

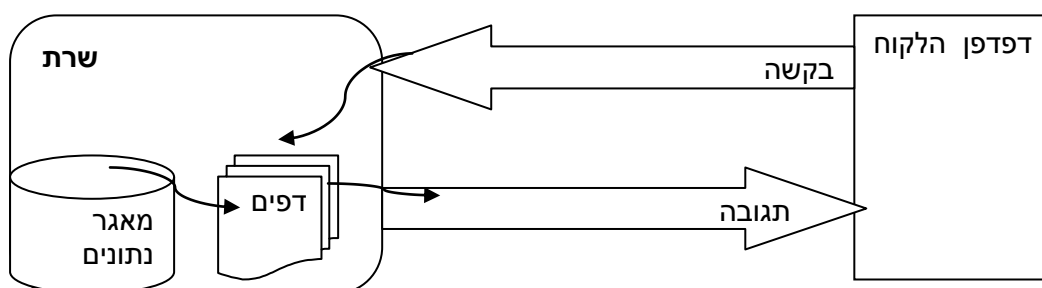
פרק 5 - תצוגה מותאמת למשתמש (שימוש בנתוני רישום)

- 5.1 - איך עובד האינטרנט
- 5.2 - קבלת הנתונים בשרת
- 5.3 - הקובץ Global.asax
- 5.4 - שליחת נתונים מהשרת למשתמש (הלקוח)
- 5.5 - התנתקות
- 5.6 - תפריט אישי מותאם למשתמש
- 5.7 - דפים מוגנים

5.1 לפני שנמשך נלמד איך עובד האינטרנט

(הסבר על המושגים בנספח ה- מושגים)

הלקוח שולח לשרת בקשה **HTTP Request** (הבקשה היא כתובת **URL**), השרת מחפש את הדף המבוקש ושולח ללקוח תגובה **HTTP Response**. הדפדפן של הלקוח מקבל את הדף ומציג אותו. השרת, לפני משלוח הדף ללקוח בודק אם יש הוראות שעליו לבצע לפני המשלוח (בדף **aspx** או בדף **aspx.cs**) השילוב של הוראות לשרת וסקריפטים, ובקשות לקוח מאפשר לנו לבנות דפים דינמיים.



תגובת השרת היא תמיד בדף **HTML**. לפעמים הדף הנשלח כלל אינו נמצא בשרת אלא נבנה במיוחד לפי בקשת הלקוח.

לדוגמה: בקשנו רשימת מתכונים לעוגות שוקולד. השרת מחפש במאגר שלו ובונה דף שבו רשימת מתכונים.

באשר השרת סיים את משלוח הדף המבוקש, נותק הקשר עם הלקוח. השרת אינו שומר מידע על ההתקשרות. גישה זו מכונה - **תיכנות חסר מצב stateless**. מצב זה יוצר בעיה כי בכל פניה לשרת על הלקוח להזדהות מחדש. (נלמד בהמשך איך פותרים בעיה זו).

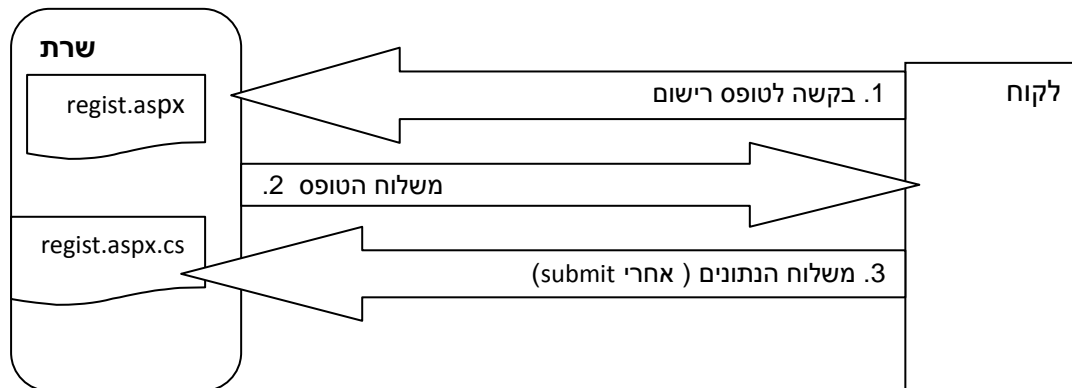
תזכורת - כותרת הטופס מכילה את ההגדרות לטיפול בטופס

```
<form id="form1" method="post" runat="server" action="" onsubmit="return Check()">
```

-action שם הקובץ בשרת אליו ישלחו הנתונים.

אם בנינו את הטופס בדף ששמו **Regist.aspx** בשיטת העבודה של **ASP.NET** ישלחו הנתונים לדף **Regist.aspx.cs**, שהוא דף "קוד מאחור" שיכיל את הפקודות הנדרשות לקבלת הנתונים וטיפול בהם. מאחר והפניה זו היא ברירת מחדל אנו לא נשתמש במאפיין **action**.

חשוב לזכור מה עושה השרת כאשר אנחנו מבקשים את דף הרישום `Regist.aspx`. בפעם הראשונה שביקשנו את הדף - עליו לשלוח אותו ללקוח כפי שהוא. בלי לבצע את הפקודות בדף `Regist.aspx.cs`.



את הפקודות לקליטת הנתונים שנשלחו שנכתוב בדף `Regist.aspx.cs` עליו לבצע רק בקריאה חוזרת לדף. מצב זה של קריאה חוזרת מכונה **PostBack**.

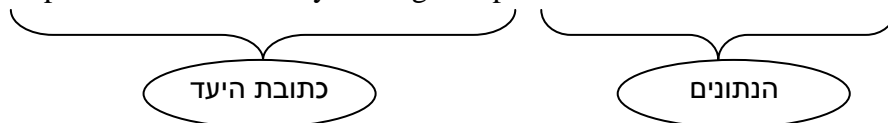
5.2 קבלת הנתונים בשרת

הנתונים שמולאו בטופס נשלחים לשרת. השרת מקבל זוגות של נתונים [שם שדה והערך] שתי שיטות קיימות למשלוח הנתונים (פרק 3 סעיף 3.1)

בשיטת GET - נשלחים הנתונים בשרת הכתובת

כאשר המשתמש לוחץ על כפתור "שלח" (`submit`) נשלח לשרת מידע הכולל את שמות השדות והערכים שלהם. המידע משורשר בשרת הכתובת. (השדות בטופס בדוגמה זו הם: `Name, Year`)

`http://localhost:4835/MySite/regist.aspx? Name=dita&Year=1987`



(בשורה המקורית יש מידע נוסף שאינו שייך לנושא)

אורך המחרוזת הנשלחת מוגבל ל-1000 תווים והמידע גלוי לעיני כל.

שיטה זו משמשת את מנועי החיפוש. אנו לא נשתמש בה.

בשיטת POST - נשלחים הנתונים כחבילה בפרוטוקול HTTP הכוללת את כתובת היעד ובתוכה ומחרוזת משורשרת של שמות השדות והערכים שלהם.

שיטה זו אין מגבלה לכמות הנתונים הנשלחים והמידע אינו גלוי. אנו נשתמש בשיטה זו.

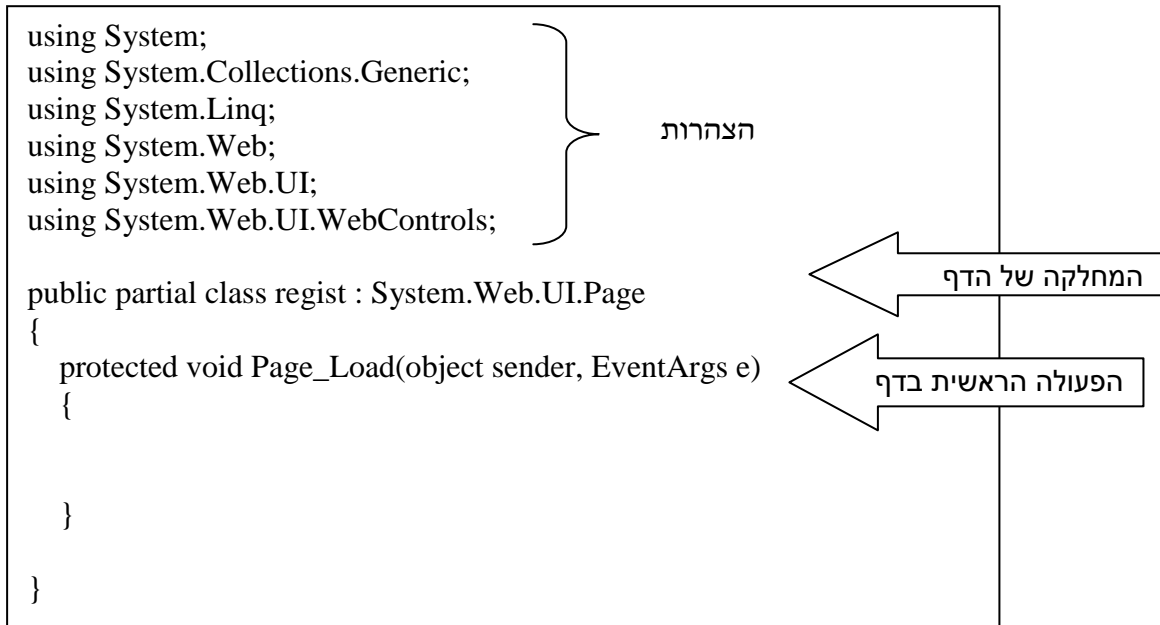
לכל קבצי aspx.cs (בשפת C#) מבנה התחלתי זהה

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class regist : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
}
    
```

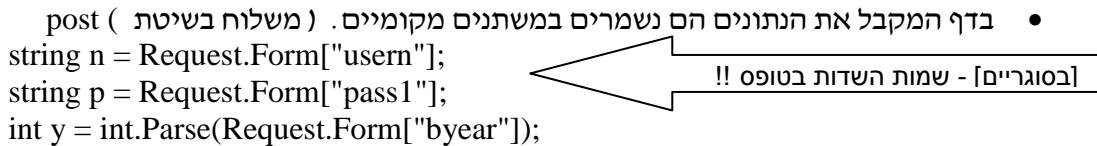


בדף מוגדרת מראש פעולה אחת המתבצעת עם טעינת הדף ושליחתו למשתמש מצב - Page_Load .

שלב א . קבלת הנתונים .

```

post ( משלוח בשיטת )
string n = Request.Form["usern"];
string p = Request.Form["pass1"];
int y = int.Parse(Request.Form["byear"]);
    
```



בקובץ זה קיימת פעולה Page_Load() המופעלת בכל פעם שהדף מוצג. אז מה הבעיה ?

קליטת נתוני רישום וההודעה יתבצעו רק בטעינה חוזרת של הדף במצב של PostBack .

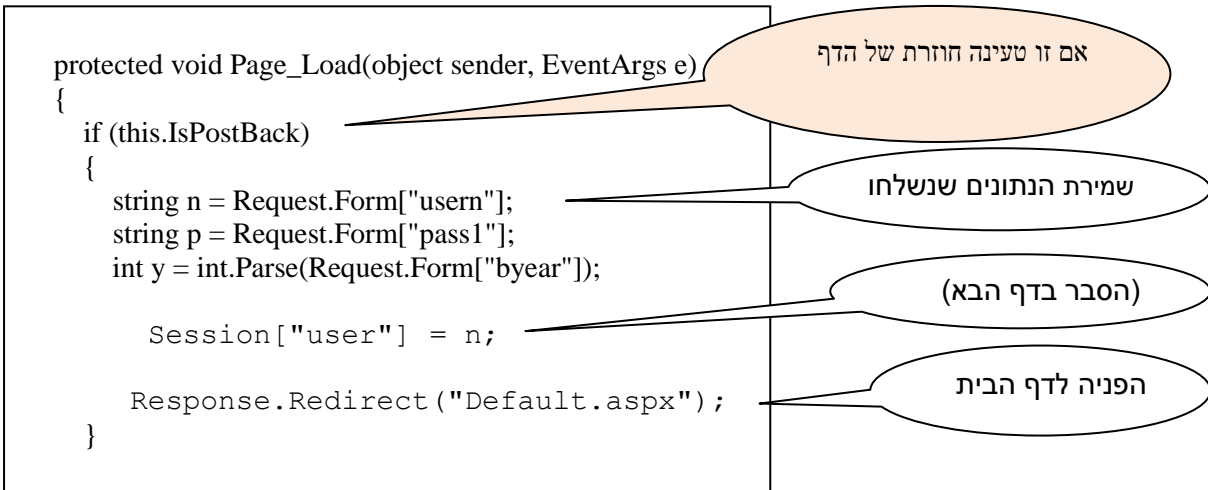
- בפעם הראשונה – מוצג הטופס
- בפעם השנייה – לאחר submit נטען הדף בשרת בפעם השנייה ורק אז יתבצעו הפקודות .

```

protected void Page_Load(object sender, EventArgs e)
{
    if (this.IsPostBack)
    {
        string n = Request.Form["usern"];
        string p = Request.Form["pass1"];
        int y = int.Parse(Request.Form["byear"]);

        Session["user"] = n;

        Response.Redirect ("Default.aspx");
    }
}
    
```



(בהמשך נלמד לשמור את הנתונים במאגר לשימוש חוזר .)

if (this.IsPostBack) - הדף (this) בודק את הסטאטוס שלו. איך הוא יודע? הרי אמרנו כי

אחרי המשלוח השרת אינו שומר מידע על ההתקשרות. (מצב Stateless)

בזמן בניית הדף למשלוח למשתמש מוסיף ASP.NET שדה מוסתר בטופס המכיל מחרוזת טקסט בה מתואר מצב הפקדים בטופס ומצב הדף. View State מאפשר לשמור את מצב הפקדים לאחר שליחתם. (ASP.NET מנהלת מצב זה אוטומטית).

למשתמש נשלח קובץ HTML דינמי המכיל שדה מוסתר.

(view source) אפשר לראות אותו בהרצה של הטופס

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
```

```
value="/wEPDwUKMTY1NDU2MTA1MmRkaTxCrCilYGH5c29ZhoFLvPBOvb4"= />
```

כאשר המשתמש שולח את הטופס לשרת נשלח גם המידע בשדה זה. כך השרת יודע כי זו שליחה חוזרת של הדף והמצב הוא **this.IsPostBack** הוא אמת.

Response.Redirect("Default.aspx"); - פעולה נוספת של האובייקט **Response** בשרת.

לנתב את המשתמש לדף אחר.

יישומי ה-web עובדים בתכנות חסר מצב - Stateless programming - הלקוח שולח לשרת בקשה, השרת שולח תגובה והקשר ניתק. אחרי שהשרת שולח טופס ללקוח הוא אינו זוכר ושומר מידע על ההתקשרות. כך שהשרת אינו "מכיר" את הגולש גם אם התחבר למערכת. לכן אנו זקוקים לדרך שבה השרת ישמור מידע על הקשרים שלו עם המשתמש, יזכור כמה ביקרו באתר, מה היו העדפות של כל גולש. (המידע המלא מטופס הרישום למשל, נשמר במאגר נתונים - בפרק הבא)



האובייקט Session ישמור מידע על כל גולש (שם, סטאטוס, כתובת IP) ויהיה נגיש מכל דף באתר לאותו גולש. אובייקט זה מאפשר לנו לכתוב בדף הבית פניה אישית לגולש לפי שמו, ולהציג לגולש תפריט המותאם לו.

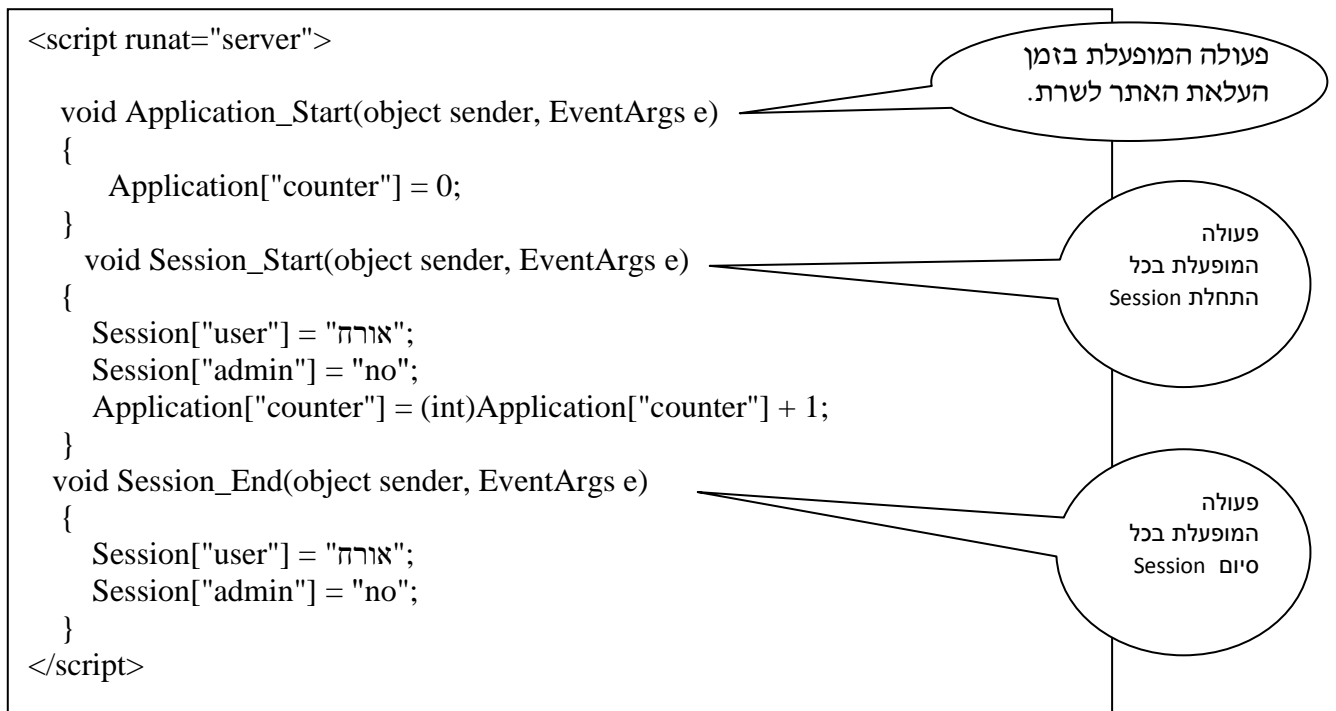
אובייקט Application מאפשר לשמור מידע שיהיה נגיש לכל הגולשים באתר. לדוגמה - מונה מבקרים.

5.3 הקובץ Global.asax

יש להוסיף לפרויקט קובץ חדש בשם Global.asax (בחלון Add New Item ...)

קובץ המכיל נתונים המשותפים לכל הקבצים באתר. משתנים הנגישים לכל קובץ ASP ביישום. הקובץ מכיל שני סוגי נתונים:

1. נתוני האתר. אובייקט Application. המכיל משתנים גלובליים לכל היישום. לדוגמה - מונה מבקרים.
2. נתוני המבקר. אובייקט Session. המכיל משתנים ומידע על כל גולש המגיע לאתר. לדוגמה: אישור כניסה, סוג מבקר.



* המשתנים של אובייקט Application מאותחלים עם העלאת האתר לשרת WEB. אפשר להוסיף משתנים נוספים לפי הצורך. למשל: מחרוזות החיבור לאובייקט ADO. (בפרק הבא)

* המשתנים של אובייקט Session מאותחלים ב- Session_Start בפניה ראשונה של גולש (דפדפן) לדף באתר.

המשתנים של אובייקט Session נמחקים בצורה פעילה, כאשר משתמש לוחץ למשל על כפתור " צא מהאתר ". גם אם המשתמש ימשיך להיות מחובר לאתר הוא לא יהיה מוכר יותר למערכת. אובייקט Session מסוים נמחק אוטומטית אחרי תקופת זמן של חוסר פעילות (כפי שהוגדר בשרת).

המשתנים המוגדרים ותפקידם:

שם	תפקיד	ערכים
Application["counter"]	מונה מבקרים לאתר	מאותחל בפעם הראשונה שהאתר עולה לאוויר. מקודם ב- 1 בכל כניסת גולש לאתר. Session חדש !!
Session["user"]	משתנה השומר את שם המשתמש המחובר.	ערך התחלתי- "אורח" ניתן לגולש אנונימי. המשתנה יכול את שם המשתמש של גולש שהתחבר לאתר.
Session["admin"]	משתנה השומר את הסטאטוס של מנהל האתר.	ערך ברירת מחדל הוא "no". כאשר המנהל מתחבר למערכת הערך ישתנה ל- "yes".

5.1 תרשיף

1. הוסיפו לאתר את הקובץ Global.asax. בחלון Add New Item לבחור "Global Application Class". ולאשר. אין לשנות את שם הקובץ.
2. הגדירו בו את המשתנים.

5.4 שליחת נתונים מהשרת אל הלקוח (הלקוח)

תגובת השרת לבקשה ממשתמש - שליחת נתונים - מתבצעת בשרת בפרוטוקול HTTP Response. השרת מכין את הדף ושולח למשתמש דף HTML. אך מה קורה אם בדף יש נתונים דינמיים המשתנים מהרצה להרצה? לאובייקט Response יש פעולה Write() המאפשרת לשרת לשלוח למשתמש נתונים התלויים במצב. הפעולה Response.Write() מקבלת מחרוזת שיכולה לכלול גם תגיות HTML.

```
<%Response.Write(" שלום " + Session["user"]+ <br />); %>
```

```
<%Response.Write(" : לאתר נכנסו עכשו עד"); %>
```

כדי להציג את שם המשתמש איננו יכולים להשתמש בתגית HTML רגילה שבה אנחנו מציגים מידע סטטי. הצגת שם המשתמש היא מידע דינמי. תוכנו משתנה בהתאם לסטאטוס של הגולש. (אנונימי או חבר מחובר).

פקודת שרת מסומנת ע"י התגית `<%` ונכתבות באזור `body` של הדף. השרת מתרגם סימנים אלו כתסריט שעליו לבצע לפני משלוח הדף למשתמש. הדף שנשלח למשתמש מכיל רק את הפלט של הפעולה.

5.2 תרשיף

1. בדף `regist.aspx.cs` הוסיפו את הקוד לקליטת שם משתמש ושנת הלידה. (ראו דוגמה לעיל)
- בדף `MasterPage` בצעו את השינויים הבאים:
2. הוסיפו את התגובה "שלום" + שם משתמש (`Session["user"]`) באזור תיבת האפשרויות לגולש בכותרת הדף (פרק 4.4 סעיף 4). מהו השם המוצג?
2. הוסיפו את מונה המבקרים באזור הכותרת התחתונה.
3. הריצו את האתר. שימו לב למספר במונה המבקרים.
- * האם רענון הדף (F5) משנה את המונה?
- * סיגרו את הדפדפן והריצו את האתר שנית. האם המונה התקדם?

כאשר מריצים את האתר בפעם הראשונה מופעל השרת של Web Developer - זהו המצב של Application_Start. ומיד מתבצע גם Session_Start ראשון. אם סוגרים את הדפדפן מתבצעת הפעולה Session_End. לכן הרצה נוספת של האתר תתחיל Session חדש. השרת אינו נסגר.

5.5 התנתקות

בסעיף הקודם ראינו כי אחרי שליחת נתוני הרישום מהטופס, הוכר המשתמש כחבר רשום באתר.

(בפרק הבא נלמד לבצע התחברות- Login - עם בדיקת הנתונים במאגר)

סיום יזום של ההתחברות נעשה ע"י סגירת ה- Session - Session.Abandon();

במקרה זה תופעל פעולה Session_End (בקובץ Global.asax), המאפסת את נתוני ה- Session.

נוסיף לפרויקט דף חדש בשם Logout.aspx.

בדף זה תהיה רק פעולה אחת בקובץ הקוד Logout.aspx.cs.

```
protected void Page_Load(object sender, EventArgs e)
{
    Session.Abandon(); ← סגירת Session
    Response.Redirect("Default.aspx"); ← חזור לדף הבית
}
```

בסעיף הבא נבצע קישור לדף ההתנתקות.

5.6 תפריט אישי מותאם למשתמש

שימוש בערך הנמצא ב-Session["user"] יאפשר לנו להציג תפריט אישי מותאם למשתמש.

אחרי הרישום
יוצג קישור
להתנתקות .



בכניסת אורח יוצג
קישור לדף הרישום



```
<table width="50" border="1">
<tr >
<td bgcolor="#EBEBEB">
<% if (Session["user"] == "אורח")
{ %>
<a href="regist.aspx">רישום</a>
<% }
else
{ %>
<a href="Logout.aspx">התנתקות</a>
<% } %>
</td>
</tr>
</table>
```

הקוד נרשם בדף MasterPage באזור תפריט אישי מותאם לגולש.

שימו לב ♥ : תגיות פקודות שרת <% %> עוטפות את כל מרכיבי if else .

תרגיל 5.3

1. הוסיפו את דף ההתנתקות Logout.aspx לפרויקט שלכם.
2. הוסיפו את הקישורים בתפריט המותאם למשתמש.
3. הריצו את האתר , הרשמו כחבר חדש , ובדקו כי התצוגה מותאמת למשתמש.

5.7 דפיט מולאניס .

אתם מכירים מצב שבו אינכם יכולים להיכנס לדפים באתר ומוצגת הודעה "לחברים בלבד".
באתר יהיו דפים המיועדים לחברים בלבד או למנהל בלבד .
שימוש בערך הנמצא ב-Session["user"] יאפשר לנו להסתיר או להציג בהתאם לסטאטוס של המשתמש.
בכל דף המיועד לחברים בלבד נוסיף בדיקה . הקוד נכתב הראש הדף .

```
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
<center>
  <% if (Session["user"] == "אורח")
  {
    Response.Write(" בלבד לחברים מיועד זה דף ");
  }
  else
  { %> </center>
                                     המשך הקוד לדף .....
  <% } %>
</asp:Content>
```

לדף המיועד למנהל בלבד נבדוק את Session["admin"]

```
<% if (Session["admin"] = "no")
{
  Response.Write("למנהל בלבד");
}
else
{ %> </center>
```

שימוש בקוד זה ימנע כניסה לדף למשתמש לא מורשה גם אם הוא יודע את כתובתו של הדף .
את הטיפול במשתנה Session["admin"] נלמד בפרק הבא.

5.4 תרדיל

1. בחרו את אחד הדפים והפכו אותו לדף מוגן לחברים בלבד . (למשל את דף האנימציה)
2. הריצו ובדקו במצב אורח ובמצב חבר .
3. כאשר אתם מריצים את דף האנימציה העתיקו את הכתובת מהדפדפן .
זוהי הכתובת בדפדפן אצלי : <http://localhost:2938/demodita1/Game.aspx>
4. פתחו לשוניית חדשה בדפדפן והדביקו את הכתובת שהעתקתם . מה מוצג ?
האם תוכלו להסביר ?

סיכום:

בפרק זה למדנו לשלוח ולקבל נתונים בשרת, ולהציג למשתמש דפים המותאמים לו.

בפרק הבא נתחיל בלימוד שימוש במאגר נתונים לשמירת הנתונים.