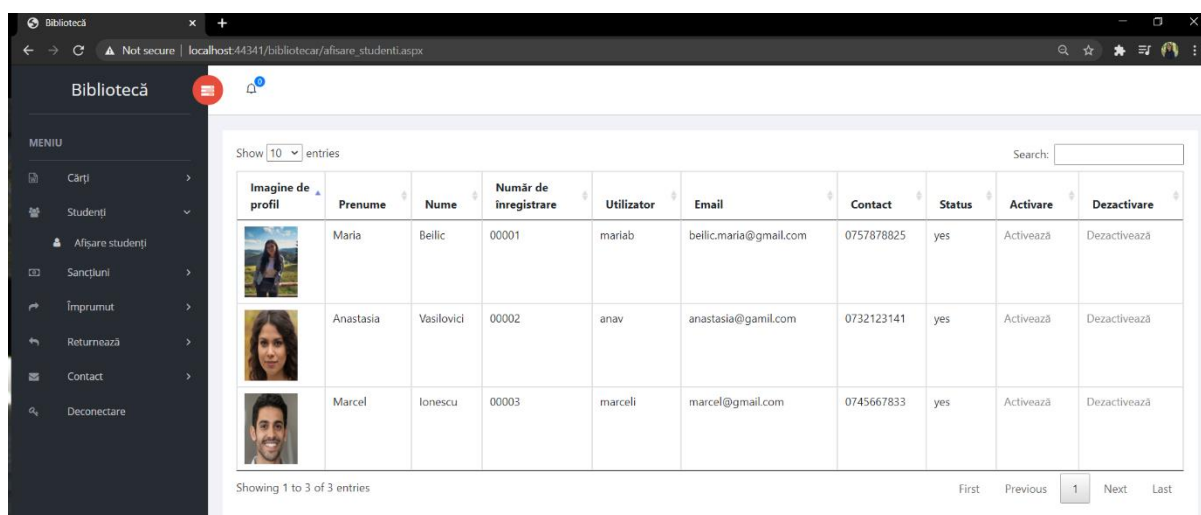


La apăsarea butonului „Modifică”, utilizatorul va fi trimis pe o pagina identică cu cea de adăugare, dar care are toate câmpurile completate cu datele cărții selectate. Modificarea se realizează printr-o comandă ce actualizează datele din baza de date cu informația introdusă ultima. Ștergerea unei cărți se obține prin comanda prezentată în Figura 12.

```
SqlCommand cmd = con.CreateCommand();  
cmd.CommandType = CommandType.Text;  
cmd.CommandText = "delete carti where id=" + Request.QueryString["id2"].ToString() + " ";  
cmd.ExecuteNonQuery();
```

Figura 12. Șterge carte

Afișarea studenților se face cu un principiu asemănător afișării de cărți, informația este preluată din baza de date. Funcționalitatea de căutare, respectiv cea de sortare a fost realizată în CSS și JavaScript, identic cu procedeul aplicat în cazul cărților. Administratorul are opțiunea de a activa sau dezactiva contul unui student. Asta înseamnă ca el poate schimba statusul studentului din baza de date prin actualizarea acestuia (Figurile 14/15). Studentul poate avea acces la bibliotecă doar dacă contul acestuia a fost aprobat de administrator.






Imagine de profil	Prenume	Nume	Număr de înregistrare	Utilizator	Email	Contact	Status	Activează	Dezactivează
	Maria	Belic	00001	mariab	belic.maria@gmail.com	0757878825	yes	Activează	Dezactivează
	Anastasia	Vasilovici	00002	anav	anastasia@gamil.com	0732123141	yes	Activează	Dezactivează
	Marcel	Ionescu	00003	marceli	marcel@gmail.com	0745667833	yes	Activează	Dezactivează

Figura 13. Afișare studenți

```
id = Convert.ToInt32(Request.QueryString["id"].ToString());  
  
SqlCommand cmd = con.CreateCommand();  
cmd.CommandType = CommandType.Text;  
cmd.CommandText = "update student_conectare set aprobat='no' where id=" + id + " ";  
cmd.ExecuteNonQuery();  
Response.Redirect("afisare_studenti.aspx");
```

Figura 14. Dezactivare cont student

```
id = Convert.ToInt32(Request.QueryString["id"].ToString());  
  
SqlCommand cmd = con.CreateCommand();  
cmd.CommandType = CommandType.Text;  
cmd.CommandText = "update student_conectare set aprobat='yes' where id=" + id + " ";  
cmd.ExecuteNonQuery();  
Response.Redirect("afisare_studenti.aspx");
```

Figura 15. Activare cont student

Utilizatorul are opțiunea de a împrumuta cărți studentului, folosind numărul de înregistrare al acestuia și ISBN-ul cărții. Pagina dedicată împrumutului este prezentată în Figura 16.

Figura 16. Pagina de împrumut.

Interfața a fost realizată cu ajutorul a doua controale de tip DropDownList, o imagine și două etichete. Datele necesare fiecărui împrumut au fost puse într-un tabel nou în SQL Server (Figura 18).

```

1 Page Title="" Language="C#" MasterPageFile="~/bibliotecar/biblioteca.Master" AutoEventWireup="true" CodeBehind="imprumut_carti.aspx.cs" Inherits="Biblioteca.biblioteca.imprumut_carti"
2 <asp:Content ID="Content1" ContentPlaceHolderID="c1" runat="server">
3 <div class="col-lg-12">
4 <div class="card">
5 <div class="card-header">
6 <strong class="card-title">Împrumut Cărți</strong>
7 </div>
8 <div class="card-body">
9
10 <div id="pay-invoice">
11 <div class="card-body">
12 <form action="" id="foi" runat="server" method="post" novalidate="novalidate">
13 <div class="form-group">
14 <label for="" class="control-label mb-1">Număr de înregistrare</label>
15 <asp:DropDownList ID="nrinr" runat="server" class="form-control"></asp:DropDownList>
16 </div>
17 <div class="form-group">
18 <label for="" class="control-label mb-1">ISBN</label>
19 <asp:DropDownList ID="isbn" runat="server" class="form-control" AutoPostBack="True" OnSelectedIndexChanged="isbn_SelectedIndexChanged"></asp:DropDownList>
20 </div>
21 <div class="form-group">
22 <asp:Image ID="i1" runat="server" Height="150" Width="100" /><br />
23 <asp:Label ID="numecarte" runat="server"></asp:Label><br />
24 <asp:Label ID="instoc" runat="server"></asp:Label><br />
25 </div>
26 <div>
27 <asp:Button ID="b1" runat="server" class="btn btn-lg btn-info btn-block" Text="Împrumută cărți" OnClick="b1_Click" />
28 </div>
29 </form>
30 </div>
31 </div>
32 </div>
33 </div>
34 </asp:Content>
35
36

```

Figura 17. Împrumut cărți – cod interfață

imprumut_carti
id
numar_inregistrare_student
isbn_carte
data_imprumut
data_returnare_aproximativa
utilizator_student
este_returnata
data_returnare

Figura 18. Tabel date împrumut

Pentru controalele de tip DropDownList s-a folosit codul din figura 19.

```
//comanda pentru preluarea numarului de inregistrare al studentului din baza de date și adăugarea lui într-un control de tip DropDownList
nrinr.Items.Clear();
SqlCommand cmd = con.CreateCommand();
cmd.CommandType = CommandType.Text;
cmd.CommandText = "select numar from student_conectare";
cmd.ExecuteNonQuery();
DataTable dt = new DataTable();
SqlDataAdapter da = new SqlDataAdapter(cmd);
da.Fill(dt);
foreach(DataRow dr in dt.Rows)
{
    nrinr.Items.Add(dr["numar"].ToString());
}

//comanda pentru preluarea isbn-ului din baza de date și adăugarea lui într-un control de tip DropDownList
isbn.Items.Clear();
isbn.Items.Add("Select");
SqlCommand cmd2 = con.CreateCommand();
cmd2.CommandType = CommandType.Text;
cmd2.CommandText = "select isbn_carte from carti";
cmd2.ExecuteNonQuery();
DataTable dt2 = new DataTable();
SqlDataAdapter da2 = new SqlDataAdapter(cmd2);
da2.Fill(dt2);
foreach (DataRow dr2 in dt2.Rows)
{
    isbn.Items.Add(dr2["isbn_carte"].ToString());
}
```

Figura 19. Controale DropDownList

Procesul de împrumut s-a realizat printr-o serie de validări urmate de actualizarea bazei de date cu informațiile corespunzătoare și actualizarea numărului de exmpare la afișarea cărților. Dacă exemplarul nu a fost selectat, cartea a fost deja împrumutată studentului sau nu există exemplare, utilizator va primi un mesaj de avertizare. Pentru o interfață mai prietenoasă s-a ales afișarea, după selectarea ISBN-ului, a numelui, numărului de exemplare și a coperti corespunzătoare împrumutului (Figura 21).

```
protected void b1_Click(object sender, EventArgs e)
{
    //verificare daca cartea este selectata sau nu
    if (isbn.SelectedItem.ToString() == "Select")
    {
        Response.Write("<script>alert('Alege o carte!');window.location.href=window.location.href+'</script>');");
    }
    else
    {
        //verificare daca studentul a împrumut deja cartea alege sau nu
        int egait = 0;
        SqlCommand cmd0 = con.CreateCommand();
        cmd0.CommandType = CommandType.Text;
        cmd0.CommandText = "select * from imprumut_carti where numar_inregistrare_student=" + nrinr.SelectedItem.ToString() + " and isbn_carte=" + isbn.SelectedItem.ToString() + " and este_returnata='no'";
        cmd0.ExecuteNonQuery();
        DataTable dt0 = new DataTable();
        SqlDataAdapter da0 = new SqlDataAdapter(cmd0);
        da0.Fill(dt0);
        egait = Convert.ToInt32(dt0.Rows.Count.ToString());
        if (egait > 0)
        {
            Response.Write("<script>alert('Carte împrumutată deja studentului selectat!');</script>");
        }
        else
        {
            //verificare daca exista cartea in stoc sau nu
            if (intoc.Text == "0")
            {
                Response.Write("<script>alert('Cartea selectată nu este disponibilă momentan!');</script>");
            }
            else
            {
                //inserarea informatiei in baza de date
                string data_imprumut = DateTime.Now.ToString("yyyy/MM/dd");
                string data_returnare_sporadice = DateTime.Now.AddDays(10).ToString("yyyy/MM/dd");
                string utilizator = "1";

                SqlCommand cmd1 = con.CreateCommand();
                cmd1.CommandType = CommandType.Text;
                cmd1.CommandText = "select * from student_conectare where numar=" + nrinr.SelectedItem.ToString() + " ";
                cmd1.ExecuteNonQuery();
                DataTable dt1 = new DataTable();
                SqlDataAdapter da1 = new SqlDataAdapter(cmd1);
                da1.Fill(dt1);
                foreach (DataRow dr1 in dt1.Rows)
                {
                    utilizator = dr1["utilizator"].ToString();
                }
                SqlCommand cmd2 = con.CreateCommand();
                cmd2.CommandType = CommandType.Text;
                cmd2.CommandText = "insert into imprumut_carti values('" + nrinr.SelectedItem.ToString() + "','" + isbn.SelectedItem.ToString() + "','" + data_imprumut.ToString() + "','" + data_returnare_sporadice.ToString() + "','" + utilizator.ToString() + "','" + intoc.Text + "')";
                cmd2.ExecuteNonQuery();
                //actualizarea numarului de exemplare dupa imprumut
                SqlCommand cmd3 = con.CreateCommand();
                cmd3.CommandType = CommandType.Text;
                cmd3.CommandText = "update carti set cantitate_carte=cantitate_carte-1 where isbn_carte=" + isbn.SelectedItem.ToString() + " ";
                cmd3.ExecuteNonQuery();
                Response.Write("<script>alert('Cartea a fost împrumutată cu succes!');window.location.href=window.location.href+'</script>');");
            }
        }
    }
}
```

Figura 20. Împrumut cărți - cod C#

```

protected void isbn_SelectedIndexChanged(object sender, EventArgs e)
{
    //comanda pentru afisarea imaginii, numelui si cantitatii (carte) la selectia isbn-ului din controlul de tip DropDownList
    i1.ImageUrl = "";
    numecarte.Text = "";
    instoc.Text = "";

    SqlCommand cmd2 = con.CreateCommand();
    cmd2.CommandType = CommandType.Text;
    cmd2.CommandText = "select * from carti where isbn_carte='"+isbn.SelectedItem.ToString()+"'";
    cmd2.ExecuteNonQuery();
    DataTable dt2 = new DataTable();
    SqlDataAdapter da2 = new SqlDataAdapter(cmd2);
    da2.Fill(dt2);
    foreach (DataRow dr2 in dt2.Rows)
    {
        i1.ImageUrl = dr2["coperta_carte"].ToString();
        numecarte.Text = dr2["titlu_carte"].ToString();
        instoc.Text = dr2["cantitate_carte"].ToString();
    }
}

```

Figura 21. Afișare date suplimentare împrumut

Dacă studentul depășește termenul de returnare (10 zile după împrumut) acesta va fi sancționat cu o sumă de bani prestabilită de către bibliotecar/administrator. Pentru sancțiune s-a adăugat un nou tabel în baza de date ce conține doar un element ce poate fi actualizat (suma). În interfață administratorul are posibilitatea de a adăuga această sancțiune. După adăugare se va calcula automat suma finală în funcție de numărul de zile întârziate (afișată pe pagina pentru returnare).

Figura 22. Sancțiune

Pentru returnare, administratorul utilizează pagina prezentată în Figura 23. Afișarea informațiilor s-a realizat cu preluarea din baza de date, am adăugat un datatable temporar pentru a prelucra noțiunile fără să se piardă informația (Figura 25).

Figura 23. Returnare carte

Număr înregistrare	ISBN	Data împrumut	Data returnare (aproximativă)	Utilizator	Returnată?	Data returnare	Întârziere	Sancțiuni (lei)	Returnează
00001	0-7475-8108-8	2020/09/02	2020/09/12	mariab	no	no	0	0	Returnează carte
00002	0-00-224585-X	2020/09/02	2020/09/12	anav	no	no	0	0	Returnează carte
00003	973-922-31-I	2020/09/02	2020/09/12	marceli	no	no	0	0	Returnează carte

```
returnare_carte.aspx* - Output
1 Page Title="" Language="C#" MasterPageFile="~/Biblioteca/Biblioteca.Master" AutoEventWireup="true" CodeBehind="returnare_carte.aspx.cs" Inherits="Biblioteca.Biblioteca.returnare_carte"
2 <asp:Content ID="Content1" ContentPlaceHolderID="c1" runat="server">
3     <div class="col-sm-4">
4         <div class="page-header float-left">
5             <div class="page-title">
6                 <h1>Returnări</h1>
7             </div>
8         </div>
9     </div>
10 </div>
11 <div class="container-fluid" style="min-height:500px; background-color:white">
12     <br />
13     <asp:DataList ID="d1" runat="server">
14         <HeaderTemplate>
15             <table class="table table-bordered">
16                 <tr>
17                     <th>Număr înregistrare</th>
18                     <th>ISBN</th>
19                     <th>Data împrumut</th>
20                     <th>Data returnare (aproximativă)</th>
21                     <th>Utilizator</th>
22                     <th>Returnată</th>
23                     <th>Data returnare</th>
24                     <th>Întârziere</th>
25                     <th>Sanctiuni (lei)</th>
26                     <th>Returnează</th>
27                 </tr>
28             </HeaderTemplate>
29             <ItemTemplate>
30                 <tr>
31                     <td>Eval("numar_inregistrare_student")</td>
32                     <td>Eval("isbn_carte")</td>
33                     <td>Eval("data_imprumut")</td>
34                     <td>Eval("data_returnare_aproximativa")</td>
35                     <td>Eval("utilizator_student")</td>
36                     <td>Eval("este_returnata")</td>
37                     <td>Eval("data_returnare")</td>
38                     <td>Eval("zile_intarziere")</td>
39                     <td>Eval("sanctiune")</td>
40                     <td>a href="returnarecarte.aspx?id=Eval("id")>Returnează carte</td>
41                 </tr>
42             </ItemTemplate>
43             <FooterTemplate>
44             </table>
45             </FooterTemplate>
46         </asp:DataList>
47     </div>
48 </asp:Content>
```

Figura 24. Returnare carte - cod interfață

În figura 24 este surprins codul pentru afișarea tabelului de returnare. Codul C# este prezentat mai jos.

```
Biblioteca Biblioteca.biblioteca.returnare_cart
38 }
39 // transmite datele in baza de date temporara
40 //datatable temporara
41 DataTable dt = new DataTable();
42 dt.Clear();
43 dt.Columns.Add("id");
44 dt.Columns.Add("numar_inregistrare_student");
45 dt.Columns.Add("isbn_carte");
46 dt.Columns.Add("data_imprumut");
47 dt.Columns.Add("data_returnare_aproximativa");
48 dt.Columns.Add("utilizator_student");
49 dt.Columns.Add("este_returnata");
50 dt.Columns.Add("data_returnare");
51 dt.Columns.Add("zile_intarziere");
52 dt.Columns.Add("sanctiune");
53 //comanda pentru completarea tabelului din interfaa
54 SqlCommand cmd1 = con.CreateCommand();
55 cmd1.CommandType = CommandType.Text;
56 cmd1.CommandText = "select * from imprumut_carti where este_returnata='no'";
57 cmd1.ExecuteNonQuery();
58 DataTable dt1 = new DataTable();
59 SqlDataAdapter dal = new SqlDataAdapter(cmd1);
60 dal.Fill(dt1);
61 foreach (DataRow dr1 in dt1.Rows)
62 {
63     DataRow dr = dt.NewRow();
64     dr["id"] = dr1["id"].ToString();
65     dr["numar_inregistrare_student"] = dr1["numar_inregistrare_student"].ToString();
66     dr["isbn_carte"] = dr1["isbn_carte"].ToString();
67     dr["data_imprumut"] = dr1["data_imprumut"].ToString();
68     dr["data_returnare_aproximativa"] = dr1["data_returnare_aproximativa"].ToString();
69     dr["utilizator_student"] = dr1["utilizator_student"].ToString();
70     dr["este_returnata"] = dr1["este_returnata"].ToString();
71     dr["data_returnare"] = dr1["data_returnare"].ToString();
72     //calcul zile intarziere = calculul sanctiunii
73     DateTime d1 = Convert.ToDateTime(DateTime.Now.ToString("yyyy/MM/dd"));
74     DateTime d2 = Convert.ToDateTime(dr1["data_returnare_aproximativa"].ToString());
75     if (d1 <= d2)
76     {
77         dr["zile_intarziere"] = "0";
78     }
79     else
80     {
81         TimeSpan t = d1 - d2;
82         nrzile = t.TotalDays;
83         dr["zile_intarziere"] = nrzile.ToString();
84     }
85     dr["sanctiune"] = Convert.ToString(Convert.ToDouble(nrzile) * Convert.ToDouble(sanctiune));
86 }
87 dt.Rows.Add(dr);
88 }
89 d1.DataSource = dt;
90 d1.DataBind();
91 }
92 }
```

Figura 25. Returnare Carte - cod C#

La acțiunea butonului „Returnează carte” se schimbă statusul din baza de date, se actualizează data returnării cu data curentă și numărul de exemplare după returnare.

```
//comanda care actualizeaza baza de date cu data returnarii si schimba statusul cartii (este_returnata= yes)
id = Convert.ToInt32( Request.QueryString["id"].ToString());
SqlCommand cmd = con.CreateCommand();
cmd.CommandType = CommandType.Text;
cmd.CommandText = "update imprumut_carti set este_returnata='yes', data_returnare='"+ DateTime.Now.ToString("yyyy/MM/dd") +"' where id="+ id +"";
cmd.ExecuteNonQuery();

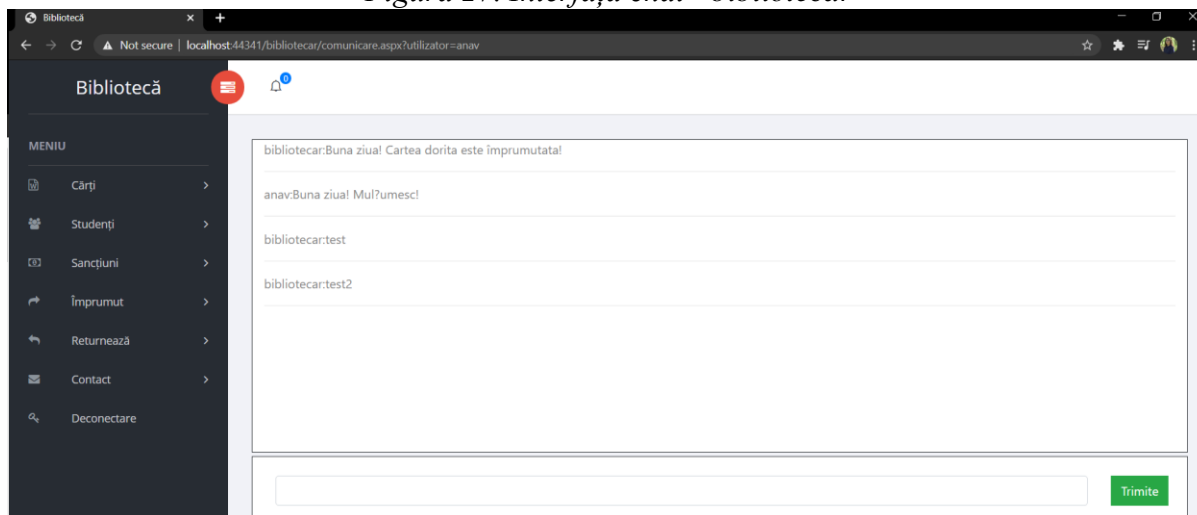
//comanda care actualizeaza stocul cartilor dupa returnarea exemplarului
SqlCommand cmd1 = con.CreateCommand();
cmd1.CommandType = CommandType.Text;
cmd1.CommandText = "select * from imprumut_carti where id="+ id +"";
cmd1.ExecuteNonQuery();
DataTable dt1 = new DataTable();
SqlDataAdapter da1 = new SqlDataAdapter(cmd1);
da1.Fill(dt1);
foreach(DataRow dr1 in dt1.Rows)
{
    isbn_carte = dr1["isbn_carte"].ToString();
}
SqlCommand cmd2 = con.CreateCommand();
cmd2.CommandType = CommandType.Text;
cmd2.CommandText = "update carti set cantitate_carte=cantitate_carte + 1 where isbn_carte='"+ isbn_carte.ToString() +"";
cmd2.ExecuteNonQuery();

Response.Redirect("returnare_carte.aspx");
```

Figura 26. Carte returnată - cod C#

Pentru funcționalitatea de contact am adăugat un nou tabel în SQL server dedicat mesajelor, ce conține sursa, destinatarul, mesajul și un status care surprinde dacă mesajul a fost primit sau nu. Codul pentru chat este scris utilizând JavaScript și conține 3 funcții, de trimitere, încărcare și actualizare a mesajelor. Interfața este prezentată în Figura 27, iar codul în Figura 29.

Figura 27. Interfață chat - bibliotecar



Pentru selectarea destinatarului, administratorului îi este prezentată la acțiunea butonului „Trimite Mesaj” din secțiunea „Contact”, aceeași listă de la afișarea studenților, cu mențiunea că în dreptul fiecăruia există opțiunea de chat. Codul pentru afișarea studenților este asemănător cu cel de la afișarea cărților.

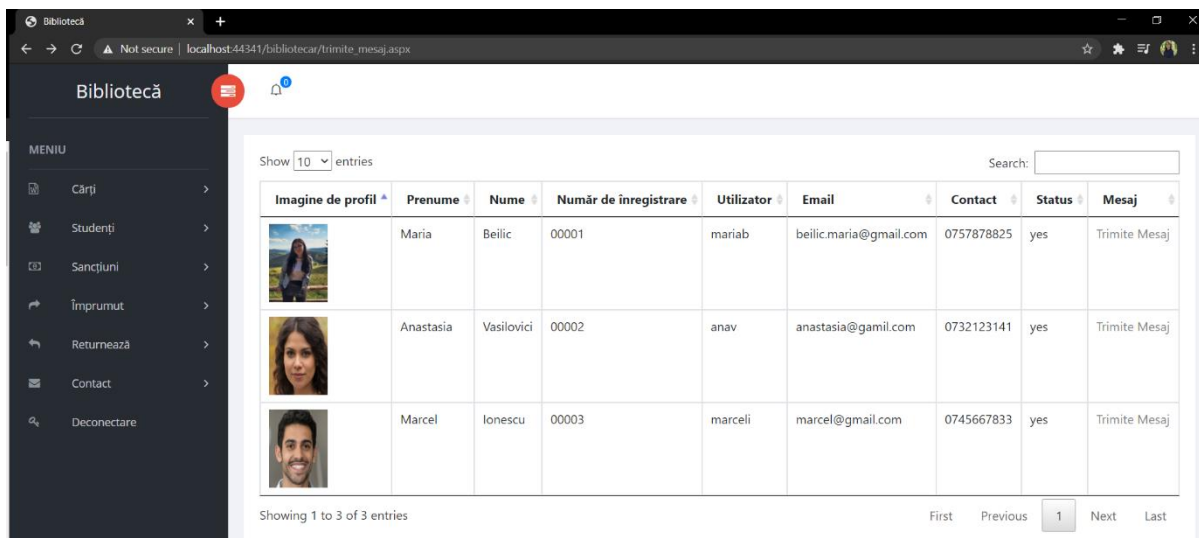


Figura 28. Selectare destinatar

Pagina de chat este simplă, conține 2 căsuțe destinate afișării, și respectiv, scrierii mesajelor urmate de un buton ce apelează funcția de trimitere a mesajelor.

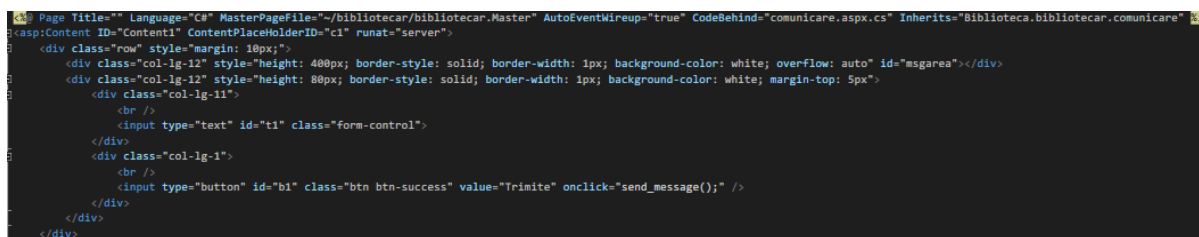


Figura 29. Interfață mesaje

Funcția „send\_message()” va solicita server-ului utilizatorul și mesajul din baza de date. Înscrierea informației în baza de date este prezentată în Figura 31.



Figura 30. Funcția trimite mesaj

```

1 using System;
2 using System.Data;
3 using System.Data.SqlClient;
4
5 namespace Biblioteca.biblioteca
6 {
7     1 reference
8     public partial class mesaje_trimise_de_biblioteca : System.Web.UI.Page
9     {
10         //initializare obiect nou de tip sqlconnection ce permite accesul la informatia din baza de date si manipularea informatiei
11         SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\beili\OneDrive\Desktop\Practica\Biblioteca\App_Data\lms.mdf;Integrated Security=True");
12         string utilizator = "";
13         string msg = "";
14
15         0 references
16         protected void Page_Load(object sender, EventArgs e)
17         {
18             //verifica daca conexiunea este deschisa, daca aceasta nu este o deschide
19             if (con.State==ConnectionState.Open)
20             {
21                 con.Close();
22             }
23             con.Open();
24             //dupa o perioada de inactivitate sau daca incercam sa ne conectam direct la pagina de adaugare suntem trimisi in pagina de conectare
25             if (Session["biblioteca"] == null)
26             {
27                 Response.Redirect("conectare.aspx");
28             }
29             //preluarea utilizatorului si a mesajului
30             utilizator = Request.QueryString["utilizator"].ToString();
31             msg = Request.QueryString["msg"].ToString();
32             //comanda pentru inserarea valorilor in baza de date
33             SqlCommand cmd = con.CreateCommand();
34             cmd.CommandType = CommandType.Text;
35             cmd.CommandText = "insert into mesaje values('biblioteca','"+ utilizator.ToString() + "','"+ msg.ToString() + "','no')";
36             cmd.ExecuteNonQuery();
37         }
38     }
39 }

```

Figura 31. Inserare mesaj în SQL Server

Funcția „load\_messages()” returnează informația din baza de date în formatul: “sursă : mesaj” în căsuța de afișare.

```

//functie incarca mesaje
function load_messages() {
    var xmlhttp = new XMLHttpRequest(); //obiect nou de tip XMLHttpRequest
    xmlhttp.onreadystatechange = function () { // eveniment declansat de schimbarea readyState
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) { //readyState tine statusul XMLHttpRequest care se schimba de la 0 la 4
            document.getElementById("msgarea").innerHTML = xmlhttp.responseText; //returneaza data sub forma de string
        }
    };
    xmlhttp.open("GET", "incarca_mesaje.aspx?utilizator=" + utilizator, true); //specifica tipul cererii, locatia si daca este asincron/sincron
    xmlhttp.send(); //trimite o cerere la server
}
load_messages();

```

Figura 32. Funcția încărcare mesaje

```

7
8
9     1 reference
10    public partial class incarca_mesaje : System.Web.UI.Page
11    {
12        //initializare obiect nou de tip sqlconnection ce permite accesul la informatia din baza de date si manipularea informatiei
13        SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\beili\OneDrive\Desktop\Practica\Biblioteca\App_Data\lms.mdf;Integrated Security=True");
14        string utilizator = "";
15
16        0 references
17        protected void Page_Load(object sender, EventArgs e)
18        {
19            //verifica daca conexiunea este deschisa, daca aceasta nu este o deschide
20            if (con.State == ConnectionState.Open)
21            {
22                con.Close();
23            }
24            con.Open();
25            //dupa o perioada de inactivitate sau daca incercam sa ne conectam direct la pagina de adaugare suntem trimisi in pagina de conectare
26            if (Session["biblioteca"] == null)
27            {
28                Response.Redirect("conectare.aspx");
29            }
30            //preluarea utilizatorului
31            utilizator = Request.QueryString["utilizator"].ToString();
32            //selectarea informatiei din baza de date
33            SqlCommand cmd = con.CreateCommand();
34            cmd.CommandType = CommandType.Text;
35            cmd.CommandText = "select * from mesaje where (utilizator='"+ utilizator.ToString() + "' and utilizator='biblioteca') or (utilizator='"+ utilizator.ToString() + "' and utilizator='biblioteca')";
36            cmd.ExecuteNonQuery();
37            //modul de afisare
38            DataTable dt = new DataTable();
39            SqlDataAdapter da = new SqlDataAdapter(cmd);
40            da.Fill(dt);
41            foreach(DataRow dr in dt.Rows)
42            {
43                Response.Write("<br>");
44                Response.Write(dr["utilizator"].ToString() + " : " + dr["msg"].ToString());
45                Response.Write("<br>");
46                Response.Write("<br>");
47            }
48            //schimbare status
49            if (dr["utilizator"].ToString() == "biblioteca")
50            {
51                SqlCommand cmd1 = con.CreateCommand();
52                cmd1.CommandType = CommandType.Text;
53                cmd1.CommandText = "update mesaje set plasat='yes' where id=" + dr["id"].ToString() + ";";
54                cmd1.ExecuteNonQuery();
55            }
56        }
57    }
58 }

```

Figura 33. Încărcare mesaje - Cod C#



Pentru actualizarea mesajelor am folosit funcția „add\_inside\_new\_message()” care la interval de 10 secunde trimite mesajele noi preluate din baza de date.

```
//functie actualizare mesaje
function add_inside_new_message() {
    var xmlhttp = new XMLHttpRequest(); //obiect nou de tip XMLHttpRequest
    xmlhttp.onreadystatechange = function () { //eveniment declansat de schimbarea readyState
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) { //readyState tine statusul XMLHttpRequest care se schimba de la 0 la 4
            if (xmlhttp.responseText != "0") { //verifica daca string-ul rezultat este 0
                var strArray = xmlhttp.responseText.split("||abcd||"); //obtinem un vector prin impartirea string ului
                for (var i = 0; i < strArray.length; i++) { //actualizarea chat ului
                    var theDiv = document.getElementById("msgarea");
                    var x = document.createElement("p");
                    var t = document.createTextNode(strArray[i]);
                    x.appendChild(t);
                    theDiv.appendChild(x);
                    var y = document.createElement("hr");
                    theDiv.appendChild(y);
                    theDiv.scrollTop = theDiv.scrollHeight;
                }
            }
        }
    };
    xmlhttp.open("GET", "incarca_mesaje_noi.aspx?utilizator=" + utilizator, true); //specifica tipul cererii, locatia si daca este asincron/sincron
    xmlhttp.send(); //trimite o cerere la server
}

//mesajele apar la 10 secunde
setInterval(function () {
    add_inside_new_message();
}, 10000);

//functie returneaza parametrii din pagina ceruta cu separarea in componente mai usor de procesat (parse)
function getUrlVars() {
    var vars = {};
    var parts = window.location.href.replace(/[?&]+([^\&]+)=([^\&]*)/gi, function (m, key, value) {
        vars[key] = value;
    });
    return vars;
}
```

Figura 34. Funcția actualizare mesaje

```
incarca_mesaje_noi.aspx.cs  incarca_mesaje.aspx.cs
Biblioteca  Biblioteca.bibliotecar.incarga_mesaje_noi

28 //preluarea utilizatorului
29 utilizator = Request.QueryString["utilizator"].ToString();
30 //selecteaza mesajele care au destinatarul: bibliotecarul si statusul: no
31 //actualizarea mesajelor
32 SqlCommand cmd = con.CreateCommand();
33 cmd.CommandType = CommandType.Text;
34 cmd.CommandText = "select * from mesaje where dutilizator='bibliotecar' and plasat='no' ";
35 cmd.ExecuteNonQuery();
36 DataTable dt = new DataTable();
37 SqlDataAdapter da = new SqlDataAdapter(cmd);
38 da.Fill(dt);
39 foreach(DataRow dr in dt.Rows)
40 {
41     contor = contor + 1;
42     if(contor==1)
43     {
44         msg = dr["sutilizator"].ToString() + ":" + dr["msg"].ToString();
45     }
46     else
47     {
48         msg = msg + "||abcd||" + dr["sutilizator"].ToString() + ":" + dr["msg"].ToString();
49     }
50     SqlCommand cmd1 = con.CreateCommand();
51     cmd1.CommandType = CommandType.Text;
52     cmd1.CommandText = "update mesaje set plasat='yes' where id=" + dr["id"].ToString() + " ";
53     cmd1.ExecuteNonQuery();
54 }
55
56 if(contor==0)
57 {
58     Response.Write("0");
59 }
60 else
61 {
62     Response.Write(msg.ToString());
63 }
64 }
65 }
66 }
```

Figura 35. Actualizare mesaje - Cod C#

La deconectare utilizatorul va fi trimis pe pagina de conectare.

## 2. Student

Pentru a accesa aplicația din punctul de vedere al studentului avem nevoie de un utilizator și de o parolă. Pentru a obține aceste date, studentul are opțiunea de a se înregistra. Interfața pentru înregistrare este asemănătoare celei de adăugare a unei cărți, dar fără Master page.

Figura 37. Înregistrare student

Datele sunt supuse unor seturi de validări ce nu permit înregistrarea unui student cu același utilizator, același număr de înregistrare și verifică dacă studentul este robot sau nu. Înregistrarea studentului se face cu introducerea datelor într-un table nou din SQL Server. Prima validare este cea de la Google, reCAPTCHA, prezentată în figurile 38 și 39.

```
//verificare robot
//reference
public bool IsReCaptchValid()
{
    var result = false;
    var captchaResponse = Request.Form["g-recaptcha-response"];
    var secretkey = "6LfadsQZAAAAAC1gr2c8JBrStvcglsgwKMDPcQ";
    var apiUrl = "https://www.google.com/recaptcha/api/siteverify?secret=" + secretkey + "&response=" + captchaResponse;
    var requestUrl = string.Format(apiUrl, secretkey, captchaResponse);
    var request = (HttpRequest)WebRequest.Create(requestUrl);

    using (WebResponse response = request.GetResponse())
    {
        using (StreamReader stream = new StreamReader(response.GetResponseStream()))
        {
            JObject jsonResponse = JObject.Parse(stream.ReadToEnd());
            var isSuccess = jsonResponse.Value("success");
            result = (isSuccess) ? true : false;
        }
    }
    return result;
}
```

Figura 38. reCAPTCHA - Cod C#

```
<script src="https://www.google.com/recaptcha/api.js?onload=renderReCaptch&render=explicit" async defer></script>
<script type="text/javascript">
    var your_site_key = '6LfadsQZAAAAAC1gr2c8JBrStvcglsgwKMDPcQ';
    var renderReCaptch = function () {
        grecaptcha.render("McCaptchContainer", {
            'sitekey': '6LfadsQZAAAAAC1gr2c8JBrStvcglsgwKMDPcQ',
            'callback': reCaptchCallback,
            'theme': 'light', //light or dark
            'type': 'image', //image or audio
            'size': 'normal' //normal or compact
        });
    };
    var reCaptchCallback = function (response) {
        if (response != '') {
            document.getElementById("lblMessage1").innerHTML = "";
        }
    };
</script>
```

Figura 39. reCAPTCHA

Dupa apelarea funcției „IsReCaptchValid()” este surprinsă verificarea numărului de înregistrare introdus, cu informația din baza de date. Dacă aceste date coincid, utilizatorul va primi un mesaj de avertizare.

```
protected void b1_Click(object sender, EventArgs e)
{
    int count = 0;
    int contor = 0;
    if (IsRecaptchValid())
    {
        //comanda care selecteaza din baza de date studentul dupa nr de inregistrare
        SqlCommand cmd1 = con.CreateCommand();
        cmd1.CommandType = CommandType.Text;
        cmd1.CommandText = "select * from student_conectare where numar='"+nrinregistrare.Text+"'";
        cmd1.ExecuteNonQuery();
        DataTable dt1 = new DataTable();
        SqlDataAdapter da1 = new SqlDataAdapter(cmd1);
        da1.Fill(dt1);
        count = Convert.ToInt32(dt1.Rows.Count.ToString());
        //daca numarul este gasit utilizator primeste un mesaj de avertizare
        if (count > 0)
        {
            Response.Write("<script>alert('Numărul ales există deja!');</script>");
        }
    }
}
```

Figura 40. Validare număr de înregistrare

Dacă numărul de înregistrare este disponibil, se trece la verificarea utilizatorului într-un mod asemănător.

```
else
{
    // comanda care selecteaza din baza de date studentul dupa utilizator
    //verificare utilizator (daca exista deja username-ul, utilizatorul este atentionat)
    SqlCommand cmd2 = con.CreateCommand();
    cmd2.CommandType = CommandType.Text;
    cmd2.CommandText = "select * from student_conectare where utilizator='" + utilizator.Text + "'";
    cmd2.ExecuteNonQuery();
    DataTable dt2 = new DataTable();
    SqlDataAdapter da2 = new SqlDataAdapter(cmd2);
    da2.Fill(dt2);
    contor = Convert.ToInt32(dt2.Rows.Count.ToString());
    //
    if (contor > 0)
    {
        Response.Write("<script>alert('Utilizatorul ales există deja!');</script>");
    }
}
```

Figura 41. Validare utilizator

După verificări, informațiile furnizate de utilizator sunt introduse în baza de date.

```
else
{
    //Modifica numele imaginii de profil cu un nume random prin functia GetRandomPassword din Class1
    string randomo = Class1.GetRandomPassword(10) + ".jpg";
    string path = "";
    fi.SavesAs(Request.PhysicalApplicationPath + "/student/imaginedeprofil/" + randomo.ToString());
    path = "student/imaginedeprofil/" + randomo.ToString();
    //aduga datele in baza de date dupa validari
    SqlCommand cmd = con.CreateCommand();
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "insert into student_conectare values('"+ prenume.Text + "','" + nume.Text + "','" + nrinregistrare.Text + "','" + utilizator.Text + "','" + parola.Text + "','" + email.Text + "','" + contact.Text + "','" + path.ToString() + "','" + no'");";
    cmd.ExecuteNonQuery();
    Response.Redirect("conectare_student.aspx");
}
}

//altfel mesaj de atentionare
else
{
    lblMessage1.Text = "Înregistrare nereușită!";
}
}
```

Figura 42. Înregistrare date student

Pentru activarea contului de student este nevoie de aprobarea administratorului. Acest subiect a fost atins în partea dedicată acestuia. Studentul cu un cont activ se va putea conecta de pe pagina următoare.



The image shows a login form titled "Conectare Biblioteca". It contains two input fields: "UTILIZATOR" with the text "utilizator" and "PAROLA" with the text "parola". Below these fields is a green button labeled "CONECTARE". At the bottom, there is a link that says "Nu ai cont? Înregistrează-te acum."

Figura 43. Conectare Student

După conectare, studentul va fi îndrumat către meniul principal. Master page-ul este asemănător celui dedicat administratorului pentru continuitate.

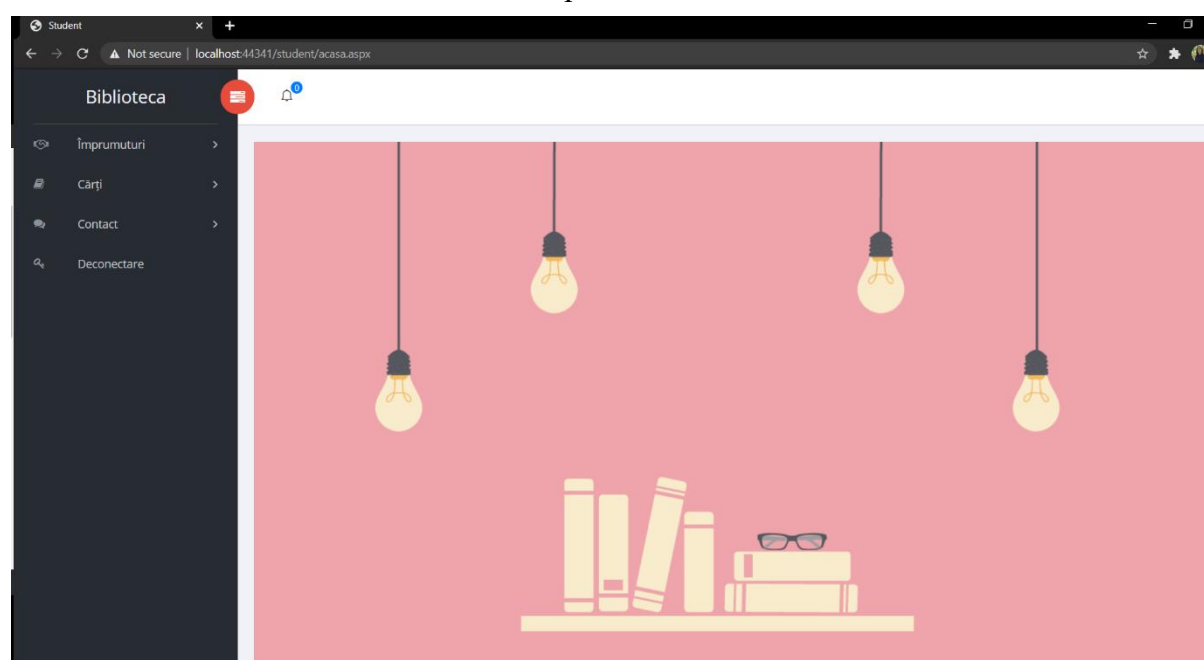


Figura 44. Meniu Student

La acțiunea butonului „Cărțile mele” din secțiunea „Împrumuturi”, studentului îi sunt afișate cărțile împrumutate, având asociate datele necesare (Figura 45). Interfața este asemănătoare cu cea de la returnări din partea dedicată administratorului, cu mențiunea că studentul nu poate prelucra împrumutul. Afișarea s-a făcut prin preluarea datelor asociate numărului de înregistrare unic.

Student

Not secure | localhost:44341/student/carti\_imprumutate.aspx

Biblioteca

Imprumuturi

Cărți

Contact

Deconectare

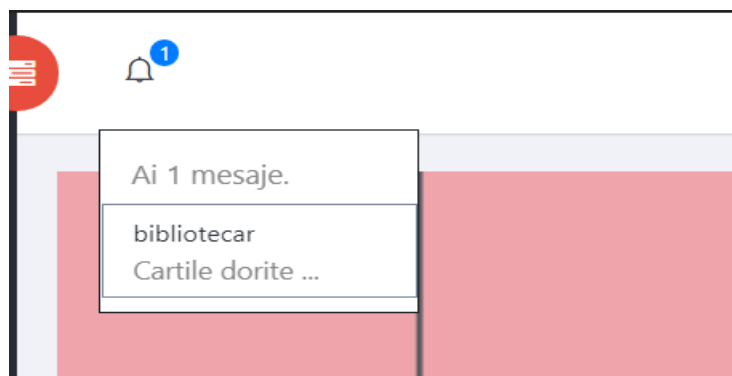
Cărțile mele

Număr înregistrare	ISBN	Data împrumut	Data returnare (aproximativă)	Utilizator	Returnată?	Data returnare	Întârziere	Sanctiuni (lei)
00001	0-7475-8108-8	2020/09/02	2020/09/12	mariaab	no	no	0	0
00001	0-7475-4624-X	2020/09/06	2020/09/16	mariaab	no	no	0	0

*Figura 45. Cărțile mele împrumutate*

Afișarea cărților este identică cu cea de la administrator (cod + interfață). Studentul are acces la toate informațiile despre cărți dar nu le poate manipula. Trimiterea de mesaje se realizează pe același principiu ca în cazul administratorului doar că, sursa în cazul actual este studentul și destinatarul este bibliotecarul.

La acționarea butonului „Deconectare” studentul este trimis pe pagina de conectare. La conectarea în cont utilizatorul (student/administrator) va avea o notificare ce îl va anunța de existența mesajelor necitite și numărul acestora (dacă este cazul). Dacă acesta va da click pe notificare va fi trimis pe pagina de chat iar notificarea va dispărea.



*Figura 46. Notificări*

```
<div id="right-panel" class="right-panel">

<!-- Header-->
<header id="header" class="header">
  <div class="header-menu">
    <div class="col-sm-7">
      <a id="menuToggle" class="menutoggle pull-left"><i class="fa fa fa-tasks"></i></a>
    <div class="header-left">
      <div class="dropdown for-message">
        <button class="btn btn-secondary dropdown-toggle" type="button"
          id="message"
          data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
          <i class="ti-bell"></i>
          <span class="count bg-primary"><asp:Label ID="notificare1" runat="server"></asp:Label></span>
        </button>
        <div class="dropdown-menu" aria-labelledby="message" style="border-style:solid; border-width:1px">
          <p class="red">Ai <asp:Label ID="notificare2" runat="server"></asp:Label> mesaj</p>
          <asp:Repeater ID="r1" runat="server">
            <ItemTemplate>
              <a class="dropdown-item media" href="trinitemesaj.aspx" style="border-style:solid; border-width:1px; border-color:lightslategray">
                <span class="message media-body">
                  <span class="name float-left"><Eval("utilizator")> </span>
                  <p><get20decaractere(Eval("msg"))> </p>
                </span>
              </a>
            </ItemTemplate>
          </asp:Repeater>
        </div>
      </div>
    </div>
  </div>
</div>

</div>
</header>
</div>
```

*Figura 47. Cod notificare*

## IV. Concluzii

În concluzie, această aplicație nu reprezintă un soft profesional, ea execută cu succes opțiunile puse la dispoziție fără erori majore dar există oricând loc de îmbunătățiri. După această experiență am reușit să asimilez un număr semnificativ de cunoștințe și să aprofundez noțiuni deja știute. Printre beneficiile lucrului în ASP.NET am surprins următoarele: un număr redus de linii de cod, tehnologie ideal server-side, informații de configurare built-in și un domeniu vast de limbaje de programare cu care poți lucra.

## V. Bibliografie

Am atașat câteva linkuri ce m-au ajutat în realizarea acestui proiect.

<https://www.w3schools.com/asp/default.ASP>

<https://www.tutorialspoint.com/asp.net/index.htm>

<https://dotnet.microsoft.com/learn/aspnet>

[https://www.youtube.com/watch?v=C5cnZ-gZy2I&ab\\_channel=freeCodeCamp.org](https://www.youtube.com/watch?v=C5cnZ-gZy2I&ab_channel=freeCodeCamp.org)

<https://www.javatpoint.com/asp-net-tutorial>

<https://www.c-sharpcorner.com/article/how-to-connect-sql-database-in-asp-net-using-c-sharp-and-insert-and-view-the-data-usi/>

<https://www.sqlservertutorial.net/>

<https://developers.google.com/recaptcha/intro>

<https://stackify.com/asp-net-performance-tuning/>